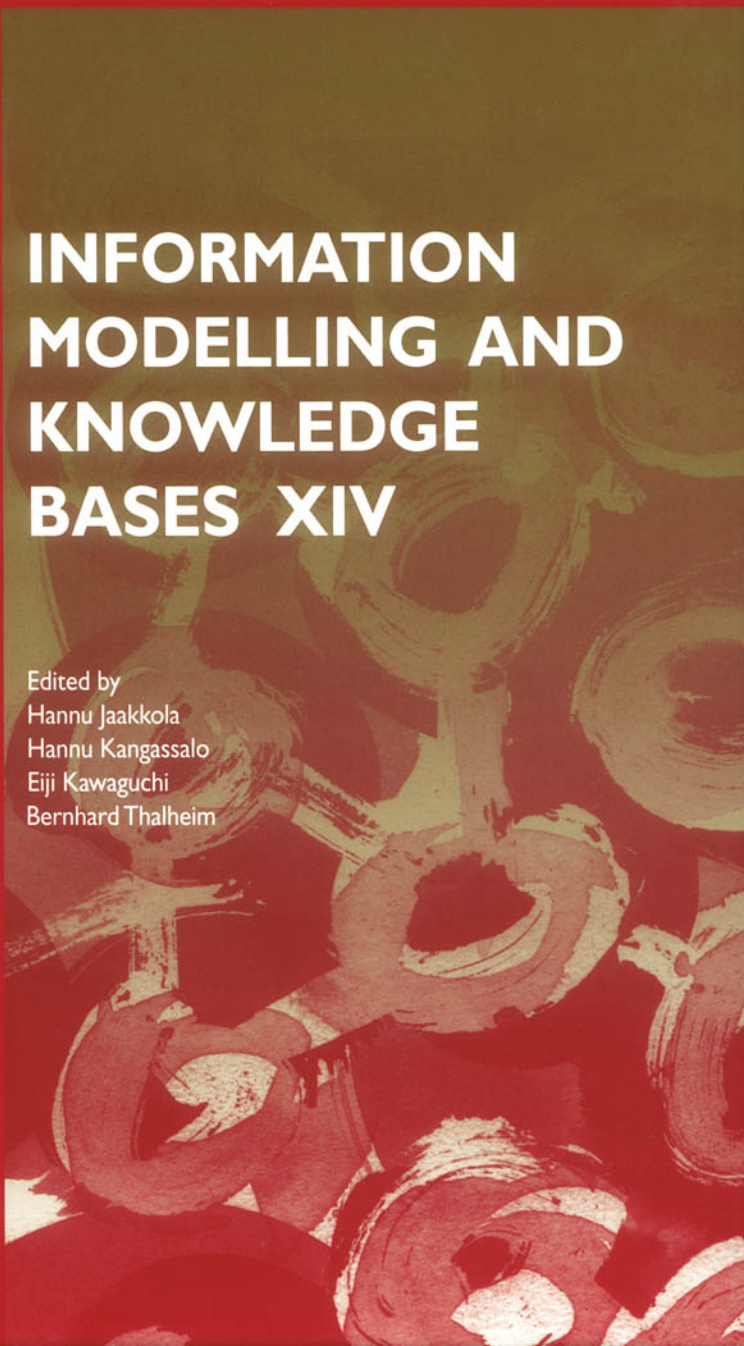
Frontiers in Artificial Intelligence and Applications

INFORMATION MODELLING AND KNOWLEDGE BASES XIV

Edited by
Hannu Jaakkola
Hannu Kangassalo
Eiji Kawaguchi
Bernhard Thalheim



IOS Press
Ohmsha

INFORMATION MODELLING AND KNOWLEDGE BASES XIV

Frontiers in Artificial Intelligence and Applications

Series Editors: J. Breuker, R. López de Mántaras, M. Mohammadian, S. Ohsuga and W. Swartout

Volume 94

Recently published in this series:

Vol. 93. K. Wang, Intelligent Condition Monitoring and Diagnosis Systems – A Computational Intelligence
Vol. 92. V. Kashyap and L. Shklar (Eds.), Real World Semantic Web Applications
Vol. 91. F. Azevedo, Constraint Solving over Multi-valued Logics – Application to Digital Circuits
Vol. 90. In preparation
Vol. 89. T. Bench-Capon et al. (Eds.), Legal Knowledge and Information Systems – JURIX 2002: The Fifteenth Annual Conference
Vol. 88. In preparation
Vol. 87. A. Abraham et al. (Eds.), Soft Computing Systems – Design, Management and Applications
Vol. 86. R.S.T. Lee and J.H.K. Liu, Invariant Object Recognition based on Elastic Graph Matching – Theory and Applications
Vol. 85. J.M. Abe and J.I. da Silva Filho (Eds), Advances in Logic, Artificial Intelligence and Robotics – LAPTEC 2002
Vol. 84. H. Fujita and P. Johannesson (Eds.), New Trends in Software Methodologies, Tools and Techniques – Proceedings of Lyee_W02
Vol. 83. V. Loia (Ed.), Soft Computing Agents – A New Perspective for Dynamic Information Systems
Vol. 82. E. Damiani et al. (Eds.), Knowledge-Based Intelligent Information Engineering Systems and Allied Technologies – KES 2002
Vol. 81. J.A. Leite, Evolving Knowledge Bases – Specification and Semantics
Vol. 80. T. Welzer et al. (Eds.), Knowledge-based Software Engineering – Proceedings of the Fifth Joint Conference on Knowledge-based Software Engineering
Vol. 79. H. Motoda (Ed.), Active Mining – New Directions of Data Mining
Vol. 78. T. Vidal and P. Liberatore (Eds.), STAIRS 2002 – STarting Artificial Intelligence Researchers Symposium
Vol. 77. F. van Harmelen (Ed.), ECAI 2002 – 15th European Conference on Artificial Intelligence
Vol. 76. P. Sinčák et al. (Eds.), Intelligent Technologies – Theory and Applications
Vol. 75. I.F. Cruz et al. (Eds.), The Emerging Semantic Web – Selected Papers from the first Semantic Web Working Symposium
Vol. 74. M. Blay-Fornarino et al. (Eds.), Cooperative Systems Design – A Challenge of the Mobility Age
Vol. 73. H. Kangassalo et al. (Eds.), Information Modelling and Knowledge Bases XIII
Vol. 72. A. Namatame et al. (Eds.), Agent-Based Approaches in Economic and Social Complex Systems
Vol. 71. J.M. Abe and J.I. da Silva Filho (Eds.), Logic, Artificial Intelligence and Robotics – LAPTEC 2001
Vol. 70. B. Verheij et al. (Eds.), Legal Knowledge and Information Systems – JURIX 2001: The Fourteenth Annual Conference
Vol. 69. N. Baba et al. (Eds.), Knowledge-Based Intelligent Information Engineering Systems and Allied Technologies – KES'2001
Vol. 68. J.D. Moore et al. (Eds.), Artificial Intelligence in Education – AI-ED in the Wired and Wireless Future
Vol. 67. H. Jaakkola et al. (Eds.), Information Modelling and Knowledge Bases XII
Vol. 66. H.H. Lund et al. (Eds.), Seventh Scandinavian Conference on Artificial Intelligence – SCAI'01

ISSN: 0922-6389

Information Modelling and Knowledge Bases XIV

Edited by

Hannu Jaakkola

Tampere University of Technology, Finland

Hannu Kangassalo

University of Tampere, Finland

Eiji Kawaguchi

Kyushu Institute of Technology, Japan

and

Bernhard Thalheim

Brandenburg University of Technology at Cottbus, Germany

IOS Press

Ohmsha

Amsterdam • Berlin • Oxford • Tokyo • Washington, DC

ISBN 1 58603 318 2 (IOS Press)
ISBN 4 274 90574 8 C3055 (Ohmsha)
Library of Congress Control Number: 2002117112

Publisher
IOS Press
Nieuwe Hemweg 6B
1013 BG Amsterdam
The Netherlands
fax: +31 20 620 3419
e-mail: order@iospress.nl

Distributor in the UK and Ireland
IOS Press/Lavis Marketing
73 Lime Walk
Headington
Oxford OX3 7AD
England
fax: +44 1865 75 0079

Distributor in the USA and Canada
IOS Press, Inc.
5795-G Burke Centre Parkway
Burke, VA 22015
USA
fax: +1 703 323 3668
e-mail: iosbooks@iospress.com

Distributor in Germany, Austria and Switzerland
IOS Press/LSL.de
Gerichtsweg 28
D-04103 Leipzig
Germany
fax: +49 341 995 4255

Distributor in Japan
Ohmsha, Ltd.
3-1 Kanda Nishiki-cho
Chiyoda-ku, Tokyo 101-8460
Japan
fax: +81 3 3233 2426

PRINTED IN THE NETHERLANDS

Preface

This book includes the papers presented at the 12th European-Japanese Conference on Information Modelling and Knowledge Bases. The conference held in May 2001 in Krippen, Germany, continues the series of events that originally started as a co-operation initiative between Japan and Finland, already in the last half of the 1980's. Later (1991) the geographical scope of these conferences has expanded to cover the whole Europe and other countries, too.

The aim of this series of conferences is to provide research communities in Europe and Japan a forum for the exchange of scientific results and experiences achieved using innovative methods and approaches in computer science and other disciplines, which have a common interest in understanding and solving problems on information modelling and knowledge bases, as well as applying the results of research to practice.

The topics of research in this conference were mainly concentrating on a variety of themes in the domain of theory and practice of information modelling, conceptual modelling, design and specification of information systems, software engineering, databases and knowledge bases. We also aim to recognize and study new areas of modelling and knowledge bases to which more attention should be paid. Therefore philosophy and logic, cognitive science, knowledge management, linguistics and management science are relevant areas, too. This time the selected papers cover many areas of information modelling, e.g.:

- concept theories
- logic of discovery
- logic of relevant connectives
- database semantics
- semantic search space integration
- context-base information access space
- defining interaction patterns
- embedded programming as a part of object design
- UML state chart diagrams.

The published papers are formally reviewed by an international program committee and selected for the annual conference forming a forum for presentations, criticism and discussions, taken into account in the final published versions. Each paper has been reviewed by three or four reviewers. The selected papers are printed in this volume.

This effort had not been possible without support from many people and organizations. In the Programme Committee there were 28 well-known researchers from the areas of information modelling, logic, philosophy, concept theories, conceptual modelling, data bases, knowledge bases, information systems, linguistics, and related fields important for information modelling. In addition, 24 external referees gave invaluable help and support in the reviewing process. We are very grateful for their careful work in reviewing the papers. Professor Eiji Kawaguchi and Professor Hannu Kangassalo were acting as co-chairmen of the program committee.

Brandenburg University of Technology at Cottbus, Germany was hosting the conference. Professor Bernhard Thalheim was acting as a conference leader. His team took care of the practical aspects which were necessary to run the conference, as well as all those things which were important to create an innovative and creative atmosphere for the hard work during the conference days.

The Editors

Hannu Jaakkola
Hannu Kangassalo
Eiji Kawaguchi
Bernhard Thalheim

Program Committee

Alfs Berztiss, University of Pittsburgh, USA
Pierre-Jean Charrel, Universite Toulouse 1, France
Valeria De Antonellis, Politecnico di Milano, Universita' di Brescia, Italy
Olga De Troyer, Vrije Universiteit Brussel, Belgium
Marie Duží, Technical University of Ostrava, Czech Republic
Yutaka Funyu, Iwate Prefectural University, Japan
Wolfgang Hesse, University of Marburg, Germany
Seiji Ishikawa, Kyushu Institute of Technology, Japan
Yukihiro Itoh, Shizuoka University, Japan
Manfred A. Jeusfeld, Tilburg University, The Netherlands
Martti Juhola, University of Tampere, Finland
Hannu Kangassalo, University of Tampere, Finland (Co-chairman)
Eiji Kawaguchi, Kyushu Institute of Technology, Japan (Co-chairman)
Isabelle Mirbel-Sanchez, Université de Nice Sophia Antipolis, France
Björn Nilsson, Astrakan Strategic Development, Sweden
Setsuo Ohsuga, Waseda University, Japan
Yoshihiro Okade, Kyushu University, Japan
Antoni Olivé, Universitat Politecnica Catalunya, Spain
Jari Palomäki, University of Tampere, Finland
Christine Parent, University of Lausanne, Switzerland
Alain Pirotte, University of Louvain, Belgium
Veikko Rantala, University of Tampere, Finland
Michael Schrefl, University of Linz, Austria
Cristina Sernadas, Instituto Superior Tecnico, Portugal
Arne Sølvberg, Norwegian University of Science and Technology, Norway
Yuzuru Tanaka, University of Hokkaido, Japan
Bernhard Thalheim, Brandenburg University of Technology at Cottbus, Germany
Takehiro Tokuda, Tokyo Institute of Technology, Japan
Benkt Wangler, University of Skövde, Sweden
Esteban Zimanyi, Université Libre de Bruxelles (ULB), Belgium

Organizing Committee

Bernhard Thalheim, Brandenburg University of Technology at Cottbus, Germany
Hannu Jaakkola, Tampere University of Technology, Pori, Finland
Karla Kersten (Conference Office), Brandenburg University of Technology at Cottbus, Germany
Thomas Kobienia (Technical Support), Brandenburg University of Technology at Cottbus, Germany
Thomas Feyer, Brandenburg University of Technology at Cottbus, Germany
Steffen Jurk, Brandenburg University of Technology at Cottbus, Germany
Roberto Kockrow (WWW), Brandenburg University of Technology at Cottbus, Germany
Vojtech Vestenický, Brandenburg University of Technology at Cottbus, Germany
Heiko Wolf (WWW), Brandenburg University of Technology at Cottbus, Germany
Ulla Nevanranta (Publication), Tampere University of Technology, Pori, Finland

Permanent Steering Committee

Hannu Jaakkola, Tampere University of Technology, Pori, Finland
Hannu Kangassalo, University of Tampere, Finland
Eiji Kawaguchi, Kyushu Institute of Technology, Japan
Setsuo Ohsuga, Waseda University, Japan (Honorary member)

Additional Reviewers

Kazuhiro Asami, Tokyo Institute of Technology, Japan
Per Backlund, University of Skövde, Sweden
Sven Casteleyn, Vrije Universiteit Brussel, Belgium
Thomas Feyer, Brandenburg University of Technology at Cottbus, Germany
Paula Gouveia, Lisbon Institute of Technology (IST), Portugal
Ingi Jonasson, University of Skövde, Sweden
Steffen Jurk, Brandenburg University of Technology at Cottbus, Germany
Makoto Kondo, Shizuoka University, Japan
Stephan Lechner, Johannes Kepler University, Austria
Michele Melchiori, University of Brescia, Italy
Erkki Mäkinen, University of Tampere, Finland
Jyrki Nummenmaa, University of Tampere, Finland
Günter Preuner, Johannes Kepler University, Austria
Roope Raisamo, University of Tampere, Finland
Jaime Ramos, Lisbon Institute of Technology (IST), Portugal
Joao Rasga, Lisbon Institute of Technology (IST), Portugal
Yutaka Sakane, Shizuoka University, Japan
Jun Sakaki, Iwate Prefectural University, Japan
Mattias Strand, University of Skövde, Sweden
Tetsuya Suzuki, Tokyo Institute of Technology, Japan
Eva Söderström, University of Skövde, Sweden
Mitsuhisa Taguchi, Tokyo Institute of Technology, Japan
Shiro Takata, ATR, Japan
Yoshimichi Watanabe, Yamanashi University, Japan

Contents

Information Modelling and Knowledge Bases XIV
H. Jaakkola et al. (Eds.)
IOS Press, 2003

A Logical Treatment of Concept Theories

Klaus-Dieter Schewe

Massey University, Department of Information Systems
Private Bag 11 222, Palmerston North, New Zealand
K.D.Schewe@massey.ac.nz

Abstract. The work reported in this article continues investigations in a theoretical framework for Concept Theories based on mathematical logic. The general idea is that the intension of a concept is defined by some equivalence class of theories, whereas the extension is given by the models of the theory. The fact that extensions depend on structures that are necessary to interpret the formulae of the logic, already provides an argument to put more emphasis on the intension.

Starting from the simple Ganter-Wille theory of formal concept analysis first-order theories that are interpreted in a fixed structure or in more than one structure are introduced. The Ganter-Wille Concept Theory turns out to be a very special case, where the logical signature contains no function symbols nor constants and only monadic predicate symbols.

It can easily be shown that first-order Concept Theories lead to lattices. Thus, they are *Kauppian Concept Theories*, i.e., it satisfies the axioms defined by the philosopher Raili Kauppi. However, not all Kauppian Concept Theories define lattices. Furthermore, in all these cases of first-order Concept Theories the extension(s) already determine the intension, which slightly contradicts the desire of concept theorists to distinguish strictly between intension and extension of concepts.

Switching from classical first-order logic to intuitionistic first-order logic removes this "contradiction". The order on intensions is defined via forcing, whereas the order on extensions is still based on set inclusion. However, the fact that we still get lattices, remains unchanged. It disappears only, if "absurd concepts", i.e., concepts with a logically contradictive intension, are excluded. Such concepts would never—under no interpretation—possess any entities that fall under it. In fact, this leads to *pseudo-Kauppian Concept Theories* by missing out exactly one of the axioms. Pseudo-Kauppian Concept Theories can be easily characterized by structures that result from duals of distributed, pseudo-complemented lattices with bottom and top elements by depriving them of the greatest element.

1 Introduction

What are concepts? Despite decades of Conceptual Modelling, a general agreement of its necessity, lots of conferences on the topic, and an IFIP task force on Information

Systems Concepts, there is still no agreement on this. In particular, there is no agreed mathematical framework for studying Concept Theories.

In this article we continue a line of thought, which aims at bringing clarity to this topic, and especially at developing a theoretical framework in which different Concept Theories can be studied. As outlined in [Feyer et al., 2002] we strongly believe that such a framework should be based on mathematical logic.

Our starting point is the informal definition in [Kangassalo, 1993]. According to this definition a *concept* is defined by its intension and its extension, where the *intension of a concept* is understood as the information content required to recognize a thing belonging to the *extension* of the concept. This is far from being a clear mathematical definition; maybe it is not intended to be one. It is not at all clear how to understand the term "information content". This remains undefined.

However, the underlying assumption is that concepts are used to characterize entities. Otherwise said, there is a fundamental relation denoted as "falls under": an entity *falls under* a concept. The *extension* of a concept C is then the set of all entities falling under C. A minor point—at least for the moment—is that we may want to talk about the concept of sets, in which case the extension cannot be a set anymore, so we have to switch to classes. We dispense with this aspect for the moment.

Characterizing entities can be done by using logic (of any kind). A set of formulae in a logic is called a *theory*. Thus, the *intension* of a concept could be defined as a logical theory. As a consequence, the falls-under-relation would become the satisfaction relation. Forgetting about the entities, the extension would become a *model* of the theory. Thus, for a given logical signature Σ a *concept* is a triple $(C, int(C), ext(C))$, where C is just a name for the concept, $int(C)$ is a theory over Σ and $ext(C)$ is a model for $int(C)$.

More formally, let C be a concept. We associate with C some logical theory ϕ_C. We would like to restrict the theory ϕ_C such that only monadic formulae, i.e., formulae with exactly one free variable, appear in ϕ_C. So we have only *monadic theories*. As a start we leave the question open which should be the underlying logic. Just think of first-order predicate logic as the first natural choice. We also leave it for the discussion, whether we should identify the concept C, at least its intention $int(C)$, with the theory ϕ_C. For instance, we could at least think of equivalence classes of theories as being the intensions of concepts.

Then we can choose a *structure* $\mathcal{S}$ that allows us to interpret the formulae in ϕ_C. In order to assign truth values to formulae, especially thos in ϕ_C, we need a valuation σ, which assign a value in the domain of the structure to each variable. Thus, we could define

$$\mathcal{E}[\![C]\!] \quad = \quad \mathcal{E}_{\mathcal{S}}[\![C]\!] \quad = \quad \{\sigma(x) \mid\models_{(\mathcal{S},\sigma)} \varphi(x) \text{ for all } \varphi \in \phi_C\} \quad .$$

as the extension of the concept C. This definition depends on the chosen structure. So, in order to be exact, we should say that $\mathcal{E}[\![C]\!]$ is the extension of C with respect to the structure $\mathcal{S}$.

In this article, we proceed with this line of thought, and try to strengthen the argumentation. We start with an alysis of Ganter's and Wille's "Formal Concept Analysis"

[Ganter and Wille, 1999]. This theory has shown to have several nice applications in bringing order into collections of empirical data. Many researchers in Concept Theory, however, consider this theory as being far too simple, and thus not sufficient for really laying the foundations of a mathematical theory of concepts. We agree with this point of view, but nevertheless think it is wortwhile to take a look into this theory from a logical point of view, which will help to understand the more general framework. In this theory the notions of intension and extension become so easy to grasp that it will give us some guidance when approaching more complicated Concept Theories. In particular, the theory of Ganter and Wille starts with a fixed structure, which they call *context*.

We shall see that we can always consider the intension of a concept in Ganter's and Wille's theory is in a logical sense restricted to atomic formulae. The obvious first generalization of the theory is to switch to general monadic, first-order formulae, but to stay first with a fixed structure. So we obtain theories of 'Concepts in a Context' with the Ganter-Wille theory being one of the easiest examples.

Then we consider the axiomatic approach to Concept Theory as defined by the philosopher Raili Kauppi [Kauppi, 1967]. According to the huge interest in Concept Theories based on her systems of axioms we will talk of *Kauppian Concept Theories*—some authors will claim that these are all Concept Theories. We briefly review these axioms and show that any first-order Concept Theory is indeed Kauppian. These theories even satisfies much stronger axioms defining a concept lattice. This results from the fact that the concept, i.e., the intention of the concept, is already determined by its extension. Obviously, not every Kauppian Concept Theory will be a theory of 'Concepts in a Context'.

We proceed with dropping the restriction to a single predefined structure. The major difference is now that we obtain several structure-dependent extensions for one concept, but we still have just one intension. Nevertheless, these Concept Theories are still Kauppian. The extensions with respect to the relevant structures will still determine the intension, i.e., that the theory will not lead out of lattices.

Finally, we leave the grounds of classical logic and consider first-order intuitionistic logic [Bell and Machover, 1977, Chapter 9], in which case we will consider forcing with respect to the intensions in order to define an order on concepts. In this case, the order on intensions will still imply an order on extensions, which is defined again via models, but the extensions will no longer determine the intensions. This was always claimed for Concept Theory, but it becomes now clear that this it is not achievable in easy cases. Surprisingly, however, we still end up with a lattice, so the resulting Concept Theory is Kauppian in a trivial sense, but raises the question, whether Kauppian Concept Theories that are not lattices make any sense.

The major problem arises from "absurd" concepts that will never—under no interpretation—have an extension. Such concepts have inconsistent intensions. If we exclude such concepts from further consideration we leave the boundaries of lattices. However, we also leave the grounds of staying within Kauppian Concept Theories. The differences are small. If we just drop one axiom, calling the result *pseudo-Kauppian*, we capture all the theories studied in this article. The resulting structures are quite close to duals of distributive, pseudo-complemented lattice with greatest and least element: we just have to drop the top element.

2 The Concept Theory by Ganter & Wille

Ganter's and Wille's theory of *formal concept analysis* [Ganter and Wille, 1999] can be seen as a simple approach to Concept Theory, in which extensional and intensional ideas are combined. Let us briefly review the main definitions of this approach.

Definition 2.1. A *context* is a triple $(\mathcal{O}, \mathcal{A}, I)$ with a set $\mathcal{O}$ of objects, a set $\mathcal{A}$ of attributes and relation $I \subseteq \mathcal{O} \times \mathcal{A}$.

The intension of the relation I is to express that an object has a property given by some attribute. Based on this idea any subset of objects $C \subseteq \mathcal{O}$ is associated with an *intension* $int(C)$:

$$int(C) \quad = \quad \{a \in \mathcal{A} \mid \forall o \in C.(o,a) \in I\} \; .$$

Analogously, each subset of attributes $B \subseteq \mathcal{A}$ is associated with an *extension* $ext(B)$:

$$ext(C) \quad = \quad \{o \in \mathcal{O} \mid \forall a \in B.(o,a) \in I\} \; .$$

Roughly speaking, intension is expressed by the set of common attributes, and extension is given by the set of objects having all the required properties. This leads to the notion of concept in Ganter's and Wille's theory:

Definition 2.2. A *concept* in a context $(\mathcal{O}, \mathcal{A}, I)$ is a pair $(obj, attr)$ with $obj \subseteq \mathcal{O}$ and $attr \subseteq \mathcal{A}$, such that $obj = ext(attr)$ and $attr = int(obj)$ hold.

Thus, according to Ganter and Wille, a concept has an intensional part, formalized by the set of attributes $attr$, and an extensional part, formalized by the set obj of objects. Both the intensional and the extensional part depend on the underlying context. On concepts, we then define a partial order by

$$(obj_1, attr_1) \sqsubseteq (obj_2, attr_2) \quad \Leftrightarrow \quad obj_1 \subseteq obj_2 \quad \Leftrightarrow \quad attr_1 \supseteq attr_2 \quad .$$

Then, it is easy to see that concepts equipped with this partial order define a lattice. This *concept lattice* has a least element $(ext(\mathcal{A}), \mathcal{A})$ and a greatest element $(\mathcal{O}, int((O)))$.

Let us rephrase the theory in logical terms. Instead of a set $\mathcal{A}$ of attributes in a context we start with a first-order relational signature, in which all predicate symbols are monadic.

Definition 2.3. A *relational signature* is a triple $(\mathcal{P}, V, ar)$ with a set $\mathcal{P}$ of predicate symbols, a set V of variables, and a function $ar : \mathcal{P} \to \mathbb{N}$ that assigns to each of the predicate symbols p their *arity* $ar(p)$.

A relational signature $(\mathcal{P}, V, ar)$ is *monadic* iff all predicate symbols are monadic, i.e., $ar(p) = 1$ for all $p \in \mathcal{P}$.

Now we can use this signature to define a logical language $\mathcal{L}$ in the usual way. We obtain the set $\mathcal{T}$ of all terms of $\mathcal{L}$ and the set $\mathcal{F}$ of all formulae of $\mathcal{L}$.

Definition 2.4. The *terms* of the language $\mathcal{L}$ are exactly the variables in V.

Atomic formulae in $\mathcal{L}$ have the form $p(t_1, \ldots, t_n)$ with terms $t_1, \ldots, t_n$ and an n-ary predicate symbol p, i.e., $ar(p) = n$.

Formulae in general are all atomic formulae and all expressions $\neg\varphi$, $\varphi \wedge \psi$, $\varphi \vee \psi$, $\varphi \Rightarrow \psi$, $\forall x.\varphi$, and $\exists x.\varphi$ with formulae φ, ψ and variables x.

In Ganter's and Wille's theory we only consider monadic relational signatures and atomic formulae, i.e., formulae always have the form $p(x)$.

Fixing a *structure* $\mathcal{S}$ for the interpretation of this restricted logic means to take a set $\mathcal{D}$—called the *domain*—and for each predicate symbol $p \in \mathcal{P}$ a subset $\omega(\mathcal{P}) \subseteq \mathcal{D}$. Note that in this is exactly what we had in a context: $\mathcal{D}$ is the set of objects and $\omega(\mathcal{P}) = \{d \in \mathcal{D} \mid (d, p) \in I\}$.

A *valuation* of $\mathcal{L}$ in $\mathcal{S}$ is just a mapping $\sigma : V \to \mathcal{D}$. A *theory* is just a set of formulae. We say that a formulae $p(x)$ is *valid* under the interpretation $(\mathcal{S}, \sigma)$ iff $\sigma(x) \in \omega(\mathcal{P})$ holds and write $\models_{(\mathcal{S},\sigma)} p(x)$ for this. A theory ϕ_C is *valid* under $(\mathcal{S}, \sigma)$ iff each formula $\varphi \in \phi_C$ is valid under $(\mathcal{S}, \sigma)$. Thus, we can define the *extension* of ϕ_C (in the structure $\mathcal{S}$) as

$$\mathcal{E}[\![\phi_C]\!] \quad = \quad \mathcal{E}_{\mathcal{S}}[\![\phi_C]\!] \quad = \quad \{\sigma(x) \in \mathcal{D} \mid \models_{(\mathcal{S},\sigma)} \varphi(x) \text{ for all } \varphi \in \phi_C\} \quad .$$

As we think of a fixed structure $\mathcal{S}$, we normally drop the subscript. Analogously, we can define the *intension* of a set of entities $E \subseteq \mathcal{D}$ as

$$\mathcal{I}[\![(E)]\!] \quad = \quad \mathcal{I}_{\mathcal{S}}[\![(E)]\!] \quad = \quad \{\varphi \mid \models_{(\mathcal{S},\sigma)} \varphi(x) \text{ for all } \sigma \text{ with } \sigma(x) \in E\} \quad .$$

This is the exact rephrasing of Ganter's and Wille's definitions. Thus, in the new terminology their definition of concept turns into the following one.

Definition 2.5. A *concept* C with respect to the structure $\mathcal{S}$ is a pair (ϕ_C, E) consisting of a theory ϕ_C called the *intension* of the concept and set $E \subseteq \mathcal{D}$ of entities called the *intension* of the concept such that $\mathcal{E}[\![\phi_C]\!] = E$ and $\mathcal{I}[\![E]\!] = \phi_C$ hold.

We write $C = (int(C), ext(C))$. It is clear that instead of a single theory ϕ_C we could have taken an equivalence class of theories (with respect to interpretation in $\mathcal{S}$) or a maximal theory. Chosing a maximal theory ϕ would give us $\mathcal{I}[\![\mathcal{E}[\![\phi]\!]]\!] = \phi$. This implies that the intension uniquely determines the extension and vice versa.

The partial order on concepts easily translates into the logical setting. We get

$$C_1 \preceq C_2 \quad \Leftrightarrow \quad int(C_2) \models int(C_1) \quad \Leftrightarrow \quad ext(C_2) \subseteq int(C_1) \quad ,$$

which underlines again that the theory is fully determined by the extension, hence by sets of entities.

3 Theories of 'Concepts in a Context'

The Ganter-Wille Concept Theory suggests an easy generalisation on the basis of first-order logic. We now take full advantage of all kinds of signatures, but still stay with monadic theories and a fixed structure $\mathcal{S}$. We briefly review the logical fundamentals and then define concepts in this slightly more general setting.

Definition 3.1. A *signature* consists of a set $\mathcal{P}$ of predicate symbols, a set $\mathcal{O}$ of function or operator symbols, and a set V of variables. Each predicate symbol p and each function symbol f has an *arity* $ar(p)$ or $ar(f)$, respectively. 0-ary function symbols are also called *constants*.

We use this signature to define a logical language $\mathcal{L}$ in the usual way. We obtain the set $\mathcal{T}$ of all terms of $\mathcal{L}$ and the set $\mathcal{F}$ of all formulae of $\mathcal{L}$.

Definition 3.2. The *terms* of the language $\mathcal{L}$ are the variables in V and expressions $f(t_1, \ldots, t_n)$ with terms $t_1, \ldots, t_n$ and an n-ary function symbol f, i.e., $ar(f) = n$.

Atomic formulae in $\mathcal{L}$ have the form $p(t_1, \ldots, t_n)$ with terms $t_1, \ldots, t_n$ and an n-ary predicate symbol p, i.e., $ar(p) = n$.

Formulae in general are all atomic formulae and all expressions $\neg\varphi$, $\varphi \wedge \psi$, $\varphi \vee \psi$, $\varphi \Rightarrow \psi$, $\forall x.\varphi$, and $\exists x.\varphi$ with formulae φ, ψ and variables x.

Free variables in formulae are defined as usual. We shall only consider formulae φ with exactly one free variable. For convenience this variable will be denoted by x, and we write $\varphi(x)$ to emphasize this. A *theory* is still a set of formulae; a *monadic theory* contains only formulae with exactly one free variable. To avoid later complications with renaming we assume that the formulae in a monadic theory all have the same free variable x.

Definition 3.3. A *structure* $\mathcal{S}$ consists of a set $\mathcal{D}$ called the *domain*, mappings $\omega(f) : \mathcal{D}^n \to \mathcal{D}$ for each n-ary function symbol f, and subsets $\omega(\mathcal{P}) \subseteq \mathcal{D}^n$ for each n-ary predicate symbol p.

A *valuation* of $\mathcal{L}$ in $\mathcal{S}$ is just a mapping $\sigma : V \to \mathcal{D}$.

We omit the standard definition of the interpretation ω of terms by elements in $\mathcal{D}$ and formulae by truth values $\mathbf{T}$ and $\mathbf{F}$ under an interpretation $(\mathcal{S}, \sigma)$ (for details see [Bell and Machover, 1977]). A formulae $\varphi(x)$ is *valid* under $(\mathcal{S}, \sigma)$ iff $\omega(\varphi) = \mathbf{T}$ holds. We write $\models_{(\mathcal{S},\sigma)} \varphi(x)$ for this. A theory ϕ_C is *valid* under $(\mathcal{S}, \sigma)$ iff each formula $\varphi \in \phi_C$ is valid under $(\mathcal{S}, \sigma)$.

Thus, we can again define the *extension* of ϕ_C (in the structure $\mathcal{S}$) as

$$\mathcal{E}[\![\phi_C]\!] \quad = \quad \mathcal{E}_{\mathcal{S}}[\![\phi_C]\!] \quad = \quad \{\sigma(x) \in \mathcal{D} \mid\mid\!\models_{(\mathcal{S},\sigma)} \varphi(x) \text{ for all } \varphi \in \phi_C\} \quad .$$

and analogously the *intension* of a set of entities $E \subseteq \mathcal{D}$ as

$$\mathcal{I}[\![E]\!] \quad = \quad \mathcal{I}_{\mathcal{S}}[\![E]\!] \quad = \quad \{\varphi \mid\mid\!\models_{(\mathcal{S},\sigma)} \varphi(x) \text{ for all } \sigma \text{ with } \sigma(x) \in E\} \quad .$$

This generalises Ganter's and Wille's definitions. As we assume a fixed structure $\mathcal{S}$, we drop the subscript.

Definition 3.4. A *concept C in the structure $\mathcal{S}$* is a pair (ϕ_C, E) consisting of a maximal theory ϕ_C called the *intension* of the concept and set $E \subseteq \mathcal{D}$ of entities called the *extension* of the concept such that $\mathcal{E}[\![\phi_C]\!] = E$ holds.

We write $C = (int(C), ext(C))$. As our definition assumes the intention to be a maximal theory with respect to interpretation in $\mathcal{S}$ we do not have to take care of theory equivalence. We get $\mathfrak{I}[\![ext(C)]\!] = int(C)$. As with the Ganter-Wille theory the intension uniquely determines the extension and vice versa. We can define a partial order on concepts by

$$C_1 \preceq C_2 \quad \Leftrightarrow \quad int(C_2) \models int(C_1) \quad \Leftrightarrow \quad ext(C_2) \subseteq int(C_1) \quad .$$

Proposition 3.5. *Concepts in a structure $\mathcal{S}$ together with the partial order $\preceq$ define a complete, distributive lattice.* □

4 Kauppian Concept Theories

Having generalized Ganter's and Wille's Concept Theory in a setting based on first-order logic, we discovered two facts:

- extensions and intensions determine each other;
- the mathematical structure behind the theories is that of a complete, distributive lattice.

The axiomatic Concept Theory introduced by Raili Kauppi, however, leads to mathematical structures that subsume lattices, but are not exhausted by them. Therefore, it is advisable to take a look at these axioms again [Kauppi, 1967]. When referring to a "Concept Theory" $\mathfrak{C}$, we mean the way concepts are defined, but we also use the notation $\mathfrak{C}$ for the set (or class) of all concepts defined by that theory.

Definition 4.1. A Concept Theory $\mathfrak{C}$ is called *Kauppian* iff it defines a partial order[1] $\preceq$ on the set of concepts $\mathfrak{C}$ satisfying the following six conditions:

(i) there exists a least concept $\bot$ with $\bot \preceq C$ for all concepts C;
(ii) for any two concepts C_1 and C_2 their meet $C_1 \sqcap C_2$ exists, i.e., $C_1 \sqcap C_2 \preceq C_1$ and $C_1 \sqcap C_2 \preceq C_2$ hold, and for any concept C satisfying $C \preceq C_1$ and $C \preceq C_2$ we get also $C \preceq C_1 \sqcap C_2$;
(iii) for each concept C there is a maximal concept $C_{\max}$ above C, i.e., $C \preceq C_{\max}$ and any concept C' with $C_{\max} \preceq C'$ satisfies $C_{\max} = C'$;
(iv) for any two concepts C_1 and C_2 with a common concept above them their join $C_1 \sqcup C_2$ exists, i.e., whenever there exists a concept C' with $C_1 \preceq C'$ and $C_2 \preceq C'$, there exist a concept $C_1 \sqcup C_2$ with $C_1 \preceq C_1 \sqcup C_2$ and $C_2 \preceq C_1 \sqcup C_2$ such that for any concept C satisfying $C_1 \preceq C$ and $C_2 \preceq C$ we get also $C_1 \sqcup C_2 \preceq C$;
(v) the meet is distributive over the join, i.e., for any concepts C_1, C_2 and C_3 we obtain (either both sides are defined are none of them)

$$C_1 \sqcap (C_2 \sqcup C_3) \quad = \quad (C_1 \sqcap C_2) \sqcup (C_1 \sqcap C_3) \; ;$$

[1] The original set of axioms defined by Kauppi does not require anti-symmetry for $\preceq$. So in principle, the axioms we formulate here refer to equivalence classes of concepts. It is a matter of taste, whether we should stay with such equivalence classes, or whether we should simply claim that equivalent concepts should be identified.

(vi) for any concept C such that there exists a concept C' without a join $C \sqcup C'$, there exist a least such concept denoted $\bar{C}$ and called the *pseudo-complement* of C, i.e., $C \sqcup \bar{C}$ is not defined, and any C', for which $C \sqcup C'$ is not defined, satisfies $\bar{C} \preceq C'$.

A Concept Theory $\mathfrak{C}$ satisfying these conditions except (iii) is called *pseudo-Kauppian*.

As an obvious consequence of our investigation in the previous sections we obtain that all first-order Concept Theories in a Context, i.e., assuming a fixed structure, are indeed Kauppian.

In many articles referencing Kauppi's system of axioms, e.g., in [Palomäki, 1994] the partial order $\preceq$ is called *intensional containment*. We dropped this notion, as the Concept Theories of the preceding sections are all Kauppian, but at the same time intensions of concepts in these theories are determined by extensions.

Also, the meet $C_1 \sqcap C_2$ is often called the *intensional product* of C_1 and C_2, the join $C_1 \sqcup C_2$ is called the *intensional sum* of C_1 and C_2 (provided it exists), and the pseudo-complement $\bar{C}$ is called the *negation* of C (provided it exists).

Pseudo-complements are unique, and it can be shown that the dual distributivity rule stating that the join is distributive over the meet also holds, i.e., for any concepts C_1, C_2 and C_3 we obtain

$$C_1 \sqcup (C_2 \sqcap C_3) \quad = \quad (C_1 \sqcup C_2) \sqcap (C_1 \sqcup C_3)\,;$$

Suppose we are given a Kauppian Concept Theory $(\mathfrak{C}, \preceq)$. We could ask what happens, if we simply add a new concept $\top$ and extend $\preceq$ in a way that $C \preceq \top$ holds for all concepts C, unless such a concept exists already, i.e., $\top$ would become a greatest concepts intensionally containing all concepts. For a mathematician this is a legitimate approach, whereas a philosopher might ask, whether this new top concept makes sense with respect to what Concept Theory should formalize.

Anyway, if we add such a greatest concept $\top$, all the axioms (i)–(vi) would still hold. We would even get the following:

(vii) there is a greatest concept $\top$;
(viii) all joins $C_1 \sqcup C_2$ exist; the join of concepts that previously were incompatible, is the new top concept $\top$;
(ix) for each concept C there would be a pseudo-complement $\bar{C}$, which is the least element satisfying $C \sqcup \bar{C} = \top$.

In summary, the extended Concept Theory defines the dual of a distributive, pseudo-complemented lattice with least and greatest elements. Let us call this a semi-Heyting lattice. This would also be the case, if our initial Concept Theory were only pseudo-Kauppian.

The term "semi-Heyting lattice" has been chosen, because a *Heyting algebra* $(\mathcal{H}, \leq)$ is a distributive, relatively pseudo-complemented lattice, with least and greatest element, i.e., for any two elements $a, b \in \mathcal{H}$ the *pseudo-complement of a with respect to b* exists, which is $a \rightarrow b = \bigsqcup\{c \mid a \sqcap c \leq b\}$. For $b = \bot$ we obtain the pseudo-complement.

Turning this easy observation around, let us start with a Concept Theory that defines the dual of a semi-Heyting lattice. Again, we could ask the question what happens, if

we remove the greatest concept $\top$. We obviously preserve properties (i), (ii), (iv) and (v) in the definition, but not (iii). Concepts C_1 and C_2 with $C_1 \sqcup C_2 = \top$ would become incompatible, so the pseudo-complement $\bar{C}$ would become the least concept that is incompatible to C, which means that (vi) is satisfied.

Thus, the mathematical structures defined by pseudo-Kauppian Concept Theories are just duals of semi-Heyting lattice that have been deprived by their greatest element.

5 Multi-Context Concept Theories

Starting from the Concept Theory defined by Ganter and Wille we developed a generalised first-order Concept Theory. However, we stayed within the framework of exactly one structure or context, in which formulae are to be interpreted. As intensions of concepts have been defined as maximal theories, these interpretations lead to the extensions. Conversely, taking all formulae that are valid for a given set of entities, leads to a maximal theory. So, we are stuck in Concept Theories that are mainly "extensional" in the sense that the intention of a concept can always be derived from its extension.

Let us briefly proceed to a further generalisation by dropping the restriction to just one structure. So we assume a first-order signature $(\mathcal{P}, \mathcal{O}, V)$ as before, but now consider a family $\{\mathcal{S}\}_{i\in I}$ of structures. In fact, these may be all structures.

Let ϕ_C be a monadic theory. We can again define the *extension* of ϕ_C (in the structure $\mathcal{S}_i$) as

$$\mathcal{E}_{\mathcal{S}_i}[\![\phi_C]\!] \quad = \quad \{\sigma(x) \in \mathcal{D} \mid\mid\!\models_{(\mathcal{S}_i,\sigma)} \varphi(x) \text{ for all } \varphi \in \phi_C\} \quad .$$

Analogously the *intension* of a set of entities $E_i \subseteq \mathcal{D}_i$, the domain of the structure $\mathcal{S}_i$, is defined as

$$\mathcal{I}_{\mathcal{S}_i}[\![E_i]\!] \quad = \quad \{\varphi \mid\mid\!\models_{(\mathcal{S}_i,\sigma)} \varphi(x) \text{ for all } \sigma \text{ with } \sigma(x) \in E_i\} \quad .$$

We may now call theories ϕ and Ψ *equivalent* iff $\mathcal{E}_{\mathcal{S}_i}[\![\phi]\!] = \mathcal{E}_{\mathcal{S}_i}[\![\Psi]\!]$ holds for all $i \in I$. We use the notation $[\phi]$ to denote an equivalence class of theories with the representative ϕ.

Definition 5.1. A *concept C in the family of structures* $\{\mathcal{S}_i\}_{i\in I}$ is a pair $([\phi_C], \{E_i\}_{i\in I})$ consisting of an equivalence class $[\phi_C]$ of theories called the *intension* of the concept and sets $E_i \subseteq \mathcal{D}_i$ of entities called the *extensions* of the concept such that $\mathcal{E}_{\mathcal{S}_i}[\![\phi_C]\!] = E_i$ and $\mathcal{I}_{\mathcal{S}_i}[\![E_i]\!] \in [\phi_C]$ hold.

We can define a partial order on concepts by

$$C_1 \preceq C_2 \quad \Leftrightarrow \quad \phi_{C_2} \models \phi_{C_1} \quad \Leftrightarrow \quad \mathcal{E}_{\mathcal{S}_i}[\![\phi_{C_2}]\!] \subseteq \mathcal{E}_{\mathcal{S}_i}[\![\phi_{C_1}]\!] \text{ for all } i \in I \quad .$$

As $\models$ refers to interpretations with respect to the chosen family of structures, we could again replace the equivalence class $[\phi_C]$ by a single maximal theory ϕ_C. It is clear that $\mathcal{I}_{\mathcal{S}_i}[\![E_i]\!] = \phi_C$ holds in this case. This allows us to write again $C = (int(C), ext(C) = (\phi_C, \{E_i\}_{i\in I})$ and stick with $\mathcal{E}_{\mathcal{S}_i}[\![\phi_C]\!] = E_i$.

In particular, it seems to be preferable to consider the intension of the concept as the primary component, as the extension depends on the chosen structures. However, the extension, i.e., the family of sets of entities $ext(C) = \{E_i\}_{i \in I}$ still determines the intension $int(C)$, so this extension to first-order Concept Theory is still "extensional". Furthermore, we still obtain complete, distributive lattices.

Proposition 5.2. *Concepts in a family of structures $\{\mathcal{S}_i\}_{i \in I}$ together with the partial order $\preceq$ define a complete, distributive lattice.* □

6 First-Order Intuitionistic Concept Theory

Finally, let us leave the grounds of classical logic and switch to first-order intuitionistic logic [Bell and Machover, 1977, Chapter 9]. The major difference to classical logic is that formulae are now interpreted in a constructive way. For instance, a $\varphi \vee \neg\varphi$ is considered to be true iff we can find a proof for φ or a proof for $\neg\varphi$; the absence of a proof for φ does not yield the truth of $\neg\varphi$. In particular, the rule of *tertium non datur* does no longer apply.

Intuitionistic logic as a basis for Concept Theory is of particular interest for Information Systems, where constructions are to be interpreted as computations. The work in [Schewe, 2000] provides an example of exploiting intuitionistic logic—in this particular case: higher-order intuitionistic logic—as for foundation for advanced database theory.

The logical language $\mathcal{L}$ is defined as for the classical logic. For simplicity we assume that the signature only allows 0-ary function symbols, i.e., constants to be used, no n-ary function symbols with $n > 0$. We start again with a monadic theory ϕ_C of the logic, i.e., a set of formulae.

We want to define the notion of *forcing* based on the famous Kripke semantics. For this we need *Kripke systems* that replace the structures used in previous sections.

Definition 6.1. A *Kripke system* $\mathcal{K}$ consists of

- a non-empty, partially ordered collection $(\mathcal{W}, \leq)$ of worlds;
- a $\mathcal{W}$-indexed family $\{\mathcal{T}_w\}_{w \in \mathcal{W}}$ of non-empty sets of terms such that $\mathcal{T}_{w_1} \subseteq \mathcal{T}_{w_2}$ holds whenever we have $w_1 \leq w_2$;
- a $\mathcal{W}$-indexed family $\{\mathcal{F}_w\}_{w \in \mathcal{W}}$ of non-empty sets of atomic formulae such that for all terms t in an atom $\varphi \in \mathcal{F}_w$ satisfy $t \in \mathcal{T}_w$, and $\mathcal{F}_{w_1} \subseteq \mathcal{F}_{w_2}$ holds whenever we have $w_1 \leq w_2$.

In order to define the notion of forcing we will need *$\pm$-formulae*. These are all formulae in $\mathcal{F}$ plus all expressions $-\varphi$ for a formula $\varphi \in \mathcal{F}$. We use $\mathcal{F}^{\pm}$ to denote the set of all $\pm$-formulae.

For a Kripke system $\mathcal{K}$ we now define $w \Vdash_{\mathcal{K}} \varphi$ for worlds $w \in \mathcal{W}$ and $\pm$-formulae $\varphi \in \mathcal{F}^{\pm}$. We say that w *forces* φ. Unless there is a need to emphasize the Kripke system, we drop the subscript $\mathcal{K}$.

Intuitively, $w \Vdash \varphi$ for a formula $\varphi \in \mathcal{F}$ means that in the world w it is known that φ is true, whereas $w \Vdash -\varphi$ means that in the world w the formula φ is understood, but it is not known that φ is true. The partial order $\leq$ in $\mathcal{K}$ can be interpreted as progression of knowledge.

Definition 6.2. The *Kripke semantics* for the language $\mathcal{L}$ and a Kripke system $\mathcal{K}$ is defined as follows:

(i) If φ is an atomic formula, then $w \Vdash \varphi$ holds iff $\varphi \in \mathcal{F}_w$ holds.
(ii) For disjunctions we have $w \Vdash \varphi \vee \psi$ iff all terms of φ and ψ are in $\mathcal{T}_w$ and at least one of $w \Vdash \varphi$ or $w \Vdash \psi$ holds.
(iii) For conjunctions we have $w \Vdash \varphi \wedge \psi$ iff both $w \Vdash \varphi$ and $w \Vdash \psi$ hold.
(iv) For implications we have $w \Vdash \varphi \Rightarrow \psi$ iff all terms of φ and ψ are in $\mathcal{T}_w$ and whenever $w' \Vdash \varphi$ holds for $w \leq w'$, then also $w' \Vdash \psi$ holds.
(v) For negations we have $w \Vdash \neg\varphi$ iff all terms of φ are in $\mathcal{T}_w$ and for all $w \leq w'$ we have $w' \nVdash \varphi$.
(vi) For existentially quantified formulae we have $w \Vdash \exists x.\varphi(x)$ iff $w \Vdash \varphi(t)$ for some term t.
(vii) For universally quantified formulae we have $w \Vdash \forall x.\varphi(x)$ iff whenever $w \leq w'$ holds, then $w' \Vdash \varphi(t)$ for any term t.
(viii) For negative formulae we have $w \Vdash -\varphi$ iff all terms of φ are in $\mathcal{T}_w$, but $w \nVdash \varphi$ holds.

We say that a $\pm$-formulae $\psi \in \mathcal{F}^{\pm}$ is *enforceable* iff there exists a Kripke system $\mathcal{K}$ and a world $w \in \mathcal{W}$ for this Kripke system such that $w \Vdash_{\mathcal{K}} \psi$ holds. We say that a set Φ of $\pm$-formulae is *enforceable* iff each $\psi \in \Phi$ is enforceable.

We say that a formula $\varphi \in \mathcal{F}$ is *Kripke-valid* (notation: $\Vdash \varphi$) iff $-\varphi$ is not enforceable. More generally, we get $\Phi \Vdash \varphi$ iff $\Phi \cup \{-\varphi\}$ is not enforceable, and $\Phi \Vdash \Psi$ iff $\Phi \vdash \psi$ holds for all $\psi \in \Psi$.

It is known that Kripke-valid formulae are always valid, but the converse in generally not true. In order to use the definition for Concept Theory, we make the assumption that the elements of a domain $\mathcal{D}$ are used as constants in the signature. Therefore, we may define

$$\mathcal{E}[\![\phi_C]\!] \quad = \quad \{\sigma(x) \in \mathcal{D} \mid\models \varphi(\sigma(x)) \text{ for all } \varphi \in \phi_C\} \quad .$$

This definition of extension relies on validity, not on Kripke-validity.

The definition above considers all Kripke systems. Same as for first order theories we may restrict our attention to a family of Kripke systems or even to a single fixed system $\mathcal{K}$.

Definition 6.3. A *concept* C *in a Kripke system* $\mathcal{K}$ is a pair (ϕ_C, E) consisting of a maximal monadic theory ϕ_C called the *intension* of the concept and sets $E \subseteq \mathcal{D}$ of entities called the *extension* of the concept such that $\mathcal{E}[\![\phi_C]\!] = E$ holds.

We write again $C = (int(C), ext(C))$. We omit the generalisation to a family of Kripke systems.

We can define a partial order on concepts by

$$C_1 \preceq C_2 \quad \Leftrightarrow \quad \phi_{C_2} \Vdash_{\mathcal{K}} \phi_{C_1} \quad .$$

With this definition $C_1 \preceq C_2$ implies again that $ext(C_2) \subseteq ext(C_1)$ holds, but the converse is no longer true.

Proposition 6.4. *Concepts in a Kripke system* $\mathcal{K}$ *together with the partial order* $\preceq^{-1}$ *define a Heyting algebra.* □

7 Conclusion

In this article we continued our thoughts that it is worthwhile to lay the foundations of Concept Theories in terms of mathematical logic. The general idea is to consider intentions of concepts to be equivalence classes of monadic theories, and the models being the extension. We clarified this view with respect to the Concept Theory by Ganter and Wille, a Concept Theory based on classical first-order logic with or without fixed structures, and a Concept Theory based on first-order intuitionistic logic. The latter seems to be equivalent to the axiomatic Concept Theory defined by Schock [Palomäki, 1992], though we did not investigate this formally.

Natural next steps would be to consider higher-order logics, especially higher-order intuitionistic logic [Bell, 1988]. The resulting theory should the suitable for capturing other Concept Theories based on versions of typed λ-calculus, e.g., the work by Materna [Materna, 1992] and his followers Duží [Duží, 2001] and Palomäki [Palomäki, 1997]. This relationship, however, has not yet been proven.

Surprisingly (or not?), all the theories investigated in this article led to concept lattices with least and greatest element, so trivially satisfied the Kauppi axioms for Concept Theories [Kauppi, 1967]. For the case of first-order logic we even obtained Boolean algebras; for intuitionistic logic we obtained the duals of Heyting algebras. Due to the nature of intuitionistic logic it seems to be the case that this property will also result, if higher order logics are studied. This raises the question, whether Kauppian Concept Theories that are not duals of Heyting algebras make any sense.

If we exclude contradictory formulae $\varphi \wedge \neg\varphi$ from the theories that define intensions of concepts for the obvious reason that it does not make sense to have such "absurd" definitions of concepts, then we lose the property of being Kauppian. The loss is small. We still obtain pseudo-Kauppian Concept Theories by missing out just one axiom. The mathematical structures behind this result from duals of distributive, pseudo-complemented lattices with zero and one that have been deprived by their top element.

With respect to Kauppi's axioms of Concept Theories we can draw the following conclusions:

- The missing greatest concept that intensionally contains all concepts is just a concept of absurdity. It is a matter of taste, whether we would like to consider a concept without any entities falling under it, really a concept. The advantage of allowing such a concept is that the mathematics of Concept Theories just becomes easier. In fact, nothing more than just "absurdity" will be added.
- The axiom stating the existence of maximal concepts, i.e., not being properly intensionally contained in any other concept (except the absurd concept, if this is permitted) above any concept, needs a justification.
- As the intuitionistic case leads to (duals of) Heyting algebras instead of (duals of) semi-Heyting lattices, we may asked the question, whether it would not be advantageous to start directly with Heyting algebras.

The open questions raised here are to be investigated in the future.

References

[Bell, 1988] Bell, J. (1988). *Toposes and Local Set Theories - An Introduction*, volume 14 of *Oxford Logic Guides*. Clarendon Press, Oxford.

[Bell and Machover, 1977] Bell, J. and Machover, M. (1977). *A Course in Mathematical Logic*. North-Holland, Amsterdam.

[Duží, 2001] Duží, M. (2001). Logical foundations of conceptual modelling using hit data model. In H. Jaakkola, H. Kangassalo, and E. Kawaguchi, editors, *Information Modelling and Knowledge Bases*, volume XII, pages 65–80. IOS Press.

[Feyer et al., 2002] Feyer, T., Schewe, K.-D., and Thalheim, B. (2002). Intensionality in concept theory. In H. Kangassalo, E. Kawaguchi, H. Jaakkola, and T. Welzer, editors, *Information Modelling and Knowledge Bases*, volume XIII, pages 422–426. IOS Press.

[Ganter and Wille, 1999] Ganter, B. and Wille, R. (1999). *Formal Concept Analysis: Mathematical Foundations*. Springer-Verlag, Berlin.

[Kangassalo, 1993] Kangassalo, H. (1993). COMIC: A system and methodology for conceptual modelling and information construction. *Data & Knowledge Engineering*, 9:287–319.

[Kauppi, 1967] Kauppi, R. (1967). Einführung in die Theorie der Begriffssysteme. *Acta Universitas Tamperensis. Ser. A*, 15.

[Materna, 1992] Materna, P. (1992). Meanings are concepts. *From the Logical Point of View*, 2:76–89.

[Palomäki, 1992] Palomäki, J. (1992). From the theory of concepts to concept theory. In S. Ohsuga, H. Kangassalo, H. Jaakkola, K. Hori, and N. Yonezaki, editors, *Information Modelling and Knowledge Bases*, volume III, pages 107–122. IOS Press.

[Palomäki, 1994] Palomäki, J. (1994). Towards the foundations of concept theory. In H. Jaakkola, H. Kangassalo, T. Kitahashi, and A. Márkus, editors, *Information Modelling and Knowledge Bases*, volume V, pages 139–154. IOS Press.

[Palomäki, 1997] Palomäki, J. (1997). Three kinds of containment relations of concepts. In H. Kangassalo, J.F. Nilsson, H. Jaakkola, and S. Ohsuga, editors, *Information Modelling and Knowledge Bases*, volume VIII, pages 261–277. IOS Press.

[Schewe, 2000] Schewe, K.-D. (2000). The type concept in OODB modelling and its logical implications. In E. Kawaguchi, H. Kangassalo, H. Jaakkola, and Issam A. Hamid, editors, *Information Modelling and Knowledge Bases*, volume XI, pages 256–274. IOS Press.

Information Modelling and Knowledge Bases XIV
H. Jaakkola et al. (Eds.)
IOS Press, 2003

3D Visual Construction of a Context-based Information Access Space

Mina AKAISHI*, Makoto OHIGASHI*, Nicolas SPYRATOS**, Yuzuru TANAKA* and Hiroyuki YAMAMOTO*
**Meme Media Laboratory, Graduate School of Engineering Hokkaido University, 060-8628 Sapporo, Japan*
***Laboratoire de Recherche en Informatique, Universite de Paris-Sud, LRI-Bat 490, 91405 Orsay Cedex, France*
E-mail: {mina|hiro|mack|tanaka}@meme.hokudai.ac.jp, Spyratos@lri.fr

Abstract. This paper proposes a framework for generating 3D Information Access Spaces from existing knowledge repositories. Recently, vast amounts of information are accumulated at accelerating paces in various forms, such as multimedia documents, application programs or service systems. New architectures for organizing and accessing this information are needed. We use a notion of context as a data modeling mechanism and we propose a framework for the construction of 3D interfaces to support intuitive access to data. Such an interface, called a Context-based Information Access space (or CIA for short) consists of multiple virtual spaces. In a CIA, virtual spaces are created automatically by accessing an information base and are then connected together by space pointers depending on context. This paper describes the CIA mechanism and its support for context traversal by users.

1. Introduction

Computers and networks are now rapidly expanding their applications through many fields, and users can share the knowledge of others and can also publish their own knowledge through the Internet. As a consequence, vast amounts of information are accumulated at accelerating paces in various forms, such as multimedia documents, application programs and service systems. We need new access architectures for organizing and accessing this information. In this paper, we propose use a notion of context as a data modeling mechanism and we propose a framework for the construction of interactive information access spaces to support intuitive access.

In computer science, a number of formal or informal definitions of some notion of context have appeared in several areas, such as artificial intelligence [1-3], software development [4-9], databases [10-14], machine learning [15,16], and knowledge representation [17-20]. In this paper, we use the notion of context introduced in [18,19,21-23] as a conceptual modeling mechanism for organizing and managing very large information bases, together with a path-language for context traversal.

A Context-based Information Access space (CIA for short) is a 3D virtual environment to support information access activities based on context. The implementation framework for the construction of CIAs is based on the IntelligentBox system [24], a constructive visual software development system aimed at interactive 3D graphic applications. A context is materialized as a virtual space represented as a spherical 3D object. This

spherical object is designed based on a special component of IntelligentBox, called the World Bottle Box [25]. Users can see the contents of the spherical object from outside, and can jump inside to see more detailed information or more relevant information.

The contents of a context are materialized using the mechanism of reification of database records [26]. Our framework reifies context data in a database as 3D components depending on predefined template models. It also supports context traversal in the form of multiple virtual spaces traversal.

Several 3D visualization systems and information navigation mechanisms [27-30] have been proposed in recent years. They visualize database records and allow users to do only predefined interactions. In our system, all functions are designed as 3D components. In addition our framework allows combining freely with any 3D components in the IntelligentBox system. This means that the user can extend the functions that are provided by adding other existing or future components. Moreover, users can reuse arbitrary parts of a CIA as new application components.

The remainder of the paper is organized as follows. In section 2, we introduce the notion of context that we use as well as a path language for context traversal. In section 3, we present the implementation framework of a 3D interface for Context-based Information Access (CIA). In section 4, we present the component-based construction of a CIA. In section 5, we explain how to explore a CIA. Finally, in section 6, we make some concluding remarks.

2. Contexts

In this chapter, we present the notion of context that we use and the query language for context traversal [18,19,21-23].

2.1 The notion of context

A context consists of an identifier c plus a *content*. The content of c is a set of triplets of the form

<names, object identifier, reference>,

where *names* is a set of descriptions for the object and *reference* is a context identifier or Nil. The context referenced by an object contains information relevant to that object.

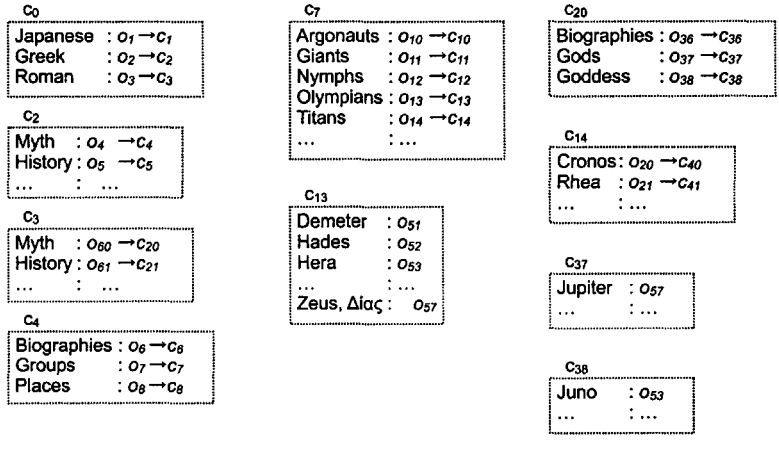


Fig. 1. Examples of context.

Figure 1 shows examples of context. The context c_0 includes three triplets. Its second triplet consists of the single name "Greek", the object identifier o_2 and the reference c_2. By following references, users can see more detailed information. It is important to note that names and references are context dependent. An object can belong to different contexts and may have different names and/or different references in each context. This feature is useful when we want to view an object from different perspectives

Contextualization can be used orthogonally to the usual abstraction mechanisms of classification, generalization and attribution, and any information base that uses the context mechanism on top of the usual abstraction mechanisms is called a contextualized information base [See 18,19,21-23].

2.2 Accessing information through paths

Accessing information in a contextualized information base often involves navigating from one object to another by following references. From an object within a given context, we can reach any object that belongs to its reference and, recursively, any object that lies on a path. Navigation is based on the notion of path. A path is a sequence of pairs of the form (c_i, t_i), where c_i is a context and t_i belongs to the content of c_i. For example, in Figure 2, the following is a path: $\{(c_0, <$ 'Greek', $o_2, c_2>)$, $(c_2, <$ 'Myth', $o_4, c_4>)$, $(c_4, <$ 'Biographies', $o_6, c_6>)\}$. Given a path p:$\{(c_1, t_1), \ldots, (c_k, t_k)\}$, a name path is a sequence of names $n_1, \ldots, n_k$ where n_i is a name in triplet t_i, $i = 1, \ldots k$. For example, the following is a name path where each name belongs to a triplet on the path given earlier: 'Greek.Myth.Biographies'. An object path is a sequence of object identifiers within a path. For example, $o_2.o_4.o_6$ is an object path. Paths form the basis for reaching objects in a context navigating through the references of objects.

2.3 Querying a Contextualized Information Base

The access to information is achieved using a path-language for context traversal. There are three groups of functions: primitive operations, fundamental operations and macro functions. The details are described in [22]. We introduce two macro-functions:

look-up(c, n): this operation takes as input a context c and a name n and returns the set of name paths n_i starting at the specified context c, and ending with name n.

cross-ref(c, o): this operation takes as input a context c and a object o and returns the set of all name paths n_i such that n_i begins with a name in the content of c and ends with one of the names of o. Consider for example, *cross-ref*(c_o, o_{57}). There are two possible paths from context c_o to context containing o_{57}. There are :

$\{(c_0, <$ 'Greek', $o_2, c_2>)$, $(c_2, <$ 'Myth', $o_4, c_4>)$, $(c_4, <$ 'Groups', $o_7, c_7>)$, $(c_7, <$ 'Olympians', $o_{13}, c_{13}>)$, $(c_{13}, <$ 'Zeus, Δίας', o_{57}, *nil*$>)\}$,

$\{(c_0, <$ 'Roman', $o_3, c_3>)$, $(c_3, <$ 'Myth', $o_{60}, c_{20}>)$, $(c_{20}, <$ 'Gods', $o_{37}, c_{37}>)$, $(c_{37}, <$ 'Jupiter', o_{57}, *nil*$>)\}$.

Using the paths we obtain all name paths that are returned by corss-ref, :

'Greek.Myth.Groups.Olympians.Zeus',

'Greek.Myth.Groups.Olympians. Δίας'

'Roman.Myth.Gods.Jupiter'.

This is useful as we can find alternative representations of the same object in different contexts, as well as the name paths to reach these representations.

3. Context-based Information Access (CIA)

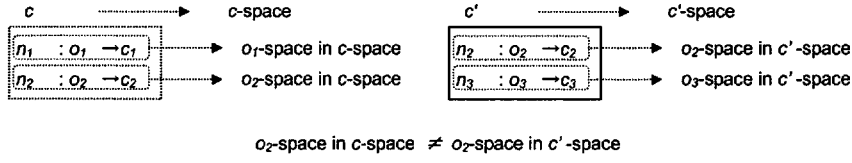


Fig. 2. Mapping the context elements into media objects.

The Context-based Information Access (CIA) is a 3D interface that allows to access information based on the context model and path-language introduced earlier. It consists of object spaces and context spaces. In this section, we explain these concepts and their implementation framework.

3.1 Context Spaces and Object Spaces

In order to create a CIA we map the components of a context into individual virtual spaces, as illustrated in Figure 2. A context identifier is mapped into an individual virtual space, called a *context space*. Each triplet in the content of the context is mapped into an individual virtual space, called an *object space*. Therefore, an object space is a 3D visualization of a triplet, a context space is a set of object spaces, and a CIA is a set of context spaces.

The creation of an object space within a context depends on that context. For example, in Figure 2, the object o_2 is materialized as two different object spaces in the two contexts c and c'. However, object o_2 is the same in the two contexts c and c'. If we want to access o_2, we will find the same thing whether we access it from c or from c'. What differs in the two contexts is the way of accessing o_2 from these contexts.

3.2 The contents of object spaces and context spaces

As we have seen, a context consists of an identifier plus a content, i.e., a set of triplets of the form *<names, object identifier, reference>*. Similarly, a context space consists of a virtual space plus a set of pointers to object spaces; an object space, in turn, is a virtual space that includes the instantiation of a single triplet. Two additional features of an object space are the name viewer and the object viewer. The name viewer represents names as text, pictures, sounds and so on. The object viewer represents an object as some media object. A pointer to a context space represents a reference.

We use *space pointers* to connect spaces. To construct a CIA, we need a mechanism that addresses efficient access to information objects within a restricted 3D display space. That mechanism allows us to embed multiple 3D spaces in a single virtual environment, and enables us to navigate through these different spaces. An embedded space can be represented as 3D media components. The user can see the contents of the embedded space from outside, and jump into the embedded space to change the current working place.

A pointer to an object space is called an Object Space Pointer (OSP) and a pointer to context space is called a Context Space Pointer (CSP). The roles of these space pointers are (i) as entrance points to access a target space and (ii) as a representation of a particular aspect of a space pointed at.

Object spaces are embedded into a context space by OSPs. The same object can be accessed from one or more different context spaces through OSPs. This feature is useful

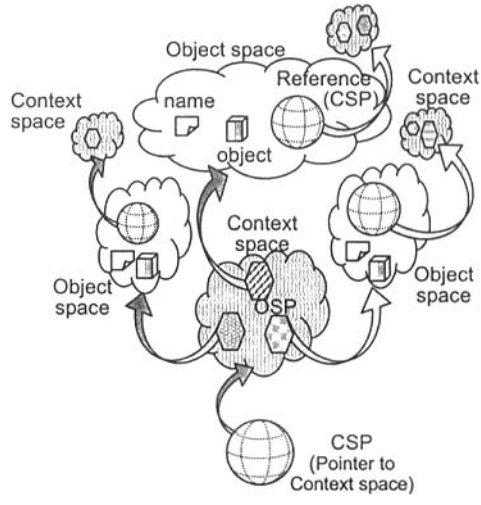


Fig. 3. An overview of a CIA .

when the user wants to view an object under different perspectives. The same object can have different names/references in different contexts in which it belongs.

Figure 3 illustrates a CIA. The spheres in the figure represent CSPs. Hexagons in a context space represent OSPs to access object spaces. Each object space includes an object, its names and a pointer to some context space.

When a user is inside a context space, he sees a set of OSPs. By selecting one of the OSPs the user is navigated to an object space and through the CSP of the object space he is navigated to the context space referenced by the object.

4. Construction of CIA

As we have already explained, a CIA consists of a set of context spaces and each context space consists of a set of object spaces. CIAs are constructed using the IntelligentBox

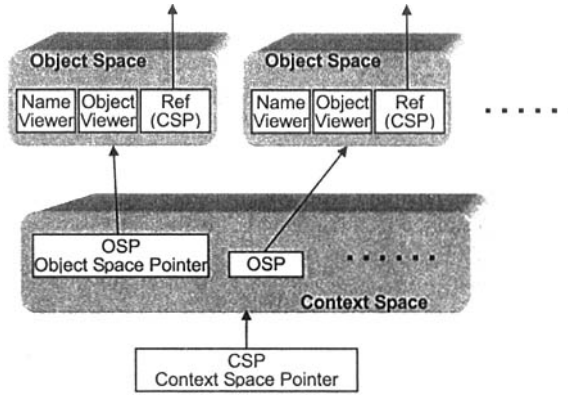


Fig. 4. The composition structure of components for a CIA.

system. In this chapter, we introduce the 3D component-ware system IntelligentBox and describe the architecture of a CIA.

4.1 A Framework for the CIA

Figure 4 shows the basic functional linkage of components to construct a CIA. A CSP connects to a context space and a context space includes a set of OSPs. Each OSP connects to an object space. In the object space, there are (i) an object viewer, (ii) a name viewer and (iii) a CSP.

4.2 An overview of the IntelligentBox system

The IntelligentBox [24] is a constructive visual software development system for interactive 3D graphic applications. The IntelligentBox represents objects as reactive 3D visual objects, called Boxes, that can be manually combined with other Boxes. It provides a uniform framework for the concurrent definition of both geometrical compound structures among Boxes and their mutually interactive functional linkages. Figure 5 shows a schematic explanation of functional linkages among boxes.

Each Box has its own state values that are stored in variables called slots. Slots are opened to connect with other Boxes. One Box can be connected with one of the slots of another Box. There is the restriction that the slot connection is available when there is a parent-child relationship between the two Boxes.

The slot connection connects a child Box slot to a parent Box slot by a message sending protocol. These messages transfer data between two mutually connected slots. This data transfer works to combine the two functions of the corresponding child Box and its parent Box because each Box has a unique function associated with its slot value.

We use the IntelligentBox system as the basis to construct the CIA.

4.3 Space Management Components

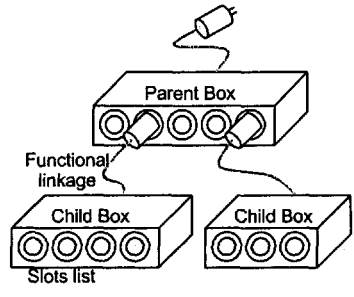


Fig. 5. A schematic explanation of functional linkages among boxes.

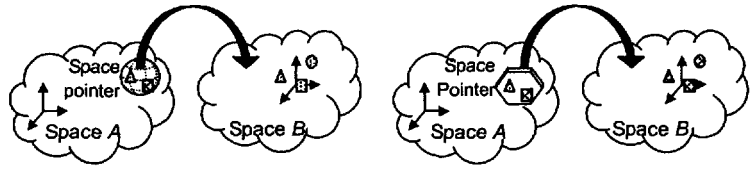


Fig. 6. The concepts of Space Pointers.

To construct a context space and an object space, we need components to manage the multiple virtual spaces. To this end, space management functions were introduced into the IntelligentBox systems by M. Itoh [25]. Since all facilities are implemented as functions of Boxes in the IntelligentBox system, these space management functions are also implemented as a particular Box called a Space Pointer Box.

Figure 6 shows the concept of a space pointer. A virtual space is embedded into its interior. The space *A* and the space *B* are independent virtual spaces. The space *B* is embedded into the space *A* by a Space Pointer Box. Any object put in space *B* can be referred from the space *A* through the space pointer box. Any object can also be transferred from space *A* to space *B* by putting the object into the Space Pointer Box. A user in space *A* can move to space *B* through the Space Pointer.

In this paper, the CSP Box is represented as a 3D arbitrarily shaped icon and the OSP Box is represented as an arbitrarily shaped board. These Space Pointer Boxes provide fundamental functions for the construction of the CIA.

4.4 OSP Generation Mechanism

When a user traverses context spaces continuously, necessary ports, object spaces and linked context spaces should be created automatically by accessing the contextualized information base. In this section, we describe the mechanism to materialize virtual context spaces and object spaces.

Figure 7 shows the functional components structure for the automatic creation of the context space based on the context data in the contextualized information base. This OSP generation mechanism provides the dynamic creation of necessary context spaces and object spaces by following the users information space traversal. Users information access activities form the virtual information access spaces.

An OSP generator creates all OSPs that belong to the specified context. The basis of

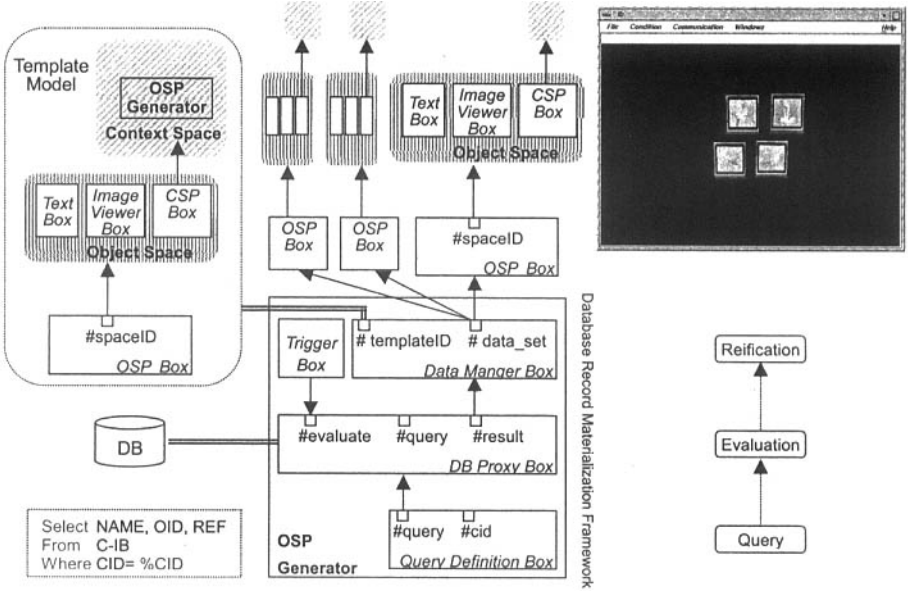


Fig. 7. The components structure to generate the object and context spaces.

C-IB

CID	NAME	OID	REF
c_0	Japanese	o_1	c_1
c_0	Greek	o_2	c_2
c_0	Roman	o_3	c_3
...	...	...	...
c_{13}	Hera	O_{53}	nil
c_{13}	Zeus	O_{57}	nil
...	...	...	...

Table 1. An example of a contextualized information base.

the OSP generator is the Database Record Materialization Framework. It gets a query as an input and output a query evaluation result as 3D media objects. First, a Query Definition Box receives the CID as an argument and defines the query. Then the query is sent to and is evaluated by a DB Proxy Box that works as an interface between a database and CIA. Finally, the result is visualized by a Data Manager Box. The template model for instantiation of an element is registered in a Data Manager. This template model consists of an OSP and an object space that includes a name viewer, an object viewer and a reference. The reference is a CSP that connects with a context space that contains OSP generator components again. This OSP generator receives the CID of a reference and creates OSPs continuously in the same procedure.

Let us assume that the context data is stored in a relational database. Table 1 is the context data table *C-IB* of the examples shown in figure 1. A user's current context is the context c_0. When the context ID c_0 is sent to the Query Definition Box, the Query Definition Box creates the following query and sends it to DB proxy Box.

Select NAME, OID, REF
From C-IB
Where CID = c_0.

The DB Proxy Box sends a query to a database and gets the results. The example of the result is a list {('Japanese', o_1, c_1), ('Greek', o_2, c_2), ('Roman', o_3, c_3)}. A Data Manager Box gets the result of the query evaluation from the DB Proxy Box. When a Data Manager Box receives the result data collection {('Japanese', o_1, c_1), ('Greek', o_2, c_2), ('Roman', o_3, c_3)}, the Data Manager Box makes copies of template Boxes. Then each copy of the template Box is instantiated with the value of the corresponding element of the collection. The Data Manager Box distributes each element of the collection to each OSP Box, that is the copy of the template model.

An OSP Box receives a triplet data from the Data Manager Box. The OSP Box connects with the object space that includes viewer Boxes of an object, names and a reference. An object is instantiated as an arbitrary Box(es). A name is instantiated by a Text Box. A reference is materialized by the CSP Box whose structure is the same as that shown in figure 7. For an example, the OSP Box gets the triplet data ('Japanese', o_1, c_1), then a Text Box shows the name 'Japanese' and an arbitrary Box(es) represents the object o_1. The reference c_1 is sent to the CSP Box whose structure is the same as shown in figure 7. Then the same procedures to create a context space are repeatedly executed.

5. Exploration in the Context-based Information Access Spaces

In this section, we explain how to explore the Context-based Information Access space.

5.1 Traversal through paths

The upper-left part of figure 8 illustrates an example of context-based information access space. The context c_0 has four objects. The OSPs in the context space c_0 show the symbol birds of Gods in the Greek mythology. If a user accesses the object o_1 through the OSP in c_0, the user will be lead to the object space that includes object o_1 and the link to context c_1.

The upper-right part of figure 8 is a display hardcopy of context space c_0. The sphere is a CSP Box that includes four OSP Boxes. When a user approaches the CSP Box, the user can enter the inside space of the CSP Box. Then the user will see the OSPs as framed pictures, which are the OSP Boxes. The user can look up the corresponding object spaces through the OSPs. The lower part of figure 8 shows the OSPs and the object spaces. The OSPs are transparent and the corresponding object spaces are seen through the OSPs.

When a user approaches the OSP cuckoo (i.e. object o_1), then the user will enter the object space os_1 that includes the object o_1. In the same way, a user can enter the object space os_2 through the OSP owl (i.e. object o_2) in the context c_0. In the same way, users can continuously trace the paths

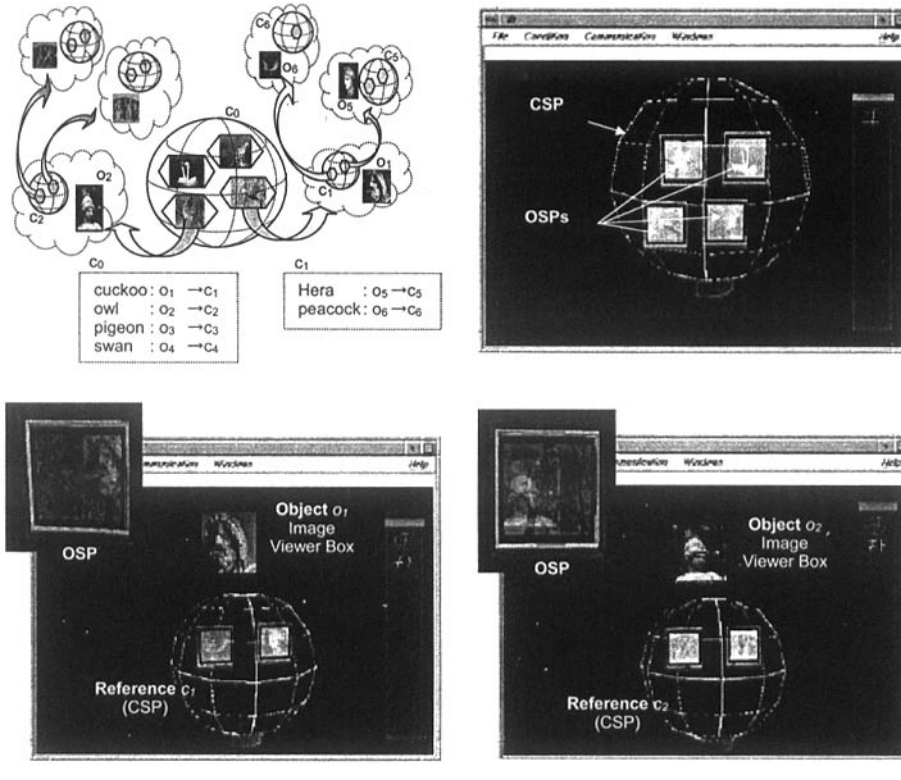


Fig. 8. An example of CIA.

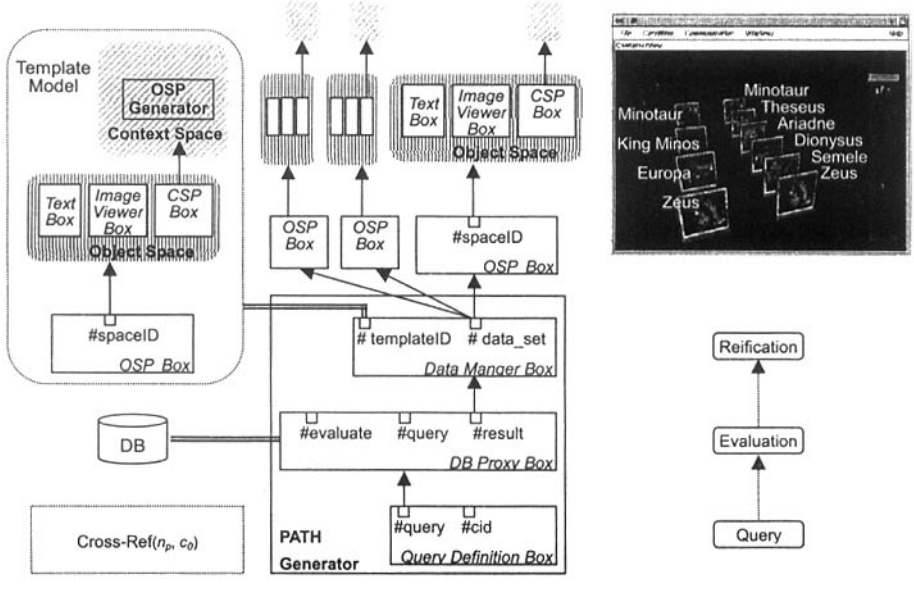


Fig. 9. The components structure to generate the object spaces to represent paths.

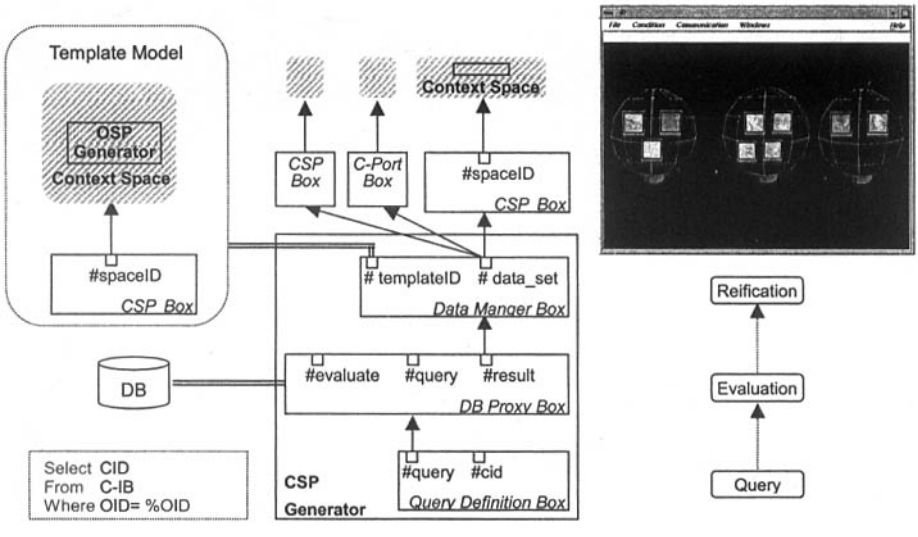


Fig. 10. The components structure to generate the context spaces.

5.2 Path Generation Mechanism

Figure 9 shows the evaluation of the path selection operation, namely the function *cross-ref*(c_0, o). It returns the name path from the current context c_0 to the object o. Let us assume that the object o is named 'Minotaur' and we get, as the results of the evaluation, the list of triplet data corresponding to the name paths:

'Zeus.Europa.King_Minos.Minotaur' and

'Zeus.Semele.Dionysus.Ariadne.Theseus.Minotaur'.

Figure 9 shows the paths as the sequences of OSPs (the OSP Boxes). Users can enter each object space through the corresponding OSP. The components structure of the Path Generator Boxes is the same as the OSP Generator Boxes in figure 7. Users can issue the query to get the path that satisfies the condition. Then users can proceed to traverse in the context-based information access space.

5.3 Context Space Generation Mechanism

Figure 10 shows the components structure to get the collection of context spaces that satisfy the given query. The Data Manager Box stores a CSP (CSP Box that connected with a context space) as a template Box. Then it gives the set of context spaces that satisfy the query condition.

For an example, let us consider the query to select all contexts that include a specified object o. We call such a set of contexts the *facets* of o. It gives users an overview of different aspects of the object o in different context spaces.

After query evaluation, the DB Proxy Box receives all context data that includes the object o, that is its facets of o. The result data is sent to the Data Manager Box. Then the Data Manager Box makes the copies of template Box whose detailed structure is shown in figure 7. Each CSP Box materializes the context space by following the aforementioned procedures in section 4.4. Users can choose the context space, enter it, and follow the links depending on their interests.

6. Conclusion

In this paper, we proposed the concept of a context-based information access space and its implementation framework. Through the OSPs in the context spaces, users can follow the reference links and get the information object and paths in which they are interested. The system creates dynamically the information spaces as the need arises. In addition the integration and collaboration of existing and future tools over a hypermedia environment would contribute to the efficient usage of the human knowledge stored in computer systems, and would enhance productivity and collaboration. In our framework of CIA, a context space and an object space are represented by 3D components, that is CSP Box and OSP Box. In the IntelligentBox system, all functions are provided as 3D components, Boxes. Then any component of CIA can be combined with other functional component Boxes such as 3D animation tools, CSCW support tools, the scientific visualization tools and so on. It might expand the application fields. For our future works, we are interested in relationships between objects in a context and we will consider such relationships as objects.

References

[1] G. Hendrix: Encoding Knowledge in Partitioned Networks, In N. Findler, editor, Associative Networks. New York: Academic Press, 1979

[2] J. McCarthy: Notes on Formalizing Context, Proc. IJCAI-93, pp.555-560, Chambery, France, 1993

[3] R. Guha: Contexts: A Formalization and Some Applications, ph. D thesis, Stanford University, 1991. Also published as Technical Report STAN-CS-91-1399-Thesis, and MCC Technical Report Number ACT-CYC-423-91.
[4] J. Richardson and P. Schwarz: Aspects: Extending Objects to Support Multiple, Independent Roles, Proc. ACM-SIGMOD Conference, pp.298-307, 1991
[5] G. Gottlob, M. Schrefl, and B. Rock: Extending Object-Oriented Systems with Roles. ACM Trans. Inf. Syst., 14(3): 268-296, 1996
[6] E. Sciore: Object Specialization, ACM Trans. Inf. Syst., 7(2): 103-122, 1989
[7] J. Shilling and P. Sweeney: Three Steps to Views: Extending the Object-Oriented Paradigm, Proc. OO Prog., Syst., Lang. And Appl. –OOPSLA, pp.353-361, 1989
[8] R. Katz: towards a Unified Framework for Version Modeling in Engineering Databases, ACM Comput. Surv., 22(4):375-408, 1990
[9] G. Kotonya and I. Sommerville: Requirements Engineering with Viewpoints, Software Engineering Journal, pp.5-19, 1996
[10] S. Abiteboul and A. Bonner: Objects and Views, Proc. ACM-SIGMOD Conference, pp.238-247, 1991
[11] F. Bancilhon and N. Spyratos: Update Semantics of Relational Views, ACM Trans. Database Syst., 6(4): 557-575, 1981
[12] N. Delisle and M. Schwartz: Contexts: A Partitioning Concept for Hypertext, ACM Trans. Office Inf. Syst., 5(2):168-186, 1987
[13] V. Kashyap and A. Sheth: Semantic and Schematic Similarities between Database Objects: A Context-Based Approach, VLDB Journal, 5(4): 276-304, 1996
[14] A. Ouksel and C. Naiman: Coordinating Context Building in Heterogeneous Information Systems, J. of Intelligent Inf. Syst., 3(2):151-183, 1994
[15] R. Michalski: How to Learn Impressive Concepts: A Method Employing a Two-Tiered Knowledge Representation for Learning, Proc. Of 4th Int. Workshop in Machine Learning, pp50-58, 1987
[16] P. Turney: robust classification with context-sensitive features, Industrial and Engineering applications of Artificial Intelligence and Expert Systems, IEA/AIE-93, pp.268-276, 1993
[17] J. Mylopoulos and R. Motschnig-Pitrik: Partitioning Information Bases with Contexts, Proc. of CoopIS'95, pp.44-55, 1995
[18] M. Theodorakis, A. Analyti, P. Constantopoulos and N. Spyratos: A Theory of Contexts in Information Bases, Technical Report 216, Institute of Computer Science FORTH, Crete Greece, 1998
[19] M. Theodorakis and P. constantopoulos: Context-Based Naming in Information Bases, International Journal of Cooperative Information Systems, 6(3&4):269-292, 1997
[20] H. Weber: Modularity in Data Base System Design: A Software Engineering View of Data Base Systems, VLDB Surveys, pp.65-91, 1978
[21] Teodorakis, M., Analyti, A., Constantopoulos, P., and Spyratos, N.: Context in Information Bases, Proc, of the 3rd Int. Conference on Cooperative Information Systems (coopIS '98), pp.260-270 (1998)
[22] M. Theodorakis, A. Analyti, P. Constantopoulos and N Spyratos: Querying Contextualized Information Bases, Proc. of the 24th Intern. Conference on Information and Communication Technologies and Programming, (ICT&P ' 99), 1999
[23] M. Theodorakis, A. Analyti, P. Constantopoulos and N. Spyratos: Contextualization as an Abstraction Mechanism for Conceptual Modelling, Proc. of the 18th Intern. Conference on Conceptual Modelling (ER '99), Paris, pp. 475-489, 1999
[24] Okada, Y. and Tanaka, Y.: IntelligentBox: A Constructive Visual Software Development System for Interactive 3D Graphic Applications, Proc. Of Computer Animation '95, pp.114-125 (1994)
[25] Itoh, M. and Tanaka, Y: WorldMirror and World Bottle: Components for Embedding Multiple Spaces in a 3D Virtual Environment, Journal of IPSJ, Vol. 42, No.10, pp.2403-2414 (2001)
[26] Ohigashi, M. and Tanaka, Y.: A Framework for the Virtual Reification of Database Records, IPSJ, Vol.42, No. SIG 1 (TOD8), pp80-91 (2001)
[27] Chalmers, M. and Chitson, P.: Bead: Explorations in Information Visualization, Proc. ACM SIGIR '92, published as a special issue of SIGIR forum, pp.330-337 (1992)
[28] Hemmje, M., Kunkel, C. and Willet, A.: LyberWorld –A Visualization User Interface Supporting Fulltext Retrieval, Proc. ACM SIGIR '94 (1994)
[29] Mariani, J. and Benford, S.: Populated Information Terrains: Virtual Environments for Sharing Data, Technical Report CSCW/4/94, Lancaster University, Computing department (1994)
[30] Massari, A., Saladini, L., Hemmja, M. and Sisinni, F.: Virgilio: A Non-Immersive VR System To Browse Multimedia Databases, Proc. Intl. Conf. on Multimedia Computing and Systems, IEEE (1997)

Information Modelling and Knowledge Bases XIV
H. Jaakkola et al. (Eds.)
IOS Press, 2003

Modelling Time-Sensitive Linking Mechanisms

Anneli Heimbürger
VTT Information Technology, Multiple Media Group
P.O. Box 12041, FIN – 02044 VTT, Finland
anneli.heimburger@vtt.fi

Abstract. Hypermedia links are becoming a common tool for information management. They play an important role in tracking the relationships within and among sets of data. In fact, links are themselves important pieces of data that need to be authored, stored, maintained, delivered and used just like other data. Methods and tools to manage the full life cycle of links are essential for developing and maintaining high-quality hypermedia applications for the wired and wireless networks of the future. The XML Linking Language (XLink) specification, which has proceeded to recommendation status in the World Wide Web Consortium (W3C), provides mechanisms for defining link traversal rules and linkbases. We are interested in how temporal linking rules related to time-sensitive applications such as web-based news services or composing document assemblies in troubleshooting situations in the industry, can be defined by means of XLink's link structures. The purpose of this exploratory study is to describe the characteristics of extended links defined in the XLink specification from our research point of view, to introduce scenarios of time-sensitive applications and identify problems related to time-sensitive linking mechanisms for more precise investigation.

1. Introduction: Links for sale!

1.1 Background

Digital convergence - the merging of computers, communications, and multimedia - is transforming our lives. Traditional industries are reorganising, new enterprises and new models of business activities will be created. Digital convergence enables user, user group and context-sensitive products and services to be developed and delivered through different platforms.

Digital convergence will be based on standardised methods for automatic content management and automatic management of relations between contents and/or fragments of contents, i.e. links. When we are developing products and services for different users, user groups, contexts and platforms, hypermedia link management is one of the key issues to be considered, and thus is at the core of digital convergence.

Link databases or linkbases offer possibilities for filtering, sorting, analysing and processing link collections. It is possible to treat collections of links as independent databases that may lead to new categories of marketable information. In the future, there will be links for sale! A link broker will be able to play an entirely new role in the business domain. He/she will develop new general and customised information services and products which are based on the management of link collections. For example, collections of customised link sets related to a specific time period could be sold as separate products.

1.2 What is the problem?

Hypermedia link management has become a matter of serious concern in the World Wide Web Consortium, and has been so in the open hypermedia research community for over ten years now. When links are embedded in documents, as they are now in the WWW, there are several disadvantages which affect the scalability and management effort of contents and their relations. Human resource requirements, link maintenance and authoring are some examples of these:

- Human resource requirements: As collections of contents become ever larger, the effort required to author the links manually grows exponentially. Automatic methods are needed.
- Link maintenance: As contents change, it becomes very difficult to maintain all the links so that the associations between the contents and fragments of contents are reliable. Links will often fail if documents are moved, edited or deleted.
- Authoring: Embedded links have to be authored explicitly. They cannot be generated automatically. Users can follow links to documents, which the author of the document is aware of and considers being relevant. Relevant links cannot be processed automatically.

All these matters are out of the question when context and time-sensitive applications are considered.

Open hypermedia research and the recent activities of the W3C indicate that the solution to many of these issues is to separate the links from the content. The solution is an external hypertext or hypermedia link database architecture. If the links are kept separate from the contents, more flexibility and control is achieved, such as:

- contents and fragments of contents can be linked automatically without changing them in any way
- alternative sets of links can be associated with the same content to suit the needs of different users, user groups, contexts and delivery platforms
- link integrity can be verified automatically.

1.3 Focus of the research

In time-sensitive applications to be used for instance in mobile environments, there is a need to develop methods for composing time-critical pieces of information on the fly according to users input. One example of the application domain is value-added services for integrated news publishing in the Web, such as financial news services for SMEs or crime-related news services. Another example is troubleshooting situations in industry, for example when a paper machine crashes. Step-by-step instructions that are related to a particular troubleshooting situation are needed quickly. Time is an essential element when composing instructions on the fly, because the different parts of the document assembly depend on how long the situation has continued.

The main focus of our research will be on the XML Linking Language (XLink), which, together with open hypermedia architecture and the concept of the Semantic Web, provide the theoretical framework for our research. We will pay special attention to the multiple linking mechanism defined in the XLink specification. At the moment, XLink is the only formal linking language that is based on the international standard [23] and defines attributes for linking elements.

This paper is an exploratory study of our research focus, and it is primarily concerned with characteristics of XLink's multiple, i.e. extended, linking mechanisms and scenarios for time-sensitive linking applications. The purpose of this exploratory study is to identify

problems related to time-sensitive linking mechanisms in the Web for more precise investigation.

1.4 Basic concepts

The basic concepts in our research are:

- Constructive research methodology: This methodology was introduced by Nunamaker et al. [33], and it provides four main avenues to approach a research problem: theory building, experimentation, observation and systems development. In our research, we will use combinations of these.
- Open hypermedia: Open hypermedia architecture stores and manages information about the links between multimedia contents and fragments of contents separately from the documents themselves. Links to and from a document can be traced by querying the link database. Links are named and have types so the user can differentiate them, for example a quote link and a reference link.
- Link database or linkbase: A set of links.
- Link management: In our research, we define link management to include the different processes from link authoring to link archiving, i.e. the whole life cycle of links (Figure 1).

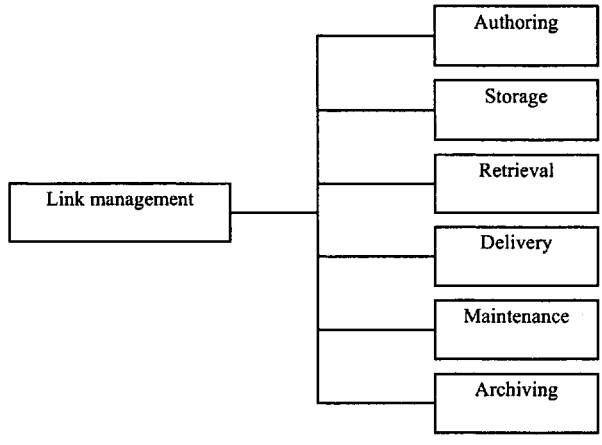


Figure 1. Different processes of link management.

- Link service: Link service supports creating, querying and maintaining the link data. A link service may operate with multiple linkbases.
- XML Linking Language (XLink) as defined by W3C [48]. XLink allows elements to be inserted into XML documents in order to create and describe links between resources.
- Resource Description Framework (RDF) as defined by the W3C [45, 52]. RDF is a general framework for describing any Internet resource. Such descriptions are often referred to as metadata or "data about data".
- Synchronized Multimedia Integration Language (SMIL) as defined by W3C [24, 46, 47]. SMIL is a mark-up language for multimedia elements on the World Wide Web.
- Multimedia Content Description Interface (MPEG-7) as defined by ISO [19, 30, 31]. MPEG-7 focuses on the standardization of a common interface for describing multimedia materials, i.e. representing information about the content.

- Structured links: Links have attributes, which define their functions and behaviour.
- Multiple linking: Links with multiple endpoints connect not only two but also a set of related nodes. When a user initiates the traversal of a link with multiple endpoints, he/she can be requested to choose between the available options (pop-up windows). Multiple links can also be used to automatically select the most decent destination by applying a filter. It would be even more desirable to filter by semantic criteria such as a user's task or profile. Some extended link functionality is already being simulated, showing pop-up menus with multiple destinations or extra information.
- The Semantic Web is an idea of World Wide Web inventor Tim Berners-Lee, Director of the World Wide Web Consortium [7, 8, 44]. The word "semantic" in the context of the Semantic Web means "machine-processable". Berners-Lee explicitly rules out the sense of natural language semantics. The aim of the Semantic Web is to have data on the Web defined and linked in such a way that it can be used by machines not just for display purposes, but for integration and reuse of information across various applications.
- Ontology: An ontology is a set of concepts - such as things, events and relations - that are defined, for example, in domain-specific controlled language in order to create an agreed-upon vocabulary for exchanging information. To standardise semantic terms, many areas use specific ontologies, which are hierarchical taxonomies of terms describing certain knowledge topics.
- Time sensitivity: The application and related information and/or pieces of information and connections between them are functions of time.

1.5 Overview of the paper

The remainder of the paper is organised as follows. Related work in link management and time ontologies are reviewed in Section 2. Features of the XML Linking Language are described in Section 3. The characteristics of XLink's extended link from our research point of view are discussed in Section 4. Scenarios for time-sensitive linking applications are introduced in Section 5. Our conclusions and issues for further research are presented in Section 6.

2. Related work

2.1 Link management

Hypertext research indicates that the solution to many link management issues is to separate the links from the content. The idea of a link service has existed since the days of Intermedia [9, 32, 54]. From a user's point of view, the need for distributed link services has grown with the development of the World Wide Web. Hyper-G and Microcosm's Distributed Link Service are two projects where support for non-embedded links to WWW pages has been developed [5, 11]. In Microcosm, SGML markup is used in the link databases and now some parts of the system have been modified to support some of the XLink-based linking facilities, such as controlling link behaviour and its presentation [11].

Grönbæk et al. [16] have described a distributed link service mechanism based on the Dexter model [18]. In this mechanism the links are maintained by a separate server, but combined with the text document by a Java applet embedded in the user's browser. Another distributed link service has been produced for the Aquarelle project [38].

Nürnberg et al. [35] have described the development of conceptual architectures of hypermedia systems, demonstrating various stages from monolithic systems to open hypermedia systems. Balasubramanian and Bashian [6] have reported an architecture for a

component-based authoring and publishing in the area of financial management and advisory services. In their architecture, information about products were separated into components. Wang and Rada [53] have investigated structured hypertext with domain semantics.

Page et al [36] have studied temporal linking in streaming applications. A hyperstructure can be viewed as metadata associated with a set of documents. In the multimedia context, the hyperstructure could be associated with multimedia documents including streamed temporal media. The hyperstructure itself can have a temporal dimension. For example, a link might be valid only for a given time interval in an audio stream. The authors have demonstrated Microcosm SoundViewer that used time intervals in the temporal media stream as anchors. Intervals could also be used to identify fragments of content from which features could be extracted for content-based navigation. The client browser can display the links with temporal relevance.

Miles-Board et al. [26] have defined an ontological hypertext that is a kind of hypertext whose structure and links are derived from the relationships between objects in the real world. According to the authors, by using the Open Hypermedia model on top of an ontological space, users can interact with such a system using simple browsing and navigation techniques. Users' actions are translated onto the ontological information space behind the scenes. They used hypermedia links that resolved to queries over the ontological information space. In this way, querying becomes a process of link following. The authors have demonstrated their ideas using the Dynamic Curriculum Vitae application.

Open hypermedia is an area that has been researched by the hypermedia community for several years and for which a number of systems have been implemented [15, 16, 28, 34, 35, 37]. In open hypermedia systems (OHS), links are managed and stored in special databases called link databases or linkbases. The idea of abstracting links from documents allows for a great deal of flexibility in link maintenance and re-use. The development of hypermedia architectures is presented in Figure 2 [35].

Common usage of the Web involves embedding links within documents in the HTML format. In this sense, the Web can be considered as a closed hypermedia system. However, there is nothing inherent in the Web infrastructure that prevents hypertext links from being abstracted away from documents and managed separately, by using mechanisms defined in the XML Linking Language (XLink) specification [48].

The recently proposed XML Linking Language (XLink) has proceeded to recommendation status in the World Wide Web Consortium (W3C). XLink is increasingly moving the Web towards the open hypermedia approach.

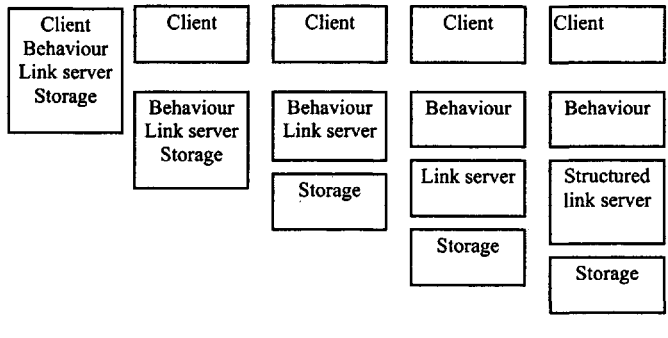


Figure 2. The development of hypermedia architectures: a) monolithic systems; b) open link service systems; c) open hyperbase; d) open hypermedia systems, and e) component-based open hypermedia systems.

2.2. Ontology of time

The recent research on ontologies and the underlying infrastructure related to the World Wide Web developments provide interesting possibilities for information and link management in the Semantic Web [7, 8, 44]. Ontologies and ontology representation languages are a popular research topic in various communities such as knowledge engineering, natural language processing, co-operative information systems, intelligent information integration, and knowledge management [4, 10, 12, 13, 17, 27, 29, 40].

Several authors have also dealt with time ontologies [1, 2, 3, 14, 25, 39, 43,] and there are some time ontologies available on the Web [55].

For humans, time is understood in terms of events. Time, as an abstract concept, means a space of time points reached from one another by before and after - like operators [14]. Time is a collection of temporal items, moments, durations and instants. Things in time are correlated somehow with these temporal items. We call this correlation temporality or temporal rules. Some major issues in the nature and in the structure of time have been identified: (a) linearity and circularity (periods); (b) finiteness and infinity; (c) openness and closure; (d) discreteness and continuity, and (e) absolute (past, present, future) and relative (before, concurrent-with, after) ordering [39].

One of the most fundamental questions when constructing an ontology of time issue is the choice of the primitive time entity. There are three options found in the literature [39]: (1) points of time (instants), (2) segments of time (intervals) and (3) occurrences in time (events). One consequence of an instant-based against an interval-based description emerges when we consider the relationship between any two entities. With a point description, a simple precedence relation is all we need, whereas with intervals we need a set of 13 different relations: before (t_1, t_2), equal (t_1, t_2), meets (t_1, t_2), overlaps (t_1, t_2), during (t_1, t_2), starts (t_1, t_2), finishes (t_1, t_2), and their inverse [1]. Relationships between two intervals are presented in Table 1 [1].

Table 1. Relationships between two intervals.

Relation	Symbol	Symbol for inverse	Example
X before Y	<	>	XXX YYY
X equal Y	=	=	XXX YYY
X meets Y	M	MI	XXXYYY
X overlaps Y	O	OI	XXX YYY
X during Y	D	DI	XXX YYYYYYY
X starts Y	S	SI	XXX YYYYYY
X finishes Y	F	FI	XXX YYYYY

In our research, time-sensitivity can be predictable temporal relationships among elements included in an application. In this case, temporal rules can be resolved and defined before the application runs. Such an application consists of a single timeline with which the different elements in the application are synchronized. It is not affected by the activation of links and as such does not provide for non-predictable changes. These can for instance be caused by user interactions or other external entities that lead to non-linear proceeding or adaptation according to user's interest. Non-predictable temporal relations can be modelled with time-sensitive link traversals. Non-predictability creates a temporal order of events that occur during runtime in time-sensitive applications. Each instantiation can be different.

3. XML Linking Language Family

3.1 HTML linking limitations

Many earlier hypermedia systems had a linking structure. For example, the Intermedia system developed by the Brown University (USA) in the early 1970's had structured links with attributes such as type and keyword [54]. Links in Intermedia were also birectional and it was possible to get an overview of the link structure in a hypermedia document. These features are still not on the Web! The linking mechanism in the Web is weak, although there are some strengths in the HTML linking. HTML has an easy syntax and it is trivial to implement.

There are several linking limitations in the HTML. Firstly, the authors and/or the users of hypermedia documents cannot create their own sub-types for links because the HTML does not contain any attribute element for linking. Secondly, the HTML link behaviour is tied to replacing a document in the same window when a user activates the link. Thirdly, there is no real link type system. There is no standard or portable set of link types. Fourthly, link databases are not allowed, which means that link sets cannot be filtered or selected on demand. Fifthly, an HTML link can point to a whole document or to a certain point inside the document but there is no mechanism to define a specified part or range of a document to be pointed at. Finally, there is no multi-ended links in HTML. Examples of uses for multi-ended links are book indexes, picking up multiple commentaries at once, creating document assemblies, parallel views of editions and translations.

3.2 XLink, XPointer, XPath and XML Base

The Standard Generalized Markup Language (SGML) is ISO's international standard for defining markup languages, and it is designed to promote text interchange [23]. Markup languages define the markup rules, which add meaning to the structure and content of documents. They are the grammar and the syntax which specify how a language should be "spoken". The Extensible Markup Language (XML) is a subset of the SGML, and it is usable over the Internet. The XML Linking Language (XLink) with recommendation status in the W3C promises to do the same thing for hypertext linking. XLink specifies how separate documents should be linked into one another, and how structures within XML documents should be addressed. XLink provides much more powerful link representation and addressing features than HTML. The XML Linking Working Group of the W3C has produced a family of linking related languages: XLink, XPointer, XPath and XML Base [48, 49, 50, 51]. The whole set is called the XML Linking Language family (Figure 3).

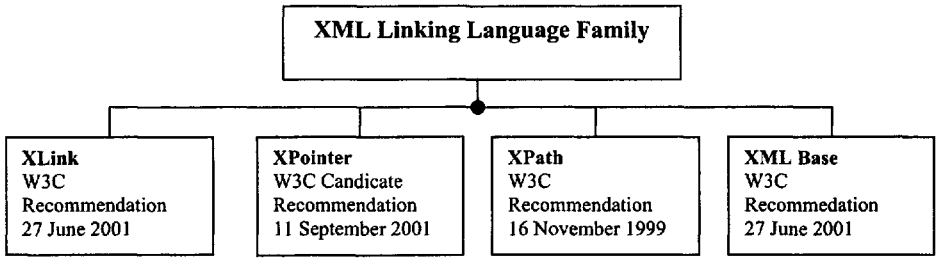


Figure 3. Members of the XML Linking Language Family and their W3C status.

XLink allows elements to be inserted into XML documents in order to create and describe links between resources. XPointer, based on XPath, identifies and locates parts of XML resources. They vary from points to complex regions and can even be distributed over the documents. XPointer gives a way to refer to a certain point or range selections inside an XML document. A future browser might load a document and scroll to or highlight the document's fragment which is defined by XPointer. The XPointer mechanism is useful for example when composing new document assemblies from existing parts. XPointer can go deep inside to fragments of an XML document. XPointer identifies sets of locations. XLink connects and describes them. XML Base defines a mechanism for providing base URI services to XLink.

From the users' point of view, the most important goals of the XML Linking Working Group are to provide:

- links from read-only documents
- attributes for links
- diverse link behaviour
- birectional (keeps links consistent and avoid broken links)
- multi-ended links
- annotation sharing
- precise link attachment in any media and possibility for link databases.

4. Extended links: XLink allows multiple linking

A link identifies a relationship between two documents or pieces of information. A linking element is an element that tells the existence and describes the characteristics of a link. A linking element can have any name, but a way to inform the XLink conforming browser that this element should be treated as a link is done by using the XML reserved and designated attribute *xlink*. A linking element can also have other attributes, such as *type*. A link type describes the relationship between the source and destination of a link, often derived from semantic categories such as "explanation" or "example". They are used to help users to get a better idea of a link target.

Links with multiple endpoints connect not only two but also a set of related nodes. In the XLink specification these are called extended links. When a user initiates the traversal of a link with multiple endpoints, he/she can be requested to choose between the available options (pop-up windows). Multiple links can also be used to automatically select the most decent destination by applying a filter. It would be even more desirable to filter by semantic criteria such as time and a user's task or profile. Some extended link functionality is already being simulated, showing pop-up menus with multiple destinations. Extended links can contain extended links as well, which makes nested tables of contents possible. A link author can also use the attributes *show* and *actuate* to describe the behaviour of the link.

An extended link is a link that associates an arbitrary number of resources. Figure 4 shows an extended link that associates five resources and provides traversal rules among them. The resources could be, for example, components of technical customer documentation. One resource is a description of the installation procedures, another resource represents operation, two resources contain matters concerning maintenance and training, and the last resource represents identified troubleshooting situations.

Without the extended link, the resources might be entirely unrelated. They might be in five separate documents, which is not desirable for example in troubleshooting situations where pieces of information from the whole documentation are needed and situation-specific assemblies should be composed. The nondirectional solid lines indicate that the

link connects the five resources. They do not have directionality. The dotted arrows indicate the traversal rules that the link provides. The traversal rules are necessary because without them the resources are associated in no particular order, with no implication as to whether and how individual resources are accessed.

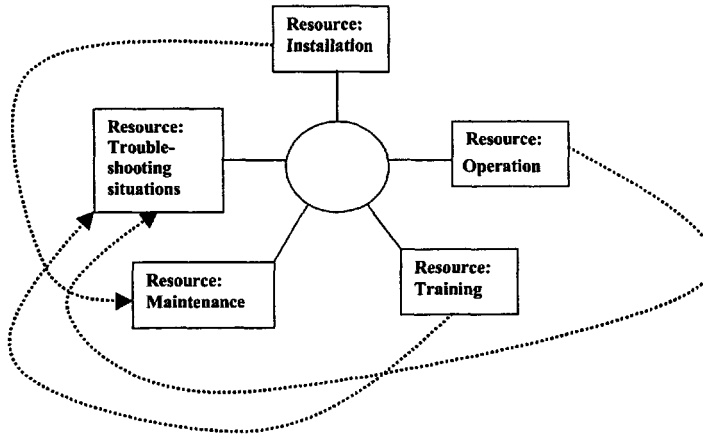


Figure 4. An extended link that associates five resources.

Following is an example of how an XLink element might look:

```
<techdocload>
    <tcd>Technical Customer Documentation</tcd>
        <installation
        xlink:href="Technical_object/installation.xml"
        xlink:label="installation "
        xlink:role="http://www.example.com/linkprops/installation "
        xlink:title="Installation"/>
        <operation
        xlink:href="Technical_object/operation.xml"
        xlink:label="operation"
        xlink:role="http://www.example.com/linkprops/operation"
        xlink:title="Operation"/>
        <training
        xlink:href="Technical_object/training.xml"
        xlink:label="training"
        xlink:role="http://www.example.com/linkprops/training"
        xlink:title="Training"/>
        <go
        xlink:type="arc"
        xlink:from="installation"
        xlink:arcrole="http://www.example.com/linkprops/maintenance"
        xlink:to="maintenance"
        xlink:show="replace"
        xlink:actuate="onRequest"
        xlink:title="Maintenace:_Code_434" />
        <go
        xlink:type="arc"
        xlink:from="operation"
        xlink:arcrole="http://www.example.com/linkprops/troubles"
        xlink:to="troubles"
```

```
            xlink:show="replace"
            xlink:actuate="onRequest"
            xlink:title="Trouble_shooting_situation:_Code_434" />
            <go
            xlink:type="arc"
            xlink:from="training"
            xlink:arcrole="http://www.example.com/linkprops/troubles"
            xlink:to="troubles"
            xlink:show="replace"
            xlink:actuate="onRequest"
            xlink:title="Trouble_shooting_situation:_Training_Code_434" />
</techdocload>
```

An extended link indicates rules for traversing among its resources by means of arc elements. The arc-type element may have the traversal attributes *from* and *to*, the behaviour attributes *show* and *actuate* and the semantic attributes *arcrole* and *title*. The traversal attributes define the desired traversal between resources that are connected by the same link. The *from* attribute defines a starting resource, and the *to* attribute an ending resources. The behaviour attributes specify how the ending resources will be shown to the user. For example "replace" means replacing windows on a screen.

The semantic attributes describe the meaning of the arc's ending resource relative to its starting resource. This contextual role can differ from the meaning of an ending resource when taken outside the context of this particular arc. For example, a resource might generically represent a "technical object", but in the context of a particular arc it might have the role of a "paper machine" and in the context of a different arc it might have the role of a "base station for mobile phones."

Linkbases are often used to make link management easier by gathering together a number of related linking elements. XLink provides a way to instruct XLink applications to access potentially relevant linkbases. Any linkbase specified as the ending resource of an *arc* with a special value must be an XML document. The traversal of a linkbase *arc* loads the ending resource (the linkbase) to extract its links for later use.

The timing of linkbase *arc* traversal depends on the value of the *actuate* attribute on the *arc*. For example, if the value is "onLoad", the linkbase is loaded and its links extracted as soon as the starting resource is loaded. If the value is "onRequest", the starting resource may consist of the words "Click here to see the list of identified troubleshooting situations", allowing the user to confirm before actuating the link. Following is an example of an XLink element loading a linkbase:

```
<loadbase>
        <startrsrc
                xlink:type="arc"
                xlink:arcrole="http://www.w3.org/1999/xlink/properties/linkbase"
                xlink:label="troubles"
                xlink:href="troubles.xml#string-range(//*,'Click here to see the list of identified trouble-
                shooting situations')" />
                <linkbase xlink:label="linkbase" xlink:href="linkbase.xml" />
                <load xlink:from="troubles" xlink:to="linkbase" actuate="onRequest" />
</loadbase>
```

We are interested in investigating more precisely the possibilities for associating temporal rules to extended link structure. Examples of these rules are presented in Figure 5. There are at least the following possibilities:

- to extend the *arc*-element's "from-to" structure with conditional time parameters
- to construct an XLink linkbase of temporal rules based on a time ontology
- a combination of these two.

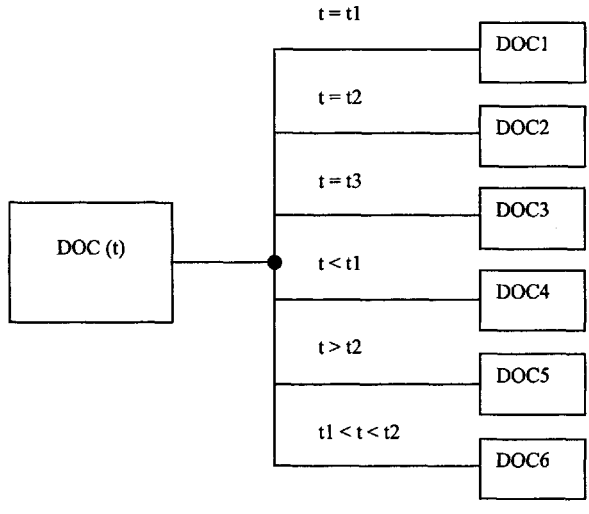


Figure 5. Examples of time-sensitive linking rules.

5. Scenarios for applications

When designing ways of organising information, it is reasonable to use metaphors which are already familiar to people. There are at least five such ways which are easy to perceive for people: (1) alphabetical order, (2) categories, (3) hierarchy, (4) location and (5) time. Alphabetical order is familiar to us from encyclopaedias. We categorise objects into logical groups. The BBC World News Site is one example of this. Hierarchy denotes to us a sequence of importance. An example of location is an atlas. Time can be the point of an actual event or a sequence of events in linear time such as hours, days, years, decades and centuries.

In our vision, the core components of a time-sensitive linking architecture are domain-specific content management with different actors and their roles in the value chain, time ontologies and linkbases (Figure 6). By means of a domain analysis, time-sensitive pieces of information related to the application concerned will be identified. The temporal rules will be defined and a time ontology created. The ontology will contain general and application-specific rules. There already exist some time ontologies which could be useful in our problem. How to convert temporal rules to extended link structures is one of our main problems.

When considering time as a link attribute, we should also pay attention to other markup languages, such as SMIL and MPEG-7. The Synchronized Multimedia Integration Language (SMIL) is used for multimedia presentations which integrate streaming audio and video with images, text or any other media type. The SMIL Linking Modules define the SMIL 2.0 document attributes and elements for navigational linking. The MPEG-7 standard (Multimedia Content Description Interface) might also offer interesting linking features i.e. composition information about how text, still pictures, graphics, audio, speech, video elements are combined in time-based applications.

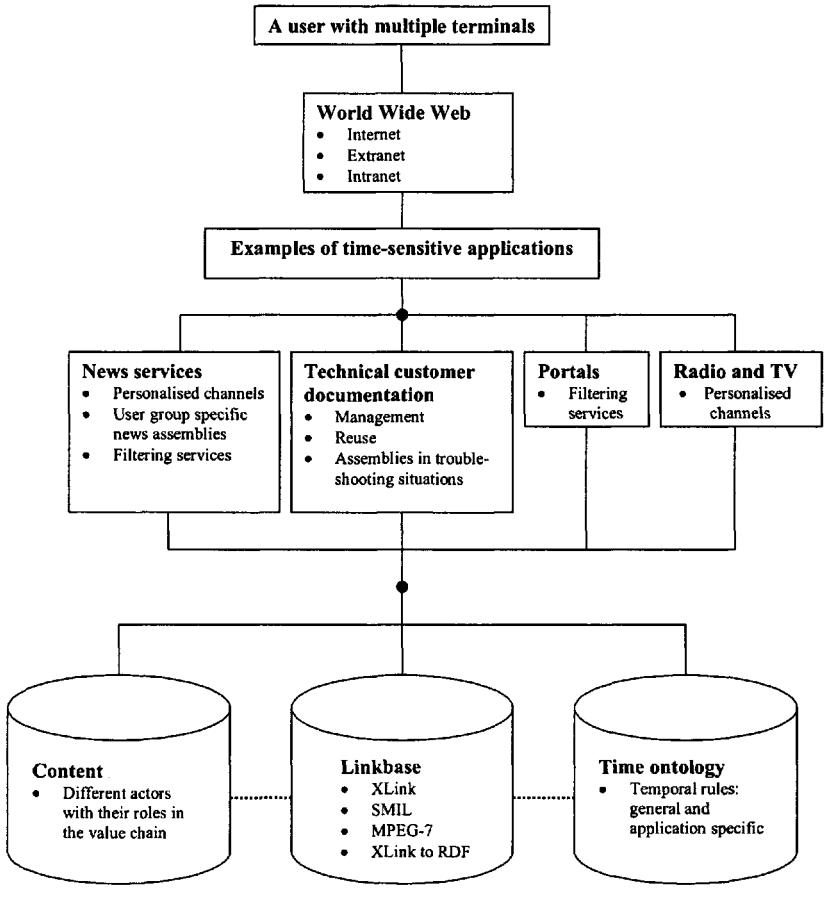


Figure 6. A vision of a time-sensitive linking architecture. Solid arrows indicate flow of information and dashed lines highlight the connections between the contents, linkbases and time ontologies.

Examples of possible applications are personalized radio and TV channels, portals, technical customer documentation and value-added services for integrated news publishing systems.

5.1 Value-added services for integrated news publishing systems

An integrated news publishing system for TV programmes and newspaper articles has been developed by VTT Information Technology together with Finnish enterprises. A continuously updated WWW-multimedia publication is created by combining the contents of online news sites from several newspapers and TV companies. Users access the service from the Internet on their PCs with normal WWW browsers or from television sets connected to the Internet through set-top boxes. The news can also be downloaded as synthesised speech into portable MP3 players, a service of particular interest to the visually impaired. The system has been designed to work with mobile devices as well.

The heart of the system is an active proxy server. Articles and TV news are deposited in its database. The system analyses the Web version of each newspaper, divides it up into structural elements, and separates off the necessary metadata. In the same way, TV news is broken down into items, using video analysis and closed caption texts. The contents are stored as objects which are then combined to form a publication tailored to the reader's preferences. News articles are tagged according to the XMLNews-Story specification. The metadata of an article is stored in the relational database. The system has two channel types, i.e. common and personal. Each channel is defined as a news composite. The common channel offers general news items as defined by the publisher, and the personal channel can be personalised according to the user's interests. A more detailed description of the technological solutions and user experiences are reported elsewhere [22, 41, 42].

One of the basic characters of news materials is time sensitivity both from producers and users' point of view. News and document sets could be composed on the fly according to some time criteria. Personalized or user group-specific news sets such as a financial news service for SMEs, according to time or time and subject criteria, is one possible application of time-sensitive linking mechanisms. Time-sensitive linking mechanisms can also support optional services such as filtering out identical or similar news stories and showing only the most relevant story with links to others.

5.2 Technical customer documentation

The XLink specification provides interesting possibilities to manage technical customer documentation in extranet and intranet environments, specially for a heterogeneous industrial project organisation [20, 21]. Producing, updating, composing, managing, testing, delivering and using the customer documentation can be done in a common network. Task, situation and user group-specific information needs can be supported. In troubleshooting situations in industry, instructions are needed quickly. Time is an essential element when composing instructions on the fly, because the different parts of the document assembly depend on how long the situation has continued.

Link management based on the XLink language makes it possible for the same information to be relinked automatically. XLink provides nested information hierarchies, many-to-one and one-to-many relationships between documents. Various sets of links can be associated with the same content to create for example user group-oriented views to information. XLink's ability to define more structure for links makes them easier to control. This leads to easier maintenance and to greater possibilities for automated processing. It is possible to treat collections of links as independent databases that may lead to new categories of marketable information.

6. Conclusions and issues for further research

The XML Linking Language family is starting to revolutionize the way links and information on links are produced and managed in the Web. XLink and XPointer are the two main members of the family. Together they provide much more powerful link representation and addressing features than HTML. XLink with its attributes provide advanced linking capabilities such as multiple linking, while the XPointer specification provides a way of identifying locations in XML documents. As a whole, the XML Linking Language gives us an interesting possibility to treat links as structured objects as we do with documents. Although XLink does not support time or time sensitivity as such, its extended link structure provides a challenging basis for developing and constructing time-sensitive linking mechanisms.

In this study, features of link management and time ontologies have been reviewed. The XML Linking Language family have been described. The exploratory study has concentrated on the issues related to extended links defined in the XLink specification. Scenarios for time-sensitive linking applications have been introduced. We have identified the following problems for further research, and they are divided into three main categories: (a) domain analysis, (b) link management in general, and (c) modelling and constructing time-sensitive linking mechanisms.

Domain analysis
- What is the most systematic way to identify time-sensitive contents and contexts?
- How does time-sensitivity effect on different actors and their roles in the value chain?

Link management in general
- What is the life cycle of links?
- Are there any general or domain-specific principles for constructing taxonomies of links?
- What kind of information should the records of a link database contain so that they support link management processes, such as maintaining, verifying, processing, sorting, filtering, analysing and reusing link collections?
- How do the XLink and similar languages support link behaviour and link user interfaces?
- How does XLink and RDF overlap and how does the mapping from XLink links to statements in an RDF model work?

Modelling and constructing time-sensitive linking mechanism
- What is the life cycle of links in time sensitivity applications?
- What does time sensitivity as a link attribute require from linking mechanisms?
- How can time-sensitive link traversal rules be presented by means of the XLink's extended link structure?
- Can temporal rules be stored in linkbases and accessed via extended link with arcrole attribute?
- How can temporal information be identified and extracted from a document itself?
- How do the linking modules of SMIL and MPEG-7 support time sensitivity?
- Are the existing ontologies of time usable or partly usable in our research?
- Which XLink editors do support extended links?

Hypermedia links are becoming a common tool for information management. They play an important role in tracking the relationships within and among sets of data. In fact, links are themselves important pieces of data that need to be authored, stored, maintained, delivered and used just like other data. Methods and tools to manage the full life cycle of links are essential for developing and maintaining high-quality hypermedia applications for the wired and wireless networks of the future.

In the future, there will be an entirely new market for links. An information broker will be able to play an entirely new role in the business domain. He/she will be able to act as a link broker, developing new general and customised information services and products which will be based on the management of link collections. Anyone will be able to publish and sell their commentary and it will be possible to filter links on demand in different contexts. XML Linking Language editors or browsers which support these functions are still in their very development stage, but these represent the future.

References

[1] Allen, J. F. 1983. Maintaining knowledge about temporal intervals. Communications of the ACM, Vol. 26, No. 11, pp. 832 – 843.

[2] Allen, J. F. 1984. Towards a general theory of action and time. Artificial Intelligence, Vol. 23, No. 2, pp. 123 – 154.

[3] Allen, J. F. 1991. Time and time again: The many ways to represent time. International Journal of Intelligent Systems, Vol. 6, No. 4, pp. 341 –355.

[4] Ankolekar, A. et al. 2001. DAML-S: Semantic markup for Web Services. In: The Proceedings of the First Semantic Web Working Symposium, July 30 - August 1, 2001, Stanford University, California, USA. Pp. 411 - 430.

[5] Andrews, K., Kappe, F. and Maurer, H. 1995. Serving information on the Web with Hyper-G. Computer Networks and ISDN Systems, Vol. 27, No.6, pp. 919 – 926.

[6] Balasubramanian, V. and Bashian, A. 1998. Document management and Web technologies: Alice marries the Mad Hatter. Communications of the ACM, Vol. 41, No. 7, pp. 107 – 115.

[7] Berners-Lee, T. 1999. Weaving the Web. New York: HarperSanFrancisco. 226 p. ISBN 0-06-251586-1.

[8] Berners-Lee, T., Hendler, J. and Lassila, O. 2001. The Semantic Web. Scientific American, vol. 284, No. 5, pp. 28 – 37.

[9] Bieber, M., Vitali, F., Ashman, H., Balasubramanian, V. and Oinas-Kukkonen, H. 1997. Fourth generation hypermedia: some missing links for the World Wide Web. International Journal of Human-Computer Studies, Vol. 47, No. 1, pp. 31 - 65.

[10] Carr, L., Bechhofer, S. Goble, C. and Hall, W. 2001. Conceptual linking: Ontology-based open hypermedia. In: Proceedings of the 10th World Wide Web Conference (referred 31.5.2001) <URL: http//www10.org>.

[11] Carr, L. A., Hall, W. and Hitchcock, S. 1998. Link services or link agents? In: Grønbæk, K., Mylonas, E. and Shipman, F. M. (eds.) Proceedings of the Ninth ACM Conference on Hypertext and Hypermedia, Pittsburgh, PA 20 – 24 June 1998, USA. The Association for Computing Machinery. Pp.113 – 122.

[12] Cover, R. 2001. Ontology Interchange Language (OIL). OASIS the XML Cover Pages (referred 5.2.2002) <URL:http://www.oasis-open.org/cover/oil.html>.

[13] Crow, L. and Shadbolt, N. 2001. Extracting focused knowledge from the semantic web. International Journal of Human-Computer Studies, Vol. 54, No. 1, pp. 155 - 184.

[14] Giumale, C. A. and Kahn, H. J. 1993. An information model of time. In: Proceedings of the 30th International Annual ACM IEEE Design Automation Conference, Dallas, Texas, USA. New York, NY: ACM Press. Pp. 668 – 672.

[15] Grønbæk, K. and Trigg, R. H. 1999. From Web to workplace: Designing open hypermedia systems. The MIT Press Books, Boston, MA. 424 pp.

[16] Grønbæk, K., Bouvin, N. O. and Sloth, L. 1997. Designing Dexter-based hypermedia services for the World Wide Web. In: Bernstein, M., Carr, L. and Østerbye, K. (eds.) Proceedings of the Eight ACM Conference on Hypertext, Southampton, UK. New York, NY: The Association for Computing Machinery, Inc. (ACM). Pp. 146 – 156.

[17] Guarino, N. 1995. Formal ontology, conceptual analysis and knowledge representation. International Journal of Human and Computer Studies, Vol. 43, No. 5/6, pp. 625 - 640.

[18] Halasz, F. and Schwartz, M. 1994. The Dexter hypertext reference model. Communications of the ACM, Vol. 37, No. 2, pp. 30 – 39.

[19] Hardman, L., Schmitz, P., van Ossenbruggen, J., ten Kate, Warner and Rutledge, L. 2000. The link vs. the event: activating and deactivating elements in time-based hypermedia. The New Review of Hypermedia and Multimedia, Vol. 6, pp. 89 - 109.

[20] Heimbürger, A. 1999. A structured link document as a new means for composing and publishing technical customer documentation in extranets and intranets. In: Smith, J. W. T, Ardö, A. and Linde, P. (eds.) Proceedings of the Third ICCC/IFIP Conference on Electronic Publishing, Ronneby 10 - 12 May 1999, Sweden. Washington, DC: ICCC Press. Pp. 90 - 102.

[21] Heimbürger, A. and Lanas, T. 1994. A practical model for computer-aided reading supports (CARS) for technical documents – The user's point of view. Microcomputers for Information Management. An International Journal for Library and Information Services, Vol. 11, No. 2, pp. 99 – 110.

[22] Heimbürger, A., Silvonen, P. and Södergård, C. 2001. A framework for automatic combination of media contents by minimising information redundancy. Case: Integrated publishing in multimedia networks. In: Hübler, A., Linde, P. and Smith, J. W. T. (eds.) 2001. Proceedings of the Fifth ICCC/IFIP Conference on Electronic Publishing, ELPUB2001 - "2001 in the Digital Publishing Odyssey". Pp. 218 - 230.

[23] ISO 8879. 1986. Information processing – Text and office systems – Standard generalized markup language (SGML). Geneva: International Organization for Standardization. 155 p.
[24] Jourdan, M. 2001. A formal semantics of SMIL: a web standard to describe multimedia documents. Computer Standards & Interfaces, Vol. 23, No. 5, pp. 439 – 455.
[25] Kitamura, Y., Ikeda, M. And Mizoguchi, R. 1997. A causal time ontology for qualitative reasoning. In: Proceedings of the 15th International joint Conference on Artifial Intelligenec (IJCAI-97). Pp. 501 – 506.
[26] Miles-Board, T., Kampa, S., Carr, L. and Hall, W. 2001. Hypertext in the Semantic Web. In: Davis, H., Douglas, Y. and Durand, D. G. (eds.) 2001. ACM Hypertext 2001, Proceeding of the 12th ACM Conference on Hypertext and Hypermedia. Aarhus, Denmark, 14 – 18 August, 2001. New York: The Association for Computing Machinery. Pp. 237 - 238. ACM ISBN 1-59113-420-7.
[27] Miller, G. A. 1995. WordNet: A lexical database in English. Communications of the ACM, Vol. 38, No. 11, pp. 39 - 41.
[28] Modha, S. And Spangler, W. 2000. Clustering hypertext with applications to web searching. San Jose: IBM Almaden Research Center. 19 p. (Research Report RJ 10160)
[29] Motta, E., Buckingham Shum, S. and Domingue, J. 2000. Ontology-driven document enrichment: principles, tools and applications. International Journal of Human-Computer Studies, Vol. 52, No. 6, pp. 1071 - 1109.
[30] Nack, F. and Lindsay, A. T. 1999. Everything you wanted to know about MPEG-7: Part 1. IEEE Multimedia, Vol. 6, No. 3, pp. 65 – 77.
[31] Nack, F. and Lindsay, A. T. 1999. Everything you wanted to know about MPEG-7: Part 2. IEEE Multimedia, Vol. 6, No. 4, pp. 64 – 73.
[32] Nielsen, J. 1995. Multimedia and hypertext. Massachusetts, MA, Cambridge: AP Professional. 480 p.
[33] Nunamaker, Jr. J. F., Chen, M. and Purdin, T. D. M. 1991. Systems development in information systems research. Journal of Management Information Systems, Vol. 7, No. 3, pp. 89 - 106.
[34] Nürnberg, P. J. and Wiil, U. K. 2001. Metadata management through open hypermedia. Tutorial 3 in ACM Hypertext 2001, the 12th ACM Conference on Hypertext and Hypermedia. Aarhus, Denmark, 14 – 18 August, 2001. 25 p.
[35] Nürnberg, P. J., Leggett, J. J. and Wiil, U. K. 1998. An agenda for open hypermedia research. In: Grønbæk, K., Mylonas, E. and Shipman, F. M. (eds.) Proceedings of the Ninth ACM Conference on Hypertext and Hypermedia, Pittsburgh, PA 20 – 24 June 1998, USA. The Association for Computing Machinery. Pp.198 – 206.
[36] Page, K. R., Cruickshank, D. and De Roure, D. 2001. It's about time: Link streams as continuous metadata. In: Davis, H., Douglas, Y. and Durand, D. G. (eds.) 2001. ACM Hypertext 2001, Proceeding of the 12th ACM Conference on Hypertext and Hypermedia. Aarhus, Denmark, 14 – 18 August, 2001. New York: The Association for Computing Machinery. Pp. 93 – 102. ACM ISBN 1-59113-420-7.
[37] Reich , S. and Anderson, K. M. (eds.) 2000. Open hypermedia systems and structural computing : Proceedings 6th International Workshop, Ohs-6, 2nd International Workshop, Sc-2, San Antonio, Texas, USA, May/June, 2000. Berlin: Springer. 183 p. (Lecture Notes in Computer Science 1903)
[38] Rizk, A. and Sutcliffe, D. 1997. Distributed link service in the Aquarelle project. In: Bernstein, M., Carr, L. and Østerbye, K. (eds.) Proceedings of the Eight ACM Conference on Hypertext, Southampton, UK. New York, NY: The Association for Computing Machinery, Inc. (ACM). Pp. 208 - 227.
[39] Schreiber, F. A. 1994. Is time a real time? An overview of time ontology in informatics. In: Halang, W. A. and Stoyenko, A. D. (eds.) Real time computing. Proceedings of the NATO Advanced Study Institute, Sint Maarten, Dutch Antilles, 5 – 17 October, 1992. Berlin, Germany: Springer-Verlag. Pp. 283 – 307. ISBN 3-540-57558-8.
[40] Staab, S., Angele, J., Decker, S., Erdmann, M., Hotho, A., Maedche, A., Schnurr, H.-P., Studer, R. and Sure, Y. 2000. Semantic community Web portals. In: Proceedings of the 9th International World Wide Web Conference, Amsterdam, The Netherlands. Elsevier Amsterdam. Pp. 473 – 491.
[41] Södergård, C., Aaltonen, M., Hagman, S., Hiirsalmi, M., Järvinen, T., Kaasinen, E., Kinnunen, T., Kolari, J., Kunnas, J. and Tammela, A. 1999. Integrated multimedia publishing: combining TV and newspaper content on personal channels. Computer Networks, Vol. 31, No. 11-16, pp. 1111 – 1128.
[42] Södergård, C. (ed.) 2001. Integrated news publishing - Technology and user experiences. Report of IMU2 project. Espoo: Technical Research Centre of Finland (VTT). 206 p. + app. 26 p. (VTT Publications 441.) ISSN 1235-0621
[43] Terenziani, P. 1996. Towards an ontology dealing with periodic events. In: Wahlster, W. (ed.) ECAI 96 12th European Conference on Artificial Intelligence, Budapest, Hungary, 11 – 16 Aug 1996. Chichester, UK: Wiley. Pp. 43 – 47. ISBN: 0-471-96809-9.
[44] W3C. 2002a. The World Wide Web Consortium: Semantic Web Activity, (referred 5.2.2002) <URL: http://www.w3.org/2001/sw/>.

[45] W3C. 2002b. The World Wide Web Consortium: The Resource Description Framework (RDF), (referred 5.2.2002) <URL: http://www.w3.org/RDF/>.
[46] W3C. 2002c. The World Wide Web Consortium: The Synchronized Multimedia Integration Language (SMIL), (referred 5.2.2002) <URL: http://www.w3.org/AudioVideo/>.
[47] W3C. 2002d. The World Wide Web Consortium: XHTML+SMIL Profile W3C Note 31 January 2002, (referred 5.2.2002) <URL: http://www.w3.org/TR/2002/NOTE-XHTMLplus/>.
[48] W3C. 2002e. The World Wide Web Consortium: XML Linking Language (XLink) Version 1.0 W3C Recommendation 27 June 2001, (referred 5.2.2002) <URL: http://www.w3.org/TR/xlink/>.
[49] W3C. 2002f. The World Wide Web Consortium: XML Path Language (XPath) Version 1.0 W3C Recommendation 16 November 1999, (referred 5.2.2002) <URL: http://www.w3.org/TR/xpath/>.
[50] W3C. 2002g. The World Wide Web Consortium: XML Pointer Language (XPointer) Version 1.0 W3C Candidate Recommendation 11 September 2001, (referred 5.2.2002) <URL: http://www.w3.org/TR/xptr/>.
[51] W3C. 2002h. The World Wide Web Consortium: XML Base W3C Recommendation 27 June 2001, (referred 5.2.2002) <URL: http://www.w3.org/TR/xmlbase/>.
[52] W3C. 2002i. The World Wide Web Consortium: Harvesting RDF Statements from Xlinks W3C Note 29 September 2000, (referred 5.2.2002) <URL: http://www.w3.org/TR/xlink2rdf/>.
[53] Wang, W. and Rada, R. 1998. Structured hypertext with domain semantics. ACM Transactions on Information Systems, Vol. 16, No. 4, pp. 372 – 412.
[54] Yankelovich, N., Haan, B. J., Meyrowitz, N. K. and Drucker, S. M. 1988. Intermedia: the concept and construction of a seamless information environment. IEEE Computer, Vol. 21, No. 1, pp. 81 – 96.
[55] Zhou, Q. And Fikes, R. 2002. A reusable time ontology (referred 5.2.2002) <URL: http://www.ksl.stanford.edu/KSL_Abstracts/KSL-00-01.html/>.

Information Modelling and Knowledge Bases XIV
H. Jaakkola et al. (Eds.)
IOS Press, 2003

Assisting Business Modelling with Natural Language Processing

Marek Labuzek
Computer Science Department,
Wroclaw University of Technology
labuzek@ci.pwr.wroc.pl

Abstract. There are many CASE tools that support software engineering but still in the requirement capture phase of the creation of a software system, there is a lot of difficult work to be done which is hardly supported and controlled. Business modelling belongs to these tasks. The paper describes an outline of a system supporting business modelling, which uses semantic modelling to automatically generate UML diagrams on the basis of a textual description. The current progress of implementation of the system is also reported.

1. Introduction

In the early phases of creating a software system it is very important to understand its context - the reality which it will, in one way, reflect and in the other - change. Various methodologies, for example [2], propose to create a conceptual model of this reality, usually called "business model". It is also often the case that a textual description is a main source of information about reality and when there is no such text, it is recommended to create it. Processing such texts is connected with several problems. Firstly, a specialised terms hamper the reading and understanding the text. Secondly, descriptions are not rarely quite thick books and processing such amount of information is inevitably liable to errors and oversights.

This paper presents an outline of a system which assists a software engineer in creating a business model on the basis of a textual description. It follows an approach described in [3] and uses natural language processing techniques to create a model of meaning of a given text, which is also a conceptual model of the described reality. Fragments of text which are erroneous or ambiguous are identified and after the corrections have been made, it generates UML ([1]) class diagram (modelling business entities) and use-case diagram (modelling business processes). The current progress of implementation of the system will also be presented together with a small example.

2. System architecture

2.1. External view

The work with the system proceeds as follows. The system analyses consecutive sentences of the text and reports problems of different kinds as explained below. The user either corrects the text or, in case of some ambiguities, can choose an appropriate interpretation from all possible ones presented as texts, which are their unambiguous paraphrases. When the sentence causes no problems, its unambiguous paraphrase is presented for final acknowledgement of interpretation. After processing whole text UML class diagram and use-case diagram are generated. The paraphrase of diagrams is also generated. It can be used to validate generated diagrams (and to some extend also the original text).

The problems with a text reported by the system belong to one of three categories: grammatical (morphological or syntactic), semantic or pragmatic ones. The first kind of grammatical problems is grammatical errors and too complicated constructions. When they are encountered, the fragment of the text must be corrected or rewritten. The second grammatical problem is ambiguity. It can be solved either through changing the text or through choosing one of presented paraphrases, which then replaces the original sentence.

Semantic problems can also be divided into errors and ambiguities. Semantic errors are simple contradictions, e.g. some entity is claimed in one place to have some property and in the other not to have it or a value of some attribute is specified inconsistently in more than one place. Some piece of text can be contradictory with some previous piece of text or with an external (to the text) knowledge base. This knowledge base is created by the user and consists of meaning of the text, which were processed earlier and/or facts entered by hand. Semantic ambiguities emerge either from deficiencies of semantic databases (described below) or from underspecification which is inherent in the text. The former can be solved through choosing an appropriate paraphrase. It will not become a part of the text, since they can be very unnatural e.g. "The person 'client' performs action of ordering goods.") but instead appropriate information is stored in a semantic database. The latter can be either avoided through adding more details to the text, or ignored, causing underspecification in generated diagrams.

Pragmatic problems occur when generated diagrams do not adhere to UML semantics (e.g. there is a circular inheritance in a class diagram) or other soundness rules e.g. testing the possibility of imprecision in the text. An example rule could indicate that mentioning "registry of sold devices" can be imprecise after mentioning "registry of devices" - there is probably one registry but maybe there is some additional condition connected with a registry and sold devices.

The mechanism of paraphrasing can also be used to integrate with the text the information acquired from other source and written directly into a UML diagram. The text of the paraphrase will serve then as a unified source of all knowledge about reality and it will be accessible to non-engineers.

2.2. Internal view

The picture below shows the operation of the system. The rounded rectangles represent activities, the remaining rectangles represent data and discs - databases. The most important data are bolded. Arrows drawn with solid lines represent the main dataflow, and those drawn with dashed lines - additional ones.

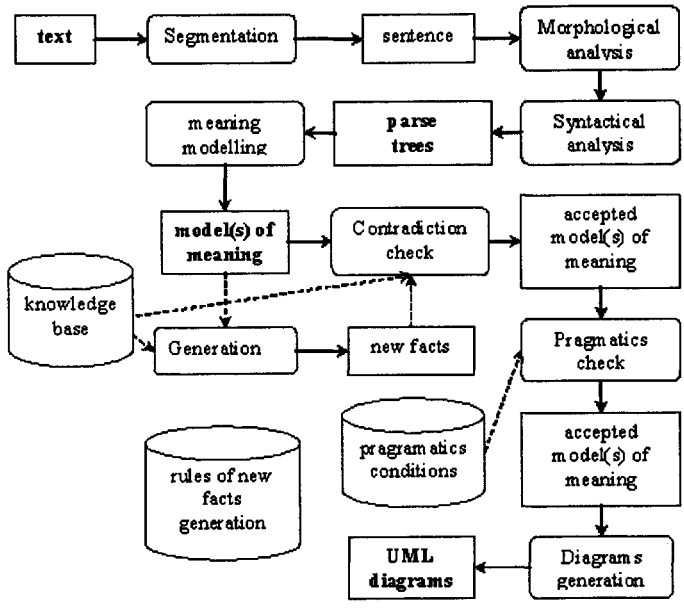


Fig. 1. The operation of the system

A text is processed in the following way. Firstly, it is segmented into words, numbers, punctuation marks and blocks of other kinds. A sequence of blocks if further divided into sentences. The syntax of each sentence is analysed and possible parse trees are generated. If no parse tree is generated then the sentence is rejected as erroneous or too complicated. If there is more than one parse tree, ambiguity is reported. The text can be changed or for each parse tree the next phases are executed. If these phases do not eliminate ambiguity, a set of interpretations is obtained. A paraphrase is generated for each interpretation and all are presented to the user, who chooses the right one.

For each processed parse tree, a model of meaning of a sentence is created. It is made in the context of a model created for all previous sentences, supporting a dynamic interpretation of a sentence, necessary for correct analysis of such phenomena as anaphora (the meaning of such expressions as personal pronouns and demonstrative pronouns), coreference (two fragments of the text relating to the same semantical being) and presupposition (conditions that must be fullfiled for an expression to be valid). In this phase, a database of semantic knowledge must be used. It contains mainly the information about semantic categories of words, for example whether a noun describes a person or other concept, whether a verb describes an action or a state. The model obtained for each processed parse tree is checked whether it is not contradictory (within itself) or contradictory with knowledge base. This process is supported by mechanism of generating new facts on the basis of the model of meaning and a knowledge base. The rules of generating new facts describe various relations in the world. Any contradictory models are eliminated. If there are no models left, the error is reported. In case of more the one model, there is an ambiguity. For each model a paraphrase is generated, and the user chooses the right one.

After successful modelling the meaning of all sentences, the whole model is checked for pragmatic problems. If there is none, UML class diagram and use-case diagram are created.

3. Semantic modelling

3.1. Meaning of a text

There are two main approaches to meaning: truth-conditional and the one that can be called "informational". In the former, the meaning is understood as conditions for a text to be true, while in the latter, the meaning is the information carried by a text. The described system uses the informational approach, since it is more natural for acquisition of a conceptual model of reality.

3.2. The ontology and structure of a business reality

In order to specify a model of meaning, one should decide what aspects of reality one wants to take into account. Considering the specificity of business models, it is assumed that in a given point of time, the reality consist of a set of entities, both physical and abstract, and a set of relationships of different kinds between them. These sets will be further called "configuration". A text is assumed to describe not a concrete configuration but a kind of template to which all possible configurations adhere and which describes possible changes from one configuration to another.

Basing on these assumptions, a meta-ontology of models of meaning can be now specified. It's skeleton is shown in fig. 2. The prefix "meta" is used here to distinguish this ontology from those described in business models. This special meta-ontology was created in order to account for various aspects connected with acquiring business models from textual descriptions (see section 3.3.) and simplicity of these models (e.g. comparing to UML meta-model, classes do not have operations and there are no interfaces).

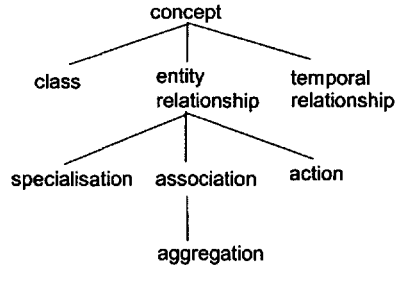


Fig. 2. Meta-ontology of a business reality

A class is a class of physical or abstract entities occurring in reality. It may have various properties referring to values of attributes of entities (e.g. "big" referring to the attribute "size"), or past or present configurations (e.g. "sold", "awaiting").

There are three kinds of relationships: specialisation, association and action. The first one holds between a more general class and a less general one. An association models a static relationship between entities. There is a special kind of association - aggregation, modelling whole-part relationships. An action refers to changes of configurations initiated by entities. Modelling actions as relationships is simplified - it takes into account an entity triggering the action and other entities taking part in it, but ignores such aspects as duration,

a scenario of action and its consequences. Such simplified account of actions usually occurs in descriptions of business reality.

Temporal relationships are modelled in a very simple way because it is assumed that in the textual description there is no scenarios but only simple temporal dependencies such as "A during B". It is proposed that temporal relationships connect two actions and can be of two kinds: *during* and *after*, which means that during action A occurs action B and after action A occurs action B, respectively. Temporal relationships have also a modality aspect, specifying whether action B occurs always or sometimes, when action A occurs.

3.3. Representation of a model of meaning

A set of connected frames was chosen as a representation of a model of meaning of a text, following Minsky's ideas [4]. A frame has three-level structure: it has an identifier of its type and a sequence of slots, they consist of an identifier of their types and a sequence of aspects and each aspect is a pair: its name being an identifier and its value. Identifiers are taken from a fixed finite set. A value of an aspect is an identifier, a string or a reference to an arbitrary frame.

The main reason for the choice of frames is that they allow to model a set of concepts and relationships which hold between them in a simple way. In particular, they make it possible to model semantics of noun phrases in a way which supports concept acquisition form a text written in a natural language. For example, in the following two sentences: "Electrical devices which have a serial number are recorded in a registry." and "Electrical devices have a serial number and are recorded in a registry.", there are two different concepts: "electrical devices having a serial number" and "electrical devices" respectively, although they have the same relationships: "has" with a concept "serial number" and "is recorded in" with a concept "registry". The difference is that the first relationship is a part of a definition of the first concept, while it is not in the case of the second concept. Using frames as a representation of meaning, it is easy to model such distinctions.

4. Generating diagrams

Thanks to the utilisation of semantic model and its similarity to UML diagrams, the algorithm of generating them is simple. It comprise two steps. In the first, all frames of entity relationships are processed and in the second - frames of temporal relationships. Processing an entity relationship depends on its type. For actions, the following elements of use case diagram are generated: an actor with the name of first participating entity, a use case with a name being deverbal noun phrase describing the action and a relation between them. For entity relationships elements of class diagram are created. Firstly, all participating entities are processed in the following way. The class for a given entity is generated, unless it was done earlier (for other relationship). If there is a generalization of this entity, then unless done earlier, a class is generated for more general entity and a relation of generalization between two generated classes is added. Of course this process can recursively create classes for more and more general entities. Secondly, an association between all classes generated for participants is created. The proper descriptions of multiplicity and aggregation are also generated. This algorithm assures that proper diagram elements are generated when one entity in semantic model participate in action and association like "customer" in a sample text given in the next section.

In the second step of diagram generation, for each temporal relationship a proper dependency between use cases is generated. If modality of relationship is "always" then the dependency has the stereotype "includes" and if modality is "sometimes", stereotype "extends" is generated. Diagrams for a sample text are given in the next section.

5. Current progress of implementation and future work

The core of a described system is already implemented. The program is prepared to process texts written in Polish. The dependence on language is very strong in case of syntactic analyser, is quite weak for the unit modelling the meaning and irrelevant for diagram generator.

Syntactical analysis is done through a grammar written in the DCG formalism [5]. It is based on a grammar described in [6], which was in some aspects simplified and in others amended. It covers many common constructions occurring in texts describing business reality. The grammar is parsed by a Prolog engine which produces all possible parse trees. There are usually just a few trees for not very complicated sentences. Some of them are still superfluous, but it is planned to eliminate them through the usage of syntactical patterns of verbs. It is also planned to implement a heuristic parser, which will accept more grammatical constructions, but probably will not produce all possible trees. Sentences not accepted by the grammar could then be parsed with this parser.

The unit modelling the meaning covers basic semantic phenomena occurring in sentences accepted by the grammar. It is implemented as a set of hand coded rules. Currently, the mechanism of inference is also being developed. It will allow for checking inconsistency between two sets of frames and deducing new frames from a given set frames on a basis of a rules represented as set of special pattern-frames. This mechanism can be used to eliminate some interpretations of a given sentence as being inconsistent with previous parts of the text or with some external knowledge stored in frames as well as to gather the acquired knowledge and utilised it while processing subsequent texts.

The unit generating diagrams produces a textual representation of diagrams. It will be developed to comply with the XMI [7] standard. The results could then be imported by many CASE tools.

The text and results given below are translated from data the system is tested on.

> The shop sells electrical devices. Each device is produced by one producer, has a name and a price. Devices having a serial number are recorded in a registry. A customer buys devices choosing them from a price-list. While choosing devices he can request all data of any device. The salesperson records the transaction writing down customer's data, a date and sold devices.

There will be generated frames for following concepts:

- entities: "shop", "electrical device", "producer", "name", "price", "serial number", "electrical device having serial number", "registry", "customer", "price-list", "data", "salesperson", "transaction", "date", "sold devices";
- specialisation: "electrical device having serial number" –> "electrical device";
- entity relations – states: "sells", "is produced", "has" (x 3), "is recorded";
- entity relations – actions: "buying", "choosing", "requesting", "recording", "writing down";

• temporal relations: "during (buying devices) always (choosing devices from a price-list)", "during (buying devices) sometimes (requesting data of device)", "during (recording transaction) always (writing down customer's data, date and sold devices)".

Finally, the diagrams shown below will be generated.

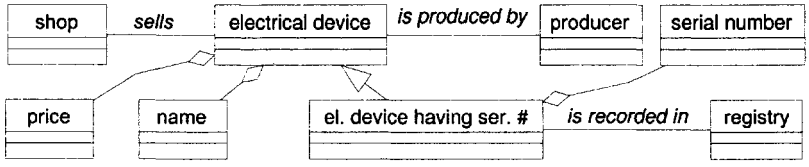


Fig. 3. A class diagram generated for the example text

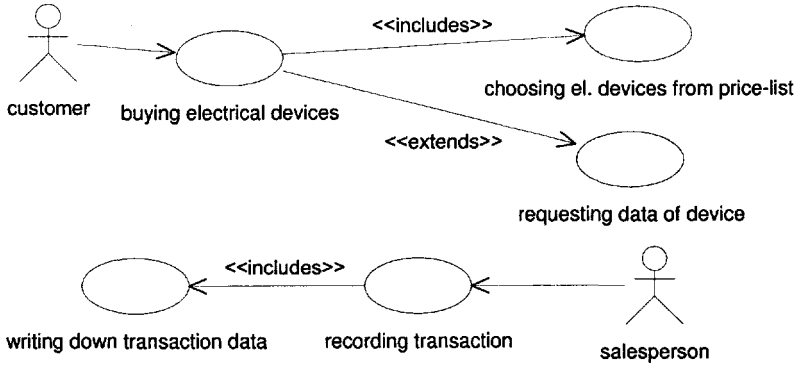


Fig. 4. A use-case diagram generated for the example text

6. Summary and conclusions

An outline of a system assisting the creation of a business model on the basis of a textual description was presented. The usage of natural language techniques allows to automatically identify possible errors and ambiguities in the text and generate UML class diagram and use-case diagram modelling a static and dynamic aspects of reality.

Benefits of using such system are the following. Firstly, a gradual analysis of a text and problem (errors and ambiguities) identification prevents oversights and misunderstandings. Automatic generation of diagrams gives a skeleton of business model and helps to learn the reality described in a text. The presented approach can also be applied to a more general problem of information extraction from a textual description.

7. References

[1] Booch G., Jacobson I., Rumbaugh J. *The Unified Modeling Language Reference Manual*, Addison-Wesley, 1999
[2] Booch G., Jacobson I., Rumbaugh J. *The Unified Software Development Process*, Addison-Wesly, 1999
[3] Huzar Z ., £abuzek M. *The Acquisition of System Specifications From Textual Descriptions*, Polish National Conference on Software Engineering, Zakopane, 2001
[4] Minsky M. *A Framework for Representing Knowledge* in: *The Psychology of Computer Vision* (P. H. Winston, ed.), Mc Craw-Hill Computer Science Series, 1975
[5] Pereira F. C. N., Shieber S. M. *Prolog and Natural-Language Analysis*, Stanford University, 1987
[6] Szpakowicz S. *Formal Syntactic Description of Polish Sentences*, Warsaw Univeristy Press, 1983
[7] XMI *XML Metadata Interchange (XMI)*, OMG Proposal, dokument ad/98-10-05

Information Modelling and Knowledge Bases XIV
H. Jaakkola et al. (Eds.)
IOS Press, 2003

Intensional Logic as a Medium of Knowledge Representation and Acquisition in the HIT Conceptual Model

Marie Duží
Technical University of Ostrava
tr. 17. listopadu 15
708 33 Ostrava
marie.duzi@vsb.cz

Pavel Materna
Masaryk University of Brno
Arne Nováka 1
602 00 Brno
materna@lorien.site.cas.cz

Czech Republic

Abstract. We present a powerful logical tool for the analysis of natural language expressions, namely Transparent Intensional Logic (TIL). Using TIL enables us to specify a knowledge base as well as the inference machine of an Information System on a high *conceptual* level of abstraction. After a brief introducing of TIL philosophy, principles and particular definitions, we present a general method of the analysis of interrogative sentences. It is shown that a particular interrogative sentence corresponds to some non-interrogative counterpart, and the two share a common *semantic core* that is (in case of empirical questions) an intension (a function from possible worlds and time points ...). To find the semantic core, we first determine the type of an answer that is the value of the denoted intension in a possible world at a time point. The whole method is applied to the HIT conceptual schema, which makes it possible to develop a *natural-language* based query language over a conceptual schema that is completely free of any implementation details.

1. Introduction

There are two research / scientific disciplines that have been originally developed separately but that have much in common, and actually in the recent days closely co-operate with each other. They are the Theory of Databases (or generally of Information Systems) and Artificial Intelligence, namely its branch called Knowledge-Based Systems. Every Knowledge / Information System consists of two basic components: Knowledge Base and an Inference Machine. Knowledge Base can be viewed as a set of statements recorded in a formal language. These statements express explicit as well as implicit information on a given part of reality. The Inference Machine enables us to obtain the (explicit / implicit) information recorded in the Knowledge Base. The whole system can be specified on various levels of abstraction, beginning from a very "low – computer like level" to a "high – conceptual level", the latter being, of course, desirable. Besides, when building up such a system, we have to strongly adhere to the principle of *the end-user involvement* in the analysis, specification and design of the system. Since the user is not bound to know any special formal apparatus of a particular system, we have to communicate with him/her in a natural language, and we have to be able to precisely, conceptually *analyse natural language expressions, idioms and jargon* used by the user.

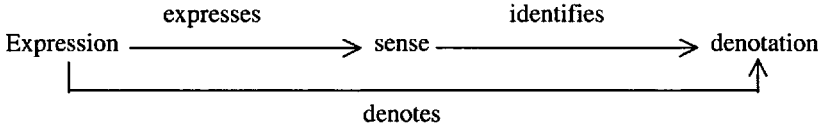
The result of such an analysis is a *conceptual schema* completed with the specification of an *inference machine* that both should replace the whole information system, see [7].

Our approach is a functional one, and the conceptual data model in use is the HIT data model that has been presented (among others) in [2], [3], where the theory and practice of the HIT conceptual schema and the HIT method of database design have been presented. Nevertheless, in these contributions we rather neglected the second component of the system, namely its Inference Machine. We have to be able not only to specify and store particular information but also to obtain the information. And, moreover, we have to be able to obtain not only the explicit information, stored in the database, but also the implicit one. In other words we have to be able to analyse particular user queries (expressed in a natural language), specify them in a formal language, and using this specification to deduce all the logical consequences of the statements stored in the knowledge/database. Thus the logical analysis of interrogative sentences is a primary task when building the Inference Machine. This paper deals just with this problem, i.e., we present a logical tool for analysing and specifying user questions on the content of the Knowledge Base.

In our opinion, one of the best logical systems used to analyse natural language expressions is the Transparent Intensional Logic (TIL) that is the underlying logical apparatus of the HIT data model. Before presenting our results we have first to introduce the basic notions, philosophical motivations and definitions of TIL.

2. Transparent Intensional Logic

TIL ([14], [10]) can be characterised as a standard logic, employing just the standard logical operators, avoiding any non-standard ones (like, e.g., the IF 'independence indicator' [6] or backwards looking operators), and its conception of a natural language analysis stems from the classical Frege-Church schema:



However, Frege's conception has been extensional and contextualistic, unlike the TIL approach which is intensional and strongly anti-contextualistic. According to Frege, an expression denotes / refers to an object (we would say extension) in 'normal' contexts, whereas in opaque contexts (like embedded clauses of propositional attitudes, e.g. knowing, believing, etc.) it denotes its sense. Thus, e.g., a sentence denotes its truth-value, 'the president of USA' denotes George Bush, 'employee' denotes a class of individuals who are employed, and so like. But George Bush has not been (and will not be) always the president of USA (temporal variability), and it is not logically necessary that just this person has been elected, anybody else could be (modal variability). Since in TIL the semantic relation between an expression and what it expresses and denotes is rigid and *a priori*, we explicate the denotation as an *intension* – function from possible worlds and time points (see Definition 2). Thus a sentence does not denote a truth value but a *proposition* – function from possible worlds and time points to the set of truth values, 'the president of USA' does not denote George Bush but a role the individual has to play to be the president of USA, i.e. an *individual office* – function from possible worlds and time

points to the set of individuals, etc. Generally speaking, *empirical expressions denote intensions*. Another weakness of Frege's approach is the fact that he has never explicated the 'sense'. He just offers a (very inspiring) characteristic – the 'mode of presentation' of the given denotation. We do not, however, confine ourselves to denotational semantics, for it is too course-grained, but adhere to a procedural fine-grained approach. The denoted intension can be obtained, *constructed* in infinitely many ways. Hence using the TIL key notion of *construction*, we explicate *sense – meaning* of an expression as the *concept* [10] represented by the expression. Summarising briefly:

Sense - meaning of an expression is explicated as a closed construction (abstract structured procedure) that specifies the *concept* represented by the expression, and that constructs (identifies)

the denotation of the expression, which is either a first-order ("flat" set-theoretical) entity, i.e. intension or extension, or a higher-order entity (involving a construction).

Note: Here by 'first-order' object we mean an object that is not a construction (and does not involve a construction). Hence from this point of view all the objects standard predicate logics (even of a higher "order") work with (like individuals, properties, functions, relations, etc.) are of the first order.

Empirical (unlike mathematical) expressions denote *intensions*, i.e. functions from possible worlds and time points. In this case we also speak about the *reference* of the expression that is the value of the denoted intension in the actual world/time. But the expression does not "speak about" its reference. To find out this reference (and its knowledge) is a matter of an empirical investigation, which is out of the scope of the logical analysis of a natural language, for such an analysis is an *a priori* discipline.

To accomplish our brief exposition of TIL, we have to characterise and precisely define the key notion of *construction*. Constructions are abstract structured procedures and are defined in a way that is partly inspired by the typed λ-calculus. They are, however, no λ-terms because they are no expressions. When they are fixed by some artificial language then it only means that using this language we speak about the procedures: not about particular symbols of the language. Therefore constructions could correspond to what Frege might have in mind, since Frege's realistic semantics presupposed that senses are extra-linguistic objects. Constructions 'work' over the collection of *types* defined as sets of partial functions over a base.

What follows is a modified version of Russell's theory of types. The "full TIL" is based on the ramified theory of types because we need constructions to be 'normal objects' possessing their respective (higher order) types as well. In other words, constructions / concepts are not only *used* to construct objects of a lower type, but also *mentioned* [1]. For the sake of simplicity, we will use in this paper just the simple theory of types, for we would need to speak about (mention) concepts only in such complicated queries like 'Which is the most frequent concept used in the department of informatics?'. Extending the whole analysis also to such cases is, however, easy and straightforward. There is another adjustment of Tichý's definitions here. We introduce the tuple type (and constructions creating tuples and their projections). Though tuples might be also defined as functions, introducing tuples as special types makes the system easier to use, for we often need to formulate and analyse questions like 'which is the only permanent address of a person X?', where address is a tuple (state, town, street, ZIP code).

Definition 1 (simple types)
Let B be a base, i.e. a collection of mutually disjoined non-empty sets.
i) Every member of the base B is an *(elementary) type over B.*
ii) Let $\beta_1, \ldots, \beta_n$ be types over B. Then $(\beta_1,\ldots,\beta_n)$ is a *(tuple) type over B,* i.e. the Cartesian product $\beta_1 \times \ldots \times \beta_n$.
iii) Let $\alpha, \beta_1, \ldots, \beta_n$ be types over B. Then $(\alpha\ (\beta_1,\ldots,\beta_n))$ is a *(functional) type over B,* i.e. the collection of all partial functions from $\beta_1 \times \ldots \times \beta_n$ to α.
An object O of a type α is called α-object, which is denoted O / α.

The base used in the discipline Logical Analysis of a Natural Language is the so-called *epistemic base* {o, ι, τ, ω}, where o is the set of truth values {True, False}, ι is the universe of discourse and its members are individuals, τ is the set of time points (or real numbers playing also the role of their surrogates) and ω is the logical space and its members are possible worlds. The term 'possible world' has been sometimes connected with false connotations due to the metaphysical character of the term. It is not a collection of things, objects. It would be probably absurd to suppose that there are more 'worlds' in this sense. Possible world is defined as a chronology of maximal collections of logically possible (mutually non-contradictory) *facts.* Over the epistemic base we can define a very important distinction between two kinds of objects, namely *intensions* vs. *extensions.*

Definition 2 (intensions, extensions)
Let α be any type. Then members of the type $((\alpha\tau)\omega)$, abbreviated by $\alpha_{\tau\omega}$, are *intensions.* Members of other types (i.e. not being functions from possible worlds ...) are *extensions.*

Intensions (denoted by empirical expressions) are actually some criteria, 'questions' on the state-of-the-world, say, in what kind of worlds and times which object is the reference of the expression (satisfies the criterion). The most common intensions are propositions / $(o_{\tau\omega})$, properties of individuals / $(o\iota)_{\tau\omega}$, individual offices / $\iota_{\tau\omega}$, magnitudes / $\tau_{\tau\omega}$ and empirical functions – attributes / $(\alpha\beta)_{\tau\omega}$ for some types α, β. Thus, e.g., a sentence does not denote a truth-value / (o), but a proposition / $(o_{\tau\omega})$. 'The highest mountain' does not denote Mount Everest / (ι), but an individual office / $(\iota_{\tau\omega})$; 'student' does not denote a set of individuals / $(o\iota)$, but a property of individuals / $((o\iota)_{\tau\omega})$, etc.

Constructions are abstract structured procedures, ways ("itineraries") of creating objects. They are objectual extra-linguistic entities. The simplest constructions mediate straight contact with objects; they are variables and *trivialisation* (that constructs a given object without any change). More complex constructions correspond to well-known λ-operations: application of a function to arguments, and creating a function *via* λ-abstraction. The remaining two constructions construct tuples and their projections. A construction constructing an α-object will be called α-construction.

Definition 3 (constructions)

i) Atomic *constructions* are *variables*. For every type α there are countably infinitely many α-variables that are incomplete constructions v-constructing an object of the respective type α dependently on valuation v.

ii) If X is any object (even a construction), then ^0X is *a construction* called *trivialisation* constructing simply X without any change.

iii) If X is a construction that v-constructs a function F / $(\alpha(\beta_1,\ldots,\beta_n))$ and $X_1,\ldots,X_n$ v-construct entities $B_1/\beta_1,\ldots,B_n/\beta_n$, then $[X\ (X_1,\ldots,X_n)]$ is a *construction* called *composition* (or traditionally application). It v-constructs the value of F on the tuple $<B_1,\ldots,B_n>$, if the function F is defined on this tuple, otherwise it does not construct anything, the composition is v-improper. (The composition fails to v-construct anything also in case that some of X_i fail.)

iv) Let $x_1,\ldots,x_n$ be pairwise distinct β_1-,…,β_n-variables and X a construction v-constructing an object of a type α. Then $[\lambda x_1 \ldots x_n\ X]$ is a *construction* called *closure* (or λ-abstraction) which v-constructs the following function F / $(\alpha(\beta_1,\ldots,\beta_n))$: Let v' be a valuation identical with v up to assigning objects $B_1,\ldots,B_n$ to variables $x_1,\ldots,x_n$, respectively. Then if X is v' improper, the function F is undefined on $<B_1,\ldots,B_n>$. Otherwise the value of F on $<B_1,\ldots,B_n>$ is the object v'-constructed by X.

v) Let $X_1,\ldots,X_n$ be β_1-,…,β_n-constructions, respectively. Then $(X_1,\ldots,X_n)$ is a *construction* called *tuple* that v-constructs an object of type $(\beta_1,\ldots,\beta_n)$.

vi) Let X be a $(\beta_1,\ldots,\beta_n)$-construction, then $X_{(1)},\ldots,X_{(n)}$ are β_1-,…,β_n-*constructions* called *projections* (which v-construct particular members of the tuple v-constructed by X).

vii) Nothing is a construction unless it so follows from i) – vi).

Notes:

Here we confine ourselves to simple hierarchy of types, so we will not use the symbol for the trivialisation and instead of writing, e.g., ^{02} we will simple write 2 because a first-order object cannot be used to construct another (lower-order) object, it can only be constructed ("mentioned") and no confusion can arise.

Intensions – unless they are constructed by trivialisation – are constructed by a construction of the 'form' $\lambda w \lambda t\ X$, where w, t are variables ranging over ω, τ, respectively, and X constructs for each valuation concerning w, t at most one object of a given type. Where C is a construction constructing an intension, particular composition $[[Cw]t]$ will be abbreviated by C_{wt}.

To make our constructions easier to read, we will use infix notation for logical connectives, identities, less (greater) than ('<', '>') signs, etc.

Definition 4

(From the view point of TIL) a *logical analysis* of an (empirical) expression E consists in finding a construction / concept expressed by E which (v-)constructs the object denoted by the expression E.

Example of TIL analysis. We will analyse the sentence
The most famous composer is bald.

a) Type-theoretical analysis

Meaningful components: *the most famous* - MF, *composer* - C, *being bald* - B, *the most famous composer* - *MFC*.

Types of the denoted objects: MF / $(\iota\ (o\iota))_{\tau\omega}$, C / $(o\ \iota)_{\tau\omega}$, B / $(o\ \iota)_{\tau\omega}$, MFC / $\iota_{\tau\omega}$.
(B, C are obviously properties of individuals. MF is an intension that dependently on worlds/times associates a class of individuals with at most one individual - the most famous one; MFC is an individual office.)

b) Synthesis
We have to compose denotations of particular components of our sentence so that to construct the proposition denoted by the sentence.
The most famous composer denotes an individual office - the role an individual can play. The sentence claims that the holder of this office in a given world/time (if any) belongs to the class of individuals that have (in that world/time) the property of being bald. To construct the office we have to combine (constructions of) MF and C. Omitting the sign for trivialisation, we have: $\lambda w\lambda t\ [MF_{wt}\ C_{wt}]$.
Applying this office to *w, t,* we obtain its holder in a given world/time:
$[\lambda w\lambda t\ [MF_{wt}\ C_{wt}]]_{wt}$
The resulting construction is:

$$\lambda w\lambda t\ [B_{wt}\ [\lambda w\lambda t\ [MF_{wt}\ C_{wt}]]_{wt}\ ]$$

or equivalently after the β-reduction:

$$\lambda w\lambda t\ [B_{wt}\ [MF_{wt}\ C_{wt}]]$$

Note that the proposition denoted by our sentence (constructed by this construction) may have no truth-value in a given world/time W/T, for the office of the most famous composer may be vacant in W/T (for instance if there were two or more equally famous composers, or if there were no composers). If our construction constructed a total proposition, true or false also in that W/T, it would imply [11] the existence of the most famous composer, which would break *the principle of the correct analysis that makes it possible to infer* ***all and only*** *the adequate consequences of our statements*, see [4]. Hence the composition $[MF_{wt}\ C_{wt}]$ may be v-improper, so that the office constructed by $[\lambda w\lambda t\ [MF_{wt}\ C_{wt}]]$ is a *partial function*, and keeping the principle of compositionality, the composition $[\lambda w\lambda t\ [MF_{wt}\ C_{wt}]]_{wt}$, as well as the whole composition
$[B_{wt}\ [\lambda w\lambda t\ [MF_{wt}\ C_{wt}]]_{wt}\ ]$ may be *v*-improper.

c) Type-theoretical checking

$\lambda w\lambda t$ $[B_{wt}$ $[\lambda w\lambda t$ $[MF_{wt}$ $C_{wt}]]_{wt}\ ]$
$(\iota\ (o\iota))$ $(o\iota)$
ι
$\iota_{\tau\omega}$
$(o\iota)$ ι
o
$o_{\tau\omega}$

We can see that the proposed analysis is correct: it constructs an $o_{\tau\omega}$-object, the proposition denoted by our sentence, and all the reasonable (sub)expressions of the sentence are taken into account, analysed as well, so that the denoted proposition is constructed by composing particular "sub-denotations".

3. Analysing interrogative sentences

Compare a pair of sentences, the first being indicative, the second interrogative (see [12]):

(1) Charles walks.
(2) Does Charles walk?

Admittedly, the sentence (1) that is normally being asserted differs syntactically from the corresponding question (2). But the syntactic difference reflects no difference in the *logic* of the two sentences; (1) and (2) denote one and the same *topic* of concern, namely the proposition that Charles walks. The sentences differ in expressing different *concerns* of the speaker (indicative and interrogative). The only function of the question mark and of the subject/auxiliary inversion in (2) is to reveal speaker's interrogative concern. Since *logic deals with topics, not concerns* [12], these syntactical devices evince no logical difference between (1) and (2). The two sentences not only denote the same proposition; they indicate it through the same logical construction: both express it as the result of predicating walkerhood of Charles. The declarative/interrogative distinction between (1) and (2) is thus not one of logic; (1) and (2) are logically indistinguishable, their difference lies entirely in the pragmatic attitude of the speaker.

Now imagine that somebody, say Tom, is looking for the President of USA. In doing so, he is related to the American presidency – the office, not to an individual. Tom can have other attitudes to the office of the President of USA. He may wish to become the President of USA, he may also wonder who in fact occupies the office, and he can ask who the President of USA is. All these attitudes on Tom's part have a common object – *the office* of the President of USA. To indicate the object of his search (e.g. when answering the question 'who are you looking for') Tom will use a form
(3) the President of USA.
To ask the corresponding question, Tom will use the interrogative form
(4) Who is the President of USA?
But the prefix "who is" and the question mark have no *logical* significance. Their function is to reveal Tom's interrogative attitude to the office. Hence (3) and (4) again denote the same entity, this time the office, and have the same 'logical form' – express the same *logical construction.*

The common entity of an interrogative sentence and its non-interrogative counterpart – the respective intension – will be further called their *semantic core.*

3.1. Kinds of Questions

The most usual classification of questions goes as follows:
A. Yes-No-questions
B. Wh-questions (whose important subclasses are, e.g., Why-questions, When-questions, Where-questions).

The common, unifying feature of *all* kinds of questions:
We want to know the present value of an intension in our world.

Note: This holds, of course, only for empirical questions. In case of analytical questions (denoting extensions) we want to know just the object constructed by particular construction, the analysis of the question. Here we are, however, interested primarily in empirical questions (on the 'state of the world', state of a knowledge base, etc.).

In the case of A. this value is a truth-value (verification of a sentence!), in the case of B. the value is an object of any kind (dependently on the type of the intension - a class of individuals (in case of properties), a definite individual (offices), a time interval, a proposition, etc.).

3.2. Semantic Core of Interrogative Sentences

Every interrogative sentence corresponds to some non-interrogative counterpart whose analysis constructs the *semantic core of interrogative sentence (SCIS).* This is always an intension; the question consists in an *attitude* to this intension: we want to know its value in the actual world now (in Information Systems the latter is mimicked by the actual state of a database).

Ad A.: The case of the Yes-No-interrogative sentences is simple: Here the SCIS of an interrogative sentence Q is the proposition constructed by the analysis of the non-interrogative counterpart Q' of Q. Its value in the actual world ('now') is therefore a truth-value, which is confirmed by the fact that the questions of this kind are to be answered by the name of a truth-value, i.e., by 'Yes' or 'No'.

Q: Are some employees widowers?
Q': Some employees are widowers.

Hence we have to construct a proposition:

$$\lambda w\, \lambda t\, \exists x\, ([\mathrm{Emp}_{wt}\, x] \wedge [\mathrm{Wid}_{wt}\, x]).$$

Yes-No-questions are the only kind where the non-interrogative counterpart is a proposition.

Ad B.: The much more varied class of the Wh-interrogative sentences contains distinct kinds of sentences; the SCIS of a Wh-interrogative sentence may be an individual office (role), a property of individuals, a magnitude, a propositional role etc. Here the type of SCIS can be relatively easily determined by the type of the object denoted by a *possible answer.* (See [8].) Where this latter type is α, the type of SCIS is $\alpha_{\tau\omega}$, of course.

Question: Which employees are widowers?
Answer: {John, Charles, Peter}

The answer denotes a class of individuals, so the type is $(o\iota)$, the type of SCIS is $(o\iota)_{\tau\omega}$. Thus we have to construct the property *(being) an employee and a widower.* Using infix notation we get

$$\lambda w\, \lambda t\, \lambda x\, ([\mathrm{Emp}_{wt}\, x] \wedge [\mathrm{Wid}_{wt}\, x]).$$

Question: Who is the oldest employee?
Answer: David

The answer denotes an individual, so the type is ι, the type of SCIS is $\iota_{\tau\omega}$. Thus we have to construct the individual role *the oldest employee.* If $O/(\iota\,(o\iota))_{\tau\omega}$, we get

$$\lambda w\, \lambda t\, [\mathrm{O}_{wt}\ \mathrm{Emp}_{wt}].$$

Question: How many employees are widowers?
Answer: Three

The answer denotes a number, so the type is τ, the type of SCIS is $\tau_{\tau\omega}$. Thus we have to construct the magnitude *the number of employees who are widowers.* If N/ (τ ($o\iota$)) ("the number of..."), we get

$$\lambda w\, \lambda t\, [\mathrm{N}\, \lambda x\, ([\mathrm{Emp}_{wt}x] \wedge [\mathrm{Wid}_{wt}x])\,].$$

Question: Is Charles a worker, or a programmer?
Answer: Charles is a worker.

This is an example of 'alternative questions'. Their analysis is a little more difficult. Our answer denotes a proposition (the question is not a Yes-No-question!), i.e., the type of what the answer denotes is $o_{\tau\omega}$. Thus we have to construct a propositional role, type $(o_{\tau\omega})_{\tau\omega}$. How could we describe this role? To satisfy this role a proposition has to be true and be exactly one of the propositions Charles is a worker and Charles is a programmer. Now there is a function denoted by the 'descriptive operator' ι , type-theoretically polymorph, whose type is – for any type α – (α ($o\alpha$)) ("the only α-object such that..."). Here we need to choose 'the only proposition such that...', so the type of ι will be $o_{\tau\omega}(o\, o_{\tau\omega})$). The construction is:

$$\lambda w\lambda t\, [\iota\, \lambda p\, [\, p_{wt} \wedge ([\, p = \lambda w\lambda t\, [\mathrm{Wor}_{wt}\ \mathrm{Ch}]] \vee [p = \lambda w\lambda t\, [\mathrm{Prog}_{wt}\ \mathrm{Ch}]])\,]].$$

There are two cases when the construction is improper when applied to W, T: if Charles is neither a worker nor a programmer or if Charles is both: a worker as well as a programmer. In both these cases there is no single proposition that would be true and be one of the two proposed propositions. Since ι is defined so that it returns a value only on singletons, it would give no value in both the mentioned cases.
We can always use pairs of Yes-No questions instead of alternative questions.

Question: When was Pavel born?
Answer: 21. 4. 1930

The answer denotes a date. We have following options. Either is date simply a number (type τ), or a tuple (type (τ,τ,τ)). In the first case the relevant types are: Born/ $(\tau\ \iota)_{\tau\omega}$, Date / τ, in the second case we have: Born/ $((\tau,\tau,\tau)\ \iota)_{\tau\omega}$, Date / (τ,τ,τ). The type of SCIS is therefore either $\tau_{\tau\omega,}$ or $(\tau,\tau,\tau)_{\tau\omega}$. In both cases we have

$$\lambda w\lambda t\, [\mathrm{Born}_{wt}\, \mathrm{P}].$$

Question: Why are more men than women employed?
Answer: (Because) women are more often absent.

This case is similar to the case of alternative questions. Here also the answer denotes a proposition, so that the type of SCIS is the type of propositional role, i.e., $(o_{\tau\omega})_{\tau\omega}$. We can define a relation, called neutrally *R(eason (of))*; it is a pseudo-causal, world dependent function that associates any event, phenomenon etc. with its reason, type $(o_{\tau\omega}\, o_{\tau\omega})_{\tau\omega}$. So we have

$$\lambda w\lambda t\, [\mathrm{R}_{wt}\, \lambda w\lambda t\, ([\mathrm{N}\, \lambda x\, ([\mathrm{M}_{wt}x] \wedge [\mathrm{Emp}_{wt}x])\,] > [\mathrm{N}\, \lambda x\, ([\mathrm{W}_{wt}x] \wedge [\mathrm{Emp}_{wt}x])\,]\,)].$$

(As an empirical function, R is not the (logical) consequence relation; its 'law-like' character has to be derivable from some constraints involved in the given information system.)

We can analyse also "recursive questions" on the transitive closure:

Question: Which are the ancestors of a given person?

Answer: John → {Alice, Bernard, Anna, Quido, ...}
Charles → {Irena, Paul, Ivana, Joseph, ... }
Peter → {Jane, Richard, Julia, William, ... }
.
.
.

The answer denotes a function that associates every individual (from the database) with the class of his/her ancestors. The type is ((οι)ι). Thus we have to construct the SCIS whose type is $((o\iota)\iota)_{\tau\omega}$. This time we use the (οι)-variable, say c, and the 'singularizer' ι of type ((οι) (ο (οι))) (selects the only class of individuals that satisfies the given conditions). Let us further assume that we can work with the relation P(arent), type $(o\iota\iota)_{\tau\omega}$. Then we have

$$\lambda w \lambda t\, \lambda x[\iota\, \lambda c\, [\forall y\, [[\mathrm{P}_{wt}(y, x)] \supset [cy]] \wedge \forall uv\, [[[cv] \wedge [\mathrm{P}_{wt}(u,v)]] \supset [cu]]]].$$

Such a construction is much more easily readable than it could seem at first sight. Informally we read: the function from worlds (w) to chronologies (t) of functions that associate every individual (x) with exactly one class ($\iota\, \lambda c$) such that it contains the parents of x (the first member of the conjunction) and, besides, every parent of every individual that is already in the class (the second member of the conjunction – *the recursive clause*). Answering means to find such a function in the conditions ('state of affairs') given by the actual information about the world and time (database).

4. Querying over HIT conceptual schema

Having explicated the way we analyse interrogative sentences using TIL, we can now illustrate the way queries are specified over HIT conceptual schema. (The idea of applying a unique functional approach to the data specification, manipulation and querying by means of the formal apparatus of the typed λ-calculus has been first formulated and published in [15].) The *HIT schema* is defined [3] as the couple $<A_S, C_A>$, where A_S is a set of constructions (concepts) of attributes specified over a base of sorts and C_A is a set of (constructions of) consistency constraints connected with A_S. The *base of sorts* is used here instead of the epistemic base to make to schema easier to understand by the user. It is the collection {E, D, τ, ω}, where E is a collection of *entity sorts* ('abstract types'), D is a collection of *descriptive sorts* ('printable types'), τ, ω as above. An entity sort E is given solely by a property (of objects of a type α, not necessarily of individuals, an $(o\alpha)_{\tau\omega}$-object) and is, therefore, not representable ('printable') [9]. A descriptive sort D is a 'normal' recursive set. The whole analysis could be, of course, performed over the epistemic base, but the "sortalisation" of the universe of discourse is in a good accordance with modern trends of an object-oriented approach, and it is very useful in practice. *Attributes* are empirical functional dependences between sorts. Attributes of a conceptual schema are functions of the so-called simple types:

a) $(\alpha\, \beta)_{\tau\omega}$ singular attributes
b) $((o\alpha)\beta)_{\tau\omega}$ multivalued attributes

where α, β are sorts or tuples of sorts.

When specifying queries over the HIT schema, we can combine (constructions of) particular attributes of the schema and (constructions of) particular logical / mathematical functions. To answer such a query actually means to find the present value of the constructed intension in the actual world. Since the actual world/time is mirrored by a database state of the system, answering a database query consists in "looking at the given database state", and possibly making use of the inference rules of the logic, TIL in our case. (For the latter see, e.g., [13].)

The simplest queries are the queries on the value of a particular attribute A at an argument B. If the attribute is the singular one, $A/(\alpha\beta)_{\tau\omega}$, the type of the answer is α, and the type of the constructed intension is $\alpha_{\tau\omega}$. Thus the respective query is (x ranges over α, B/β):

$$\lambda w \lambda t\, \iota x\, (x = [A_{wt}\, B]), \text{ or shortly } \lambda w \lambda t\, [A_{wt}\, B]$$

(The former construction is preferable, for we explicitly specify that we want 'the only x, such that ...' .)

If A is a multivalued attribute, $A/((o\alpha)\beta)_{\tau\omega}$, the type of the answer is $(o\alpha)$ and the intension constructed by the query is of the type $(o\alpha)_{\tau\omega}$:

$$\lambda w \lambda t\, \lambda x\, [[A_{wt}\, B]x], \text{ or shortly } \lambda w \lambda t\, [A_{wt}\, B]$$

(The former is again preferable, for we explicitly state that we wish to obtain a *set* - λx of values of the attribute A.)

Generally, a typical query over a HIT conceptual schema consisting of (concepts of) attributes $A_1,\ldots,A_n$ is a construction of the 'form'

$$\lambda w \lambda t\, [F\, ([A_{1wt}\, B_1],\ldots,[A_{n\,wt}\, B_n])],$$

where F is a (construction of) a mathematical / logical function, $B_1,\ldots,B_n$ are members of the respective types of arguments. And even more generally, the type of the answer may be again a function, so that the intension constructed by a query is again a (definable – redundant) attribute [3]. In this case an analysis of the query will get the 'form'

$$\lambda w \lambda t\, \lambda\, x_1 \ldots x_n\, \iota y\, (y = [F\, ([A_{1wt}\, x_1],\ldots,[A_{n\,wt}\, x_n])])$$

Example:

Concluding this section we adduce an outline of a (very simple) example of the HIT conceptual schema, and formulate some queries over the schema.

Entity sorts: #STUDENT, #LECTURER, #PERSON, #ROOM, #COURSE, #DEPARTMENT, #GRANT, #SEMESTER, ... (in the 'real' schema all the entity sorts have to be defined, i.e. particular defining properties have to be expressed).

ISA relationships:
#STUDENT ISA #PERSON
#LECTURER ISA #PERSON

Descriptive sorts:
DAY-IN-WEEK = {Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday},
TEACHING-HOUR = {(7.00,7.45), (8.00,8.45), ..., (20.00,20.45)},
GRADE = {1,2,3,4},
KIND = {lecture, seminar, laboratory, ...},

POSITION = {assistant, assistant-professor, docent, professor, ...},
DEGREE = {PhD, RNDr., PhDr, CSc., DrSc., ...},
ROOM-ID = {A-1011, A-1012, E-401, E-402, D-117, ...},
COURSE-ID = {ML, LANL, AI, ...}
...

Attributes

a) Descriptive attributes:

D1: (POSITION) of a given (#LECTURER)
D2: (DEGREE)-s of a given (#LECTURER)
D3: (SALARY/τ) of a given (#LECTURER)
D4: (NAME) of a given (#PERSON)
... (personal identification, address, ...)
D5: (ROOM-ID) of a given (#ROOM)
D6: (CAPACITY/τ) of a given (#ROOM)
D7: (COURSE-ID) of a given (#COURSE)
D8: (NUMBER/τ) of credits assigned to a (#COURSE)
D9: (ANNOTATION) of a given (#COURSE)
...

b) Relationship attributes:

R1: (#LECTURER)-s who are employed in a given (#DEPARTMENT)
R2: (#COURSE)-s enrolled by a given (#STUDENT) in a given (#SEMESTER)
R3: (GRADE) that has been obtained by a (#STUDENT) passing out a (#COURSE)
R4: (#GRANT)-s that have been solved by a given (#PERSON)
R5: (TIME-SCHEDULE)-s = (DAY-IN-WEEK, TEACHING-HOUR) when a (#COURSE) is lectured by a (#LECTURER) in a (#ROOM) as a (KIND) of lecturing
...

To make our schema intelligible for users, graphical support is provided (see [3]).

Queries:

Q1: Is Mr. Eastow a professor?
(Is there in our knowledge base a person named Eastow who is a professor?)
Q1': $\lambda w \lambda t\, \exists p\, ([D4_{wt}\, p] = \text{'Eastow'}) \wedge ([D1_{wt}\, p] = \text{'professor'})$
(Variable p ranges over #LECTURER, using inheritance between subtypes-supertypes makes it possible to apply D4 to p.)
Note that when answering such a question the existential quantifier $\exists$ corresponds to the search of a person whose identity is given by name. If D4 (name) were an id-attribute of #PERSON, the query could be specified in a more natural shortened way:
Q1'': $\lambda w \lambda t\, ([D1_{wt}\, \text{'Eastow'}] = \text{'professor'})$

Q2: How many professors are there in particular departments?
Q2': $\lambda w \lambda t\, \lambda d\, \iota n\, (n = [\text{Card}\ \lambda p\, ([[R1_{wt}\, d]\, p] \wedge [D1_{wt}\, p] = \text{'professor'})]$
(Variable d ranges over #DEPARTMENT, p over #LECTURER, Card is the function 'cardinality of a set')

Q2' can be read as follows: Dependently on the database state ($\lambda w \lambda t$) assign to each department (λd) the number (ιn) that is equal to the cardinality of the set of lecturers who are employed in the department ([[$R1_{wt}$ d] p]) and whose position is a professor ([$D1_{wt}$ p] = 'professor'). Note that this is an example of a query that is a derivable (redundant) attribute.

Q3: How many professors are there in the Department of informatics?
Assuming that 'Name of a department' is the id-attribute of #DEPARTMENT, we can formulate the shortened analysis:
Q3': $\lambda w \lambda t\ \iota n\ (n = [\text{Card}\ \lambda p\ ([[R1_{wt}\ \text{'Informatics'}]\ p] \wedge [D1_{wt}\ p] = \text{'professor'})]$

Q4: Is the room D-117 free on Mondays from 12.00 to 12.45? (Yes-no question)
Q4': $\lambda w \lambda t\ \neg\exists\ (c,l,k,s)\ (\ [[R5_{wt}\ (c,l,\text{'D-117'},k)]\ s\ ] \wedge$
$(s_{(1)} = \text{'Monday'}) \wedge (s_{(2)} = (12.00,12.45))\ )$
Variables: c ranges over #COURSE, l over #LECTURER, k over KIND, s over the couple TIME-SCHEDULE; $s_{(1)}$ returns day in week, $s_{(2)}$ the teaching hour.

Q5: When is the room D-117 occupied by teaching? (Wh-question, the answer being of the type o(DAY-IN-WEEK,TEACHING-HOUR)) – a set of 'time-schedules')
Q5': $\lambda w \lambda t\ \lambda s\ \exists\ (c,l,k)\ [[R5_{wt}\ (c,l,\text{'D-117'},k)]\ s\ ]$

Q6: Which rooms are free on Mondays from 12.00 to 12.45? (Wh-question)
Q6': $\lambda w \lambda t\ \lambda r\ \neg\exists\ (c,l,k,s)\ (\ [[R5_{wt}\ (c,l,r,k)]\ s\ ] \wedge (s_{(1)} = \text{'Monday'}) \wedge (s_{(2)} = (12.00,12.45))\ )$
Variables c,l,k,s *as* above, r ranging over #ROOM

5. Conclusion

In general, the advantage of HIT-like approach as compared with relational approaches can be characterised as follows:

Using HIT data model, we design a conceptual schema as well as the inference machine that both accurately and completely define business rules and demands in a way that our users can understand. This conceptual layer is free of any implementation details. Using HIT approach we communicate with the user in a *natural language*, still the results of the conceptual analysis are precisely recorded in the form of TIL logical constructions. On the other hand, relational approaches are always somewhat implementation-biased, using less powerful logical notation, they often use even an obscure language that most people don't understand.

HIT is a representative of *functional* approach. While there are some theoretical assets of functional approaches (in particular, they are friendly to *procedural* semantics of natural languages), we can state in the context of creating question-answering systems that the functional approach makes it easy to analyse very complicated questions; in terms of functional systems every query can be conceived of as offering a function – intension whose value we want to get. In particular, the values of such a function are dependent on the state of the world that is represented by the actual data in the database. This is implicitly recognised also in the other systems: the queries are, of course, fed by actual data – otherwise no answer can be obtained. In HIT-like systems the functional approach is continued – the value of any intension that represents the question may be *again a*

function (cf. HIT attributes). Hence using the only modelling construct of HIT model, i.e. HIT attribute, an empirical functional dependency, makes it possible to *conceptually* specify the schema as well as 'data manipulations' and 'data querying' over the schema by a unique functional 'language of TIL constructions'. The functional approach is in good harmony with the frequently occurring functional form in the natural language. Our examples show that such natural language queries – even much more complex – are easily analysed as TIL logical constructions, i.e. *concepts* expressed by the queries.

One should be aware that the main asset of such a conceptual approach consists just in the precise *specification* that is free of any 'implementation ballast'. Such a specification proved to be very useful in practice: It serves as a basic setting for a programmer, as a thorough *documentation* of the system, and last but not least, it is a common ground of a user-analyst-designer-programmer communication. Generally, HIT method of database design is based on transformation of the HIT conceptual schema (including data manipulation and queries) via the central level (Chen's like C-schema) into a relational schema. However, one may wonder whether a *direct implementation* of such a system has been realized. The answer is positive. Though still not on a commercial level, there are research projects that accomplish the automated analysis of natural language statements and queries based on TIL; we quote at least one of the most successful attempts (realized in C++), which is the PhD thesis [5]. Another such a system is just being realized in Prolog as a student research project.

References

[1] M. Duží, P.Materna: Non-reasonable Sentences. In: *Logica'94, FILOSOFIA,* Academy of Sciences of the Czech Republic, 1994, pp. 107-124.

[2] M.Duží, J. Pokorný: Semantics of General data Structures. *Information Modelling and Knowledge Bases* IX, IOS Press 1998, pp. 115-130.

[3] M. Duží: Logical Foundations of Conceptual Modelling using HIT Data Model. *Information Modelling and Knowledge Bases* XII, IOS Press 2000, pp. 65-80.

[4] M. Duží, P.Materna: Parmenides Principle (The Analysis of Aboutness). To appear in *Philosophia*, Israel, 2002.

[5] A. Horák: *The Normal Translation Algorithm in Transparent Intensional Logic for Czech.* PhD Thesis, Masaryk University of Brno, 2001.

[6] J.Hintikka: The Interrogative Model of Knowledge Acquisition as a Framework for Concept identification. *Information Modelling and Knowledge Bases* III, IOS Press, 1992, pp. 174-181.

[7] H.Kangassalo: Conceptual description for Information Modelling Based on Intensional Containment Relation. University of Tampere, 1998.

[8] P.Materna, E.Hajicová, P.Sgall: Disambiguation of Interrogative Sentences through Logical Analysis of Answers. In: *Prague Studies in Mathematical Linguistics* 9, 1985.

[9] P.Materna: Entity Sorts. What are they? *Computers and Artificial Intelligence,* 6,4,1987, pp. 321-324.

[10] P. Materna: *Concepts and Objects.* Acta Philosophica Fennica, Helsinki, 1998.

[11] P.F. Strawson: On Referring. *Mind* 59, 1950, pp.320-344.

[12] P.Tichý: Questions, answers and logic. *American Philosophical Quaterly,* Vol.15, N.4, 1978, pp. 275-284.

[13] P.Tichý: Indiscernibility of Identicals. *Studia Logica* 45 (1986), pp. 257-273.

[14] P.Tichý: *The Foundations of Frege's Logic.* De Gruyter, 1988.

[15] J.Zlatuška: Data bases and the Lambda Calculus. In: *Proc. IFIP'86 World Computer Congress,* Dublin, 1986, pp. 97-104.

Information Modelling and Knowledge Bases XIV
H. Jaakkola et al. (Eds.)
IOS Press, 2003

Logic of Relevant Connectives for Knowledge Base Reasoning

Noriaki YOSHIURA
yoshiura@lab.cc.gunma-u.ac.jp

Computer Center, Gunma University
1-5-1 Tenjincho Kiryu City Gunma 376-8515, Japan

Abstract. The formalization of human knowledge base reasoning is a main issue in artificial intelligence. Although logic is one of most useful ways of such formalization, inference of logic does not correspond to that of knowledge base reasoning. The reason is that the aim of logical inference is to deal with the truth values of propositions while that of human reasoning is to infer useful information. As far as logical connectives are concerned, whether there is some relation between two propositions, we can connect them by logical connectives such as conjunction and so on in classical logic. However, in daily speech or in knowledge base reasoning, when we introduce logical connectives into propositions, we give consideration to relation between the propositions. In this paper, we discuss relevance of logical connectives on formalization of knowledge base reasoning and propose logic LRC (Logic of Relevant Connectives). LRC is based on relevant logic ER from which fallacies of relevance and validity are removed. ER has two merits. One is that it is one of the stronger relevant logical systems, and the other is that ER is decidable. These characteristics are the reason why LRC is based on ER.

1 Introduction

Formalization of knowledge reasoning is one of the main issues of philosophy and artificial intelligence. Logic is a useful method for this formalization, however, classical logic is not sufficient because the meaning of implication in classical logic is different from that in daily speech [3]. For example, for an arbitrary formula A, $A \rightarrow B$ can be inferred from B in classical logic, even if there is no relation between A and B. However, from the viewpoint of daily speech, some relation is necessary for the inference of $A \rightarrow B$. We give an example of such an inference by daily speech; in classical logic, "that snow is black implies $3+5=8$" can be inferred from "$3+5=8$", however, it seems strange. The cause of this strange inference is the meaning of implication in classical logic and this strangeness of implication is called "fallacy". Relevant logic has been researched as a logic from which fallacies of implication are removed. As a method of formalization of knowledge reasoning, relevance logic is more suitable than classical logic in the sense that the fallacies of implication are removed[1].

Relevant logic ER is one of relevant logics[4, 5]. Since relevant logic ER is free from fallacies of implication, it is suitable for formalizing knowledge reasoning. However, there are still several differences between inferences of relevant logic ER and knowledge

base reasoning. One of the important differences is how to introduce logical connectives into propositions. In ER, when we introduce conjunction, disjunction or negation into propositions, we do not consider relation between these propositions. On the other hand, in knowledge base reasoning, when we introduce logical connectives into propositions, we consider relation between the propositions.

For example, in relevant logic ER, $A \wedge C \rightarrow B \wedge D$ can be inferred from $A \rightarrow B$ and $C \rightarrow D$. However, there is no guarantee that the conjunctive connection between A and C is relevant. We can introduce any logical connective into any two propositions even if these two propositions have some relations. Consider this example by using concrete propositions as follows:

$$\text{it rains} \rightarrow \text{picnic is canceled}$$

$$\text{Ken is human} \rightarrow \text{Ken will die at some future.}$$

From these propositions, we can infer the following proposition.

$$(\text{it rains} \wedge \text{Ken is human}) \rightarrow (\text{picnic is canceled} \wedge \text{Ken will die at some future})$$

However, from the viewpoint of daily speech, there is no relation between "it rains" and "Ken is human". This inference does not seem to obtain useful information.

On the other hand, $A \rightarrow B \wedge C$ can be inferred from $A \rightarrow B$ and $A \rightarrow C$. In this case, we can think that relevance of conjunctive connection of B and C is guaranteed by the same proposition A, which is a premise of each implication formula. We can introduce the concept of relevance of disjunction by using "common propositions".

As above examples show, relevant logic ER does not have the concept of relevance of all logical connectives. This paper discusses relevance of logical connectives for formalization of knowledge base reasoning. This paper also proposes logic LRC (Logic of Relevant Connectives). LRC is based on relevant logic ER. Since LRC has relevance concept of all logical connectives, LRC is more suitable than ER for formalization of knowledge reasoning. We explain the logic LRC by showing provable and non provable formulas. Table 1 shows the provability of LRC In this table, $\Gamma \vdash A$ means that A can be inferred from Γ. We denote Γ by "premise" and A by "conclusion".

Example1.	$A \vdash (A \rightarrow B) \rightarrow B$
Example2.	$A \rightarrow B, C \rightarrow D \vdash A \wedge C \rightarrow B \wedge D$
Example3.	$A \vdash A \vee B$
Example4.	$A, A \rightarrow B \vdash B$
Example5.	$A \rightarrow B, A \rightarrow C \vdash A \rightarrow B \wedge C$
Example6.	$A \vee B, A \rightarrow C, B \rightarrow D \vdash C \vee D$
Example7.	$(A \rightarrow B) \rightarrow C, A \vdash (A \rightarrow B) \rightarrow B$

Table 1: Provability of LRC

In Example 1, since relevance of implication connection of $A \rightarrow B$ is not guaranteed, $(A \rightarrow B) \rightarrow B$ is not a theorem in LRC. In Example 2, an example of conjunction, relevance of $A \wedge C$ which is included in the conclusion is not guaranteed. Thus, $A \wedge C \rightarrow B \wedge D$ is not a theorem in LRC. In example 3, an example of disjunction, relevance of $A \vee B$ which occurs in the conclusion is not guaranteed. Thus, $A \vee B$ is not a theorem in LRC. In example 4, since the conclusion B does not include new connection of logical connectives, B can be inferred from A and $A \rightarrow B$. In example 5, $B \wedge C$ occurring in the

$$A:e \vdash A:e \qquad \neg A:r \vdash \neg A:r \quad \text{Axiom}$$

$$\frac{\Gamma, \neg A:r \vdash}{\Gamma \vdash A:e}\ RAA \qquad \frac{\Gamma \vdash A:\varphi_1 \quad \Delta \vdash B:\varphi_2}{\Gamma,\Delta \vdash A \wedge B:i}\ \wedge I$$

$$\frac{\Gamma \vdash A \wedge B:e}{\Gamma \vdash A:e}\ \wedge E1 \qquad \frac{\Gamma \vdash A \wedge B:e}{\Gamma \vdash B:e}\ \wedge E2 \qquad \frac{\Gamma \vdash A:\varphi}{\Gamma \vdash A \vee B:i}\ \vee I1 \qquad \frac{\Gamma \vdash B:\varphi}{\Gamma \vdash A \vee B:i}\ \vee I2$$

$$\frac{\Gamma \vdash A \vee B:e \quad \Delta_1, A:e \vdash C:e \quad \Delta_2, B:e \vdash C:e}{\Gamma,\Delta_1,\Delta_2 \vdash C:e}\ \vee E1$$

$$\frac{\Gamma \vdash A \vee B:e \quad \Delta_1, A:e \vdash C:i \quad \Delta_2, B:e \vdash C:\varphi}{\Gamma,\Delta_1,\Delta_2 \vdash C:i}\ \vee E2$$

$$\frac{\Gamma \vdash A \vee B:e \quad \Delta_1, A:e \vdash C:\varphi \quad \Delta_2, B:e \vdash C:i}{\Gamma,\Delta_1,\Delta_2 \vdash C:i}\ \vee E3$$

$$\frac{\Gamma \vdash A \vee B:e \quad \Delta_1, A:e \vdash \quad \Delta_2, B:e \vdash}{\Gamma,\Delta_1,\Delta_2 \vdash}\ \vee E4$$

$$\frac{\Gamma, A:e \vdash B:\varphi}{\Gamma \vdash A \to B:i}\ \to I \qquad \frac{\Gamma \vdash A \to B:e \quad \Delta \vdash A:\varphi}{\Gamma,\Delta \vdash B:e}\ \to E$$

$$\frac{\Gamma \vdash A:\varphi \quad \Delta \vdash \neg A:e}{\Gamma,\Delta \vdash}\ \neg E1 \qquad \frac{\Gamma \vdash A:e \quad \neg A:r \vdash \neg A:r}{\Gamma, \neg A:r \vdash}\ \neg E2$$

$$\frac{\Gamma, A:e \vdash}{\Gamma \vdash \neg A:i}\ \neg I \qquad \frac{\Gamma, A:\varphi, A:\varphi \vdash}{\Gamma, A:\varphi \vdash}\ C1 \qquad \frac{\Gamma, A:\varphi, A:\varphi \vdash B:\phi}{\Gamma, A:\varphi \vdash B:\phi}\ C2 \qquad \frac{\Gamma, \neg A:e, \neg A:r \vdash}{\Gamma, \neg A:r \vdash}\ C3$$

$$\frac{\Gamma, \neg A:e \vdash B:\varphi}{\Gamma \vdash A \vee B:i}\ EM1 \qquad \frac{\Gamma, \neg B:e \vdash A:\varphi}{\Gamma \vdash A \vee B:i}\ EM2$$

$$\frac{\Gamma \vdash A \vee B:e \quad \Delta, A:e \vdash}{\Gamma,\Delta \vdash B:e}\ DS1 \qquad \frac{\Gamma \vdash A \vee B:e \quad \Delta, B:e \vdash}{\Gamma,\Delta \vdash A:e}\ DS2$$

Figure 1: Inference rules of ER

conclusion is relevant connection of conjunction because A is the common proposition of B and C and because A guarantee the relation between B and C. In example 6, relevance of $C \vee D$ in the conclusion is guaranteed by $A \vee B$ in the premise. In example 7, since $A \to B$ which occurs in the conclusion also occurs in the premise of inference, the connection of $A \to B$ is relevant.

This paper is organized as follows. Section 2 gives ER. Section 3 discusses relevance of logical connectives. Section 4 defines LRC (logic of relevant connectives). Section 5 proves some properties of LRC. Section 6 concludes this paper and tells some future works.

2 Relevant logic ER

This section presents relevant logic ER and its properties. [4, 5]

2.1 Definitions of relevant logic ER

This section presents the relevant logic ER. As showed in Fig. 1, ER is defined as a sequent style natural deduction system[4]. The different point from usual natural deduction is that a formula has an attribute value. ER is a kind of labeled deduction system[2]. Attribute values show how a formula is inferred and which kind of rule can be applied to the formula. By using attribute values, we restrict the applicability of

inference rules in order to remove the fallacies of implication from ER. In ER, there are two restrictions; one is that formulas which are inferred by I-rule [1] can not be used as major premise[2] of E-rule [3]. The other is that some kinds of formulas can not be discharged by *Reductio ad absurdum rule* (RAA). These two restrictions are realized in ER by attaching attribute value to each formula in proof and by using this attribute value for deciding applicability of inference rules, Because of these two restrictions, formulas including fallacies do not become theorems of ER.

In the following, we present definitions of formulas, attribute value, proof and theorem of ER. After that, We explain properties of ER.

Definition 1 (Formulas of ER) *Atomic propositions are formulas of ER. If A and B are formulas of ER, then* $\neg A$, $A \wedge B$, $A \wedge B$, $A \vee B$ *and* $A \rightarrow B$ *are formulas of ER.*

Definition 2 (Attribute values of formula) *"e", "i" and "r" are defined to be attribute values of formula. "e" indicates that a formula with such an attribute value can be major premise of elimination rules. "i" indicates that a formula with such an attribute value can not be major premise of elimination rules. "r" indicates that a formula with such an attribute value can be discharged by the rule RAA.*

In the following, φ, $\varphi_1, \cdots$, ϕ,$\phi_1, \cdots$ are used as meta variables of attribute value. $A : \varphi$ is called *attribute formula*, where A is a formula and φ is an attribute value.

Definition 3 $\Gamma \vdash \mathcal{A}$ *is called sequent, where* Γ *is a multiset of attribute formulas and* $\mathcal{A}$ *is an attribute formula or an empty formula. A proof of ER is a finite sequence of sequent* $F_1, \cdots, F_n$, *where* F_i $(1 \leq i \leq n)$ *is an axiom or a conclusion which one of the rules in Fig. 1 infers by using some of* $F_1, \cdots, F_{i-1}$ *as premise.*

Definition 4 (Theorem of ER) *A is a theorem of ER if and only if there exists a proof* $F_1, \cdots, F_n$ *of ER and* F_n *is of the form* $\vdash A : \varphi$. *We say that A is inferred from a sets* Γ *of formulas if and only if* $\Gamma \vdash A : \varphi$ *is proved.*

2.2 Properties of ER

ER has the following properties.

1. Removal of fallacy of relevance[4]

   Suppose that A is an atomic proposition and $B_1, B_2, \cdots$ and B_n $(n \geq 1)$ are atomic propositions different from A. $A \rightarrow (B_1 \rightarrow (B_2 \rightarrow \cdots (B_n \rightarrow A) \cdots)$ is not a theorem in ER.

2. Removal of fallacy of validity[4]

   Suppose that A and B are different atomic propositions. $A \rightarrow (B \vee \neg B)$ and $(A \wedge \neg A) \rightarrow B$ are not theorems in ER.

[1] I-rule is an inference rule which introduces logical connectives. In ER, $\wedge I, \vee I1, \vee I2, \rightarrow I, \neg I, EM1$ and $EM2$ are I-rule.

[2] Major premise is a premise $A \vee B$ of $\vee E1, \vee E2, \vee E3$ or $\vee E4$, a premise of $\wedge E1$ or $\wedge E2$, a premise $A \rightarrow B$ of $\rightarrow E$ or a premise $\neg A$ of $\neg E1$ or $\neg E2$.

[3] E-rule is an inference rule which eliminate logical connectives. In ER, $\vee E1, \vee E2, \vee E3, \vee E4$, $\wedge E1, \wedge E2, \rightarrow E, \neg E1$ and $\neg E2$ are E-rule.

3. Variable-sharing[4]

   If $A \rightarrow B$ is a theorem, A and B contain an atomic proposition in common.

4. Disjunctive Syllogism[4]

   In ER, Disjunctive Syllogism(DS) holds. In almost all relevant logics, this rule does not hold.

$$\frac{A \vee B \quad \neg A}{B} \; DS$$

5. Provability[5]

   ER is stronger than relevant logic R, which is the typical relevant logic in which fallacies of relevance and validity are removed.

6. Decidability[4]

   ER is decidable, while almost all relevant logics are undecidable.

Since variable-sharing is a necessary property of relevant logic and since fallacies of implication are removed from ER as much as from R, ER is more suitable for formalization of knowledge base reasoning than R.

3 Relevance of logical connectives

This section defines relevance of each logical connective from the viewpoint of knowledge base reasoning. In knowledge base reasoning, all logical connectives which occur in formulas in knowledge base are supposed to be relevant. For example, if $A \rightarrow B \wedge C$ is in knowledge base [4], then, it is guaranteed that $B \wedge C$, conjunction of B and C, is relevant connection because of existence of $A \rightarrow B \wedge C$ in knowledge base. We define this concept of connectives relevance as follows:

Definition 5 (Relevance based on knowledge base) *All logical connective connections which occur in knowledge base are relevant.*

Several inference rules introduce new connections of logical connective and these connections do not occur in knowledge base. Relevance of these connections of logical connectives must be defined. We give this definition which is based on proof of ER, that is to say, how to deduce a formula in ER decides relevance of connectives occurring in the formula.

3.1 Relevance connections of conjunction and disjunction

This subsection discusses relevance connections of conjunction and disjunction.

[4]The connectives of highest binding power are "$\neg$", followed by "$\wedge$", followed by "$\vee$", followed finally by "$\rightarrow$".

3.1.1 Partial order of information quantity

In classical logic or relevant logic, $A \wedge (A \wedge B)$ is semantically equal to $A \wedge B$ and $A \vee (A \wedge B)$ is semantically equal to A. This means that inferring $A \wedge (A \wedge B)$ and from A and $A \wedge B$ and inferring $A \vee (A \wedge B)$ from A or $A \wedge B$ obtain only complex formulas equivalent to A or $A \wedge B$. It follows that these connections of disjunction and conjunction are not relevant. We introduce partial order of information quantity in order to define these non-relevant connections formally.

Definition 6 (Partial order of information quantity) *Partial order of information quantity $\geq_m$ is partial order of formulas. This order consists of reflexive law, transitive law, antisymmetric law and the following axioms.*

1. $\neg(A \vee B) =_m \neg A \wedge \neg B$
2. $A \wedge B \geq_m A, \quad A \geq_m A \vee B$
3. $A \wedge B =_m B \wedge A, \quad A \vee B =_m B \vee A$
4. $A \wedge (B \vee C) \geq_m (A \wedge B) \vee (A \wedge C)$
5. *If* $A \geq_m B$ *and* $C \geq_m B$, *then* $A \vee C \geq_m B$
6. *If* $A \geq_m B$ *and* $A \geq_m C$, *then* $A \geq_m B \wedge C$
7. *If* $A \geq_m B$, *then* $\neg B \geq_m \neg A$

This partial order is lattice in which "$\wedge$" corresponds to "meet", "$\vee$" corresponds to "join" and "$\neg$" corresponds to "complement". Distributive law and De Morgan law holds in this partial order. In relevant logic, implication can not be abbreviated by "$\vee$" and "$\neg$", that is to say, $A \rightarrow B$ is not equivalent to $\neg A \vee B$. $A \rightarrow B$ does not refer to the truth value of A or B in relevant logic. This property of implication in relevant logic is reflected in the definition of the partial order "$\geq_m$" and there is no order between $A \rightarrow B$ and A or B. $A \rightarrow B$ is considered to be one atomic proposition in the partial order "$\geq_m$".

By using partial order of information quantity, we define a necessary condition of relevance connection of conjunction and disjunction; $A \not\geq_m B$ and $B \not\geq_m A$ are necessary so that $A \wedge B$ which is inferred from A and B is relevant connection of conjunction. In the following, we denote "$A \not\geq_m B$ and $B \not\geq_m A$" by $A \asymp B$ and we use this condition in order to define the relevance connection of conjunction and disjunction.

3.1.2 Conjunction

We discuss relevance connection of conjunction, which does not occur in knowledge base.

$A \wedge B$ is inferred from A and B by using $\wedge I$ in ER, some relation between A and B are necessary so that $A \wedge B$ is relevance connection of conjunction. If the same premise are used in proves of A and B, this same premise is supposed to guarantee some relation between A and B. In brief, in the case that $\Gamma, C : e \vdash A : \varphi_1$ and $\Delta, C : e \vdash B : \varphi_2$ are proved, the same premise $C : e$ is supposed to guarantee the existence of some relation between A and B. It follows that $A \wedge B$ which occurs in $\Gamma, \Delta, C : e \vdash A \wedge B : i$ is relevant connection of conjunction.

On the other hand, Suppose that $\Gamma, A : e, B : e \vdash C : e$ is proved. In this case, A and B are used for inferring C and this fact is supposed to guarantee the relation

between A and B. Therefore, $A \wedge B$ which occurs in $\Gamma, A \wedge B : e \vdash C : e$ is relevant connection of conjunction.

In the above, we discuss the two cases of connection of conjunction and we propose the conditions for relevance connection of conjunction. In addition to these conditions, we have to consider information quantities of A and B; if $A \leq_m B$ or $B \leq_m A$, $A \wedge B$ has the same meaning as A or B. In this case, inference $A \wedge B$ from A and B does not obtain new information and this connection of conjunction does not seem relevant.

The above discussion leads the following definition of relevance connection of conjunction.

Definition 7 (Relevance connection of conjunction) *Relevant connection of conjunction is defined as follows.*

1. *If* $\Gamma, A : e, B : e \vdash C : \varphi$ *is provable and* $A \asymp B$, *then* $A \wedge B$ *which occurs in* $\Gamma, A \wedge B : e \vdash C : \varphi$ *is relevant connection of conjunction.*
2. *If* $\Gamma_1 \vdash A : \varphi_1$ *and* $\Gamma_2 \vdash B : \varphi_2$ *is provable,* $\Gamma_1 \cap \Gamma_2 \neq \emptyset$ *and* $A \asymp B$, *then* $A \wedge B$ *which occurs in* $\Gamma_1, \Gamma_2 \vdash A \wedge B : i$ *is relevant connection of conjunction.*

3.1.3 Disjunction

We discuss relevance connection of conjunction, which does not occur in knowledge base.

Like conjunction, $A \vee B$ $A \vee B$ is inferred from A and B by using some inference rules, some relation between A and B are necessary so that $A \vee B$ is relevant connection of disjunction. For example, in the case that $\Gamma, C : e \vdash A : \varphi_1$, $\Delta, D : e \vdash B : \varphi$ and $\Pi \vdash C \vee D : e$ are provable, we can prove $\Gamma, \Delta, \Pi \vdash A \vee B : \varphi_2$ by using some inference rules and $A \vee B$ which occurs in this sequent seems relevant connection of disjunction because $C \vee D$ is supposed to guarantee relevance of $A \vee B$.

On the other hand, in the case that $\Gamma, A : e \vdash C : \varphi_1$ and $\Delta, B : e \vdash C : \varphi_2$ are provable, we can prove $\Gamma, \Delta, A \vee B : e \vdash C : \varphi_3$ by using $\vee E1$ and $A \vee B$ which occurs in this sequent seems relevant connection of disjunction because the common formula C which occurs in $\Gamma, A : e \vdash C : \varphi_1$ and in $\Delta, B : e \vdash C : \varphi_2$ is supposed to guarantee relevance of $A \vee B$.

The above discussion leads the definition of relevance connection of disjunction as follows:

Definition 8 (Relevance connection of disjunction) *Relevant connection of disjunction is defined as follows.*

1. *If* $\Gamma, A : e \vdash C : \varphi_1$ *and* $\Delta, B : e \vdash C : \varphi_2$ *are provable and* $A \asymp B$, *then* $A \vee B$ *which occurs in* $\Gamma, \Delta, A \vee B : e \vdash C : \varphi_3$ *is relevant connection of disjunction.*
2. *If* $\Pi \vdash A \vee B : e$, $\Gamma, A : e \vdash C : \varphi_1$ *and* $\Delta, B : e \vdash D : \varphi_2$ *are provable and* $C \asymp D$, *then* $C \vee D$ *which occurs in* $\Gamma, \Delta, \Pi \vdash C \vee D : i$ *is relevant connection of disjunction.*
3. *If* $\Gamma, A : e \vdash B : \varphi_1$ *and* $\Delta, \neg A : e \vdash C : \varphi_2$ *are provable and* $B \asymp C$, *then* $B \vee C$ *which occurs in* $\Gamma, \Delta \vdash B \vee C : i$ *is relevant connection of disjunction.*

3.2 Negation

We discuss relevance connection of disjunction, which does not occur in knowledge base.

In ER, negation formula $\neg A$ is inferred by the inference rule $\neg I$ and we can infer several sequents from $A_1 : e, \cdots, A_n : e \vdash$ in many ways of applying $\neg I$ as follows:

$$\frac{A_1 : e, \cdots, A_n : e \vdash}{A_2 : e, \cdots, A_n : e \vdash \neg A_1 : i}\ \neg I$$

$$\vdots$$

$$\frac{A_1 : e, \cdots, A_n : e \vdash}{A_1 : e, \cdots, A_{n-1} : e \vdash \neg A_n : i}\ \neg I$$

We can obtain negation formulas as many as the number of formulas which occur in the left side of the sequent $A_1 : e, \cdots, A_n : e \vdash$.

Recall the meaning of $A_1 : e, \cdots, A_n : e \vdash$. This sequent means that $A_1, \cdots, A_n$ do not hold in the same time. Thus, that the negation formula occurring in the following proof seems more relevant connection than negation of each formula of $A_1 \cdots A_n$.

$$\begin{array}{c} A_1 : e, \cdots, A_n : e \vdash \\ \vdots \\ \hline \vdash \neg(A_1 \wedge \cdots \wedge A_n) \end{array}\ \neg I$$

From the above discussion, we define relevance connection of negation as follows:

Definition 9 (Relevance connection of negation) *If $A : e \vdash$ is provable, then $\neg A$ which occurs in $\vdash \neg A : i$ is relevant connection of negation.*

3.3 Implication

We discuss relevance connection of disjunction, which does not occur in knowledge base.

In ER, implication is introduced by the inference rule $\rightarrow I$.

$$\frac{\Gamma, A : e \vdash B : \varphi}{\Gamma \vdash A \rightarrow B : i}\ \rightarrow I$$

On the other hand, Since there are many ways of applying $\rightarrow I$ to sequent $A_1 : e, \cdots, A_n : e \vdash B : \varphi$, we obtain many sequents as follows:

$$\frac{A_1 : e, \cdots, A_n : e \vdash B : \varphi}{A_2 : e, \cdots, A_n : e \vdash A_1 \rightarrow B : i}\ \rightarrow I$$

$$\vdots$$

$$\frac{A_1 : e, \cdots, A_n : e \vdash B : \varphi}{A_1 : e, \cdots, A_{n-1} : e \vdash A_n \rightarrow B : i}\ \rightarrow I$$

By applying $\rightarrow I$ to these conclusions repeatedly, we can obtain many sequents such as $\vdash A_1 \rightarrow (A_2 \cdots B) \cdots) : i$. However, we think that $A_1 : e, \cdots, A_n : e \vdash B : \varphi$ means that A_1, A_2, $\cdots$ and A_n are used equally for inferring B. Therefore, we think that $A_1 \wedge \cdots A_n \rightarrow B$ is relevance connection of implication as follows:

$$\begin{array}{c} A_1 : e, \cdots, A_n : e \vdash B : \varphi \\ \vdots \\ \hline \vdash A_1 \wedge \cdots A_n \rightarrow B : i \end{array}\ \rightarrow I$$

In knowledge base reasoning, it is not necessary to infer trivial propositions as result. Thus, if relation between premise and conclusion of implication formula is too trivial, then such an implication formula is not relevance connection of implication. For example, $A \rightarrow A$ is too trivial and it is relevance connection of implication.

$$\dfrac{\dfrac{\dfrac{}{A:e \vdash A:e} \quad \dfrac{\dfrac{\vdash B \rightarrow C:e}{A:e \vdash B:e} \quad \dfrac{\vdash A \rightarrow B:e \quad A:e \vdash A:e}{}}{A:e \vdash C:e}}{A:e, A:e \vdash A \wedge C:i}}{\dfrac{A:e \vdash A \wedge C:i}{\vdash A \rightarrow A \wedge C:i}}$$

Figure 2:

Figure 2 is a proof of $A \rightarrow A \wedge C$. This formula is an implication formula and its premise is A and its conclusion is $A \wedge C$. Since the formula A exists in the premise of implication the formula A in the conclusion of implication hold trivially if the premise of this implication formula is true. Therefore, we do not think that such an implication formula as $A \rightarrow A \wedge C$ is relevant connection of implication. To define these implication formula is not relevant, we introduce pseudo tautology as follows.

Definition 10 (pseudo tautology) *Pseudo tautology is defined inductively as follows:*

1. $A \rightarrow A$ *is pseudo tautology.*
2. *If* $A \rightarrow B$ *is pseudo tautology, then* $A \wedge C \rightarrow B$, $(C \wedge A) \rightarrow B$, $(A \vee C) \rightarrow B$ *and* $(C \vee A) \rightarrow B$ *are pseudo tautologies.*
3. *If* $A \rightarrow B$ *is pseudo tautology, then* $A \rightarrow B \wedge C$, $A \rightarrow (B \vee C)$, $A \rightarrow C \wedge B$ *and* $A \rightarrow (C \vee B)$ *are pseudo tautologies*

Now, we present important properties of pseudo tautology.

Proposition 1 (Decidability of pseudo tautology) *Whether a formula is pseudo tautology is decidable.*

By using pseudo tautology, we can introduce a necessary condition for relevance of implication. This condition is as follows; if $A \rightarrow B$ is relevant connection, then it is not pseudo tautology.

We can define relevance of implication connection from above discussion.

Definition 11 (Relevance of implication connection) *If* $A : e \vdash B : \varphi$ *is provable in ER and* $A \rightarrow B$ *is not pseudo tautology, then* $A \rightarrow B$ *is relevant connection of implication in* $\vdash A \rightarrow B : i$.

4 Logic *LRC*

This section defines logic *LRC*, which satisfies the definition of relevance of connectives

As above described, *LRC* is based on *ER*. To construct *LRC* satisfying condition of relevance connections, we introduce some admissible inference rules and impose some restrictions.

$$\frac{\Gamma, A : e \vdash C : e \quad \Delta, B : e \vdash C : e}{\Gamma, \Delta, A \vee B : e \vdash C : e} \vee L1 \qquad \frac{\Gamma, A : e \vdash C : i \quad \Delta, B : e \vdash C : e}{\Gamma, \Delta, A \vee B : e \vdash C : i} \vee L2$$

$$\frac{\Gamma, A : e \vdash C : e \quad \Delta, B : e \vdash C : i}{\Gamma, \Delta, A \vee B : e \vdash C : i} \vee L3 \qquad \frac{\Gamma, A : e \vdash \quad \Delta, B : e \vdash}{\Gamma, \Delta, A \vee B : e \vdash} \vee L4$$

Figure 3:

$$\frac{\Gamma \vdash B \vee C : e \quad \dfrac{\Delta_1 B : e \vdash D : \varphi}{\Delta_1, B : e \vdash D \vee E : i} \vee I1 \quad \dfrac{\Delta_2, C : e \vdash E : \varphi_2}{\Delta_2, C : e \vdash D \vee E : i} \vee I2}{\Gamma, \Delta_1, \Delta_2 \vdash D \vee E : i} \vee E3$$

Figure 4:

4.1 Addition of inference rules

By using only inference rules of ER, we can not obtain the logic which has the concepts of relevance connection of conjunction and disjunction. For example, If $\Gamma, A : e, B : e \vdash C : \varphi$ is provable in ER, then $A \wedge B$ which occurs in $\Gamma, A \wedge B : e \vdash C : \varphi$ are relevant connection of conjunction. Therefore, we need inference rule which infers $\Gamma, A \wedge B : e \vdash C : \varphi$ from $\Gamma, A : e, B : e \vdash C : \varphi$. However, such an inference rule does not exists in ER, even if $\Gamma, A \wedge B : e \vdash C : \varphi$ is provable in the case that $\Gamma, A : e, B : e \vdash C : \varphi$ is provable. To construct LRC, we add some admissible inference rules.

4.1.1 Inference rules for conjunction

To reflect the first condition of relevance connection of conjunction on LRC, we add some inference rules into ER to construct LRC. In brief, we need inference rule which infers $\Gamma, A \wedge B : e \vdash C : \varphi$ from $\Gamma, A : e, B : e \vdash C : \varphi$ because this rule does not exist in ER.

$$\frac{\Gamma, A : e, B : e \vdash C : e}{\Gamma, A \wedge B : e \vdash C : e} \wedge L1 \qquad \frac{\Gamma, A : e, B : e \vdash}{\Gamma, A \wedge B : e \vdash} \wedge L2$$

It is trivial that this inference rule is admissible in ER. $A \wedge B$ occurring in the conclusions of these inference rules is relevant connection of conjunction by the definition. Thus, by using these inference rules, we can reflect the concept of relevant connection of conjunction into LRC.

4.1.2 Inference rules for disjunction

To reflect the first condition of relevance connection of disjunction on LRC, we add some inference rules into ER to construct LRC like the case of conjunction. By the first item of the definition of relevant connection of disjunction, we need inference rule which infers $\Gamma, \Delta, A \vee B : e \vdash C : \varphi_3$ from $\Gamma, A : e \vdash C : \varphi_1$ and $\Delta, B : e \vdash C : \varphi_2$. because this inference rule does not hold in ER. Thus, we introduce the inference rule in Figure 3.

By the second item of the definition of relevant connection of disjunction, we need inference rule which infers $\Gamma, \Delta_1, \Delta_2 \vdash D \vee E : i$ from $\Gamma \vdash B \vee C : e$, $\Delta_1, B : e \vdash D : \varphi_1$ and $\Delta_2, C : e \vdash E : \varphi_2$. This inference rule holds in ER as shown in Figure 4. However, $C \vee E$ is inferred from C and from E indecently in the proof of Figure 4 and we have to

decide relevance of $C \vee E$ from the whole proof and not from the conclusion of inference rules. Therefore, we introduce new inference rules to decide relevance of $C \vee E$ only from the conclusion of inference rules not from the whole proof.

$$\frac{\Gamma \vdash B \vee C : e \quad \Delta_1, B : e \vdash D : \varphi_1 \quad \Delta_2, C : e \vdash E : \varphi_2}{\Gamma, \Delta_1, \Delta_2 \vdash D \vee E : i} \vee I3$$

4.2 Logic LRC

This subsection defines syntax and inference rules of *LRC*. With regard to formulas, attribute values and attribute formulas, we use the syntax of *ER* in *LRC*.

LRC is logic for knowledge base reasoning and theorems of *LRC* are decided by knowledge base $\mathcal{K}$; if $\mathcal{K}$ is an empty set, then there is no theorem of *LRC* based on $\mathcal{K}$ and if $\mathcal{K}$ is different, then theorems of *LRC* is different. Therefore, theorems of *LRC* is defined as "theorems based on knowledge base $\mathcal{K}$. In the following, we explain the formal definitions of *LRC*.

Definition 12 *In LRC, we say that* $\Gamma \vdash \mathcal{A}$ *is a sequent, where* Γ *is a multiset of attribute formulas and* $\mathcal{A}$ *is an attribute formula or an empty formula.*

In LRC, a proof based on knowledge base $\mathcal{K}$ *is a finite sequence* $F_1, \cdots F_n$ *of sequent, where* F_i $(1 \leq i \leq n)$ *is an axiom or a conclusion which one of the rules in Fig. 1 infers by using some of* $F_1, \cdots, F_{i-1}$ *as premise.*

Each inference rule must satisfy the following conditions. To describe the conditions, we use some notations; $A \sqsubseteq P$ means that A is a subformula of P and $A \sqsubset P$ means that A is a proper subformula of P.

1. Axiom

   $\mathcal{K}$ includes P such that $A \sqsubseteq P$.

2. $\wedge I$

   One of the following conditions must be satisfied.

   (1) $\mathcal{K}$ includes P such that $A \wedge B \sqsubseteq P$.

   (2) $\Gamma \cap \Delta \neq \emptyset$ and $A \asymp B$.

3. $\vee I1, \vee I2$

   $\mathcal{K}$ includes P such that $A \vee B \sqsubseteq P$.

4. $\rightarrow I$

   One of the following conditions must be satisfied.

   (1) $\mathcal{K}$ includes P such that $A \rightarrow B \sqsubseteq P$.

   (2) $A \rightarrow B$ is not pseudo tautology and Γ is empty.

5. $EM1, EM2$

   One of the following conditions must be satisfied.

   (1) $\mathcal{K}$ includes P such that $A \wedge B \sqsubseteq P$.

$$A:\varphi \vdash A:\varphi \quad \text{Axiom 1} \quad \vdash A:e \quad \text{Axiom 2} \quad \frac{\Gamma,\neg A:r\vdash}{\Gamma\vdash A:e}\ RAA$$

$$\frac{\Gamma\vdash A\wedge B:e}{\Gamma\vdash A:e}\ \wedge E1 \quad \frac{\Gamma\vdash A\wedge B:e}{\Gamma\vdash B:e}\ \wedge E2 \quad \frac{\Gamma\vdash A:\varphi_1 \quad \Delta\vdash B:\varphi_2}{\Gamma,\Delta\vdash A\wedge B:i}\ \wedge I$$

$$\frac{\Gamma\vdash A\vee B:e \quad \Delta_1,A:e\vdash C:e \quad \Delta_2:e\vdash C:e}{\Gamma,\Delta_1,\Delta_2\vdash C:e}\ \vee E1 \quad \frac{\Gamma\vdash A\vee B:e \quad \Delta_1,A:e\vdash C:i \quad \Delta_2:e\vdash C:\varphi}{\Gamma,\Delta_1,\Delta_2\vdash C:i}\ \vee E2$$

$$\frac{\Gamma\vdash A\vee B:e \quad \Delta_1,A:e\vdash C:\varphi \quad \Delta_2:e\vdash C:i}{\Gamma,\Delta_1,\Delta_2\vdash C:i}\ \vee E3 \quad \frac{\Gamma\vdash A\vee B:e \quad \Delta_1,A:e\vdash \quad \Delta_2:e\vdash}{\Gamma,\Delta_1,\Delta_2\vdash}\ \vee E4$$

$$\frac{\Gamma\vdash A:\varphi}{\Gamma\vdash A\vee B:i}\ \vee I1 \quad \frac{\Gamma\vdash B:\varphi}{\Gamma\vdash A\vee B:i}\ \vee I2 \quad \frac{\Gamma,A:e\vdash B:\varphi}{\Gamma\vdash A\to B:i}\ \to I \quad \frac{\Gamma\vdash A\to B:e \quad \Delta\vdash A:\varphi}{\Gamma,\Delta\vdash B:e}\ \to E$$

$$\frac{\Gamma\vdash A:\varphi \quad \Delta\vdash\neg A:e}{\Gamma,\Delta\vdash}\ \neg E1 \quad \frac{\Gamma\vdash A:e,\phi \quad \neg A:r\vdash\neg A:r}{\Gamma,\neg A:r\vdash}\ \neg E2 \quad \frac{\Gamma,A:e\vdash}{\Gamma\vdash\neg A:i}\ \neg I$$

$$\frac{\Gamma,\neg A:e\vdash B:\varphi}{\Gamma\vdash A\vee B:i}\ EM1 \quad \frac{\Gamma,\neg B:e\vdash A:\varphi}{\Gamma\vdash A\vee B:i}\ EM2 \quad \frac{\Gamma\vdash A\vee B:e \quad \Delta,A:e\vdash}{\Gamma,\Delta\vdash B:e}\ DS1 \quad \frac{\Gamma\vdash A\vee B:e \quad \Delta:e\vdash}{\Gamma,\Delta\vdash A:e}\ DS2$$

$$\frac{\Gamma,A:e,B:e\vdash C:\varphi}{\Gamma,A\wedge B:e\vdash C:\varphi}\ \wedge L1 \quad \frac{\Gamma,A:e,B:e\vdash}{\Gamma,A\wedge B:e\vdash}\ \wedge L2 \quad \frac{\Gamma\vdash A\vee B:e \quad \Delta_1,A:e\vdash C:\varphi_1 \quad \Delta_2:e\vdash D:\varphi_2}{\Gamma,\Delta_1,\Delta_2\vdash C\vee D:i}\ \vee I3$$

$$\frac{\Gamma,A:e\vdash C:e \quad \Delta,B:e\vdash C:e}{\Gamma,\Delta,A\vee B:e\vdash C:e}\ \vee L1 \quad \frac{\Gamma,A:e\vdash C:i \quad \Delta,B:e\vdash C:e}{\Gamma,\Delta,A\vee B:e\vdash C:i}\ \vee L2$$

$$\frac{\Gamma,A:e\vdash C:e \quad \Delta,B:e\vdash C:i}{\Gamma,\Delta,A\vee B:e\vdash C:i}\ \vee L3 \quad \frac{\Gamma,A:e\vdash \quad \Delta,B:e\vdash}{\Gamma,\Delta,A\vee B:e\vdash}\ \vee L4$$

$$\frac{\Gamma,A:\varphi,A:\varphi\vdash}{\Gamma,A:\varphi\vdash}\ C1 \quad \frac{\Gamma,A:\varphi_1,A:\varphi_1\vdash B:\varphi_2}{\Gamma,A:\varphi_1\vdash B:\varphi_2}\ C2 \quad \frac{\Gamma,\neg A:e,\neg A:r\vdash}{\Gamma,\neg A:r\vdash}\ C3$$

Figure 5: Inference rules of LRC

(2) $A \asymp B$.

6. $\wedge L1, \wedge L2$

   One of the following conditions must be satisfied.

   (1) $\mathcal{K}$ includes P such that $A \wedge B \sqsubseteq P$.

   (2) $A \to B$ is not pseudo tautology and the left side of sequent is empty.

7. Other inference rules.

   There is no restriction.

Definition 13 (Theorems of LRC) *If $\Gamma \vdash A : \varphi$ is proved and $\Gamma \subseteq \mathcal{K}$, then A is a theorem of LRC based on $\mathcal{K}$.*

5 Properties of LRC

This section proves the following properties of LRC.

1. Soundness

   All LRC theorems based on knowledge base $\mathcal{K}$ are theorems of ER and include only relevant logical connectives.

2. Completeness

   All theorems of ER which include only relevant logical connectives are theorems of LRC.

5.1 Soundness

This subsection proves the following theorem.

Theorem 1 *All LRC theorems based on knowledge base $\mathcal{K}$ are theorems of ER and all logical connectives occurring in theorems of LRC are relevant.*

To prove this theorem, we prove the following lemma.

Lemma 1 *Suppose that sequent $\Gamma \vdash A : \varphi$ is provable in LRC. $\Gamma \vdash A : \varphi$ is provable in ER and all logical connectives occurring in A and in Γ are relevant.*

Proof: We prove this lemma by induction of complexity of proof of LRC. Because of shortage of space, we describe only the cases of $\vee L1$ and $\rightarrow I$.

1. $\vee L1$

   Suppose that $\Gamma, \Delta, A \vee B : e \vdash C : e$ is proved by $\vee L1$ in LRC under the knowledge base $\mathcal{K}$. By the definition of $\vee L1$, $A \asymp B$. By induction hypothesis, $\Gamma, A : e \vdash C : e$ and $\Delta, B : e \vdash C : e$ are proved in ER and all connectives occurring in these sequents are relevant. In ER, $\Gamma, \Delta, A \vee B : e \vdash C : e$ is inferred as follows:

$$\frac{A \vee B : e \vdash A \vee B : e \quad \Gamma, A : e \vdash C : e \quad \Delta, B : e \vdash C : e}{\Gamma, \Delta, A \vee B : e \vdash C : e} \vee E1$$

   $A \vee B$ occurring in the conclusion of this proof is relevant because it satisfies the definition of relevance of disjunction.

2. $\rightarrow I$

   Suppose that $\Gamma \vdash A \rightarrow B : i$ is proved by $\rightarrow I$ in LRC under the knowledge base $\mathcal{K}$. By induction hypothesis, $\Gamma, A : e \vdash B : \varphi$ can be proved in ER and all connectives occurring in $\Gamma \cup \{A, B\}$ are relevant. Thus, $\Gamma \vdash A \rightarrow B : i$ can be inferred in ER. In the following, we prove that $A \rightarrow B$ is relevant connection of implication. By the definition of $\rightarrow I$, One of the following conditions must be satisfied.

   (1) $\mathcal{K}$ includes P such that $A \rightarrow B \sqsubseteq P$.

   (2) $A \rightarrow B$ is not pseudo tautology and Γ is empty.

   In each case, $A \rightarrow B$ occurring in $\Gamma \vdash A \rightarrow B : i$ is relevant. Thus, we have proved the claim in the case of $\rightarrow I$. ■

By lemma 1, we can prove the theorem 1 easily.

5.2 Completeness

This subsection prove the following theorem.

Theorem 2 *Suppose that A is inferred from some subset of knowledge base $\mathcal{K}$ in ER and that all connectives occurring in A are relevant. A is a theorem based on $\mathcal{K}$ in LRC.*

First we prove the following lemma.

Lemma 2 *If sequent $\Gamma \vdash A : \varphi$ is provable in ER and A does not include irrelevant connections with respect to knowledge base $\mathcal{K}$ then S is also provable in LRC*

Proof: We prove this lemma by induction of complexity of proof of ER. Because of shortage of space, we describe only the cases of $\wedge I$ and $EM1$.

1. $\wedge I$

   Suppose that $\Gamma \vdash A \wedge B : i$ is inferred by $\wedge I$ in ER and that $A \wedge B$ is relevant connection of conjunction. Since $A \wedge B$ is relevant connection of conjunction, the following hold

   - $A \asymp B$ or $\mathcal{K}$ includes P such that $A \wedge B \sqsubseteq P$.
   - $\Pi_1 \vdash A : \varphi_1$ and $\Pi_2 \vdash B : \varphi_2$ can be inferred in ER, where $\Pi_1 \cap \Pi_2 \neq \emptyset$ and $\Gamma = \Pi_1 \cup \Pi_2$ [5].

   Thus, $\Gamma \vdash A \wedge B : i$ can be proved in LRC as follows:

$$\frac{\Pi_1 \vdash A : \varphi_1 \quad \Pi_2 \vdash B : \varphi_2}{\Gamma \vdash A \wedge B : i} \wedge I$$

   Since this inference holds in LRC, we have proved the claim in the case of $\wedge I$.

2. $EM1$

   Suppose that $\Gamma \vdash A \vee B : i$ is inferred by $EM1$ in ER and that $A \vee B$ is relevant connection of disjunction. By the definition of relevant connection of disjunction, $A \asymp B$ or $\mathcal{K}$ includes P such that $A \vee B \sqsubseteq P$. By induction hypothesis, $\Gamma, \neg A : e \vdash B : \varphi$ is provable in LRC. Therefore, we can obtain the following proof of LRC and $A \vee B$ satisfies the restriction of $EM1$.

$$\frac{\Gamma, \neg A : e \vdash B : \varphi}{\Gamma \vdash A \wedge B : i} EM1$$

   we have proved the claim in the case of $EM1$. ∎

By lemma 2, we can prove theorem 2.

6 Conclusion

This paper proposed concept of relevant connection of logical connectives from the viewpoint of knowledge base reasoning and introduced logic LRC in which relevance concept is reflected. We also proved that soundness and completeness of LRC.

Future works are to research semantics of LRC and decidability of LRC and to construct a method of constructing set of theorems from knowledge base.

[5] Recall that Π_1, Π_2 and Γ are multiset and that $\cup$ is an union of multiset.

References

[1] Jingde Cheng:The Fundamental Role of Entailment in Knowledge Representation and Reasoning, Journal of Computing and Information, Vol.2, No.1, pp.853–873 (1996).

[2] Dov M.Gabbay, Labelled Deductive Systems, Vol 1, Oxford LABELLED Guides 33, Oxford U.P., (1996)

[3] Anderson and Belnap:Entailment. The logic of relevance and necessity, Princeton University Press (1975).

[4] Noriaki Yoshiura and Naoki Yonezaki, A Decision Procedure for the Relevant Logic *ER*, Tableaux 2000, published in University of St. Andrews Research Report CS/00/01, pp.109-123 (2000)

[5] Provability of Relevant Logic *ER*, Noriaki Yoshiura, Naoki Yonezaki, Proc. of The 11th European-Japanese Conference on Information Modelling and Knowledge Bases pp.90-104 (2001)

Information Modelling and Knowledge Bases XIV
H. Jaakkola et al. (Eds.)
IOS Press, 2003

A Model of Anonymous Covert Mailing System Using Steganographic Scheme

Eiji Kawaguchi*, Hideki Noda*, Michiharu Niimi* and Richard O. Eason**

*) Kyushu Institute of Technology
Kitakyushu, 804-8550 Japan

**) University of Maine
Orono, Maine 04469-5700 USA

(* kawaguch@know.comp.kyutech.ac.jp)

Abstract

In this paper we show a model of an anonymous covert mailing system for Internet communication. It is the system scheme of an on-going project to develop a real-life secure mailing system in our group. In this system a sender can send a secret message even to a non-acquainted person in an anonymous way. The users of this system are assumed to be members of a closed organization. But it is not primarily limited only to such users. It is a quite easy-to-use system with very cheap cost. This project as of now is at a prototype-developing stage.

1. Introduction

Human beings have long hoped to have a technique to communicate with a distant partner anonymously and covertly. We may be able to realize this hope by using steganography. Modern steganography has a relatively short history because people did not pay much attention to this skill until Internet security became a social concern. Most people did not know what steganography was because they did not have any means to know the meaning. Even today ordinary dictionaries do not contain the word "steganography." Books on steganography are still very few [1], [2].

In 1997 the authors invented a new steganographic method named "BPCS-Steganography." The most important feature of this steganography is that it has a very large data hiding capacity [3], [4]. It normally embeds 50% or more of a container image file with information without increasing its size . We made an experimental program (for Windows) and located it on a Web site for free downloading [5]. We also have several introductory Web pages to BPCS-Steganography and its applications [6].

Steganography can be applied to variety of information systems. Some key is used in these systems when it embeds/extracts secret data. One natural application is a secret mailing system [7] that uses a symmetric key. Another application pays attention to the nature of steganography whereby the external data (e.g., visible image data) and the internal data (any hidden information) cannot be separated by any means. We will term this nature as an "inseparability" of the two forms of data.

In the present paper we will show our basic model of an anonymous and covert e-mailing

system. The structure of the present paper is as follows. In Section 2 we will make a short discussion on the problems of an encrypted mailing system. Section 3 describes the scheme of the Anonymous Covert Mailing System. We will show how we can make it a safe system in Section 4. Finally, in Section 5, we show our future schedule to make it a real life system.

2. Problems of an encrypted mailing system

There are two types of cryptography systems: (1) symmetric key systems, and (2) asymmetric key systems.

In a symmetric system a message sender and receiver use a same encoding/decoding key. In this system, however, the sender and the receiver must negotiate on what key they are going to use before they start communication. Such a negotiation must be absolutely secret. They usually use some second channel (e.g., fax or phone). However, the second channels may not be very secure. There is another problem in this situation in that if the sender is not acquainted with the receiver, it is difficult to start the key-negotiation in secret. Further more, the more secure the key system is, the more inconvenient the system usage is.

An asymmetric system uses a public key and a private key system. The public key is open to the public, and it is used for message encoding when a sender is sending a message to the key owner. However, if the public key is counterfeited, this system does not work at all. So, the public key must be guaranteed as authentic. Therefore, this system needs a special authentication bureau, which is an organization that all the people in the world can trust in. In reality, it can only exist if it commercially pays. Therefore, this system is expensive and time consuming for users.

3. A model of an anonymous covert mailing system

The authors' research group at Kyushu Institute of Technology started to develop a secure and easy-to-use e-mailing system according to the BPCS-Steganography method. We do not intend to develop a new "message reader-and-sender" or "message composer", but we are developing three system components that make an Anonymous Covert Mailing System (ACMS). A message sender inserts (actually, embeds) a secret message in an envelope using steganography and sends it as an e-mail attachment. The receiver receives the attached envelope and opens it to receive the message. An "envelope" in this system is actually an image file that is a container, vessel, cover, or dummy data in the terminology of steganography. This system can solve all the problems mentioned above.

The following items are the conditions we have set forth in designing the system.

(1) The name of the message sender can be anonymous.

(2) The message is hidden in the envelope and only the designated receiver can open it.

(3) Sender can send a secret message even to an unacquainted person.

(4) It is easy to use for both sender and receiver.

(5) The system is inexpensive in installing and using.

We expect this ACMS is used in a closed organization (such as in a company), but there are no restrictions for any group (or the general public) to use it.

3.1 Components of the system

ACMS is a steganography application system (or program). It makes use of the inseparability of the external and internal data. The system can be implemented differently according to different programmers or different specifications. Different ACMS' are incompatible in operation with others. In other words, an organization using ACEM must use one single ACMS. However, each member in the organization uses it in a customized way. The customization is made when the user installs it on his/her own computer. In the following description, M_i denotes a member i, and M_j denotes a member j.

An ACMS consists of the three following components.

(1) Envelope Producer (EP) (2) Message Inserter (MI) (3) Envelope Opener (EO)

We denote M_i's ACMS as $ACMS_i$ (i.e., customized ACMS by M_i). So, it is described as $ACMS_i = (EP_i, MI_i, EO_i)$.

EP_i is a component that produces M_i's envelope (E_i). E_i is the envelope (actually, an image file) which is used by all other members in the organization when they send a secret message to M_i. E_i is produced from an original image (E_0). M_i can select it according to his preference. E_i has both the name and e-mail address of M_i on the envelope surface (actually, the name and address are "printed" on image E_i). Fig. 1 illustrates an envelope. It will be placed at an open site in the organization so that anyone can get it freely and use it any time. Or someone may ask M_i to send it directly to him/her.

MI_i is the component to insert (i.e., embed according to the steganographic scheme) M_i's message into another member's (e.g., M_j)'s envelope (E_j) when M_i is sending a secret message ($Mess_i$) to M_j. One important function of MI_i is that it detects a key (Key_j) that has been hidden in the envelope (E_j), and uses it when inserting a message ($Mess_i$) in E_j.

Fig. 1 An example of an envelope E_i

EO_i is a component that opens (extracts) E_i's "message-inserted" envelope $E_i(Mess_j)$ which M_i received from someone as an e-mail attachment. The sender (M_j) of the secret message ($Mess_j$) is not known until M_i opens the envelope by using EO_i.

3.2 Customization of an ACMS

Customization of an ACMS for member M_i takes place in the following way. M_i first decides a key (Key_i) when he installs the ACMS onto his computer. Then he types in his name

($Name_i$) and e-mail address ($EAdrs_i$). Key_i is secretly hidden (according to a steganographic method or some other method) in his envelope (E_i). This Key_i is eventually transferred to a message sender's MI_j in an invisible way. $Name_i$ and $EAdrs_i$ are printed out on the envelope surface when M_i produces E_i by using EP_i (cf. Fig. 1). Key_i is also set to EO_i at the time of installation. $Name_i$ and $EAdrs_i$ are also inserted (actually, embedded) automatically by MI_i any time M_i inserts his message ($Mess_i$) in another member's envelope (E_j). The embedded $Name_i$ and $EAdrs_i$ are extracted by a message receiver (M_j) by EO_j. Fig. 2 illustrates the scheme of this AC-Mailing system.

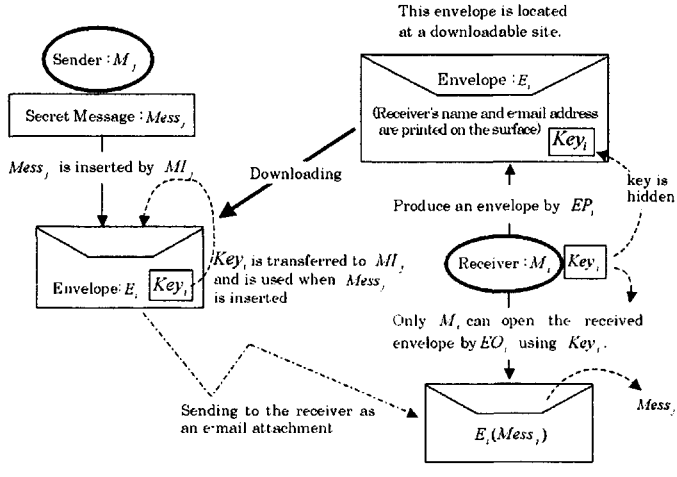


Fig. 2 The scheme of the AC-Mailing system.

3.3 How it works

When some member (M_j) wants to send a secret message ($Mess_j$) to another member (M_i), whether they are acquainted or not, M_j gets (e.g., downloads) M_i's envelope (E_i), and uses it to insert his message ($Mess_j$) by using MI_j. When M_j tries to insert a message, M_i's key (Key_i) is transferred to MI_j automatically in an invisible manner, and is actually used. M_i can send $E_i(Mess_j)$ directly, or ask someone else to send it to M_i as an e-mail attachment.

M_j can be anonymous because no sender's information is seen on $E_i(Mess_j)$. $Mess_j$ is hidden, and only M_i can see it by opening the envelope. It is not a problem for M_j and M_i to be acquainted or not because M_j can get anyone's envelope from an open site. ACMS is a very easy-to-use system because users are not bothered by any key handling, as the key is always operated automatically. As ACMS doesn't need any authorization bureau, this system

can be very low cost. All these features overcome the drawbacks of an encrypted mailing system (cf. Section 2).

4. Anti reverse-engineering strategy

This ACMS is secure only if the key is not stolen (i.e., not disclosed by anyone). The location of the hidden key in the envelope is kept secret by the system developer, but some people may be interested in reverse-engineering the execution programs (*EP* and *MI*). General techniques to make a program difficult to reverse-engineer include the following.

(1) To make the structure of the program a very tangled one, or a "spaghetti program."

(2) To make it "manually untraceable" by way of inserting very complicated subprograms.

(3) To set a lot of branches in the program-flow according to non-algorithmic conditions.

One practical method for (3) is to use "time intervals" between two (or more) instructions along a program-flow. In a normal running mode of the computer, the flow branches into a correct direction, but in a reverse-engineering mode the computer speed may be shifted down for tracing and the flow strays into incorrect directions. This confuses the reverse-engineers completely. It is difficult to make a theoretical analysis of how secure the system is, but it is practically safe if the programs are made very carefully.

5. Future schedule

Our project is still at a prototype-implementation stage. The basic system design has already been finalized. We will test and investigate the actual usage of the system within a software company of 700 employees in the Tokyo area. We may also test this system for "inter-organization" use by introducing a "user-group identifying capability" in the system. The final goal of the project is to develop a "world-wide use" version as soon as possible.

References

[1] Stefan Katzenbeisser and Fabien A.P. Petitcolas (eds) :"Information hiding techniques for teganography and digital watermarking", Artech House, 2000.

[2] Neil F. Johnson, Zoran Duric and Sushil Jajodia :"Information Hiding", Kluwer Academic Publishers, 2001.

[3] M. Niimi, H. Noda and E. Kawaguchi :"An image embedding in image by a complexity based region segmentation method", Proceedings of International Conf. on Image Processing'97, Vol.3, pp.74-77, Santa Barbara, Oct., 1997.

[4] E. Kawaguchi and R. O. Eason :"Principle and applications of BPCS-Steganography", Proceedings of SPIE: Multimeda Systems and Applications, Vol.3528, pp.464-463, 1998.

[5] URL http://www.know.comp.kyutech.ac.jp/BPCSe/Dpenv-e/DPENVe-pro_down.html

[6] URL http://www.know.comp.kyutech.ac.jp/BPCSe/

[7] E. Kawaguchi, et al : "A concept of digital picture envelope for Internet communication" in Information modeling and knowledge bases X, IOS Press, pp.343-349, 1999.

Information Modelling and Knowledge Bases XIV
H. Jaakkola et al. (Eds.)
IOS Press, 2003

A Semantic Search Space Integration Method for Meta-level Knowledge Acquisition from Heterogeneous Databases

Yasushi Kiyoki* and Saeko Ishihara**
*Faculty of Environmental Information
Keio University
**Graduate School of Media and Governance
Keio University
Fujisawa, Kanagawa 252-8520, Japan
phone: 81+466-47-5111, fax: 81-466-47-5041
e-mail: kiyoki@sfc.keio.ac.jp

Abstract.
In this paper, we present a semantic search space integration method for a heterogeneous database environment. This method realizes integration among various semantic search spaces for meta-level knowledge acquisition from heterogeneous databases. This semantic space integration is performed with the interpretation of meanings in terms of common concepts between different databases in heterogeneous research fields. In this paper, we also present an implementation method for applying our integration method to actual semantic search spaces. We have implemented an actual space integration system for accessing environmental and medical information resources. We clarify the feasibility and effectiveness of our method and system by showing several experimental results for environmental and medical document databases.

1 Introduction

The most important objective of our study is to develop a meta-level knowledge base system for realizing a creative environment in new research fields by integrating information resources in various research fields, such as cultural, social, and natural sciences. The meta-level knowledge base system leads to highly creative activities for human beings over various research fields by sharing, retrieving, editing and integrating databases through wide-area computer networks.

A number of legacy databases for individual scientific research fields are connected to wide-area computer networks. As the existing legacy databases for those research fields have been designed and created with their own data structures, data representations and languages, it is difficult to obtain global knowledge by sharing, retrieving, editing and integrating those databases. In such a heterogeneous database environment, semantic heterogeneity poses problems in integrating different databases. Knowledge sharing, semantic retrieving and integrating those databases are essentially important for dynamically creating new research fields over various research fields[4, 9, 10, 13]. We have proposed a meta-level database system which realizes an intelligent database integration environment[4, 6, 8]. In this system, databases for various fields are connected to the meta-level layer, and those databases are integrated by semantic functions. By the connection among those databases from different fields, this system provides a knowledge integration environment for new scientific research related to various scientific fields.

We have also proposed a metadatabase system with a new semantic associative search method based on a mathematical model of meaning (MMM) [2, 6]. This method

makes it possible to extract and obtain significant information from multidatabases with a machinery for semantic associative search. In this method, the acquisition of information in multidatabases is performed by semantic computations.

It is complicated to deal with the meanings of data items in a multidatabase environment. One of the hardest problems is that it is difficult to identify the semantic equivalence, similarity and difference between data items which are extracted from different databases [1, 6, 9, 14]. It is not easy for users to select the appropriate databases and extract significant information for their requests. To provide the facilities for selecting the appropriate databases and extracting the significant information from those databases, a methodology for realizing semantic interoperability is an important part of database integration technology [1, 2, 10, 14]. The problematic relationships between data items for realizing semantic interoperability are classified into two types: "homonyms" and "synonyms." Homonym means the same data item is used for different concepts. Synonym means the same concept is described by different data items in different databases.

Our mathematical model of meaning (MMM) provides a function to compute the semantic equivalence, similarity and difference between data items which are included in different databases and realizes semantic interoperability among the data items. This model is used to find semantically equivalent or similar data items with different data representations and to recognize the different meanings of a data item. The main feature of this model is that the specific meaning of a data item can be dynamically fixed and unambiguously recognized according to the context. In this method, the data items of multidatabases are mapped into an orthogonal image space and selected by an intelligent semantic associative search mechanism[2, 6].

Several information retrieval methods, which use the orthogonal space created by mathematical procedures like SVD (Singular Value Decomposition), have been proposed. The MMM is essentially different from those methods using the SVD (e.g. the Latent Semantic Indexing (LSI) method [15, 16]). The essential difference is that the MMM provides the important function for semantic projections which realizes the dynamic recognition of the context. That is, in our model, the context-dependent interpretation is dynamically performed for computing the distance between words by selecting a subspace from the entire orthogonal semantic space. In our model, the number of phases of the contexts is almost infinite (currently 2^{2000} and 2^{800}, approximately). Other methods do not provide the context dependent interpretation for computing equivalence and similarity in the orthogonal space, that is, the phase of meaning is fixed and static.

We have applied the MMM to several multimedia database applications, such as image and music data retrieval by impressionistic classification. We have introduced these research results in [2, 6] and the book "Multimedia Data Management - using metadata to integrate and apply digital media -," McGraw Hill, Chapter 7, 1998 [5]. Through these studies, we aim to create a new meta-level knowledge base environment by applying those methods to data retrieval, data integration and data mining [3, 7].

In this paper, we present a new method of semantic retrieval space integration(SSI) for heterogeneous fields. This method makes it possible to integrate semantic retrieval spaces with the interpretation of meanings by using common concepts (common terms) for matrices of heterogeneous fields. This method realizes the information retrieval from viewpoints related to semantically integrated fields. In this paper, we also present an implementation method for applying our integration method to semantic associative search spaces. We clarify the feasibility and applicability of our method by several experiments for environmental fields.

In this method, it is assumed that common concepts (common terms) between heterogeneous fields are detected in advance before applying this method to the semantic associative search spaces corresponding to those fields. It is assumed that the semantic equivalence and similarity between terms in different fields are recognized by using our mathematical model of meaning (MMM) or the concept of ontology[1, 6, 9, 14].

The SSI method is used for integrating orthogonal spaces created by mathematical procedures like MMM[2, 6] and SVD(Singular Value Decomposition: e.g. the Latent Semantic Indexing (LSI) method [15, 16]). It can be applied to various semantic information retrieval methods using vector spaces. In those methods, a vector space is created for information retrieval for a single field or several fields which are fixed in advance. Although it is possible to create several vector spaces for several fields, those vector spaces are not integrated into a single space. Our SSI method dynamically integrates arbitrary vector spaces into a single space for realizing knowledge acquisition from information resources related to various research fields.

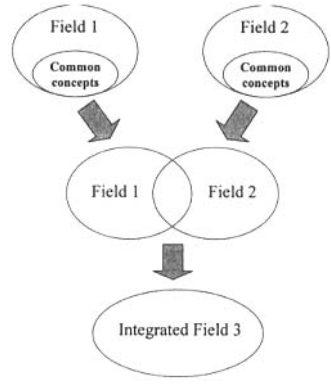


Fig.1: Integrated fields by using common concepts

2 Outline of the Semantic Space Integration Method

We propose a new semantic space integration method (SSI) for obtaining information related to multiple research fields. This method realizes semantic search space integration from different semantic search spaces as shown in Fig. 1. The procedure for semantic space integration and semantic search consists of the following processes:

Process-1: Creation for individual matrices (Original matrix creation),

Process-2: Semantic Space Integration (SSI) for matrices (Space Integration),

Process-3: Orthogonal semantic space creation for MMM and SVD (Integrated and Orthogonal Semantic Space creation).

Our SSI method defines a set of functions and data structures to realize Process-2. Process-1 and Process-3 are dependent on semantic search methods. In Section 4 we explain Process-1 and Process-3 in the case for applying the mathematical model of meaning (MMM) to orthogonal semantic space creation.

2.1 Data structure

The data structure is defined as a set of basic words and features in the form of a matrix with basic words and feature words as shown in Fig. 2. The data structure is referred to as "space matrix with words".

A set of basic words which characterizes the data items to be used is given in the form of an m by n matrix. That is, for given m basic words, each word is characterized by n features.

2.2 Basic function for semantic space integration

The semantic space integration function is defined for integrating individual spaces corresponding to two different fields. This method can be applied to integration of various semantic spaces created in different research fields independently of the order of the space sequence in semantics.

This function integrates two space matrices as shown in Fig.3 and Fig.4. That is, this function performs the semantic space integration between two different research fields. This new function is very important for integrating semantic spaces originally created in different research fields independently.

Although this function is not commutative between two space matrices M1 and M2, the integrated space M3 is not dependent on the order of the space sequence in semantics. That is, the order of the result spaces does not change the semantics in the

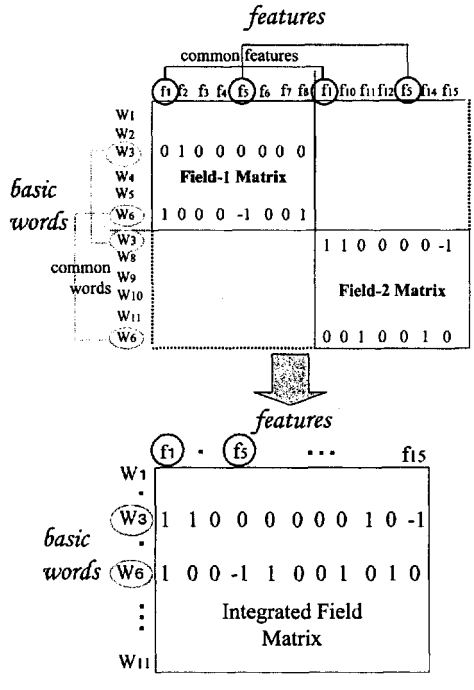


Fig.2: Integrated matrices

integrated space. Therefore, this function can be applied repeatedly to integration of various semantic spaces.

This function consists of the following three steps:

Step-1: Feature word integration:

Each feature word in the space matrix M2 is checked whether it exists commonly in the feature words of the space matrix M1 with the interpretation of synonymy. It is assumed that the semantic equivalence and similarity between words are recognized in advance by using ontology research results, such as in [1, 6, 9, 14], before applying this step to the space matrices M1 and M2. If a synonym or a common concept exists between feature words in M1 and M2, it is removed from the set of feature words of M2. The feature words of the integrated space matrix M3 consist of the feature words of M1 and the reduced feature words of M2, as shown in Fig. 3 and 4.

Step-2: Basic word integration:

Each basic word in the space matrix M2 is checked whether it exists commonly in the basic words of the space matrix M1 with the interpretation of synonymy. If a synonym or a common concept exists between basic words in M1 and M2, it is removed from the set of basic words of M2. The basic words of the integrated space matrix M3 consists of the basic words of M1 and the reduced basic words of M2, as shown in Fig. 3 and 4.

Step-3: Value settings to the integrated space matrix M3:

The basic words and feature words are set as vertical and horizontal words in M3 as shown in Fig. 4. Each element of M3 is set in this step. M3 consists of four submatrices M1', M2', M2" and M2"' as shown in Fig. 4. M3 is the matrix integrating two different space matrices which are created independently from different research fields.

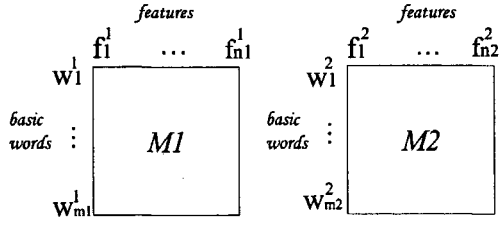


Fig.3: Two Original Single-Field Matrices (M1, M2)

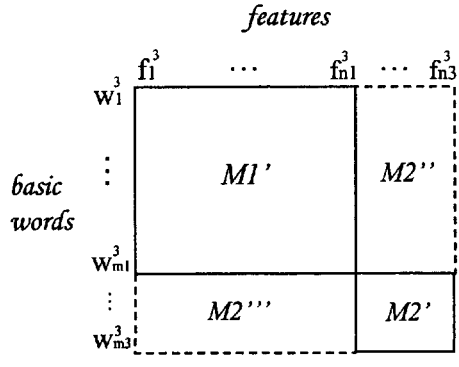


Fig.4: Integrated Matrix M3

M1' is the submatrix corresponding to the original M1. The basic and feature words in M1' are the same as those words in M1. Each element of M1' is set to the same value as the value in the original M1 if the basic word and the feature word corresponding to this element are not commonly existing between the original M1 and M2. If both of the basic word and the feature word in M1' are commonly existing between the original M1 and M2, the element corresponding to these words is set to the value computed by the integrating operation(integrator)between the M1 and M2 elements corresponding to the common basic and feature words.

M2' is the submatrix corresponding to the reduced M2 after eliminating both of common basic and feature words to M1 from M2. Each element of M2' is set to the same value as the value in the original M2 in terms of the reduced basic and feature words neither of which has common words to the original M1.

M2" is the submatrix corresponding to the elements where the common basic words are existing between the original M1 and M2 and the feature words between M1 and M2 are different. Each element of M2" is set to the same value as the value corresponding to the basic and feature words in the original M2.

M2"' is the submatrix corresponding to the elements where the common feature words are existing between the original M1 and M2 and the basic words between M1 and M2 are different. Each element of M2"' is set to the same value as the value corresponding to the basic and feature words in the original M2.

3 Integration Function

In this section, we define the integration function for our semantic space integration model. The data structures and the basic operations explained in Section 2 are formulated in the following expressions.

The semantic space integration function F is expressed as

$$M3 = F(M1, M2)$$
$$\{e^3_{i,j}\} = F(\{e^1_{i,j}\}, \{e^2_{i,j}\}),$$

where $e^1_{i,j}$ and $e^2_{i,j}$ are elements of the original space matrices M1 and M2, respectively, and $e^3_{i,j}$ is an element of the integrated space matrix M3.

The space matrix integration function F is expressed in the following formulation.

The schema of the space matrix M3 is represented as an ordered set of basic words and an ordered set of feature words as shown in Fig. 4. The ordered set of basic words is the vertical elements in M3, and the ordered set of feature words is the horizontal elements in M3.

The feature words in the space matrix M3 to be created as an integrated space matrix are extracted from the feature words of the original space matrices M1 and M2. $\bar{O}Set$ is the operator which makes an ordered set for words. The numbers of feature words in M1 and M2 are represented as $n1$ and $n2$, respectively. The number of feature words in M3 is represented as $n3$,

$$n2' = count\ (Difference\ (Set_{i=1,n2}(f^2_i), Set_{j=1,n1}(f^1_j))$$
$$\bar{O}Set_{k=1,n2'}(f^{2'}_k) \equiv$$
$$\quad Difference\ (\bar{O}Set_{i=1,n2}(f^2_i), \bar{O}Set_{j=1,n1}(f^1_j))$$
$$n3 = n1 + n2'.$$

The basic words in the space matrix M3 are extracted from the basic words of the original space matrices M1 and M2. The numbers of basic words in M1 and M2 are represented as $m1$ and $m2$, respectively. The number of basic words in M3 is represented as $m3$,

$$m2' = count\ (Difference\ (Set_{i=1,m2}(w^2_i), Set_{j=1,m1}(w^1_j))$$
$$\bar{O}Set_{k=1,m2'}(w^{2'}_k) \equiv$$
$$\quad Difference\ (\bar{O}Set_{i=1,m2}(w^2_i), \bar{O}Set_{j=1,m1}(w^1_j))$$
$$m3 = m1 + m2'.$$

The integrated matrix M3 consists of four submatrices M1', M2', M2" and M2"'.

The matrix M1' is the submatrix corresponding to the original M1. The basic and feature words in M1' are the same as those words in M1. Each element $e^{1'}_{i,j}$ of M1' is defined as follows:

$$e^{1'}_{i,j} = \begin{cases} integrator((e^1_{i,j}), (e^2_{i',j'})) & if((i \le n1) \wedge (j \le m1)) \wedge (f^1_i = f^2_{i'}) \wedge (w^1_j = w^2_{j'}) \\ e^1_{i,j} & \text{otherwise} \end{cases}$$

The domain of the matrix elements and the integrator are application-dependently fixed. This model does not give restriction for defining them. In our current application study to environmental and medical semantic spaces, the domain of the elements is {-1, 0, 1}, and the integrator is defined as a three-valued logical OR operator where the OR operator between "-1" and "1" gives "0", and that between "-1" and "0" gives "-1".

M2" is the submatrix corresponding to the elements where the common basic words are existing between the original M1 and M2 and the feature words between M1 and M2 are different. Each element $e^{2''}_{i,j}$ of M2" is defined as follows:

$$e^{2''}_{i,j} = \begin{cases} e^2_{i,j'} & if((i > n1) \wedge (j \le m1)) \wedge (w^1_{j'} = w^2_{j'}) \\ \emptyset & \text{otherwise} \end{cases}$$

M2"' is the submatrix corresponding to the elements where the common feature words are existing between the original M1 and M2 and the basic words between M1 and M2 are different. Each element $e^{2'''}_{i,j}$ of M2"' is defined as follows:

$$e^{2'''}_{i,j} = \begin{cases} e^{2}_{i',j'} & if((i \leq n1) \wedge (j > m1)) \wedge (f^1_i = f^2_{i'}) \\ \emptyset & \text{otherwise} \end{cases}$$

M2' is the submatrix corresponding to the reduced M2 after eliminating both of common basic and feature words to M1 from M2. Each element $e^{2'}_{i,j}$ of M2' is defined as follows:

$$e^{2'}_{i,j} = \{ e^{2}_{i',j'} \quad if((i > n1) \wedge (j > m1)) \wedge (f^1_i \neq f^2_{i'}) \wedge (w^1_j \neq w^2_{j'})$$

The space matrix M3 is created by combining the submatrices M1', M2', M2" and M2"'. Each element $e^3_{i,j}$ of M3 is defined as follows:

$$e^3_{i,j} = \begin{cases} e^{1'}_{i,j} & if(i \leq n1) \wedge (j \leq m1) \\ e^{2'}_{i,j} & if(i > n1) \wedge (j \leq m1) \\ e^{2''}_{i,j} & if(i \leq n1) \wedge (j > m1) \\ e^{2}_{i,j} & if(i > n1) \wedge (j > m1) \end{cases}$$

4 Outline of the Mathematical Model of Meaning

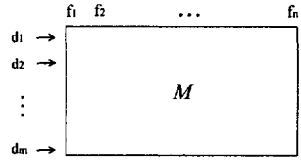


Fig.5: Metadata represented in data matrix M

In this section, the outline of the mathematical model of meaning (MMM) is briefly reviewed. This model has been presented in [2, 6] in detail.

In MMM, Process-1 (Creation for an individual matrix) and Process-3 (Orthogonal semantic space creation) are realized by the following methodology. Process-2 (Semantic Space Integration(SSI) for matrices) is realized by the function defined in Section 2.

1. Assumption :

   As Process-1 in Section 2, a set of basic words which characterizes the data items to be used is given in the form of an m by n matrix. That is, for given m words, each word is characterized by n features, as shown in Fig. 5.

2. Defining the image space $\mathcal{I}$:

   As Process-3 in Section 2, first we construct the correlation matrix with respect to the features. Then we execute the eigenvalue decomposition of the correlation matrix and normalize the eigenvectors. We define the image space $\mathcal{I}$ as the span of the eigenvectors which correspond to nonzero eigenvalues. We call such eigenvectors semantic elements hereafter. We note that since the correlation matrix is symmetric, the semantic elements form orthonormal bases for $\mathcal{I}$. The dimension ν of the image space $\mathcal{I}$ is identical to the rank of the data matrix A. Since $\mathcal{I}$ is ν dimensional Eucledian space, various norms can be defined and a metric is naturally introduced.

3. Defining a set of the semantic projections Π_ν:

   We consider the set of all the projections from the image space $\mathcal{I}$ to the invariant subspaces (eigen spaces). We refer to the projection as the semantic projection

and the corresponding projected space as the semantic subspace. Since the number of i dimensional invariant subspaces is $(\nu(\nu-1)\cdots(\nu-i+1))/i!$, the total number of the semantic projections is 2^ν. That is, this model can express 2^ν different phases of the meaning.

4. Constructing the Semantic Operator S_p:

   Suppose a sequence s_ℓ of ℓ words (context words) which determines the context is given. We construct an operator S_p to determine the semantic projection according to the context. Context words are given as a sequence of several keywords which are defined with n-dimensional vectors to specify the query for information retrieval. Several examples for context words and vectors were presented in [4, 6].

   We call the operator a semantic operator.

   (a) First we map the ℓ context words in databases to the image space $\mathcal{I}$. This mathematically means that we execute the Fourier expansion of the sequence s_ℓ in $\mathcal{I}$ and seek the Fourier coefficients of the words with respect to the semantic elements. This corresponds to seeking the correlation between each context word of s_ℓ and each semantic element.

   (b) Then we sum up the values of the Fourier coefficients for each semantic element. This corresponds to finding the correlation between the sequence s_ℓ and each semantic element. Since we have ν semantic elements, we can constitute a ν dimensional vector. We call the vector normalized in the infinity norm the semantic center of the sequence s_ℓ.

   (c) If the sum obtained in (b) for a semantic element is greater than a given threshold ε, we employ the semantic element to form the projected semantic subspace. We define the semantic projection by the sum of such projections.

   This operator automatically selects the semantic subspace which is highly correlated with the sequence s_ℓ of the ℓ context words which determines the context.

   This model makes dynamic semantic interpretation possible. We emphasize here that, in our model, the "meaning" is the selection of the semantic subspace, namely, the selection of the semantic projection and the "interpretation" is the best approximation in the selected subspace.

As an example, we have implemented an experimental system of the mathematical model of meaning [4, 6, 8]. As an example of the m by n matrix, we used basic words in an English dictionary "General Basic English Dictionary [12]" in which approximately 871 basic words are used to explain each English word[12]. Those English words are used as features, that is, they are used as the features corresponding to the columns in the matrix. That is, 871 features are provided to make the image space. And, approximately 2115 words are used to represent the words corresponding to the rows in the matrix. Those words are used as the basic words in the English dictionary "Longman Dictionary of Contemporary English [11]." The 2115×871 matrix is used to create the image space. By using this matrix, an image space is computed within the framework of the mathematical model of meaning. This space represents the semantic space for computing meanings of the keywords and data items which are used in a multidatabase environment. A given keyword and data items are mapped into this space, and semantic equivalence and similarity between the keyword and data items are computed in a mathematical way. This image space consists of 868 dimensional orthogonal vectors, approximately.

Similarly, the 2115×2115 matrix is used to create another image space. In this matrix, approximately 2115 words of "Longman Dictionary of Contemporary English [11]" are used to represent not only the basic words corresponding to the rows but also the features the columns in the matrix. This image space consists of 2000 dimensional orthogonal vectors, approximately. In those image spaces, the number of phases of the contexts is almost infinite (currently 2^{868} and 2^{2000}, approximately).

5 Application to Integration for two Semantic Spaces for Environmental Information and Medical Information

To clarify the feasibility and effectiveness of our space integration method, we performed several experiments by using two different semantic spaces which include environmental

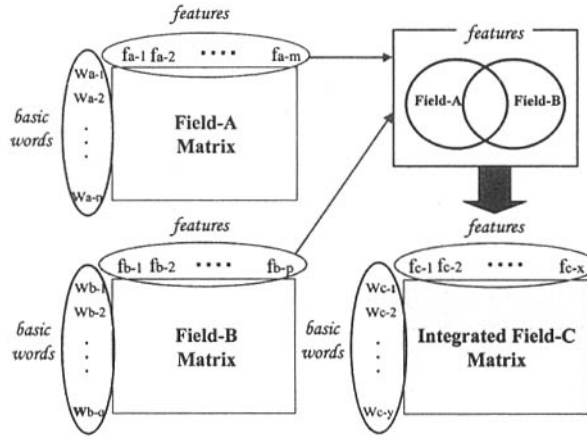


Fig.6: Integration for features and words

and medical information, respectively. The objective of these experiments is to evaluate the effectiveness and applicability of our method to combine the actual different semantic spaces. We have actually created two semantic spaces for environmental and medical information, and integrated those spaces by applying our space integration method.

5.1 Experimental environment

As a multidatabase environment including environmental and medial information, the semantic spaces for those fields were created and integrated into an integrated semantic space for an environmental and medical information by using our SSI method, as shown in Fig. 6 and 7.

For creating the environmental semantic space, we have referenced a dictionary for the environmental field [17, 18, 19]. From the dictionary, we have extracted basic words and feature words for the environmental semantic space(environmental metadata space).

For the medial semantic space, we have referenced a dictionary for the medical field [20, 21] and extracted basic words and feature words for the medical semantic space(medical metadata space).

Three semantic spaces(M1, M2 and M3) have been created for retrieving documents related to environmental, medical and integrated environmental-medical semantic spaces. Each semantic space is independently used for retrieving those documents.

(1) environmental semantic space for retrieving environmental documents as the space matrix M1 in Section 2.

(2) medical semantic space for retrieving medical documents as the space matrix M2 in Section 2.

(3) integrated environmental-medical semantic space for retrieving environmental and/or medical documents as the space matrix M3 in Section 2.

Our space integration method is applied to create the integrated environmental-medical semantic space (M3) from the space matrices M1 and M2. We have implemented our space integration method by developing the programs in Perl programming language.

The space matrices for those spaces are shown in Table 1.

As retrieval candidate documents for the environmental semantic space, we mapped 500 newspaper articles into the space.

As retrieval candidate documents for the medical semantic space, we mapped 500 newspaper articles into the space.

We also mapped those 1000 documents into the integrated environmental-medical semantic space.

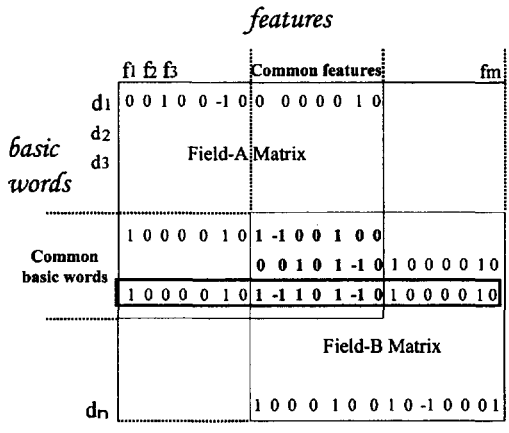


Fig.7: Integration for vector elements

Table 1: Semantic spaces

	Number of feature	Number of words	Space dimension
Environmental semantic space	425	469	415
Medical semantic space	437	690	436
Integrated semantic space	806	1127	794
Number of common terms	56	32	-

For mapping each of those documents into semantic spaces, we created a vector for each document by extracting a set of metadata from the document. Several metadata sets are shown in Table 2 as examples.

5.2 Experiment-1

In Experiment-1, we evaluate the quality of the retrieval results in a single semantic space created from a single field. That is, the environmental semantic space mapping the 500 environmental documents is used for evaluating retrieval results in this space. Similarly, the medical semantic space is used for evaluation.

5.2.1 Evaluation method

In Experiment-1, 15 queries have been given to our experimental system with the environmental semantic space. Similarly, other 15 queries have been given to our experi-

Table 2: Examples of metadata

Document-ID	metadata
980122259	influenza, virus, infection, tuberculosis, cold
980210202	cancer, virus, gene, inflammation, infection, hepatitis, hepatic, - -
971023101	dioxin, hormone, environment, pesticide
980623148	ulcer, cancer, gastritis, antibiotic, bowel
980723299	stress, hormone, excitation, diabetes, brain, obesity, hospital, senescence
001224107	greenhouse gas, warming, sea, limate change, Kyoto protocol, Forest, - -
001117222	dioxin, contamination, environmental quality, heavy metal, soil pollution
000528158	forest, bionomics, ecosystem, tree planting, forestry
980422027	ozone, ozone layer, chlorofluocarbon, greenhouse effect, warming, - -
981207252	contamination, sewer, river, environmental hormones, lake, water pollution, - -

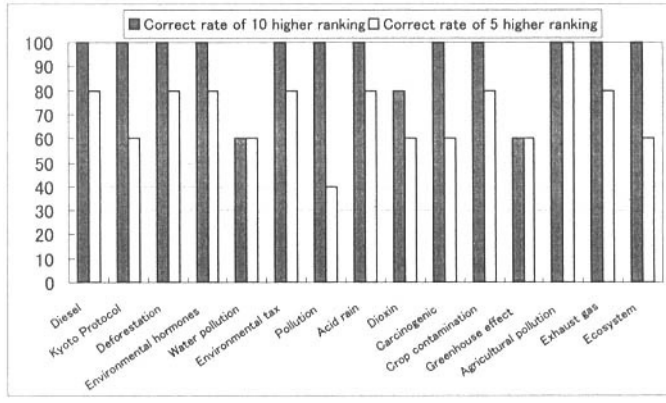


Fig.8: The result for the Environmental Space

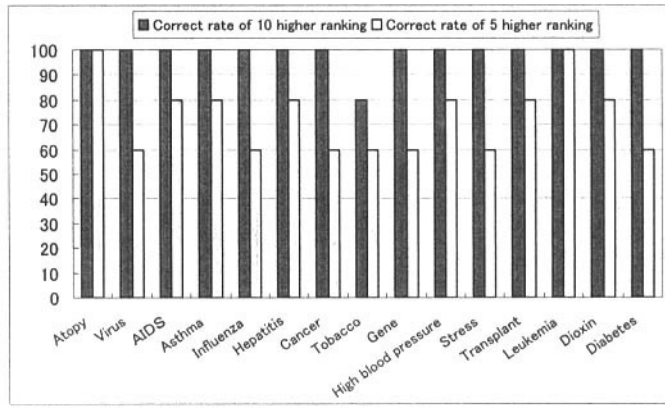


Fig.9: The result for the Medical Space

mental system with the medical semantic space. We have fixed 5 documents as correct answers for each query in advance. In the experimental results, we show the ratio (recall evaluator) of the correct answers included in the top 5 and top 10 retrieval results in Fig.8 for the environmental semantic space, and in Fig.9 for the medical semantic space.

5.2.2 Experimental results

Those experimental results show that our system realizes high quality retrieval for documents in terms of the recall evaluator. For all queries, the ratio (recall ratio) of the correct answers included in the top 10 retrieval results is in 80 to 100%. And, for a half of queries, the ratio (recall ratio) of the correct answers included in the top 5 retrieval results is in more than 80%.

In the query "environmental hormones", the document, whose metadata do not include "environmental hormones" itself, is selected in the high ranking in the retrieval results. The metadata of this document are "PCB ,hormone, contamination, agricultural chemicals, insecticide, industrial waste". This experiment shows that our semantic retrieval system realizes the dynamic computation for semantic correlation between a

given query word and metadata of documents.

In Experiment-2, we evaluate the quality of retrieval results in the integrated semantic space created by our semantic space integration method, in comparing with retrieval results in the single semantic spaces. The main advantage of our method is that it realizes the acquisition of the documents which are related to two or more different fields, but are not selected in each single semantic space because they do not have high correlation to each single field.

We mapped 1000 documents into the integrated environmental-medical semantic space. Those documents, which are used in Experiment-1, are related to the environmental and/or medical fields.

5.2.3 Evaluation method

In this experiment, the same queries are given to the environmental semantic space, the medical semantic space and the integrated environmental-medical semantic space, respectively.

The experimental results are shown in Tables 3, 4 , and 5.

The left-side numbers in each column are document identifiers (Document ID's), and the right-side values represent semantic correlations between a given query and each document.

5.2.4 Experimental results

Table 3 shows the retrieval results to the query "air-pollution". For this query, in the environmental semantic space, the document [01128042] is in the ranking 157th, that is, it is not selected in the high ranking. However, in the integrated environmental-medical semantic space, this document is in the ranking 4th. This result shows that this document can be selected in the integrated semantic space. The metadata for this document are "SPM, diesel, smoke, contamination, contaminant, warming, sea, environmental quality, environment agency, a standard, light oil, health, healthy damage, pollution, Oxide, The automobile NOx law, air pollution, Nitrogen, Nitrogen Oxide, Nitrogen dioxide, Exhaust gas, emission control, Exhaust, floating particle-like substance, Sulfur". The subset of the metadata "SPM, diesel, smoke-contamination, contaminant, warming, air pollution, Nitrogen oxide, Exhaust gas" of the metadata is highly related to "air pollution" in the environmental field, and the other subset of the metadata "health, healthy damage, pollution" are highly related to "air pollution" in the medical field. That is, this document is related to "air pollution" in the integrated environmental-medical semantic space, rather than in the single semantic spaces. This result shows that our method makes it possible to acquire information which is related to several fields and is not extracted in the semantic space created for a single field. Unlike our method, other methods using semantic spaces cannot extracted this kind of documents.

Similarly, Table 4 shows the retrieval results to the query "Environmental hormones". For this query, in the environmental semantic space, the document [000828030] is in the ranking 346th, that is, it is not selected in the high ranking. However, in the integrated environmental-medical semantic space, this document is in the ranking 9th. This result shows that our method realizes acquisition of information which is related to several fields and is not extracted in the semantic space created for a single field.

The metadata for this document are "SPM, cancer, Diesel, contamination, Greenhouse effect, sea, environmental standards, Environment Agency, a standard, Light oil, health, healthy damage, pollution, Oxide, automobile NOx law, the measure against an automobile exhaust gas, petroleum, air pollution, Carbon, Nitrogen, Nitrogen oxide, Copper, Carbon dioxide, Nitrogen dioxide, Exhaust gas, Emission control, the amount of discharge, Carcinogenic, Floating particle-like substance, fog, Sulfur".

In this document, the subset of the metadata "cancer, health, healthy damage, pollution, Carcinogenic" is highly related to "Environmental hormones" in the medical field. This document is not extracted in the environmental semantic space because the metadata are not highly related to "Environmental hormones" in the space, but it is extracted in the integrated environmental-medical semantic space.

Table 5 shows the retrieval results to the query "rehabilitation". For this query, in the medical semantic space, the document [980924091] is in the ranking 193th, that is, it is not selected in the high ranking. However, in the integrated environmental-medical semantic space, this document is in the ranking 5th. The metadata of this document

Table 3: The result for the query "Air pollution"

rank	Integrated Space	Environmental Space	Medical Space
1	001217119 - 0.757872	000625162 - 0.395851	971004314 - 0.532862
2	000826025 - 0.746731	000622083 - 0.365665	980514260 - 0.495517
3	990506132 - 0.746605	000120132 - 0.353257	980508179 - 0.483583
4	001128042 - 0.732719	000119220 - 0.349681	980606209 - 0.461891
5	980617296 - 0.729570	980903276 - 0.341292	971225110 - 0.455851
6	001106140 - 0.722226	980417025 - 0.324443	980417168 - 0.454193
7	980822158 - 0.721012	971015081 - 0.318102	980417134 - 0.428897
8	001015134 - 0.719486	001220024 - 0.313622	981223001 - 0.427919
9	000828030 - 0.718638	001201344 - 0.312631	981207252 - 0.407963
10	990630163 - 0.715068	000201030 - 0.312409	000503002 - 0.367019
-	-	-	-
155	990128019 - 0.550391	000630141 - 0.166195	980731226 - 0.202048
156	990705164 - 0.549977	990523208 - 0.164348	000603039 - 0.201249
157	000317080 - 0.549225	001128042 - 0.163946	980318249 - 0.201079
158	990817003 - 0.548865	000914166 - 0.161578	980609034 - 0.200232
159	990430215 - 0.548798	990409037 - 0.160712	980226112 - 0.199906
160	000916016 - 0.548206	000430155 - 0.160669	980108022 - 0.199830
-	-	-	-
495	981008357 - 0.240430	980421066 - 0.074441	981030300 - 0.112964
496	971023101 - 0.238382	001115316 - 0.073632	981105322 - 0.111208
497	000718090 - 0.237932	000522195 - 0.073462	990320218 - 0.111173
498	000820076 - 0.236857	980124137 - 0.072628	980518076 - 0.105826
499	980514260 - 0.236689	000815116 - 0.071398	980702379 - 0.103438
500	980606209 - 0.231906	001217119 - 0.070788	000519231 - 0.102851
-	-		
995	980122259 - 0.000000		
996	980119050 - 0.000000		
997	990129172 - 0.000000		
998	981201341 - 0.000000		
999	980320232 - 0.000000		
1000	990328037 - 0.000000		

Table 4: The result for the query "Environmental hormones"

rank	Integrated Space	Environmental Space	Medical Space
1	001217119 - 0.608303	990622146 - 0.490062	971004314 - 0.833046
2	990506132 - 0.602118	991209025 - 0.488231	990207146 - 0.805216
3	000826025 - 0.597233	001008152 - 0.482937	001203184 - 0.798642
4	001128042 - 0.588013	001219087 - 0.479456	980226374 - 0.796882
5	980617296 - 0.588009	990319268 - 0.471976	000228033 - 0.796095
6	001015134 - 0.583953	000917206 - 0.462762	980315082 - 0.789393
7	980822158 - 0.580817	000420149 - 0.459485	001018015 - 0.788370
8	001106140 - 0.579942	000724149 - 0.458495	980827211 - 0.784964
9	000828030 - 0.579025	000308112 - 0.455430	000208028 - 0.783548
10	990630163 - 0.575484	991008090 - 0.450773	000111020 - 0.783444
-	-	-	-
345	991002024 - 0.397573	990529311 - 0.165141	001023062 - 0.676683
346	990206130 - 0.396505	000828030 - 0.164548	981216292 - 0.676617
347	971023101 - 0.396302	001217119 - 0.164532	990403229 - 0.676071
348	980302004 - 0.396166	001112159 - 0.164246	971120001 - 0.676011
349	980904187 - 0.395591	001213222 - 0.164041	980518076 - 0.675793
350	990708027 - 0.395425	001015140 - 0.163888	001128142 - 0.675600
-	-	-	-
495	000827163 - 0.299448	990429123 - 0.108627	980404225 - 0.600715
496	980417363 - 0.299043	000615024 - 0.106554	980926161 - 0.598681
497	970522131 - 0.297935	990504190 - 0.105574	981211355 - 0.590334
498	990212137 - 0.297796	000622083 - 0.105187	980129109 - 0.585872
499	000730157 - 0.297184	980110032 - 0.099752	980926176 - 0.562776
500	000329078 - 0.296400	980307043 - 0.097569	000424258 - 0.545890
-	-		
995	000929221 - 0.053021		
996	990212158 - 0.052136		
997	990608208 - 0.051044		
998	001202168 - 0.050939		
999	000901028 - 0.047600		
1000	000518111 - 0.047534		

Table 5: The result for the query "Rehabilitation"

rank	Integrated Space	Environmental Space	Medical Space
1	000212252 - 0.343179		000212252 - 0.376446
2	990608334 - 0.306642		001217040 - 0.299593
3	980731226 - 0.305199		980731226 - 0.299593
4	001217040 - 0.305199		990510190 - 0.289963
5	980924091 - 0.304567		990129248 - 0.288849
6	990201257 - 0.300692		000503002 - 0.287556
7	001216154 - 0.295039		990101224 - 0.284870
8	980123363 - 0.292607		980713306 - 0.282685
9	990101224 - 0.292605		000721226 - 0.282491
10	001206315 - 0.289999		000727218 - 0.282465
-	-		-
190	990319234 - 0.223382		001102187 - 0.233054
191	990429202 - 0.223382		971228015 - 0.232904
192	980803175 - 0.223363		981016057 - 0.232896
193	980212131 - 0.222834		980924091 - 0.232876
194	000408035 - 0.222228		001023282 - 0.232871
195	000131170 - 0.222139		000914216 - 0.232793
-	-		-
495	990515229 - 0.162435		981007173 - 0.157683
496	001117222 - 0.161544		980421173 - 0.153499
497	001203184 - 0.161426		970919029 - 0.152549
498	000611151 - 0.161363		000402194 - 0.152527
499	980209031 - 0.160666		001114034 - 0.151237
500	980109222 - 0.160655		980801168 - 0.146089
-	-		
995	000818040 - 0.093834		
996	980901233 - 0.093075		
997	000713104 - 0.092315		
998	001023061 - 0.091412		
999	990429041 - 0.089518		
1000	980824168 - 0.087680		

are "MRI, epilepsy, paralysis, rehabilitation, disturbance of consciousness, hepatic cirrhosis, trachea, thrombus, hypercholesterolemia, hypertension, hyperlipemia, bleeding, tongue, apoplexy, position, bowel, hypotension, diabetes, head, arteriosclerosis, urine, Brain, cerebral thrombosis, stroke, Lungs, pneumonia, attack, hospital, pulse". These metadata are related to both environmental and medical fields. This result shows that our method makes it possible to extract information which is related to various fields.

These experimental results have shown that our semantic space integration method realizes a new semantic information acquisition and retrieval environment. This method can be applied repeatedly to integration of various semantic spaces created in different research fields.

5.3 Experiment-3

In Experiment-3, we evaluate the ability in terms of field independency in the integrated semantic space.

5.3.1 Evaluation method

As retrieval candidate documents for the integrated semantic space, we mapped 1000 documents (newspaper articles) into the space, where the 500 documents are from the environmental field and the other 500 documents are from the medical field.

Queries are classified into three kinds: (1) queries related to the environmental field, (2) queries related to the medical field, and (3)queries related to both environmental and medical fields.

For each query, query results are represented in the ranking for documents. Points are given to each document in the top 100, as 100 points for the 1st ranking document, 99 points for 2nd ranking, - - -, and 1 point for the 100th ranking.

For the queries related to the environmental field, it is expected that the total points for the documents related to the environmental field should be high in the query result. The experimental results are shown in Fig. 10, 11 and 12.

5.3.2 Experimental results

For the queries related to the environmental field, the documents highly related to the environmental field have been extracted in high ranking, and for the queries related to the medical field, the documents highly related to the medical field have been extracted in high ranking. Furthermore, for the queries related to both environmental and medical fields, the documents have been extracted from both environmental and medical fields.

Those experimental results show that the integrated semantic space has ability for realizing field independency, that is, for field-specific queries, documents related to the field itself can be extracted sharply. Furthermore, for queries related to various fields, our integrated semantic space realizes information acquisition from various fields.

Our experimental study has shown that our semantic space integration method and integrated semantic space realize new semantic space creation for integrating various research fields.

6 Conclusion

Our metadatabase system has been designed for dealing with semantic search space integration between heterogeneous databases in a multidatabase environment. In this system, the machinery for semantic associative search is realized as the essential semantic search system for extracting the semantically related information in a multidatabase environment.

In this paper, we have also presented the applicability of the semantic space integration method to the actual multidatabases for environmental and medical information. We have shown several experimental results which have been obtained by semantic search for two different databases in a multidatabase environment. In this experimental study, we have implemented an actual system for environmental and medical fields which are becoming important in the world-wide areas and clarified the feasibility and effectiveness of the metadatabase system in the actual multidatabases.

As the future work, we will extend the current integrated semantic space to include various databases in heterogeneous research fields. Furthermore, we will integrate the

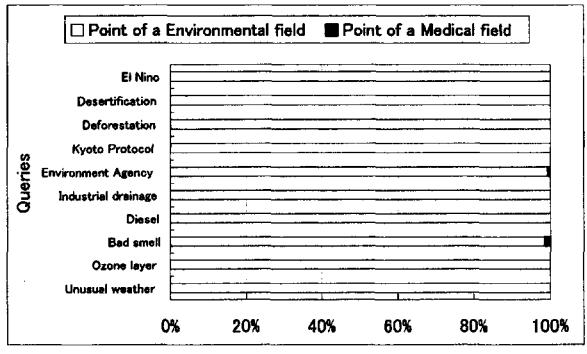


Fig.10: The result of queries related to the Environmental field

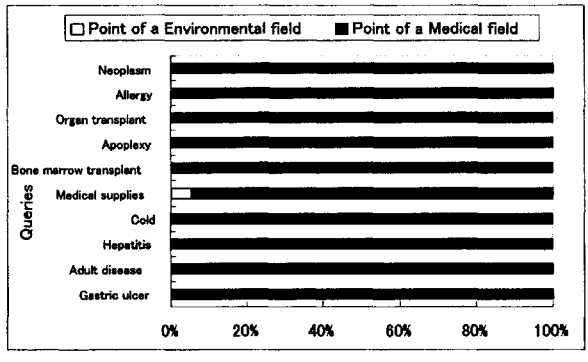


Fig.11: The result of queries related to the Medical field

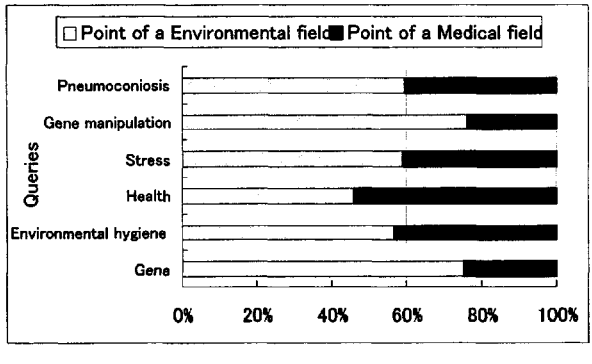


Fig.12: The result of queries related to both environmental and medical fields

semantic associative search system with the multimedia database systems[3, 4] in a distributed and network database system to realize multimedia databases for various research fields.

References

[1] Bright, M.W., Hurson, A.R., and Pakzad, S.H.(1992), "A Taxonomy and Current Issues in Multidatabase System," *IEEE Computer, Vol.25, No.3*, pp.50-59.

[2] Kitagawa, T. and Kiyoki, Y., *"The mathematical model of meaning and its application to multidatabase systems,"* Proceedings of 3rd IEEE International Workshop on Research Issues on Data Engineering: Interoperability in Multidatabase Systems, pp.130-135, April 1993.

[3] Kiyoki, Y. and Kitagawa, T., *"A metadatabase system for supporting semantic interoperability in multidatabases,"* Information Modelling and Knowledge Bases, IOS Press, Vol. V, pp. 287-298, 1994.

[4] Kiyoki, Y., Kitagawa, T., and Hayama,T., *"A metadatabase system for semantic image search by a mathematical model of meaning,"* ACM SIGMOD Record, (refereed as the invited paper for special issue on metadata for digital media), Vol.23, No. 4, pp.34-41, Dec. 1994.

[5] Kiyoki, Y., Kitagawa, T., and Hayama,T.,*"A metadatabase system for semantic image search by a mathematical model of meaning,"* Multimedia Data Management – using metadata to integrate and apply digital media –, McGrawHill(book), A. Sheth and W. Klas(editors), Chapter 7, 1998.

[6] Kiyoki, Y., Kitagawa, T. and Hitomi, Y., *"A fundamental framework for realizing semantic interoperability in a multidatabase environment,"* International Journal of Integrated Computer-Aided Engineering, Vol.2, No.1(Special Issue on Multidatabase and Interoperable Systems), pp.3-20, John Wiley & Sons, Jan. 1995.

[7] Kiyoki, Y. and Kitagawa, T., *"A semantic associative search method for knowledge acquisition,"* Information Modelling and Knowledge Bases (IOS Press), Vol. VI, pp.121-130, 1995.

[8] Kiyoki, Y. and Kitagawa, T. , *"Application of a Semantic Associative Search Method to Multidatabases for Environmental Information,"* Information Modelling and Knowledge Bases (IOS Press), Vol. XI, May, 1999.

[9] Larson, J.A., Navathe, S.B., Elmasri, R.(1989), *"A theory of attribute equivalence in database with application to schema integration,"* IEEE Transaction on Software Engineering, Vol.15, No.4, pp.449-463, 1989.

[10] Litwin, W., Mark, L., and Roussopoulos, N.(1990), *"Interoperability of Multiple Autonomous Databases,"* ACM Comp. Surveys, Vol.22, No.3, pp.267-293., 1990.

[11] *"Longman Dictionary of Contemporary English,"* Longman, 1987.

[12] Ogden, C.K.(1940), *"The General Basic English Dictionary,"* Evans Brothers Limited, 1940.

[13] Sheth, A. and Larson, J.A.(1990), *"Federated database systems for managing distributed, heterogeneous, and autonomous databases,"* ACM Computing Surveys, Vol.22, No.3, pp.183-236, 1990.

[14] Sheth, A. and Kashyap, V.(1992), *"So far (schematically) yet so near (semantically),"* Proc. IFIP TC2/WG2.6 Conf. on Semantics of Interoperable Database Systems, pp.1-30.

[15] Deerwester, S., Dumais, S. T., Landauer, T. K., Furnas, G. W. and Harshman, R. A., *"Indexing by latent semantic analysis,"* Journal of the Society for Information Science, vol.41, no.6, 391-407, 1990.

[16] Dumais, S. T., Furnas, G. W., Landauer, T. K., and Deerwester, S., *"Using latent semantic analysis to improve information retrieval,"* Proc. CHI'88: Conference on Human Factors in Computing, New York: ACM, 281-285.. p

[17] *"Dictionary for Environmental Problem Information, "* Nichigai Associates, Tokyo, 477p., 2001.

[18] Ueda, T., *"Handy Dictionary for Environmental vocabularies, "* Kyoritsu, Tokyo, 332p., 2000.

[19] Araki, S., *"Environmental Scientific Dictionary, "* Tokyo-Kagaku-Dojin, 1015p., 1985.

[20] Goto, C., *"Medical Dictionary, "* Version 2, Ishi-yaku, Tokyo, 1999.

[21] Hino, S., *"Medical Dictionary, "* Igaku-Shoin, Tokyo, 1107p., 1992.

Information Modelling and Knowledge Bases XIV
H. Jaakkola et al. (Eds.)
IOS Press, 2003

Generation of Server Page Type Web Applications from Diagrams

Mitsuhisa Taguchi, Tetsuya Suzuki, and Takehiro Tokuda
{mtaguchi,tetsuya,tokuda}@tt.cs.titech.ac.jp
Department of Computer Science, Tokyo Institute of Technology
Meguro, Tokyo 152-8552, Japan

Abstract. Web applications such as database query systems and transaction systems are widely used on the Internet. There are two types of Web applications, server program type and server page type. From the viewpoint of automatic generation, server page type Web applications are not easy to generate. In this paper, extending the model for automatic generation of server program type Web applications, we propose diagrams called Web transition diagrams to represent server page type Web applications such as ASP and JSP. Then we design a Web application generator for server page type Web applications. Typical server page type Web applications can be generated using our generator without procedural programming. The standard level of security management in Web applications is also provided automatically.

1 Introduction

Currently, the demand of Web applications such as database query systems and transaction systems is rapidly increasing on the Internet. In order to support development of Web applications, technologies such as CGI programs and Java servlets appeared [1, 2, 3]. Source codes of Web applications based on these technologies consist of processing program codes in which hypertexts are embedded. We call these Web applications *server program type Web applications.* However, this approach has following problems. The readability of source codes is not good and a change of a page design is not easy. Then, technologies such as ASP (Active Server Pages) and JSP (Java Server Pages) appeared [2, 4]. Documents of Web applications based on these technologies consist of hypertexts in which processing program codes are embedded. We call these Web applications *server page type Web applications.* Server page type technology enables us to separate appearances of Web pages from processing programs.

However, the difficulty of systematic development of Web applications is still a big problem for non-programmers and unexperienced engineers. A Web application generator called T-Web system based on server program type technology has been proposed [5, 6, 7, 8]. T-Web system is a system to generate Web applications from diagrams automatically. However, to design a Web application generator based on server page type technology is not easy, because processing program codes are dispersed in documents of Web pages.

In this paper, we extend the model for automatic generation of server program type Web applications, and design T-Web system for server page type Web applications. We propose diagrams called Web transition diagrams to represent Web applications based on server page type technology, especially with ASP and JSP architecture in mind. However, these diagrams can represent overall behavior of Web applications based on

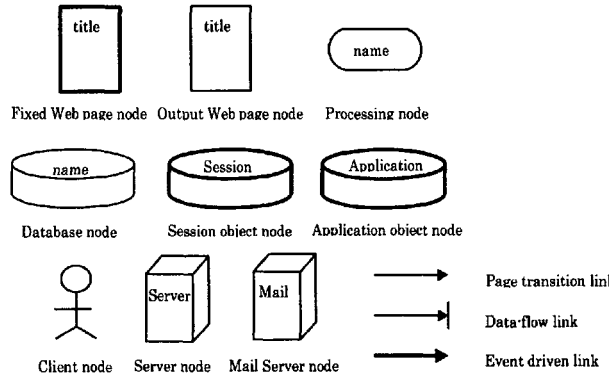


Figure 1: Nodes and links of Web transition diagrams

any server page type architecture, because these diagrams are independent of specific architectures. Our T-Web system can be used by non-programmers or unexperienced engineers to generate server page type Web applications having the standard level of security management without procedural programming.

The organization of the rest of this paper is as follows. We give definitions and notation of Web transition diagrams for server page type Web applications in Section 2. We present T-Web system for server page type Web applications in Section 3. We give an example of Web applications, and compare Web application generator with Microsoft's Visual InterDev in Section 4. We give concluding remarks and our future work in Section 5.

2 Web Transition Diagrams

2.1 Definitions

Web transition diagrams for server page type Web applications are diagrams that can describe overall behavior of general server page type Web applications. We designed Web transition diagrams, considering execution of Web applications as execution of programs based on pipes and filters architecture. These diagrams consist of eight types of nodes and three types of links (Fig.1).

1. Nodes

   The meaning of each node is as follows.

   (a) **A fixed Web page node** represents a static Web page which can be reached by a certain URL. All Web users can see the same contents in a fixed Web page at any time. A title written on the upper side of a square identifies a fixed Web page node. Generally, this node corresponds to a document written in only HTML or XML.

   (b) **An output Web page node** represents a dynamic Web page which is generated by a processing program. According to results of the processing program, a part of this Web page is dynamically changed. A title identifies

an output Web page node. Generally, this node corresponds to a document based on server page type technology such as ASP and JSP.

(c) **A processing node** represents a processing program which can be activated by Web users or server events to perform transactions. A processing program generates an output Web page using input data and software components. A name written in an oval identifies a processing node.

(d) **A database node** represents a relational database table on a database server, or a local file storing data on a Web server. A name written in an oval identifies a database node. A list of names of variables between "{" and "}" represents a schema of data.

(e) **A session object node** represents memory storage on a Web server, which is temporarily assigned to each user for every session. Stored data are thrown away at the end of each session. In ASP and JSP architecture, this node corresponds to Session Objects.

(f) **An application object node** represents memory storage on a Web server which can be accessed by all processing programs. This node keeps common information for all Web users. Data are stored until the termination of the Web application. In ASP and JSP architecture, this node corresponds to Application Objects.

(g) **A client node** represents a user's Web browser which has user's individual information. This node has two roles. One is to keep information of the browser environment and user certification. Another role is to hold cookies.

(h) **A Web server node** represents a Web server application which can raise events affecting a Web application. This node raises events when a Web application starts and terminates, and a session of a user starts and ends.

(i) **A mail server node** represents a smtp server. This node has a role only to send e-mail and not to receive e-mail.

In the rest of this paper, we call both a fixed Web page node and an output Web page node *a Web page node.*

2. Edges

The meaning of each edge is as follows.

(a) **A page transition link** shows a transition from a Web page node to another fixed Web page node with no data flow. A page transition means action of a Web user tracing a link and changing a Web page he looks at.

(b) **A data-flow link** shows a data transaction between two system components. Data flow from the source of an arrow to the destination along the arrow. Parameters are represented as a list of parameter names between "<" and ">".

(c) **An event driven link** shows a transition caused by an event. A type of the event is represented as a parameter between "<" and ">".

3. Page Elements

   Furthermore, a Web page node has additional elements as shown in Fig.2. These fifteen types of page elements are page elements in HTML related to page transitions and data flows between Web components. The meaning of each element is as follows.

   (a) **A page title** represents a name of the Web page.

   (b) **A link to a fixed Web page node** represents a hyperlink. As tracing the link, we can get to another fixed Web page.

   (c) **A link to a processing node** represents a hyperlink to a processing program with no data flow.

   (d) **A link to a processing node with parameters** represents a hyperlink to a processing program with values of parameters.

   (e) **A visible parameter** represents a place in which a value of the variable should be embedded.

   (f) **A database information by a processing program** represents a place in which values of data records should be embedded.

   (g) **A text input** represents a text field which has only a line to input in.

   (h) **A password input** represents a text field which has a line to input a password in. Characters of a password are invisible.

   (i) **A textarea input** represents a text field which has lines to input in.

   (j) **A check box** represents input space to select one or more values from a list.

   (k) **A radio button** represents input space to select only a value from a list.

   (l) **A selection list** represents a scroll menu to select only a value from a list.

   (m) **A hidden parameter** represents a place in which a value of the variable should be embedded. However, the value is invisible on the Web page.

   (n) **A submit button** represents a button to press when a Web user wants to send input data.

   (o) **A reset button** represents a button to press when a Web user wants to delete input data.

2.2 Composition of Web Transition Diagrams

We have ten types of links between nodes as follows.

1. **A page transition link from a link element inside a Web page node to a fixed Web page node** represents tracing a hyperlink. As the link is activated, a Web user gets to a fixed Web page corresponding to the destination of the arrow.

2. **A data-flow link from a link element with parameters or a submit button inside a Web page node to a processing node** represents activation of a processing program. When the link has parameters, it means values of the parameters are passed from a Web page to a processing program.

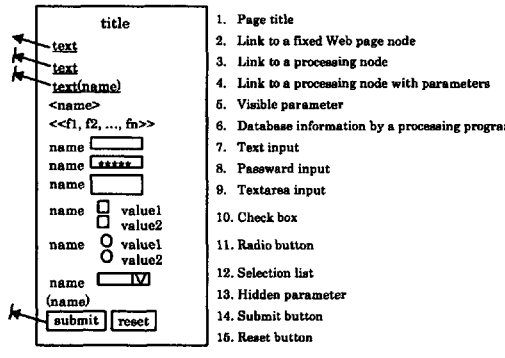


Figure 2: Elements in fixed/output Web page nodes

3. **A data-flow link from a processing node to an output Web page node** represents a generation of an output Web page by a processing program. When a processing node has a branch of links in which one has a label "OK" and the other has a label "error", it means as follows. If all processes are successful, then the link labeled "OK" is activated. If processes go wrong, for example for lack of input data, the link labeled "error" is activated.

4. **A data-flow link between a processing node and a database node** represents writing or reading values of parameters. A link from a processing node to a database node means writing data to a database table or a local file, and the opposite direction means reading values of parameters from a database table or a local file. When a variable among parameters has a mark "*", it means a value of the variable may include a number of records.

5. **A data-flow link between a session object node and a processing node** represents storing or loading values of parameters. A link from a processing node to a session object node means storing data to temporary memory storage, and the opposite direction means loading values of parameters from memory storage. Loading data is allowed only to processes in the same session who stored the data.

6. **A data-flow link between an application object node and a processing node** represents storing or loading values of parameters. A link from a processing node to an application object node means storing data to memory storage, and the opposite direction means loading values of parameters from memory storage. Storing and loading data are allowed to all processing programs.

7. **A data-flow link between a client node and a processing node** represents storing or loading values of parameters. A link from a processing node to a client node represents storing data as client cookies, and the opposite direction means loading values of parameters from cookies.

8. **A data-flow link from processing node to a mail server node** represents sending an e-mail message by a processing program. A receiver address and message contents of e-mail are set by a processing program.

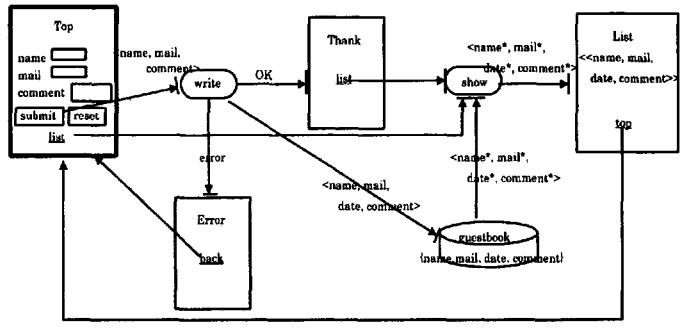


Figure 3: A simple example of Web transition diagrams

9. **An event driven link from a Web server node to a processing node** represents an occurrence of an event. As an event specified in the parameters takes place, the processing program is activated.

10. **An event driven link from a client node to a fixed Web page node or a processing node** represents the start of a session. A link from a client node to a fixed Web page node means Web users should enter from the Web page at the start of each session, and the Web application can make users enter only from the page. A link from a client node to a processing node means that the processing program is activated when the link is activated by an event of the start of a session.

A simple example of Web transition diagrams is shown in Fig.3. This diagram represents the overall behavior of a simple guestbook system. The diagram consists of a fixed Web page node, three output page nodes, two processing nodes, a database node, two page transition links, and eight data-flow links. This diagram represents the following behavior of a Web application. As a Web user inputs data into three text fields of the "Top" page and pushes the submit button, the processing program "write" processes input data (link type 2), writes results into the database table "guestbook" (link type 4), and generates the output page "Thanks" (link type 3). If one or more text fields are blank, the "write" program generates the output page "Error" (link type 3). When a user traces the hyperlink "list" from the "Thank" page (link type 2), the processing program "show" reads data from the database table "guestbook" (link type 4), and generates the output page "List" (link type 3). When a user traces the hyperlink "back" from the "Error" page or the hyperlink "top" from the "List" page, the user can get to the "Top" page.

Web transition diagrams are intuitively so clear that even non-programmers and unexperienced engineers can understand and compose diagrams easily.

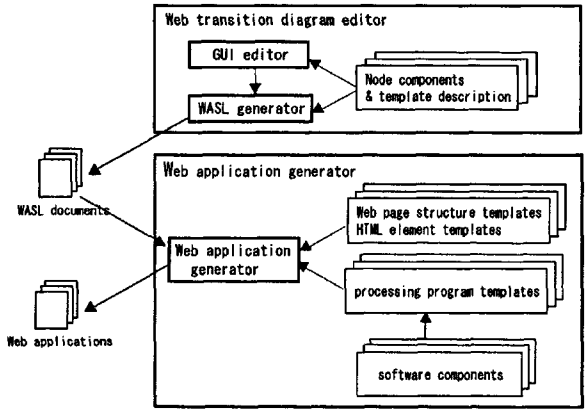


Figure 4: Structure of T-Web system for server page type Web applications

3 T-Web System

3.1 System Structure

T-Web system for server page type Web applications is a system to generate server page type Web applications from Web transition diagrams automatically. As a way to generate actual Web applications, we adopt a template method. We prepare templates, some of them include program codes, as a part of our system. Selecting appropriate templates and setting parameters, developers can produce Web applicaitons without writing even a line of program codes. Thus, even non-programmers and unexperienced engineers can generate server page type Web applications easily by using our system.

Our system consists of two parts: a Web transition diagram editor and a Web application generator as shown in Fig.4. To generate Web applications from diagrams, diagrams should have consistency. To support composition of consistent Web transition diagrams, we present a Web transition diagram editor as a part of our T-Web system. By using a GUI editor, we can compose Web transition diagrams selecting templates and setting parameters visually. After we complete composing diagrams, the editor generates documents written in an intermidiate language called WASL, which we designed to describe specification of Web applicaitons. A Web appliation generator generates server page type Web applications from WASL documents, using templates which are prepared as a part of a generator. A Web appliation generator is separated from a Web transition diagram editor and depends on only the definition of WASL.

3.2 Intermediate Language WASL

WASL (Web Application Specification Markup Language) is a language based on XML, which we designed to describe specification of server page type Web applications. The definition of WASL is shown in Fig.5 as a DTD (Document Type Definition). WASL documents describe components appearing in a Web application and relationships between components. Components such as Web pages, processing programs, server storages and client storages are specified by a name, a type of the component, its version

```
<!ELEMENT web(component+,connection*)>
<!ELEMENT component(property,reference?,input?,output?)>
<!ELEMENT connection(link+,reference*)>

<!ELEMENT link(output,input)>
<!ELEMENT property(name,type,version)>
<!ELEMENT reference((port+|(object*,process*)))>
<!ELEMENT input((name,type)*,port)>
<!ELEMENT output((name,type)*,port)>
<!ELEMENT port((name,type,(io|argument))?)>
<!ELEMENT object(name,type)>
<!ELEMENT process(name,type,port)>
<!ELEMENT argument(name+)>

<!ELEMENT name(#PCDATA)>
<!ELEMENT type(#PCDATA)>
<!ELEMENT version(#PCDATA)>
<!ELEMENT io(#PCDATA)>
```

Figure 5: DTD for WASL documents

and an interface which represents types of input/output data. WASL is independent of specific architectures, and can describe essential behavior of Web applications.

3.3 Web Transition Diagram Editor

A Web transition diagram editor is a part of T-Web system to support composition of consistent Web transition diagrams, and generates WASL documents. Furthermore, the editor consists of two parts: a GUI editor to compose Web transition diagrams and a WASL generator to generate WASL documents from Web transition diagrams.

3.3.1 GUI Editor

A GUI editor is a part of a Web transition diagram editor to construct diagrams, select templates and set parameters visually. A process to compose new Web transition diagrams is as follows.

1. **Putting nodes on drawing fields**
   First, we put nodes which are necessary to represent the behavior of a Web application on drawing fields of the editor. We can select a node to put on from a list of nodes and put a node on any places we like in drawing fields, using a pointing device.

2. **Specifying details of nodes**
   As we select a node, a window having input and selection fields appears. We call this window *a property window*. We can specify details of a node from a property window, for example a title of a Web page, a schema of a database table and a name of variables in a session object. We can also set page elements in a Web page node from a property window. Forms of property windows are arranged for each type of nodes, so eight types of property windows are prepared.

3. **Giving relationships between nodes**
   As we give a relationship between two nodes from a property window, an arrow appeares on a drawing field. Usually, we can select only a type of links defined in section 2.2 in a property window, and we select the other node related to a link from a list. Because of this method, consistency of composed diagrams is maintained.

4. **Selecting processing program templates and setting parameters**
   In a property window of a processing node, there is a list of names of processing program templates and their summaries. Reading summaries, we can select an appropriate processing program template applied to a processing node from a list. After selecting a template, we also set parameters of the template from a property window, for example names of columns which a processing program reads from a database table.

5. **Setting global parameters**
   Finally, we set global parameters necessary to execute the Web application, which are a base URL of the Web application, a title of the system, a name of a database server, a name of a database and a name of a smtp server.

Furthermore, to compose consistent Web transition diagrams easily, we designed a GUI editor so that we can simulate dynamic data flows of a Web application. Each node is self-controlled to check consistency with adjacent nodes. As we select a node, possible links to transit from the node are activated and color of the arrows is changed on the editor.

The previous version of T-Web system has a problem as follows. A Web transition diagram becomes more complicated and its readability becomes poor, as the scale of a Web application becomes larger. We designed a Web transition diagram editor so that we can compose smaller Web transition diagrams for each function, and then construct a larger Web transition diagram by stratifying these diagrams. As a consequence, a whole Web transition diagram has a tree structure. We call the children diagrams in a tree structure *sub Web transition diagrams*, and a diagram having no parent diagram *a top Web transition diagram.*

We add three types of nodes for Web transition diagrams: a starting node, an ending node and a folder node. A starting node and an ending node are represented by the same shapes of nodes used in a state chart of UML. A folder node is represented by the shape of a folder. Each sub Web transition diagram needs to have a starting node and one or more ending nodes, while a top Web transition diagram needs not necessarily to have a starting node and ending nodes. To connect a parent diagram with a number of children diagrams, the parent diagram needs to have folder nodes. When a folder node is the destination of an arrow and the link is activated, it means an execution flow moves to a sub Web transition diagram related to the folder node and the execution starts from a starting node of the sub diagram. When an execution flow comes to an ending node, the flow moves to the parent diagram and the execution restarts from the folder node where the execution flow comes in.

3.3.2 WASL Generator

A WASL generator is a part of a Web transition diagram editor. It takes complete Web transition diagrams and global parameters as input and generates WASL documents as

output. If Web transition diagrams are stratified, WASL documents are generated as divided files for each sub diagram. As a consequence, we can flexibly replace a part of Web transition diagrams and reuse them.

First, a WASL generator writes description of nodes as XML documents according to the DTD of WASL. The description consist of parameters which are set from property windows, and information which is preset in a node such as a node type and a version of the node. Then, the generator writes description of relationships between nodes. Usually, input/output port names in WASL documents are assigned automatically to connect two nodes. Finally, the global parameters are written to WASL documents.

3.4 Web Application Generator

A Web application generator takes WASL documents as input and generates necessary components for execution of a Web application as output. A document of a Web page is generated by applying given values of parameters to a selected processing program template, and constructing a Web page from the template and Web page structure templates such as a text field. The order of templates in a Web page is as given in Web transition diagrams.

There is a sufficient number of processing program templates provided as a part of a Web application generator. Most of templates are templates concentrating on data transactions using database query commands such as `INSERT`, `DELETE`, `UPDATE` and `SELECT`. The others are templates concentrating on other necessary business processes such as sending an e-mail message. Useful software components such as COM (Component Object Model) and JavaBeans components are used in templates. However, developers need only to know summaries and parameters of templates and don't need to know which components are required to achieve the function, because software components are encapsulated by a template.

The language to describe templates and types of output Web components are dependent on architectures the generator supports. For example, a generator for ASP applications with XML has templates described in XML, and generates ASP documents described in XML, XSL (eXtensible Style Language) files, a CSS (Cascading Style Sheet) file and a DTD file. On the other hand, a generator for ASP architecture with HTML has templates described in HTML, and generates ASP files described in HTML and a CSS file.

The standard level of security management such as checking restricted length of input parameters is provided. Using components prepared in T-Web system, only acceptable values of parameters can be passed to server pages. For session management, all templates have codes to check conditions of input values of parameters before performing main processes. Condition checking is a part of session management. A session can be also manually managed using general methods provided in a Web transition diagram editor, such as cookies.

3.5 Flexibility

We can add a new processing program template to T-Web system at anytime. If a user of T-Web system can not find an appropriate processing program template to design a Web application, there are two ways for developers to make a new processing program template:

1. Writing whole codes of a new template, and

2. Reusing existing software components and processing program templates, and construcing a template.

In the former way, developers have to know about the architecture of which he wants to generate Web applications, and needs to be able to write program codes. In the latter way, developers may be able to make a new template writing very small program codes. As well as reusing software components existing templates use, we can handle a part of a processing program template as a software component such as a scriptlet and reuse it in a new template.

We can replace a Web application generator of T-Web system with another generator, which is independent of a Web transition diagram editor and dependent on only the definition of WASL. As a consequence, we can generate Web applications based on various architectures using T-Web system. For example, we can generate Web applications based on two different architectures, ASP and JSP, respectively from the same WASL documents by using generators for ASP architecture and for JSP architecture. We believe that our T-Web system can genenerate Web appliations based on any server page type architecture. This shows our system has very high flexibility.

4 Evaluation

4.1 An Example

The example diagram in Fig.6 shows overall behavior of an online shopping system. The system allows a Web user to see a list of goods, put goods into a shopping cart and pay for the goods. A Web user should enter the top page "menu" at the start of a session. A Web user can select a category of goods, "book" or "computer", in the page "menu" and see a list of goods contained in the category in the page "list". A user can put goods into a shopping cart, whose data are stored in a session object. A user can confirm contents of his shopping cart and change quantity of them in the page "change". When a user pays for goods, he has to input his private information in the page "account". If all processes of payment are successful, the page "thanks" appears.

This diagram consists of a fixed Web page node, six output Web page nodes, six processing nodes, three database nodes, a session object node, a client node, four page transition links, twenty eight data-flow links, and a event driven link. This diagram uses five types of processing program templates. The processing node "show" has a function to select data of goods in a given category from the database table "pmaster" and show a list of them. The processing node "write1" has a function to store data of selected goods into a session object. The processing node "write2" has a function to store data into the database table "master" which has information about decided orders and the database table "person" which has information of customers. The processing node "delete" has a function to delete a number of data in a shopping cart. The processing node "showWithComb1" and "showWithComb2" have a function to take data of goods and a shopping cart as input and generate a list of goods which exist in a shopping cart as output.

When we generate WASL documents from this diagram, and generate Web applications using a Web application generater for ASP architecture with XML and a generator for JSP architecture with HTML respectively, we get Web components for ASP architecture and JSP architecture as shown in Table.1.

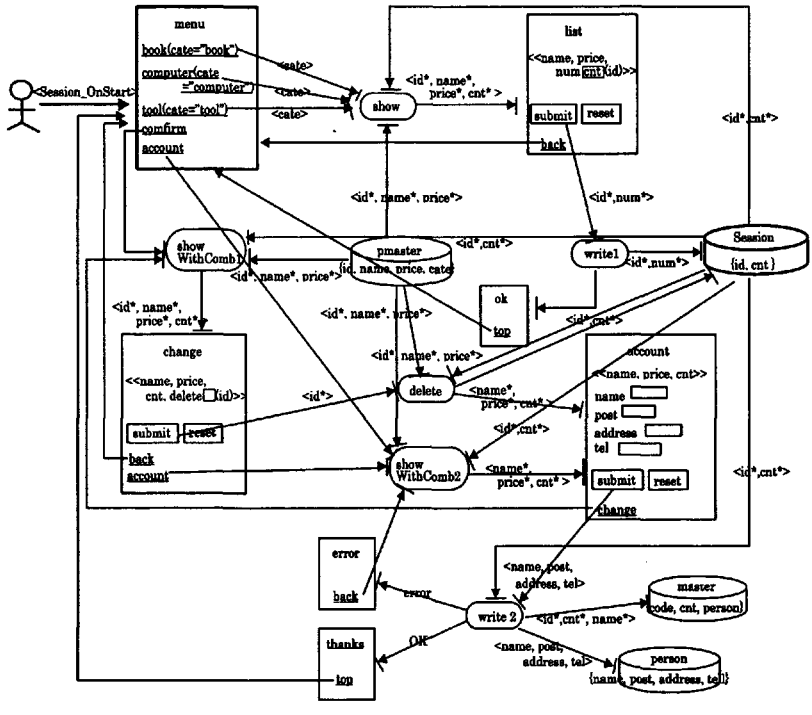


Figure 6: Web transition diagram of an online shopping system

4.2 Comparisons

A standard Web application generation method is manual writing of processing programs and composing Web pages using a Web page composer [9]. Web page composers can help developers visually compose Web pages and associate Web pages with processing programs. However, they cannot be used to develop any processing program and take care consistency between Web pages and processing programs. A number of development environments may be helpful for the above problems. In this section, we compare our system with Microsoft's Visual InterDev which is a widely used Integrated Development Environment.

Visual InterDev is a development environment to help developers generate Web applications based on ASP architecture. Table 2 overviews comparison results of our system with Visual InterDev at main features. Visual InterDev has facility to describe a site diagram which represents relationships between Web pages. It is helpful for developers to implement and maintain Web applications. However, it does not represent data flows or databases, while Web transition diagrams do. Visual InterDev also helps developers write processing programs, using components such as ActiveX controls plugged in the environment. However, it does not generate whole Web applications automatically, while T-Web system does from Web transition diagrams. Visual InterDev has a visual composer which enables developers to visually compose graphical Web pages. T-Web system does not support high level graphical interfaces. But we can compose high level

Table 1: Example of components that will be generated by T-Web system

	ASP with XML	JSP with HTML
Static page files	menu.xml	menu.html
Dynamic page files	list.asp, change.asp account.asp, ok.asp error.asp, thanks.asp	list.jsp, change.jsp account.jsp, ok.jsp error.jsp, thanks.jsp
Style files	ec.css, menu.xsl, list.xsl, change.xsl account.xsl, ok.xsl error.xsl. thanks.xsl	ec.css
Other files	COM components, ec.dtd	JavaBeans components

Table 2: Summary of Comparisons of T-Web system with Visual InterDev

	Visual InterDev	T-Web system
Lifecycle coverage	Implementation, maintenance	Design, implementation
Automation	Generation of HTML	Generation of HTML or XML, processing programs, database connection
Reuse	Plug-in components	Diagrams, page templates, components
Architecture	3-tiers, dynamic	3-tiers, dynamic
Support to usability	Canned interfaces, low customization	Low graphical interfaces

interfaces by using other tools such as a visual composer for documents generated by T-Web system.

5 Conclusions

We extended the model for server program type Web applications to represent overall behavior of server page type Web applications, and designed T-Web system as a tool for Web transition diagram composition and server page type Web application generation.

We believe that our Web transition diagrams can represent Web applications based on any server page type technologies. Our T-Web system can generate Web applications based on any server page type technology, by replacing a Web application generator of T-Web system with another generator, which is independent of a Web transition diagram editor. Web applications based on non-sever page type technologies may be also generated from our Web transition diagrams by a similar method.

We have implemented basic parts of T-Web system for server page type Web applications. Our current diagram may not support atomic transactions consisting of two or more actions. It will be our future work to handle such transactions and to improve overall consistency and security of Web applications.

References

[1] Shinshir Gundavaram. CGI Programming on the World Wide Web. O'Reilly & Associates, 2000.

[2] Marty Hall. Core Servlets and JavaServer Pages. Prentice Hall, 2000.

[3] James Gosling, Bill Joy, Guy Steele, Gilad Bracha. The Java Language Specification. Addison-Wesley, Massachusetts, 2000

[4] Alex Homer. Dave Sussman and Brian Francis and others. Professional Active Server Pages 3.0. Wrox Press, 1998.

[5] K. Jamroendararasame, T. Suzuki and T. Tokuda, A Generator of Web-based Transaction Systems Using Web Transition Diagrams. In *Proc. 17th Japan Society for Software Science and Technology*, Tokyo, 2000

[6] K.Jamroendararasame, T. Matsuzaki, T.Suzuki, T.Tokuda. Generation of Secure Web Applications from Web Transition Diagrams. Proceedings of the IASTED International Symposia Applied Informatics, pages 496-501, 2001.

[7] K.Jamroendararasame, T. Matsuzaki, T.Suzuki, T.Tokuda. Two Generators of Secure Web-Based Transacion Systems. Proceedings of the 11th European-Japanese Conference on Information Modelling and Knowledge Bases, pages 348-362, 2001.

[8] K.Jamroendararasame, T.Suzuki, T.Tokuda. JSP/Servlet-Based Web Application Generator. 18th Conference Proceedings Japan Society for Software Science and Technology, 2C–1, 2001.

[9] Piero Fraternali. Tools and Approaches for Developing Data-Intensive Web Applications: A Survey. ACM Computing Surveys, Vol. 31, No. 3, pages 227-263, 1999.

[10] Jim Conallen. Building Web Application with UML. Addison-Wesly, 2000.

Information Modelling and Knowledge Bases XIV
H. Jaakkola et al. (Eds.)
IOS Press, 2003

Unifying Various Knowledge Discovery Systems in Logic of Discovery

Toshiyuki KIKUCHI[1] Akihiro YAMAMOTO[1,2]

[1]MemeMedia Laboratory,
Hokkaido University
N 13 W 8, Sapporo 060-8628 JAPAN
Email : {kikuchi, yamamoto}@meme.hokudai.ac.jp

[2]"Information and Human Activity", PRESTO
Japan Science and Technology Corporation (JST)

Abstract. The main process of Knowledge Discovery from Databases (KDD) is to generate general rules (hypotheses) from given data. When different KDD systems generate different hypotheses from a same dataset, we have to compare them. The system EVLD, which we are proposing in this paper, enables us to apply various KDD systems to one dataset simultaneously, and to obtain various hypotheses from it. The system EVLD is designed according to the three-level architecture, with adopting Plotkin's logic of discovery for the description of its conceptual level. It thanks to the architecture that each of the hypotheses generated in EVLD is represented in a common format and therefore comparable. By this property we can say that EVLD is a tool for regarding every hypothesis as a view to data.

1 Introduction

Today academic research is driven by data. Researchers collect more and more data, store them in database systems, and apply various software systems to them in order to discover new academic knowledge. Some of such software systems, which we call *KDD systems* or *discovery systems*, have been developed in the area of Machine Learning (ML) and Knowledge Discovery from Databases (KDD). Every output of a KDD system from a dataset is a general rule. It is also called a hypothesis, because it varies from systems to systems. Sometimes researchers need to compare hypotheses generated by some different KDD systems from a same dataset. For such requirement we propose a system named EVLD (Environment for Various systems in Logic of Discovery), which enables us to apply various KDD systems to one dataset simultaneously, and to obtain various hypotheses from the systems.

In order to compare several hypotheses, all of them must be represented in a common format. In this research we treat data stored in a relational database, but readers must notice that this restriction is not equivalent to the fact that every hypothesis from one dataset is represented in a unique format. For example, some KDD systems generate hypotheses in the form of decision trees [9], some generate association rules [1], and some generate logic programs [6, 10], and so on. Note that every hypothesis in the formats listed above can be transformed into a logic program. Moreover, Plotkin, one of the frontiers of KDD, gave a philosophical foundation to KDD by formalizing every

datum and knowledge in first-order logic [8], and called it *logic of discovery*. From these reasons, we adopt logic programs in EVLD, as a common format of hypotheses generated by various KDD systems.

Our system EVLD is designed according to the three-level architecture where the logic of discovery is the description of its conceptual level. The role of the logic of discovery in EVLD can be explained with comparing it to that of the relational algebra in the relational database theory. The relational algebra gives mathematical foundations not only to representing every datum in a uniform manner but also to creating useful views to data. Analogously, the logic of discovery gives a mathematical background not only to representing hypotheses by various KDD systems in a uniform format but also to assuring the proposal by Maruyama et al. [4, 5] that every hypothesis is a view to data.

The logic of discovery consists of two parts, declarative and procedural. The declarative part defines the space of searching by preparing predicate symbols and representing data and background theories in logic programs. The procedural part consists of mechanisms deriving appropriate hypotheses from data. A user of EVLD declares predicate symbols and how to represent every dataset in a logic program for the logic of suggestion. He/She also chooses KDD systems supported by EVLD in order to realize the logic of induction.

In the following readers are assumed to have fundamental terminology and concepts on logic programming, inductive logic programming [7] and relational database [12]. In Section 2 we give a formal definition of KDD systems in terms of the logic of discovery. We explain the architecture of the system EVLD in Section 3, and how to build existing KDD systems into EVLD in Section 4. In Section 5 we give a sample application of EVLD. We conclude the paper with some remarks on our research in Section 6.

2 Formal Definition of KDD Systems

The logic of discovery given by Plotkin [8] formalizes discovery as solving hypothesis finding problems. We define the hypothesis finding problems and KDD systems in terms of logic programming.

Definition 1 ([2]) An instance of the *hypothesis finding problem* is a triplet (B, F, V) where

- $F = \{f_1, f_2, \ldots, f_n\}$ is a set of ground atoms each of which represents a *fact*,
- $V = \{v_1, v_2, \ldots, v_n\}$ is a set of conjunctions of ground atoms such that v_i represents an *evidence* relevant to f_i, and
- B is a definite program representing a *background theory* relevant to all of the f_i's and v_i's.

We put $E = \{f_1 \leftarrow v_1, \ldots, f_n \leftarrow v_n\}$, and assume that $B \not\models E$. The instance is denoted by $HFP(B, F, V)$. A solution to $HFP(B, F, V)$ is given by any logic program H such that $B \cup H \models E$. A *discovery system* or a *KDD system* S is a system which takes F, V and B as inputs and generates solutions of $HFP(B, F, V)$.

We are aiming to knowledge discovery from relational databases, but the definition above does not give any methods how to create F and V from a relation. As is illustrated in the following two examples, we have to consider two methods.

ID	Class	Size	Color	Animal
a	dangerous	small	black	bear
b	dangerous	medium	black	bear
c	dangerous	large	black	dog
d	safe	small	black	cat
e	safe	medium	black	horse
f	dangerous	large	black	horse
g	dangerous	large	brown	horse

Figure 1: A relation R_{animal} in a relational database

Example 1 ([8]) Let us consider a relation R_{animal} in a relational database depicted in Figure 1. From the first tuple we define

$$\begin{aligned} f_1 &= class(a, dangerous), \text{ and} \\ v_1 &= size(a, small) \wedge color(a, black) \wedge animal(a, bear). \end{aligned}$$

In the same way we define f_i and v_i from the i-th tuple in the relation as follows:

$$\begin{aligned} f_2 &= class(b, dangerous), \\ v_2 &= size(b, medium) \wedge color(b, black) \wedge animal(b, bear), \\ f_3 &= class(c, dangerous), \\ v_3 &= size(c, large) \wedge color(c, black) \wedge animal(c, dog), \\ &\ldots \end{aligned}$$

Let $F_{animal} = \{f_1, f_2, \ldots, f_n\}$ and $V_{animal} = \{v_1, v_2, \ldots, v_n\}$. An example of solutions of $HFP(\emptyset, F_{animal}, V_{animal})$ is a logic program.

$$H_{animal} = \left\{ \begin{array}{l} class(X, dangerous) \leftarrow size(X, small), color(X, black), animal(X, bear) \\ class(X, dangerous) \leftarrow size(X, large), color(X, Y), animal(X, Z) \\ class(d, safe) \leftarrow size(d, small), color(d, black), animal(d, cat) \\ class(e, safe) \leftarrow size(e, medium), color(e, black), animal(e, horse) \end{array} \right\}.$$

The relation R_{animal} represents seven animals, and every hypothesis derived by KDD systems is expected to classify the animals w.r.t. the attribute *class*. In the terminology of the entity-relationship model [12], we intend that every tuple of the relation R_{animal} represents an entity with a list of attributes, and a KDD system is fed a list of entities. The system is expected to output some classification rules for the entities represented by some attributes in the list. We say that such a KDD system is *attribute-oriented*. An example of attribute-oriented systems is C4.5 [9].

Example 2 ([2]) Consider the relations R_{parent}, R_{male} and R_{gf} in Figure 2. We represent each of relations in a logic program:

$$\begin{aligned} DB_{parent} &= \left\{ \begin{array}{ll} p(a, b) \leftarrow, & p(b, c) \leftarrow \\ p(d, e) \leftarrow, & p(e, f) \leftarrow \end{array} \right\}, \\ DB_{gender} &= \left\{ \begin{array}{l} m(a) \leftarrow \\ m(b) \leftarrow \end{array} \right\}, \\ DB_{gf} &= \left\{ \begin{array}{l} gf(a, c) \leftarrow \\ gf(d, f) \leftarrow \end{array} \right\}. \end{aligned}$$

Parent	Child
a	b
b	c
d	e
e	f

Male
a
d

Gfather	Gson-and-Gdaughter
a	c
d	f

Figure 2: Relations R_{parent}, R_{male}, and R_{gf}

The predicate symbols p, m, f, and gf represent relations of is_a_parent_of, is_male, is_the_father, and is_a_grandfather, respectively. We also consider a logic program

$$R_{father} = \{f(X,Y) \leftarrow p(X,Y), m(X)\}.$$

When we let a background theory $B_{gf} = DB_{parent} \cup DB_{gender} \cup R_{father}$ and $F_{gf} = DB_{gf}$, a logic program

$$H_{gf} = \{gf(X,Y) \leftarrow f(X,Z), p(Z,Y)\}$$

is an example of the correct hypotheses of $HFP(B_{gf}, F_{gf}, \emptyset)$.

The relation R_{parent} is regarded as a part of instance of a relationship *parent*, and KDD systems are expected to generate a general definition of the relationship. We say that such KDD system is *relationship-oriented.* Such systems as Progol [6] and FOIL [10] are relationship-oriented. Note that relationship-oriented discovery systems can be used for attribute-oriented discovery if we prepare some proper predicate symbols, but the converse does not hold in general.

3 Unifying KDD Systems

3.1 Overview of EVLD

The structure of our system EVLD (Environment for Various systems in Logic of Discovery) is illustrated in Figure 3.

We assume that users of EVLD are aiming to discover rules from relations stored in a relational database DB. Such a user proceed to their discovery along the following steps:

Step 1 : The user must decide whether they like attribute-oriented discovery or relationship-oriented discovery.

Step 2 : Next the user selects some KDD systems from a list of KDD systems available in EVLD.

Step 3 : Then EVLD displays the list of relations in DB, and the list of attributes for each relation. The user selects one relation, and creates some sets of attributes and assigns a predicate symbol for each of the sets. He/She also defines which predicates are for facts and which are for evidences. EVLD provides these operations by displaying wizard-style menus. The user can choose some rules in RB in order to use them as a background theory B.

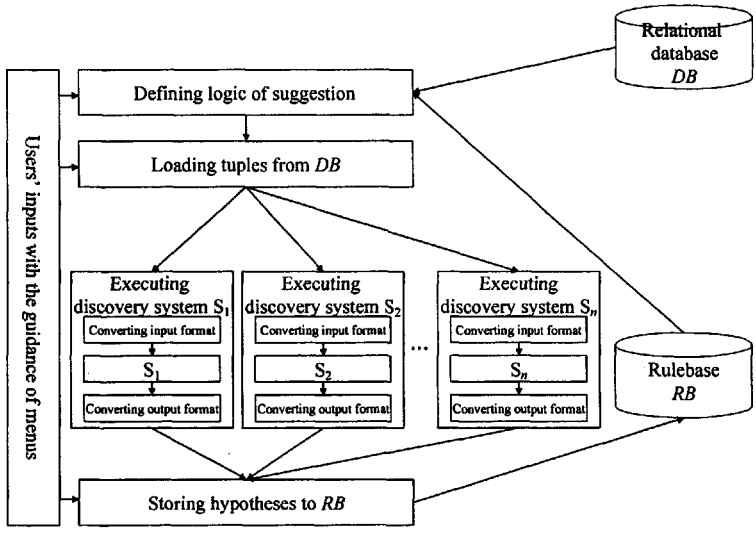


Figure 3: The Structure of EVLD

Step 4 : By pressing a start button the user requests all of the KDD systems to generate hypotheses. Then EVLD generates relations from DB with SQL queries, and feeds them to the selected KDD systems as their inputs.

Step 5 : After each KDD system outputs a rule in the form of a logic program using the predicate symbols prepared on Step 3, EVLD displays the rules to the user so that one of the rule can be compared with another. The user can store some of the obtained rules in RB for the use of the following round of discovery.

For example, by using EVLD to apply C4.5, Progol, and FOIL to the relation R_{animal} in Example 1 , the user will obtain three hypotheses in the form of the following logic programs:

$$H_{C4.5} = \left\{ \begin{array}{l} safe(X) \leftarrow horse(X), medium(X) \\ safe(X) \leftarrow horse(X), small(X) \\ safe(X) \leftarrow cat(X) \\ dangerous(X) \leftarrow horse(X), large(X) \\ dangerous(X) \leftarrow dog(X) \\ dangerous(X) \leftarrow bear(X) \end{array} \right\},$$

$$H_{Progol} = \left\{ \begin{array}{l} safe(X) \leftarrow cat(X) \\ dangerous(X) \leftarrow large(X) \\ dangerous(X) \leftarrow bear(X) \end{array} \right\}, \text{ and}$$

$$H_{FOIL} = \left\{ \begin{array}{l} safe(X) \leftarrow cat(X) \\ safe(X) \leftarrow horse(X), medium(X) \\ safe(X) \leftarrow horse(X), small(X) \\ dangerous(X) \leftarrow large(X) \\ dangerous(X) \leftarrow bear(X) \end{array} \right\}.$$

The hypotheses $H_{C4.5}$, H_{Progol}, and H_{FOIL} are generated by C4.5, Progol and FOIL, respectively. Predicates in the hypotheses are not the same ones in Example 1 . The difference will be explained in the next subsection.

3.2 Defining Predicate Symbols

In order to represent every tuple obtained from DB as a ground atom, users have to prepare predicate symbols. On Step 3 they are allowed to assign any predicate symbol to a set S of attributes, but one predicate symbol might be used for two different attributes. In order to avoid such confliction EVLD provides a unique predicate symbol p_Q for every SQL query Q used in Step 4. The predicate is called the *standard predicate* for Q. Every predicate symbol assigned by users is treated as an *alias* of the standard predicate. More precisely, if a user created a set $S = \{A_1, \ldots, A_n\}$ of attributes of a relation R and assigned a predicate symbol p to S, the symbol is an alias of the standard predicate symbol $p_{\text{SELECT } A_1,\ldots,A_n \text{ FROM } R}$ because an SQL query

$$\text{SELECT } A_1, \ldots, A_n \text{ FROM } R$$

is used on Step 4. In EVLD the standard predicate is displayed as

$$R(A_1, \ldots, A_n)$$

because it appears so often.

Example 3 Consider the relation in Figure 1 . The set obtained with a query

$$\text{SELECT ID, Class FROM } R_{animal}$$

contains a tuple $(a, dangerous)$. With the standard predicate for the query the tuple is regarded as a ground atom

$$p_{\text{SELECT ID,Class FROM } R_{animal}}(a, dangerous).$$

and is sometimes displayed as

$$R_{animal}(\text{ID, Class})(a, dangerous).$$

If a user declare an alias *class* for the standard predicate, the tuple is regarded as $class(a, dangerous)$ in Example 1 .

In the logic of discovery we have to distinguish facts and evidences. Our system assumes that they are distinguished with predicate symbols. That is, users must declare each predicate symbol whether it should be used for facts or for evidences. This declaration is similar to that of two modes *modeh* and *modeb* in Progol[6].

From the logical point of view, attribute-oriented discovery needs some constraints on designing predicate symbols. The constraints are given as follows.

Definition 2 ([11]) A relational schema S is *for classification of entities* if it consists of

1. one attribute name for the *key*,
2. one attribute name for *classes* of entities, and

3. several attribute names for *explanations* of entities.

In the following, we assume that the key attribute name, the class attribute name are respectively ID and A_0, and that A_j $(j = 1, 2, \ldots, n)$ are for attribute names for explanations.

If we assume every value of each A_j $(j = 0, 1, \ldots, n)$ is an element of a finite set D_j, two types of assignment of predicates can be considered in attribute-oriented discovery. Let R be an instance of the schema S.

Type 1 : Users declare the predicate $R(ID, A_0)$ for facts and $R(ID, A_1)$, ..., $R(ID, A_n)$ for evidence. That is, if $R = \{(i, a_{i0}, a_{i0}, \ldots, a_{in}) \mid i = 1, 2, \ldots, k\}$, then the set of facts and evidences are respectively

$$F_R = \{R(ID, A_0)(i, a_{i0}) \mid i = 1, 2, \ldots, k\}, \text{ and}$$
$$V_R = \{R(ID, A_1)(i, a_{i1}) \wedge \ldots \wedge R(ID, A_n)(i, a_{in}) \mid i = 1, 2, \ldots, k\}.$$

We may use alias of the predicate symbols provided by the system.

Type 2 : Users declare a predicate symbol p_{jh} for each value v_{jh} in D_j $(j = 0, 1, \ldots, n, h = 1, 2, \ldots, \sharp(D_j))$ of one arguments. They declare p_{0k}'s are for facts and the rests are for evidences. For the relation R above, the sets of facts and evidences are respectively

$$F_R = \bigcup_{i=1}^{k} \{p_{0h_i}(i) \mid a_{i0} = v_{0h_i}\}, \text{ and}$$
$$V_R = \bigcup_{i=1}^{k} \{p_{1h_1}(i) \wedge \ldots \wedge p_{nh_n}(i) \mid a_{i1} = v_{ih_1}, \ldots, \text{ and, } a_{in} = v_{nh_n}\}.$$

Example 4 The facts and evidences in Example 1 are defined with predicate symbols of Type 1, with *class*, *color*, *size*, and *animal* are aliases for the standard predicates. Facts and evidences with predicate symbols of Type 2 are the followings:

$$\begin{aligned} f_1 &= dangerous(a), \\ v_1 &= small(a) \wedge black(a) \wedge bear(a), \\ f_2 &= dangerous(b), \\ v_2 &= middle(b) \wedge black(b) \wedge bear(b), \\ &\ldots \end{aligned}$$

4 Building KDD Systems into EVLD

EVLD gives sets of tuples obtained from the database DB to KDD systems, while it receives hypotheses in the form of logic programs. The system FOIL can easily be built into EVLD because it takes sets of tuples as its inputs and outputs logic programs. When we connect such a KDD system to EVLD that its inputs must be ground atoms (e.g., Progol) and its outputs are logic programs, we give a wrapper to the system. The wrapper translates every tuple into a ground atom using the predicate symbols which users have prepared.

If KDD systems such as FOIL and Progol require negative examples require, users generate them either by creating a relation explicitly with declaring that every tuple in it is for negative examples, or by assuming the Closed World Assumption.

For a KDD system whose outputs are not logic programs (e.g., C4.5) we also have to prepare some wrapper for it. As an example of such wrapping, we explain the transformation of decision tree into a logic program. Let us assume a logic of suggestion for attribute-oriented systems and Type 1 of the assignment of predicates.

Definition 3 A tree T is a *decision tree* for classification of entities represented in a relation R w.r.t. an attribute A_0 if it satisfies the following conditions:

- Every node N of T which is not a leaf is labeled with an attribute name in $\{A_1, \ldots, A_n\}$.
- For every path N_0, N_1,...,L where N_0 is the root and L is a leaf, $A_{j(N_h)} \neq A_{j(N_{h'})}$ if $h \neq h'$.
- If a node is labeled with an attribute $A_{j(N)}$, each edge E^N_M from N to a child M is labeled with a subset of $D_{j(N)}$. If $M_1, \ldots, M_k$ are the children of N, the sets $S^N_{M_1}, \ldots, S^N_{M_k}$ are mutually distinct and $S^N_{M_1} \cup \ldots \cup S^N_{M_k} = D_{j(N)}$.
- Every leaf L of T is labeled with a subset S_L of D_0.

The decision tree T *explains* the relation R if for every path P from the root N_0 of D to a leaf L and for every tuple $(k, a_{k0}, a_{k1}, \ldots, a_{kn})$, $a_{kj(N_h)} \in S^{N_h}_{N_{h+1}}$ $(h = 0, 1, \ldots, l)$ implies $a_{k0} \in S_L$ where $N_0, N_1, \ldots, N_l, L$ are all the nodes on P.

A decision tree can be represented as a logic program even if either type of representation is adopted.

Definition 4 For every leaf L we define

$$H_L = \left\{ p_0(X, a) \leftarrow p_{j(N_0)}(X, a_0), p_{j(N_1)}(X, a_1), \ldots, p_{j(N_l)}(X, a_l) \;\middle|\; \begin{array}{l} a \in S_L, a_h \in S^{N_h}_{N_{h+1}} \\ (h = 0, 1, ..., l) \end{array} \right\},$$

where the path from the root to L is $N_0, N_1, \ldots, N_l, L$. Moreover, if $L_1, \ldots, L_n$ are the leaves of D, we put $H_T = H_{L_1} \cup \ldots \cup H_{L_n}$.

Proposition 1 *If a decision tree T explains R, H_T is a solution of $HFP(\emptyset, F_R, V_R)$.*

Example 5 In the table illustrated in Figure 1 , a decision tree w.r.t. the attribute *class* is illustrated in Figure 4. The tree is represented as a logic program

$$H_{C4.5} = \left\{ \begin{array}{l} class(X, safe) \leftarrow animal(X, horse), size(X, medium) \\ class(X, safe) \leftarrow animal(X, horse), size(X, small) \\ class(X, safe) \leftarrow animal(X, cat) \\ class(X, dangerous) \leftarrow animal(X, horse), size(X, large) \\ class(X, dangerous) \leftarrow animal(X, dog) \\ class(X, dangerous) \leftarrow animal(X, bear) \end{array} \right\}.$$

5 A Simple Application of EVLD

As a simple example of applications of EVLD we developed a web service ANIMAL. We illustrate the service in Figure 5. The service displays the results (hypotheses) of applying C4.5, Progol, and FOIL to the relation R_{animal}. It represents each of hypotheses $H_{C4.5}$, H_{Progol}, and H_{FOIL} not in the form of a logic program, but in the form of conclusions derived from each hypothesis and the premises which users input. In the first page of the service (Figure 5 (a)), users select premises for all of three hypotheses, and the next page shows the conclusion derived by each hypothesis from the premises. For example, if a user selects that the intended animal is a cat, its size is large, and its color is brown, the second page displays the conclusions derived with $H_{C4.5}$, H_{Progol}, and H_{FOIL}. Because EVLD adopts the logic of discovery as conceptual model, all that the designer of the web page must do is connecting every predicate symbol to an operation on web pages.

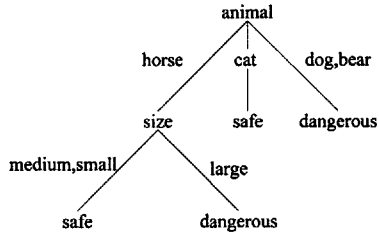
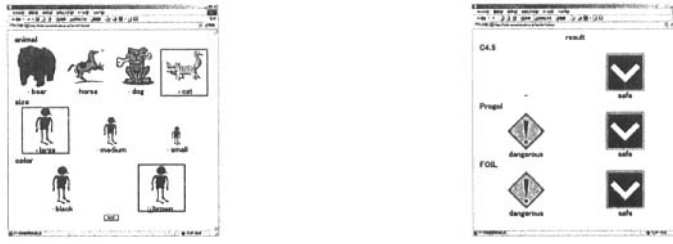


Figure 4: A decision tree D_{animal} w.r.t. the attribute $class$



(a) (b)

Figure 5: An web application using EVLD

6 Concluding Remarks

In this paper we propose the logic of discovery can be a model for the conceptual level in unifying various KDD systems, with implementing the system EVLD. Since EVLD allows databases and KDD systems to work on distributed host computers, we can regard it as a system for distributed discovery.

EVLD is implemented in Java. One reason why we chose Java is we would like to connect the database system DB and EVLD with the application programming interface JDBC [13]. Another reason is that our work is strongly affected with two previous systems which are also implemented in Java. One of the systems is appeared in the book by Bigus and Bigus [3]. It makes users enable to try three different learning algorithms, with a window-based interface like a wizard. However the algorithms provided by the system treat only relations consisting of bit vectors. EVLD can treat relations whose attribute values are in arbitrary discrete sets. The second system which affected us is the WEKA system by Witten's group [14]. It provides various KDD systems on Java Virtual Machine. We use its implementation of C4.5 in EVLD, but we cannot find any relationship-oriented systems in WEKA. EVLD might be regarded as an extension of the two systems, but it is based on our original idea.

We have two problems to solve in the future. The first one is criteria for comparing hypotheses from various KDD systems. Some users may merge several obtained hypotheses into one logic program. For such usage of EVLD we have to give to criteria for justifying such operation. The second appears when we use EVLD as a distributed KDD system. If a database and KDD systems are on different host computers, the data and passed from the database to KDD systems would be huge and make the network traffic heavy. Moreover, the data might be confidential. Therefore we will need some technology to divide huge data into small parts and to send them in confidence.

7 Acknowledgement

This research is partly supported by the Okawa Foundation for Information and Telecommunications.

References

[1] R. Agrawal, T. Imielinski, and A. Swami. Mining Association Rule between Sets of Items in Large Databese. In *Proceedings of the ACM SIGMOD Conference on Management of Data*, pages 207–216, 1993.

[2] H. Arimura and A. Yamamoto. Inductive Logic Programming : From Logic of Discovery to Machine Learning. *IEICE Transactions on Information and Systems*, E83-D(1):10–18, 2000.

[3] J. P. Bigus and J. Bigus. *Constructing Intelligent Agents with Java.* Wiley, 1998.

[4] O. Maruyama, T. Uchida, T. Shoudai, and S. Miyano. Toward genomic hypothesis creator: View designer for discovery. In *Proceedings of the First International Conference on Discovery Science (LNAI 1532)*, pages 105–116. Springer, 1998.

[5] O. Maruyama, T. Uchida, K. L. Sim, and S. Miyano. Designing views in hypothesis creator: System for assisting in discovery. In *Proceedings of the Second International Conference on Discovery Science (LNAI 1721)*, pages 115–127. Springer, 1999.

[6] S. Muggleton. Inverse Entailment and Progol. *New Generation Computing*, 13:245–286, 1995.

[7] S.-H. Nienhuys-Cheng and R. de Wolf. *Foundations of Inductive Logic Programming (LNAI 1228).* Springer, 1997.

[8] G. D. Plotkin. A Further Note on Inductive Generalization. In *Machine Intelligence 6*, pages 101–124. Edinburgh University Press, 1971.

[9] J. R. Quinlan. Learning Logical Definitions from Relations. *Machine Learning*, 5:239–266, 1990.

[10] J. R. Quinlan and R. M. Cameron-Jones. Induction of Logic Programs : FOIL and Related Systems. *New Generation Comuting*, 13:287–312, 1995.

[11] M. Terabe, O. Katai, T. Sawaragi, T. Washio, and H. motoda. Attribute generation based on association rules. *The Jornal of JSAI*, 15(1):187–193, 2000. (in Japanese).

[12] J. D. Ullman. *Princeples of Dadabase and Knowledge-base Systems, Volume I, II.* Computer Science Press, 1988.

[13] S. White et al. *JDBC API Tutorial and Reference, Second Edition; Univerasal Data Access for the Java 2 Platform.* Addison-Wesley, 1999.

[14] I. H. Witten and E. Frank. *Data Mining : Practical Learning Tools and Techniques with Java Imprementations.* Morgan-Kaufmann, 2000.

Information Modelling and Knowledge Bases XIV
H. Jaakkola et al. (Eds.)
IOS Press, 2003

Intensional vs. Conceptual Content of Concepts

Jari Palomäki
Tampere University of Technology
Pohjoisranta 11, P.O.Box 300
FIN-28101 Pori
Finland
E-mail: jari.palomaki@pori.tut.fi

Abstract: In traditional approach, i.e. in the *Port Royal Logic*, the rule of inverse relation between extension and conceptual content of concept holds. However, Bolzano was able to show the rule invalid, and thus his concept of conceptual content of concept differed from the traditional one. Raili Kauppi, influenced by Leibniz's logic, developed her intensional concept theory in (1967). It is shown that her conception of intensional content of concepts, differs from the traditional one as well as Bolzano's conception of conceptual content of concepts.

Keywords: Concepts, concept theory, extension, intension, intensional logic, relational concepts.

Introduction

A famous logic text, the *Port Royal Logic,* composed by two leaders of the Port Royal movement Antoine Arnauld and Pierre Nicole in 1662, made a distinction between the comprehension [*compréhension*] and the extension [*étendue* or *extension*] of an idea. The comprehension of an idea consists of "the attributes which it includes in itself, and which cannot taken away from it without destroying it." The extension of an idea consists of "the subjects with which that idea agrees," or which contain it. Both the comprehensions of ideas and the extensions of ideas are used in the *Port Royal Logic* in justifying the basic rules of traditional logic, (Adams 1994, 58)

Leibniz, in turn, distinguished these two types in terms of ideas [*secundum ideas* or *per ideas*] on one hand, and in terms of instances [*secundum individua* or *per exempla subjecta*] or individuals belonging to the terms [*per individuis terminorum*] on the other hand, (ibid., 59). Nowadays this distinction is usually made in terms of "the intension of a concept" and "the extension of a concept".

In the *Port Royal Logic* "the extension of an idea" constituted both the species and the individuals that fall under it, whereas in Leibniz the extensional treatment is almost always in terms of individuals that falls under the idea, (Kauppi 1960, 43). Now "the extension of a concept" is taken to be a class (or a set) of all those individuals which fall under it. However, nowadays there are at least two different way to interpret "the comprehension of an idea", i.e. either as "the intension of a concept" or as "the conceptual content of a concept". These two are things to be distinguished.

1. Limits of the Traditional Conceptual Content of a Concept

In traditional approach the conceptual content and the extension of a concept can be defined as follows:

I The *conceptual content* of a concept consists of all those attributes (i.e. concepts) which are contained in it.

II The *extension* of a concept consists of all those objects which fall under it.

From these two definitions the rule of inverse relation between extension and conceptual content of concept follows:

The lesser the extension of a concept, the greater its conceptual content, and *vice versa*.

However, Bernard Bolzano in his *Wissenschaftslehre* (1837, §120) gives the following example in order to show that (#) is not always the case:

1. The concept of 'man, who understands every European language',

and

2. the concept of 'man, who understands every living European language'.

The conceptual content of the concept (1) is lesser than the conceptual content of the concept (2), for the concept (2) has in addition the concept of 'living' as its conceptual content. Also, the extension of the concept (1) is also lesser than the extension of the

concept (2), for there are fewer people who understand every European language (including e.g. Latin) than who understands every living European language. Thus, according to Bolzano, (1) and (2) contradicts (#).

A reason why Bolzano was able to show the rule (#) invalid was that his concept of conceptual content of concept differed from the traditional one. On the other hand, under the traditional definition of conceptual content of concept, where (I) is constrained only on the conjunctive form of the conceptual content of concepts, the rule (#) is still valid.

However, the conceptual content of concept can be defined so that the traditional definition will be the special case of the general one - including Bolzano's conception. This was done in Palomäki (1997).

2. The Non-Traditional Conceptual Content of a Concept

In (1998) Pavel Materna appraises Bolzano's theory of concepts. According to him, for Bolzano the conceptual content of a concept is the sum of all simple concepts, which are parts of the concept. Since the "conceptual content of a concept" is only the "sum" of its parts, and the conjunctive is not the only way of holding these parts together, Bolzano is not to be classified with traditional logicians. Especially, it is now possible to distinguish between the conceptual content of a concept and the concept itself, where different concepts may have the same conceptual content, cf. Bolzano's examples (1837, §56):

3. 'A learned son of a non-learned father' *vs.* 'a non-learned son of a learned father',

and

4. '5^{3}' *vs.* '3^{5}',

Moreover, according to Materna, "[t]hese examples show, at the same time, that the traditional scheme has broken down", where the example (3) "is most important, for Bolzano's analysis is not even imaginable in the traditional doctrine", (Materna 1998, 72). Materna's own theory of concepts follows this Bolzanian line.

3. Intensional Content of a Concept

In a letter to Arnauld 14 July 1786 Leibniz wrote, (Leibniz 1997, 62):

> "[I]n every affirmative true proposition, necessary or contingent, universal or singular, the notion of the predicate is contained in some way in that of the subject, *praedicatum inest subjecto* [the predicate is included in the subject]. Or else I do not know what truth is."

This view may be called the conceptual containment theory of truth, (Adams 1994, 57), which is closely associated with Leibniz's preference for an "intensional" as opposed to an "extensional" interpretation of categorical propositions. Leibniz worked out a variety of both intensional and extensional treatments of the logic of predicates (i.e. concepts), but preferring the intensional approach, (Kauppi-1960, 220, 251, 252).

Raili Kauppi, influenced by Leibniz's logic, developed her intensional concept theory in (1967). Kauppi's axiomatic concept theory, denoted by **KC**, is presented in a first-order language L that contains individual variables a, b, c,..., which range over the *concepts*, and one non-logical 2-place *intensional containment relation,* denoted by "„". When $a \mathrel{„} b$, we say that a concept b is intensionally contained in a concept a, or that the intension of concept a contains the intension of concept b. This theory **KC** is based purely on the intensional content of concepts, which is studied also in Palomäki (1994a,b).

Given the Bolzano's examples (1), (2), (3), and (4), I shall now show how they are presented by means of Kauppi's concept theory, i.e. by means of her relational concept theory **RC**, where **KC** $\subseteq$ **RK**. This would also show, that Kauppi's conception of concepts differs from the traditional one, too.

3.1. Bolzano's examples (1) and (2):

Bolzano's 'counter-examples' (1) and (2) to the rule (#), is treated in Kauppi's theory as follows:

The intension of (1) is greater that the intension of (2), since the man, who understands every European language, understands also every living European language, whereas the man, who understands every living European language, does not necessarily understand every European language. That is, the concept of 'every' has more intension than the concept of 'every living', i.e. the restriction of the concept of 'every' to the concept of 'every living'. More generally, the concept of 'every' contains intensionally the concept of 'some'. On the other hand, the extension of (2) is greater that the extension of (1), for there are fewer people who understand every European language (including e.g. Latin) than who understand every living European language. Thus, the rule (#) holds for (1) and (2) in Kauppi's theory.

However, the concept of 'every' contains intensionally the concept of 'some', but the extension of the concept of 'some' is a subset of the extension of the concept of 'every'. Thus, indeed, the rule (#) is violated in Kauppi's theory.

3.2. Bolzano's examples (3) and (4):

Concerning Bolzano's examples (3) and (4), relational concepts are needed. The theory of relational concepts was left unfinished by Kauppi. However, in Kauppi (1967) she gave the basic outline how this theory of relational concepts should be developed. She even analysed Bolzano's example (3) in order to show how her theory differs from Bolzano's theory.

In the example (3), firstly, we need a relational concept 'x is a son of y'. Secondly, we need the range-designation for the variables of the relational concept, i.e. for example both the ranges of x and y are designated to be the concept of 'human'. Thirdly, we need the operation of restriction of the relational concept, i.e. where the "positions" of the relational concept are restricted by given the values to them. For example, in the relational concept 'x is a son of y', the "positions" of it, denoted by "x" and "y", are restricted by the concepts 'learned' and 'non-learned', respectively, we get the restricted relational concept 'a learned son of a non-learned father'. On the other hand, if we restrict the relational concept 'x is a son of y' by the concepts 'non-learned' and 'learned', respectively, we get the restricted relational concept 'a non-learned son of a learned father'. Moreover, since the concepts 'learned' and 'non-learned' are *incompatible*, also the restricted relational concepts 'a learned son of a non-learned father' and 'a non-learned son of a learned father' are incompatible, (Kauppi 1967, 108). Accordingly, we are not able to apply the rule (#) for (3) to test its validity.

The example (4) is treated similarly. Firstly, we have the concepts '3' and '5'. Secondly, we have the relational concept 'x is the exponent of y'. Thirdly, we give the range-designation to the variables as being 'natural numbers'. Fourthly, restricting this relational concept by the concepts '3' and '5', respectively, we get the restricted relational concept '5^3'. On the other hand, restricting this relational concept by the concepts '5' and '3', respectively, we get the restricted relational concept '3^5'. However, since the concepts '3' and '5' are *compatible*, we are to compare the restricted relational concepts '5^3' and '3^5'. Since the concept of '3' contains intensionally the concept of '5', the relational concept of the exponential function '5^3' has a greater intension than the relational concept of the exponential function '3^5'. On the other hand, the extension of '3^5' is greater than the extension of '5^3'. Thus, the rule (#) holds for (4).

4. Conclusion

In traditional approach, i.e. in the *Port Royal Logic*, the rule of inverse relation between extension and conceptual content of concept holds. However, by an example Bolzano was able to show that rule invalid, and his own concept of conceptual content of concept differed from the traditional one. On the other hand, Raili Kauppi, influenced by Leibniz's logic, developed her intensional concept theory in (1967). It was shown that her conception of intensional content of concepts differs from the traditional one as well as Bolzano's conception of conceptual content of concepts.

References

Adams, R. M., 1994: *Leibniz: Determinist, Theist, Idealist.* New York, Oxford: Oxford University Press.

Bolzano, B., 1837: *Wissenschaftslehre* **I**. Sulzbach.

Kauppi, R., 1960: *Über die Leibnizsche Logic mit besonderer Berücksichtigung des Problems der Intension und der Extension.* Acta Philosophica Fennica, Fasc. XII. Helsinki: Societas Philosophica Fennica.

Kauppi, R., 1967: *Einführung in die Theorie der Begriffssysteme.* Acta Universitatis Tamperensis. Ser. A. Vol. 15. Tampere: Tampereen yliopisto.

Leibniz, G. W., 1997: *Philosophical Writings.* Ed. G. H. R. Parkinson. Trans. M. Morris and G. H. R. Parkinson. London: The Everyman Library.

Materna, P., 1998: *Concepts and Objects*. Acta Philosophica Fennica **63**. Helsinki: Societas Philosophica Fennica.

Palomäki, J., 1994a: "Towards the Foundations of Concept Theory", *Information Modelling and Knowledge Bases V: Principles and Formal Techniques*. Eds. H. Jaakkola, H. Kangassalo, T. Kitahashi, and A. Markus. Amsterdam, Oxford, Washington, Tokyo: IOS Press, 139-154.

Palomäki, J., 1994b: *From Concepts to Concept Theory: Discoveries, Connections, and Results*. Acta Universitatis Tamperensis, ser. A, vol. 416. Tampere: Tampereen yliopisto.

Palomäki, J., 1997: "Three Kinds of Containment Relations of Concepts". *Information Modelling and Knowledge Bases VIII*. Eds. H. Kangassalo, J.F. Nilsson, H. Jaakkola, and S. Ohsuga. Amsterdam, Berlin, Oxford, Tokyo, Washington, DC.: IOS Press, 261-277.

Information Modelling and Knowledge Bases XIV
H. Jaakkola et al. (Eds.)
IOS Press, 2003

Flexible Association of Varieties of Ontologies with Varieties of Databases

Vojtech Vestenický[1], Bernhard Thalheim[2]

Computer Science Institute
Brandenburg University of Technology at Cottbus, FRG
{vojtech[1], thalheim[2]}@Informatik.TU-Cottbus.DE

Abstract

Data access requires a sufficient knowledge about the underlying data and especially their structures. Databases in use may be based on very different schemata what increases difficulties for casual users.

User understanding of a particular application domain can be captured by means of ontologies. There are often several application oriented domains and data sources involved in one information system. The flexible assignment between them becomes crucial.

We propose an approach which supports flexible associations between ontologies and databases. Instead of directly associating ontology elements with database elements, we first associate ontology with concepts and then concepts with query forms. The approach allows greater independence and flexibility.

The practical deployment is illustrated on our *Intelligent query generator.* It supports a natural language access and an easy adaptation of queries to user profiles.

1 Introduction

We have sometimes problems with finding information using information services (IS). The users have different understanding of the world. Furthermore IS often consist of very different data schemata. This makes the understandability of such IS even more difficult for casual users.

Major issues when building an efficient query generator come from the following restrictions: *general algorithms for a generation of arbitrary queries can not exist* (non-recursive) [AHV95] and *the query generation with the schema knowledge is too broad.*

The query generators where users have to specify a database schema as an input are restrictive, since only those who are familiar with the schema can use them.

A natural language access is better, because it allows users to formulate their queries in their own natural language.

A classical approach is shown in Figure 1a), the user selects the natural language query through the menu interface. The corresponding SQL query representation is then sent to the database and the user receives the result. If such systems will satisfy all user

needs then the implementation of this approach requires a huge number of menus and is thus infeasible.

A better approach has been proposed in [TK01] and applied to information sites. The tool supports the natural language query transformation into a set of SQL query candidates, see Figure 1b) (*Query Liquefaction* is the main component. It allows melting the request into parts, to integrate the parts with the database and the meta-data contained in the *DB schema manager*).

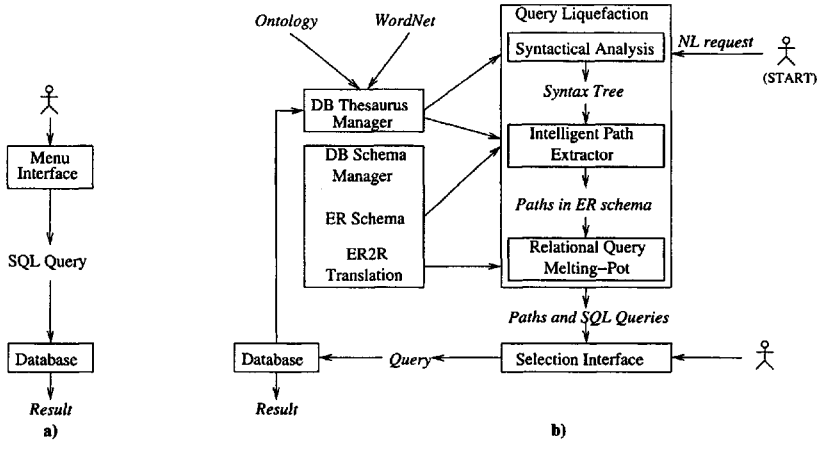


Figure 1: a) Classical approach b) Cottbus intelligent NL request transformer

This approach suffers from the following main problems:

- *pragmatics of sentence semantics*, feasibility of translation from natural language to its relational representation (NL2R)
- given a schema, the candidate sets for queries, obtained through NL2R, might be too large for arbitrary users and for programmers too, which are unexperienced in SQL programs details
- to reason whether the generated candidate sets of queries is appropriate to answer the NL request is a difficult task and there is no general algorithm for selection

We extend this approach to eliminate the above problems. Its functionality will be illustrated by describing our *Intelligent query generator.*

2 Our approach

Motivation scenario: a user asks a website "Where can I find professor Sigmund?" or "What are the publications of professor Sigmund?". The user of the local university website needs at least 5 clicks in order to be successful, but he will not be successful on the fly. Even search engines don't help as much as necessary.

In our approach, the motivation scenario proceeds as follows: the user clicks on ontology object *Person* then on *Publication*, provides the name of the professor as input and

lets the engine work. Figure 2 shows the main principles of our approach (*subjective conception* is based on user-biased ontologies, *objective concept* covers the knowledge about data and *subschema* represents underlying data structures).

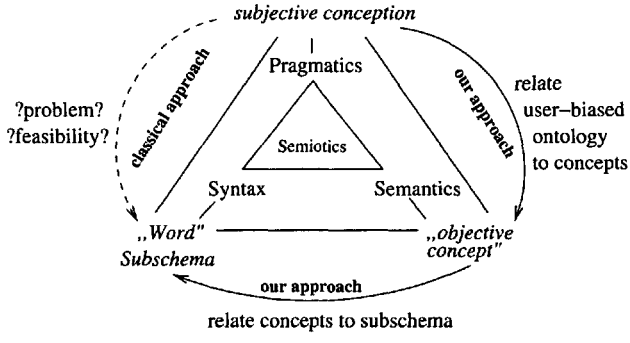


Figure 2: Approach overview

We propose an extension of the *Cottbus intelligent NL request transformer* [TK01].

- In order to eliminate the problems with *pragmatics of the sentences* we propose an **ontology with concepts** to become a part of the system.

Ontologies concentrate on meaningful names and a very limited number of associations. They may change. The hierarchy of ontology objects depends on the application area currently considered. Thus we have chosen the specialization of concepts from ontologies. Ontology objects and concepts are related through a relation. This approach provides more independence, allows a simple change of hierarchies and is thus better for evolving applications, e.g. in website applications.

- To avoid large *candidate sets* we relate the **concepts with query forms** [Tha00] which are independent of the DB schema.

Concepts form a relatively stable piece of knowledge. Concepts are covering the linguistic knowledge, e.g. associations among them (homonyms, hyponyms, etc.). The meaning of a concept is captured by descriptions. Concepts can be related to databases by general queries, which must be independent on the database system currently chosen. Therefore it is better to specify queries in a very general form (so-called query form) and to choose the corresponding SQL query at run time or better at compile time using an SQL query generator.

The general view of our proposed extended architecture is in Figure 3. The user selects the ontology object. For this object the concept is determined through the relation and for the concept the corresponding query form. This query form is used as an input to the NL request transformer (see Figure 1b)).

Our architecture has the following restrictions in order to keep it manageable:

- Query forms are orthonormal representation of queries. (Orthonormal language is restricted of natural language, which can be directly mapped to query forms, see Example 1).

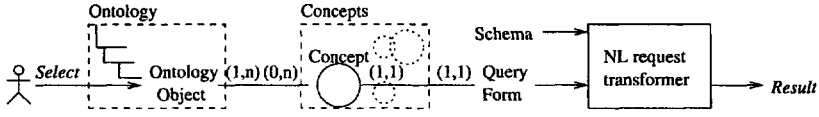


Figure 3: General view

- Concepts and query forms are tightly coupled through one-to-one relationship (see Figure 3).

These restrictions aren't restricting the approach entirely since we could use libraries of preferenced associations between query forms and concepts.

2.1 Specialization of concepts from ontologies

Ontologies are defined independently of the actual data, reflect a common understanding of the semantics of the domain of discourse and are used to share and exchange information between sources [Gru93, NFF+91].

We use for our purpose the *terminological ontology*, i.e. an ontology whose categories need not be fully specified by axioms and definitions. An example of a terminological ontology is *WordNet*, whose categories are partially specified by relations such as subtype-supertype or part-whole, which determine the relative positions of the concepts with respect to one another but do not completely define them [Sow02].

Since we need to associate ontologies with concepts we have adapted the definition (see Definition 1).The association is done through the generic function. The user can choose the suitable concept based upon its description and associate it with the particular ontology object. For example, the components of Figure 4 are : O={*Person, Publication, ...*}, R={*authorOf, worksIn ,As, ...*}, H={*(Person, Worker), (Publication, Article), ...*},F*(Publication)*={*Person authorOf Publication*}.

Definition 1 *The ontology is a tuple* (O, R, H, F) *where:* O *is a set of names used for ontology objects;* R *is a set of binary relations over* O*;* H *is a hierarchy over* O*,* $H \subseteq O \times O$*;* F *is a generic function,* $F : O \rightarrow C$ *and* C *is a set of Concepts.*

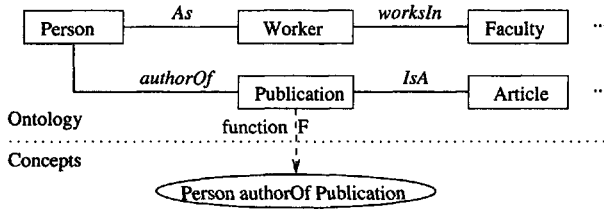


Figure 4: The ontology and concepts

2.2 Concepts and query forms

A concept is a discrete unit of knowledge that exists relatively independently of minds and objects. In an application domain, some objects can *fall under* a given concept C. In this case, those objects are called the *extension* of C in the application domain [Nii99].

In our approach we want concepts to be related with databases through query forms. For this reason we extend the definition of concept (see Definition 2).

Definition 2 *The concept is a tuple* $(Name, Desc, QF)$ *where: Name is a concept name; Desc is a concept description; QF is a generic query form.*

Query forms are a parametric set of QBE generalizations independent from the DB-schema [Tha00], see Example 1.

Example 1 *Query form: Find the titles of publications written by an author with the name* Y. *(The output parameter is denoted by a variable with the prefix '!'. The input parameter is specified by a variable with the prefix '?'.)*

Author	
Publication	Name
Title	?Y
!	

The relation between concepts and query forms is not enough for a successful instantiation of generic query forms. We use therefore additionally an *explicit cooperation schema.* This schema explicitly relates the parts of ER-Schema with the concepts through the relation $Eq(C, S)$, where C is the concept name and S is the part of ER-Schema. For example, *Eq(Title, Article.Title), Eq(Title, Publication.Title), Eq(Person, Person)*, etc.

The parts of ER-Schema can then be determined by the *query form instantiation* (see Figure 5 based on Example 1). If the match can not be found, e.g. *written*, we can state the *Eq* relation manually *Eq(written, publishes)* or automatically (see Figure 6, part of NL request transformer). The resulting *query form instantiation* may look like this: *SELECT Publication.Title FROM Publication, Publishes, Author WHERE Author.Name=Y.*

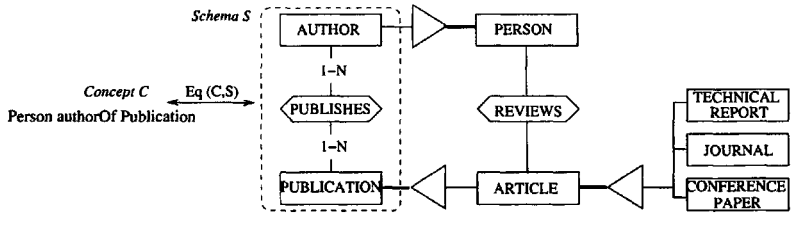


Figure 5: Concepts and ER-Schema

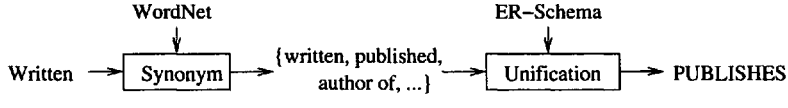


Figure 6: Relating concepts and ER-Schema

2.3 Query generation

The queries can be generated in two ways: at compile time or at run time.

Query generation at compile time

1. A user navigates through the ontology and selects the ontology object O. Since the ontology objects are associated with the concepts, the concept C is extracted for the object O.
2. The concepts are connected through *explicit cooperation schema* with the parts of ER-Schema. The schema S for the concept C is determined.
3. The query form QF for the concept C is selected and used as an input to *Query Liquefaction* (see Figure 1b)).
4. The query candidates $\{q_1, q_2, \ldots, q_n\}$ are generated.
5. The user can select the suitable candidate query q_i or it will be selected according to his profile.
6. The concept C is then related with the chosen candidate q_i for schema S.

Query generation at run time

1. The user chooses the ontology object O and for O the concept C is automatically determined.
2. The query q for the concept C is selected.
   or
2. The user can open a concept editor for the concept C and modify it. The appropriate query must be compiled for the query form.
3. The new query candidates can be generated using the new query form.
4. The user picks up the suitable query candidate q_i.
5. The query q_i is then related with the concept C.

3 Implementation

We have implemented the prototype of the intelligent query generator using the Borland Delphi for Windows. The prototype supports the user when extracting the information or constructing the queries. It can handle different user requirements using the profiles. However, it has following limitations: one ontology object is connected with one concept and only one database schema at a time is supported.

An application scenario is depicted in Figure 7. The user has selected the ontology object *Publication*. The associated concept *Person authorOf Publication* has been determined. The query form for this concept allows to choose a parameter *AuthorName* and set its value to *Sigmund*. The user can save the corresponding query in his profile *MyProfile*. Next time he chooses this profile he will be directly presented with the query results and will not need to go through the ontology again.

4 Outlook

In the future work we try to further develop the current approach. We have not dealt with an assignment of different ontologies. In the ontology has not been possible to combine existing ontology objects in order to define new ones, e.g. *Person* and *Article* in *Person who wrote an article*. The composition of concepts has been absent to simplify the realization and needs further study. The restriction to use orthonormal language is necessary to reduce the complexity of the approach and therefore will not be easily eliminated. The prototype is limited in its functionality and needs to be extended in order to become a part of our information system, i.e. it supports just one ER-Schema.

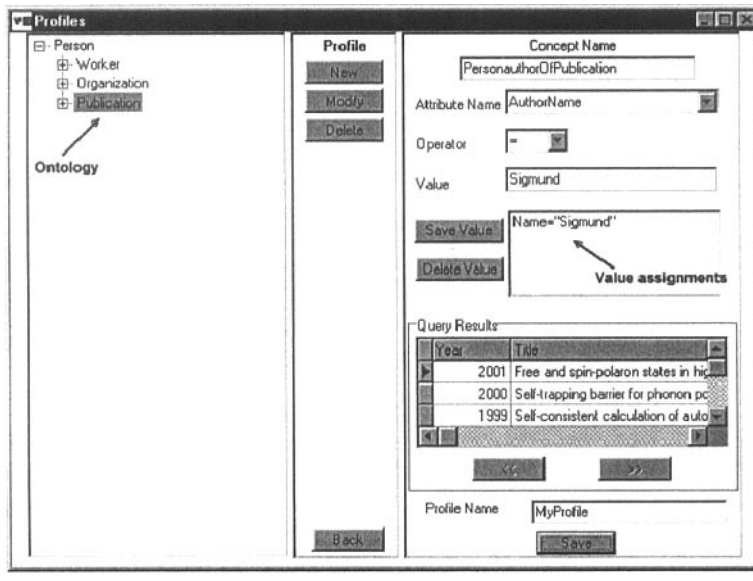


Figure 7: Defining a new profile

Acknowledgement: We are grateful to Thomas Kobienia for the implementation of the *NL Request Transformer* and Jens Wölkerling for implementing the prototype of the *Intelligent Query Generator.*

References

[AHV95] Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of Databases.* ISBN 0-201-53771-0. Addison-Wesley, 1995.

[Gru93] Thomas. R. Gruber. A translation approach to portable ontology specification. Technical Report Technical Report KSL 92-71, Knowledge Systems Laboratory, Computer Science Department, Stanford University, April 1993.

[NFF+91] R. Neches, R. Fikes, T. Finin, T. Gruber, R. Patil, T. Senator, and W. Swartout. Enabling technology for knowledge sharing. In *AI Magazine*, Fall 1991.

[Nii99] Marko Niinimäki. Semantic Data Models and Concepts - A Comparison of IFO and COMIC. Technical report, Department of Computer Science, University of Tampere, Finland, October 1999.

[Sow02] J. F. Sowa. Ontology Working Group, 2002.

[Tha00] B. Thalheim. *Fundamentals of Entity-Relationship Modeling.* ISBN 3-540-65470-4. Heidelberg, February 2000.

[TK01] B. Thalheim and T. Kobienia. From NL DB Request to Intelligent NL DB Answer. In *NLDB Conference*, Madrid, 2001.

Information Modelling and Knowledge Bases XIV
H. Jaakkola et al. (Eds.)
IOS Press, 2003

UML as a First Order Transition Logic[1]

Love EKENBERG
lovek@dsv.su.se

Paul JOHANNESSON
pajo@dsv.su.se

Department of Information Technology and Media
Mid Sweden University
SE-851 70 SUNDSVALL, SWEDEN
and
Department of Computer and Systems Sciences
Royal Institute of Technology and
Stockholm University
Forum 100
SE-164 40 KISTA, SWEDEN
Telefax: +46-8-703 90 25

Abstract. The work presented herein suggests a working method for the transformation of UML specifications to an event driven logic based formalism. The result models a fully expressive set of UML language constructs from the source language, in a first order model at meta level, and extend this with subclasses together with their relations in accordance with an input specification. The translation of any specification within this subset of the language is then achieved by just extending the meta model. The resulting model is expressed in first order logic. In this way the transformation preserves the conceptual integrity of the source specifications and provides a basis for further integration and verification.

1 Introduction

The Unified Modelling Language, UML, has gained increased popularity in recent years. It is now regularly used not only for systems analysis and design, for which it was originally conceived, but also for other phases in the systems life cycle such as requirements engineering and business analysis. The success of UML can to a large extent be attributed to two factors. First, UML has received extensive industry support from IT suppliers as well as users and has been effectively standardised. Secondly, UML makes use of intuitive and visual modelling constructs as the main components of the language, which facilitates its adoption among large user groups. However, this reliance on graphical constructs poses problems when it comes to precise and unambiguous semantics. The constructs of UML are typically informally defined, which leaves room for ambiguities, loose interpretations and misunderstandings. An important task is, therefore, to formalise the basic notions and constructs of UML.

The main purpose of this work is to describe how specifications in UML language can be translated into a logic based formalism so as to prepare a unifying view of a general class of conflict detection. Translating specifications into first order logic has the added advantage that by introducing 2nd order analyses, it becomes possible to analyse a broader spectrum of conflicts, i.e. using first order logic extended with transaction mechanisms

[1]This work was supported by the Lyee International Project.

provides tools for systematically classifying conflicts, including: individual inconsistency; protocols for multi-agent behaviour; incongruence when events are incompatible; whether certain combinations of initial states are incongruent; whether event paths are incompatible, etc. cf. [8], [9]. This kind of unified view is made possible through translations of the kind suggested in this paper, even when the processes are described in an event-driven representation. We argue that first order logic, when used in conjunction with conceptual modelling, provides a sound basis on which specifications written in a process based language can be transformed, merged, and verified for the purpose of detecting interference.

Different varieties of temporal logic [6] and BDI logic [15] on the other hand are strong contenders for the target language. In fact, much previous work on formalising UML has been based on different versions of temporal logic, e.g. [13], who defines the semantics of UML interactions in temporal logic. Another approach has been to map UML constructs to some formal specification language, e.g. [12], who maps class diagram to Object-Z. Dynamic logic has been used as a basis for UML semantics, e.g. in [14]. One of the most complete formalisations of UML is given in [16], who bases the semantics on π-calculus and labelled transition systems. However, first order logic remains the choice for this work. This is in agreement with [11] and [3], that argue for the advances of first order logic to model dynamics in contrast to approaches based on temporal logic. Furthermore, the representation in first-order logic has some convenient features from a theorem proving perspective.

The paper is organised as follows. In section 2, we introduce a first-order logic framework for conceptual schemata and define basic notions for taking into account dynamic aspects, which are modelled by the event concept. In section 3, we show how class diagrams in UML can be translated into the framework. In section 4, we show how dynamic properties of UML specifications can be translated, and in the final section we summarise the work.

2 Preliminaries

We give a short overview of the basic concepts used in the sequel. An extended treatment of the various concepts can be found in [8], [10].

In the definitions below, we assume an underlying *language* L of first-order formulae.[2] By a *diagram* for a set R of formulae in a language L, we mean a Herbrand model of R, extended by the negation of the ground atoms in L that are not in the Herbrand model. Thus, a diagram for L is a Herbrand model extended with classical negation.[3]

Definition 1

A *schema* S is a structure ⟨R,ER⟩ consisting of a *static part* R and a *dynamic part* ER. R is a finite set of closed first-order formulae in a language L. ER is a set of *event rules*. Event rules describe possible transitions between different states of a schema and will be described below. *L(R)* is the *restriction of L to R*, i.e. L(R) is the set {p | p ∈ L, but p does not contain any predicate symbol, that is not in a formula in R}. The elements in R are called *static rules* in L(R). *A static assertion in S,* F(**x**), is a closed first order formula in L(S).

[2] As we will explain below, the assumption of a particular language is not necessary, since the concepts can be more generally defined.

[3] For our purposes, this is no loss of generality by the well-known result that a closed formula is satisfiable iff its Herbrand expansion is satisfiable. For a discussion of this expansion theorem and its history, see, e.g. [7].

Example

Assume that the static part of the schema S is the following: [4]

$$R = \{\neg r(a) \vee r(b),\ r(c) \leftrightarrow r(a)\}$$

The diagrams for schema S are then:

$$\{r(a),r(b),r(c)\}$$
$$\{\neg r(a),r(b),\neg r(c)\}$$
$$\{\neg r(a),\neg r(b),\neg r(c)\}$$

The dynamic part of a schema requires a definition of the concept of event rule, which is given its semantics through the event concept. Intuitively, an event is an instance of an event rule, i.e. a transition from one diagram to another one. An event rule may be initialised from the environment of the agent system or from another agent included in the system. Initialised event rules map on possible events that may change the state of an instance of a specification. Below, L denotes a language.

Definition 2

An *event rule* in L is a structure $\langle P(\mathbf{z}), C(\mathbf{z})\rangle$. $P(\mathbf{z})$ and $C(\mathbf{z})$ are first-order formulae in L, and $\mathbf{z}$ is a vector of variables in the alphabet of L.[5] In terms of conceptual modelling, $P(\mathbf{z})$ denotes the precondition of the event rule, and $C(\mathbf{z})$ the post condition. ■

Definition 3

The set of *basic events* for a schema $\langle R, ER\rangle$ is a relation $E \subseteq D \times D$, where D is the set of diagrams for R. An element (σ, ρ) belongs to E iff

(i) $\rho = \sigma$, or

(ii) there is a rule $\langle P(\mathbf{z}), C(\mathbf{z})\rangle$ in ER, and a vector $\mathbf{e}$ of constants in L, such that $P(\mathbf{e}) \in \sigma$ and $C(\mathbf{e}) \in \rho$. In this case we will also say that the *basic event* (σ, ρ) *results* from the event rule.[6] ■

Note that E is not necessarily a function. Intuitively, this means that an event rule is non-deterministic: if an instance of the precondition of an event rule belongs to a diagram σ, the event rule results in events (σ, ρ) for every diagram ρ, such that the corresponding post condition belongs to ρ. Thus, the approach to the dynamics of a schema differs from the traditional transactional approach in the database area, where an event deterministically specifies a minor modification of a state [1]. Depending of the particular information processor used, this event concept can be restricted in a variety of ways.

Example (cont.)

The event rules for the schema S_1 are:

$$Er_{1a} = \langle r(a), r(b)\rangle$$
$$Er_{1b} = \langle \neg r(a), (\neg r(b) \wedge \neg r(c))\rangle$$
$$Er_{1c} = \langle (\neg r(a) \wedge r(b)), r(a)\rangle$$

[4] To simplify the presentation, a, b, and c are constants. As can be seen from definition 1, the static part of a schema may have a more general form.

[5] The notation A(x) means that x is free in A(x).

[6] In [8] an approach using event messages is described. However this concept is not necessary in a model not presupposing a particular information processor.

Let:

$$\sigma_1 = \{r(a), r(b), r(c)\}$$
$$\sigma_2 = \{\neg r(a), \neg r(b), \neg r(c)\}$$
$$\sigma_3 = \{\neg r(a), r(b), \neg r(c)\}$$

Then the set of basic events for schema is:

$$\{(\sigma_1,\sigma_1), (\sigma_2,\sigma_2), (\sigma_3,\sigma_3), (\sigma_1,\sigma_3), (\sigma_3,\sigma_1), (\sigma_3,\sigma_2)\},$$

where (σ_1,σ_1), (σ_2,σ_2), and (σ_3,σ_3) follows trivially from (i) in definition 3, (σ_1,σ_3) is due to Er_{1a}, (σ_3,σ_1) is due to Er_{1c}, and (σ_3,σ_2) is due to Er_{1b}. ■

A description of a schema is a structure consisting of all diagrams for a schema, together with all possible transitions that are possible with respect to the basic events for the schema.

Definition 4

The *description of a schema s* is a digraph ⟨D, E⟩, where D is the set of diagrams for s, and E is the set of basic events (i.e. arcs in the digraph) for s. ■

Example (cont.)

Fig. 1 illustrates a description of schema S_1. The arrows in the figure represent basic events and the circles represent the diagrams for the schema.

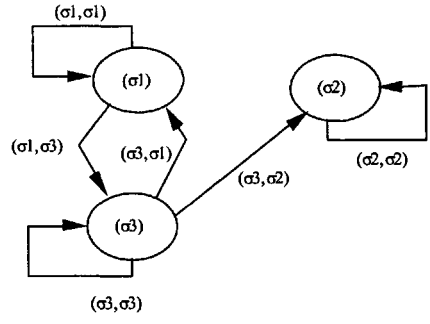


Fig. 1 The description of a schema

3 Translating UML Class Diagrams into Logic

Static concepts and relations modelled in UML can straightforwardly be expressed in terms of first order formulae. In this section, we suggest such a translation for static properties of class diagrams.

3.1 UML Class Diagrams

A class diagram is the standard modelling concept for static properties in UML. The components of a class diagram are classes that represent concepts in the world. These can have one or more lexical attributes. Classes can be related to each other, which is represented in UML by associations. Associations can be further specified by cardinality constraints expressing properties such as, e.g., injectivity, surjectivity and totality. Compositions are particular kind of associations, and are expressed by aggregations in UML. Furthermore, various kinds of subset relations can be represented.

Using definition 5 below, a class diagram in UML can readily be translated into a set of first order formulae.

Definition 5

Given a set *C* of class diagrams in a UML specification, where the strings *agg* or *lex* do not occur. R_C is the least set of first order formulae defined by the following clauses.

1. Alphabet

If *r* is a name of a class definition in C, then *r* is a predicate symbol of arity one in $L(R_C)$.

a) If *t* is a name of an association in C, then *t* is a predicate symbol of arity two in $L(R_C)$.
b) If *t* is a name of an attribute in C, then *t* is a predicate symbol of arity two in $L(R_C)$.
c) *agg* is a predicate symbol of arity two in $L(R_C)$.
d) *lex* is a predicate symbol of arity one in $L(R_C)$.

2. Typing constraints for associations

If *r* and *s* are names of class definitions in C, and *t* is a name of an association from *r* to *s* in C, then $\forall x \forall y(t(x,y) \rightarrow (r(x) \wedge s(y)))$ is in R_C.

3. Typing constraints for attributes

If *r* is a name of a class definition in C and *t* is a name of an attribute of *r* in C, then $\forall x \forall y(t(x,y) \rightarrow (r(x) \wedge lex(y)))$ is in R_C.

4. Aggregation constraints

If *r* and *s* are names of class definitions in C, and *t* is a name of an aggregation from *r* to *s* in C, then $\forall x \forall y(t(x,y) \rightarrow (r(x) \wedge s(y) \wedge agg(x,y)))$ is in R_C.

5. ISA constraints

If *r* and *s* are names of class definitions in C, and the statement *r ISA s* belongs to C, then $\forall x(r(x) \rightarrow s(x))$ is in R_C.

6. Subclass constraints

Assume that *p, r* and *s* are names of class definitions in C, and that *p ISA s* and *r ISA s* belong to C. If *p* and *r* are disjoint in C, then $\forall x \neg(p(x) \wedge r(x))$ is in R_C. If *p* and *r* are exhaustive wrt *s* in C, then $\forall x(s(x) \rightarrow (p(x) \vee r(y)))$ is in R_C.

7. Cardinality constraints

If *r* and *s* are names of class definitions in C, and *t* is a name of an association from *r* to *s* in C, with cardinality $((min_r .. max_r), (min_s .. max_s))$, then the formulae below are in R_C.

7.1. Minimum number of associations for the domain

$$\forall y \exists x_1 \ldots \exists x_{min_r}((s(y)) \rightarrow (t(x_1, y) \wedge \ldots \wedge t(x_{min_r}, y))) \wedge$$
$$\neg(x_1 = x_2) \wedge \ldots \wedge \neg(x_1 = x_{min_r}) \wedge$$
$$\neg(x_2 = x_3) \wedge \ldots \wedge \neg(x_2 = x_{min_r}) \wedge \ldots \wedge$$
$$\neg(x_{min_{r-1}} = x_{min_r})$$

7.2. Maximum number of associations for the domain

$$\forall y \forall x_1 ... \forall x_{\max_r} \forall x_{\max_r+1} [((t(x_1, y) \wedge ... \wedge t(x_{\max_r}, y) \wedge t(x_{\max_r+1}, y))$$
$$\rightarrow$$
$$((x_1 = x_2) \vee ... \vee (x_1 = x_{\max_r}) \vee (x_1 = x_{\max_r+1}) \vee$$
$$(x_2 = x_3) \vee ... \vee (x_2 = x_{\max_r}) \vee (x_2 = x_{\max_r+1}) \vee ... \vee$$
$$(x_{\max_r} = x_{\max_r+1}))]$$

7.3. Minimum number of associations for the range

$$\forall y \exists x_1 ... \exists x_{\min_s} ((r(y)) \rightarrow (t(y, x_1) \wedge ... \wedge t(y, x_{\min_s}))) \wedge$$
$$\neg(x_1 = x_2) \wedge ... \wedge \neg(x_1 = x_{\min_s}) \wedge$$
$$\neg(x_2 = x_3) \wedge ... \wedge \neg(x_2 = x_{\min_s}) \wedge ... \wedge$$
$$\neg(x_{\min_{s-1}} = x_{\min_s})$$

7.4. Maximum number of associations for the range

$$\forall y \forall x_1 ... \forall x_{\max_s} \forall x_{\max_s+1} [((t(y, x_1) \wedge ... \wedge t(y, x_{\max_s}) \wedge t(y, x_{\max_s+1}))$$
$$\rightarrow$$
$$((x_1 = x_2) \vee ... \vee (x_1 = x_{\max_s}) \vee (x_1 = x_{\max_s+1}) \vee$$
$$(x_2 = x_3) \vee ... \vee (x_2 = x_{\max_s}) \vee (x_2 = x_{\max_s+1}) \vee ... \vee$$
$$(x_{\max_s} = x_{\max_s+1}))]$$

Example

Consider the UML class diagram below.

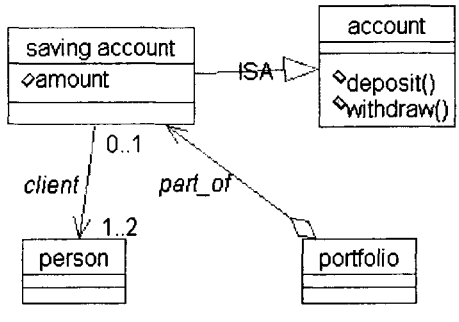


Fig. 2 A UML class diagram

Thus, *saving_account*, *account*, *person* and *portfolio* are classes in UML, representing concepts. An association represents the relation *client*, from *saving_account* to *person*, meaning that a client is a person with an account. The attribute *amount* represents the current balance. An account must be owned by a person, and a person can have at most one saving account. Furthermore, at maximum two clients can share an account. The class *saving_account* is a subclass to *account*. Furthermore, *saving_account* is a component of the portfolio of the bank. This is represented by the aggregation form *saving_account* to *portfolio*. The methods *deposit* and *withdraw* represent possible transactions for an account and will be treated in section 4 below.

Using the rules in definition 5, the class diagram is translated to the following set of formulae:

Translation of static properties	UML components
$\forall x(saving\_account(x) \rightarrow account(x))$,	ISA relation
$\forall x \forall y(client(x,y) \rightarrow (saving\_account(x) \wedge person(y)))$,	association *client*
$\forall x \forall y(amount(x,y) \rightarrow (saving\_account(x) \wedge lex(y)))$,	attribute *amount*
$\forall x \forall y(part\_of(x,y) \rightarrow (saving\_account(x) \wedge portfolio(y) \wedge agg(x,y)))$,	aggregation *part_of*
$\forall x \exists y(saving\_account(x) \rightarrow client(x,y))$,	cardinality 1
$\forall x \forall y \forall z \forall w((client(x,y) \wedge client(x,z) \wedge client(x,w)) \rightarrow ((y=z) \vee (y=w) \vee (z=w)))$,	cardinality ..2
$\forall x \forall y \forall z((client(y,x) \wedge client(z,x)) \rightarrow (y=z))$	cardinality ..1

4 Translating Dynamics in UML into Logic

In UML, dynamics can be modelled in three different ways: by methods in classes, by state diagrams, and by activity diagrams. In this section, we consider the translation of methods and state diagrams into logic.

4.1 Translating Methods into Logic

Methods in UML describe operations within specifications and can be arbitrary automata. A method is labelled by a signature with a vector of variables. In the sequel, we assume that the semantics of a signature g(y) can be formulated by a pair <pre(x), post(x)>, where *pre(x)* is a first order formula expressing the precondition of the method and *post(x)* is a first order formula expressing the postcondition of the method.[7]

Definition 6

Given a set of methods *M* in a UML specification, where the string *inv* does not occur. ER_M is the least set of expressions defined by the following clauses.

1. Alphabet

a) If g(y) =<pre(x), post(x)> is a method in *M*, then the corresponding predicate symbols with the same arities are in $L(ER_M)$.
b) *inv* is a predicate symbol of arity one in $L(ER_M)$.

2. Methods

Let *k* be a class and g(y) =<pre(x), post(x)> a method in *k*. If g(y) is a method in *M*, Then $<inv(g(y)) \wedge pre(x), post(x) \wedge \neg inv(g(y))>_k$ is in ER_M, where pre(x) and post(x) are as above.[8]

4.2 Translating State Diagrams into Logic

A UML state diagram is associated to a specific class in the static specification and is a state diagram in the usual sense. Thus, it consists of a set of nodes and conditions for when transitions between these states may occur.

Definition 7

A *state diagram for a class k* is a triple <N, A, G>, where

a) N is a set of nodes.
b) A is a set of directed arcs, i.e. pairs over N.
c) G is a set of triples $<a, g(y), v(x)>_k$, where a is an arc, g(y) is the signature of a method in *k*, and v(x) is an open first order formula.

---

[7] The expressiveness of the translation could be extended by the addition of recursive techniques for handling more general types of methods, but this is beyond the scope of this article.

[8] inv(g(a)) expresses that the method g(y) is invoked with the vector a.

Intuitively, such a triple in G specifies an arc, its associated event, and its guard.

Example

Consider the diagram below.

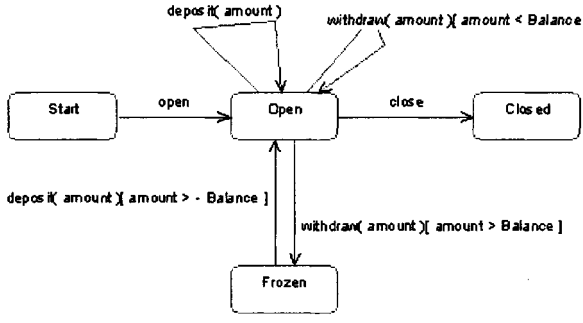


Fig. 3 A UML class diagram for the class *account*

For this diagram

N = {Start, Open, Frozen, Closed}

A = {<Start, Open>, < Open, Open>, <Open, Frozen>, <Frozen, Open>, <Open, Closed>}

G = {<<Start, Open>, open, True>,

<<Open, Open>, deposit(amount), true>,

<<Open, Open>, withdraw(amount), amount < Balance>,

<<Open, Closed>, close, True>,

<<Open, Frozen>, withdraw(amount), amount > Balance>,

<<Frozen, Open>, deposit(amount), amount > - Balance>}

Definition 8

Let <N, A, G> be a state diagram for a class *k*. Let <<t_i, t_j>, g(y), v(x)> ∈ G. The *arc event* of <t_i, t_j> is

<inv(g(y)) ∧ state(x, t_i) ∧ v(x), state(x, t_j) ∧ ¬inv(g(y))>$_k$.[9]

Example

An example of an arc event in the diagram above is:[10]

<inv(deposit(amount)) ∧ state(x, open), state(x, open) ∧ a=balance+amount ∧ balance(a) ∧ ¬inv(deposit(amount))>[11]

Definition 9

Given a set *S* of state diagrams <N, A, G>$_k$ in a UML specification, where the string *state* does not occur. A *schema* for S is constructed as follows.

---

[9] State(x, t) means that x is in state t.

[10] In the sequel, we omit the index *k* if the class is obvious from the context.

[11] The function symbols + and – as well as the predicate symbol = have the obvious semantic.

1. Alphabet

a) If $\langle\langle t_i, t_j\rangle, g(y), v(x)\rangle \in G$, then the corresponding predicate symbols and constants in t_i, t_j, $g(y)$ and $v(x)$ with the same arities are in $L(ER_S)$ and $L(R_S)$.

b) *state* is a predicate symbol of arity two in $L(ER_S)$ and $L(R_S)$.

2. Rules for an object in a state

$R_S = \{\forall x(state(x, t_i) \rightarrow \neg state(x, t_j)) \mid t_i, t_j \in N \wedge i \neq j\} \cup \{\forall x \exists t(state(x, t)\} \cup \{\forall x(state(x, t_i) \rightarrow k(x)) \mid t_i \in N\}$, where t_i is a state in a class *k*.

3. Arcs

If $\langle\langle t_i, t_j\rangle, g(y), v(x)\rangle$ is an arc in *G*, $\langle inv(g(y)) \wedge state(x, t_i) \wedge v(x), state(x, t_j) \wedge \neg inv(g(y))\rangle_k$ is in ER_S.

4. Schema

The *schema for S* is the structure $\langle R_S, ER_S\rangle$.

Example

The translation of the diagram in Fig. 2 is the following.

Translation of dynamic properties	UML concept
R_S={∀x(state(x, start) *xor* state(x, open) *xor* state(x, frozen) *xor* state(x, closed))},	An object can be in one state only[12]
{∀x((state(x, start) ∨ state(x, open) ∨ state(x, frozen) ∨ state(x, closed))→account(x))}	An object in a state must belong to the class *account*
ER_S={<inv(open()) ∧ state(x, start), state(x, open) ∧ ¬inv(open())>},	The method open().
<inv(deposit(amount)) ∧ state(x, open), state(x, open) ∧ a= balance+amount ∧ balance(a) ∧ ¬inv(deposit(amount))>,	The method deposit().
<inv(withdraw(amount)) ∧ state(x, open) ∧ (amount < balance), state(x, open) ∧ a= balance–amount ∧ balance(a) ∧ ¬inv(deposit(amount))>,	The method withdraw() when the amount is less than the balance.
<inv(deposit(amount)) ∧ state(x, frozen) ∧ (amount > –balance), state(x, open) ∧ a= balance+amount ∧ balance(a) ∧ ¬inv(deposit(amount))>,	The method deposit() when the amount is greater than the negative balance.
<inv(withdraw(amount)) ∧ state(x, open) ∧ (amount > balance), state(x, frozen) ∧ a= balance–amount ∧ balance(a) ∧ ¬inv(withdraw(amount))>,	The method withdraw() when the amount is greater than the balance.
<inv(close()) ∧state(x, open), state(x, closed) ∧ ¬inv(close())>	The method close().

Definition 10

Let *C* be a set of class diagrams, *M* a set of methods and *S* a set of state diagrams of a UML specification *U*. Furthermore, let R_C, R_S, ER_S and ER_M be as in the definitions above regarding these components. A schema for *U* is the structure <R, ER>, where

- $R = R_C \cup R_S$, and
- $ER = ER_M \cup ER_S$.

Consequently, a structure <R, ER> constructed as in definition 10 is a translation of the UML specification into a schema in the sense of definition 1. From the construction, it follows that this schema has the same properties as the UML specification from which it was derived.

[12] Because of the authors' laziness, *A xor B* is used as an abbreviation for $(A \vee B) \wedge \neg(A \wedge B)$.

5 Concluding Remarks

We have presented some possible transformations between UML specifications and first order transition model. Our approach takes into account static as well as dynamic features of UML constructs, where the dynamics is modelled by means of the event concept. The work is motivated by our earlier research in schema integration and the integration of multi-agent architecture designs to handle problematic cases of global inconsistency in distributed information systems. In our further work, we will use the translation algorithms with our earlier results in conflict detection, and by this enabling an elaborate analysis of UML specifications.

References

[1] S. Abiteboul and V. Vianu, "Equivalence and Optimization of Relational Transactions," *Journal of ACM*, vol. 35, pp. 130–145, 1988.

[2] M. D'Augostino, Investigations into the Complexity of some Propositional Calculi, Technical Monograph PRG-88, Ph. D. Thesis, Oxford University, 1990.

[3] J. van Benthem and J. Bergstra, "Logic of Transition Systems," *Journal of Logic, Language and Information*, vol. 3, pp. 247-283, 1995.

[4] E.W. Beth, The Foundations of Mathematics, North Holland, 1959.

[5] S. Cook and R. Reckhow, "The Relative Efficiency of Propositional Proof Systems," Journal of Symbolic Logic, vol.44, pp.36-50, 1979.

[6] C. Dixon, M. Fisher and M. Wooldridge, "Resolution for Temporal Logics of Knowledge," *Journal of Logic and Computation*, Vol. 8:3, pp. 345-372, 1998.

[7] B. Dreben and W. D. Goldfarb, The Decision Problem: Solvable Classes of Quantification Formulae: Reading, Mass, Addison-Wesley, 1979.

[8] L. Ekenberg and P. Johannesson, "A Formal Basis for Dynamic Schema Integration," Proceedings of 15th International Conference on Conceptual Modelling ER'96, pp. 211–226, Lecture Notes in Computer Science, 1996.

[9] L. Ekenberg, " The Logic of Conflicts between Decision Making Agents," *Journal of Logic and Computation*, vol. 10, No. 4, pp.583–602, 2000.

[10] P. Johannesson et al, Conceptual Modelling, Prentice Hall, 1997.

[11] P. Johannesson and P. Wohed, "Modelling Agent Communtication in a First Order Logic," *Accounting Management and Information Technologies*, 8, 5-22, 1998.

[12] S. Kim and D. Carrington, "Formalising the UML Class Diagram Using Object-Z, in *Proceedings of UML'99*, eds. R. France and B. Rumpe, LNCS 1723, Springer, 1999.

[13] A. Knapp, "A Formal Semantics for UML Interactions", in *Proceedings of UML'99*, eds. R. France and B. Rumpe, LNCS 1723, Springer, 1999.

[14] C. Pons, G. Baum, and M. Felder, "Foundations of Objec-oriented Modeling Notations in a Dynamic Logic Framework", in *Foundatios of Models and Languages for Data and Objects*, Kluwer, 1999.

[15] A. S. Rao and M. P. Georgeff, "Decision procedures for BDI logics," *Journal of Logic and Computation*, Vol. 8:3, pp. 293-342, 1998.

[16] G. Övergaard, *Formal Specification of Object-Oriented Modelling Concepts*, Ph.D. thesis, Department of Teleinformatics, Royal Institute of Technology, Stockholm, 2000.

Information Modelling and Knowledge Bases XIV
H. Jaakkola et al. (Eds.)
IOS Press, 2003

Consistency Checking of Behavioural Modeling in UML Statechart Diagrams

Takenobu Aoshima Takahiro Ando Naoki Yonezaki
Department of Computer Science,
Graduate School of Information Science and Engineering,
Tokyo Institute of Technology
2-12-1 Ookayama, Meguro-ku, Tokyo 152-8552, Japan
E-mail:{aoshima, ando, yonezaki}@fmx.cs.titech.ac.jp

Abstract. The UML statechart diagram is widely used modeling language for describing behavioural aspect of software systems. Its rich notions of structurization make the description brief but difficult to be well-defined and consistent with the describer's intention. We present a procedure to check the validity of UML statechart diagrams. The procedure checks the existence of conflicting transitions, inappropriate history indicators, inappropriate complete transitions and dead states while constructing model graphs of UML statechart diagrams for LTL model checking. These checks contribute to removal of both ambiguous and undefined descriptions of the behaviour. After the construction of the well-defined model, the procedure executes model checking with specifications written in LTL for the consistency between the behaviour of the system and the describer's intention.

1 Introduction

The Unified Modeling Language(UML)[1] is the most widespread object-oriented modeling language for specifying software systems. A statechart diagram in the UML can be used to describe the behavioural aspect of a system. It has very rich notions of structurization such as history, hierarchy and concurrency. These extensions for a simple state transition diagram contribute to efficient system designs.

Rich notations can also, however, be a cause of errors in a system design. Often, the behaviour description of the system is contrary to the describer's intention. For instance, concurrency is prone to produce unexpected combination of states. Thus, it is necessary to verify the correspondence between the meaning of the described model and the intention of the describer. Complex and often implicit rules of the statechart diagram cause another problem. The diagram must follow the rule of the UML statechart diagram. Most description rules given in the UML official document are formalized in OCL formulae. For instance, the following OCL formula means that initial state has at most one outgoing transition, and does not have any incoming transitions:

```
(self.kind = #initial) implies
                ((self.outgoing->size <= 1) and self.incoming->isEmpty))
```

This kind of constraint can be easily checked by static analysis. Some CASE tools such as Rational Rose[2] and Microsoft Visio[3] have that function and show the violations while designers are editing diagrams. There are, however, violations that we need to

detect utilizing dynamic analysis of the diagrams. For instance, the history indicator in Figure 1 is inappropriate. There is an execution path to enter the indicator with no corresponding history information. In this case, the indicator must have one transition connected to a default history state.

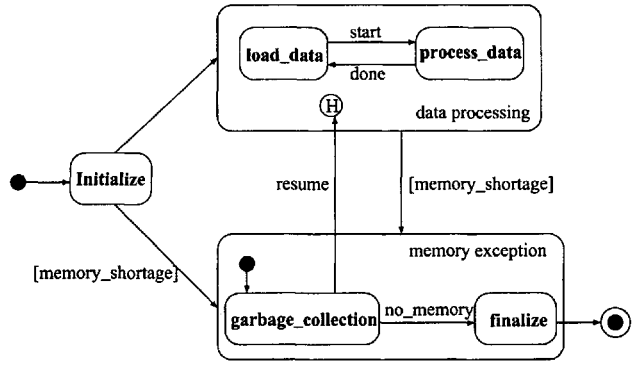


Figure 1: A statechart diagram

In this paper, we present a formal method to check the validity of statechart diagrams in the UML version 1.4. It performs several checks, such as detection of inappropriate history indicators, detection of conflicting transitions, detection of states that are never entered. These detection procedures contribute to getting well-defined behavioural model of the diagrams. It also performs model checking with Linear Temporal Logic. This contributes to detection of contradiction between the behaviour and the describer's intention. In the first step of the checking procedure, we translate an input statechart diagram into a flat finite transition system. While flattening the diagram, we check for the existence of inappropriate completion transitions, inappropriate history indicators, and conflicting transitions. After the flattening process, we do unreachable state detection and model checking on the flat transition system.

We are developing a tool that supports the describing processes of statechart diagrams based on verification methods that we present in this paper. Input of our tool is statechart diagram generated by third-party utilities, such as Rational Rose and Argo/UML[4], which output the data of diagrams in the standard XML modeling language for UML, called XMI[5].

Model checking for statechart diagrams in the UML have been intensively studied. In [6], Quaroni, Lavazza, and Venturelli gave an translation of the diagrams into TRIO logic[7]. The translation was straightforward, because they treated only a simple subset of structures of the state diagrams, and TRIO logic is very expressive. Because of the expressiveness of the logic, even a simple safety property cannot be verified. They say their main aim is not to build a verification method for the diagram, but just a translation method for the diagram into a formal language.

In [8], Kwon gave an translation of the diagrams into the SMV input language SML[9] by term writing techniques. This work enables the diagram to be verified by CTL model checking. But he did not treat either whole hierarchical structure or concurrency of states. In the translation, a composite state containing a history state is

just be encoded SML using SML variables instead of expansion of trace of states. This simple translation for history could be advantageous to model checking procedure for the diagram, since the translated representation could utilize some techniques for reduction of number of states in SMV verification process. But analysis for inappropriate use of history indicators cannot be done in the translation process. Similar translation approaches such as vUML[10], which is a translator for SPIN, and a translator for JACK[11] have the same problem.

The rest of this paper is structured as follows. In Section 2, we introduce some definitions, notations and procedures for the following discussion. Section 3 presents how to flatten statechart diagrams and how to check inappropriateness of the diagrams during the flattening process. Section 4 gives a translation procedure of flattened statechart diagrams into a finite transition system which can be utilized for model checking. Section 5 presents an LTL model checking procedure with the finite transition systems and an example of model checking on a UML statechart diagram. In Section 6, we conclude this paper with some remarks.

2 Preliminaries

In this section, we introduce some formal notations and procedures to use for verification of UML statechart diagrams. Linear Temporal Logic is used to specify designers' intention. Tableau method is applied to transform the diagrams into an Finite transition system, and is also used for model checking. Finite transition system is a graph that can be utilized in a well-known model checking procedure. Finally, we present a syntactic format of UML statechart diagram for the verification.

2.1 Language for Specification

We use Linear Temporal Logic(LTL) as a specification language for reactive systems. LTL is a classical propositional linear temporal logic which has "Until" operator[] as a modality. The assignment of each proposition corresponds to the occurrence of each event in a system. In this logic, it is possible to express infinite event sequence and eventuality properties of events.

Definition 2.1 (Formula)
$\mathbb{PV}$ is a set of proposition. $p \in \mathbb{PV}$ is a formula. If A and B are formulas, then $\neg A$, $A \wedge B$, and $[A]B$ are formulas. $A \vee B$, $A \rightarrow B$, $\langle A \rangle B$, $\Box A$, and $\Diamond A$ are abbreviations of $\neg(\neg A \wedge \neg B)$, $\neg A \vee B$, $\neg[A]\neg B$, $[\bot]A$, and $\langle \bot \rangle A$ respectively, where $\bot = p \wedge \neg p, p \in \mathbb{PV}$.

Definition 2.2 (Model)
$\mathbb{N}$ is a set of the natural numbers, $\geq$: $\mathbb{N} \times \mathbb{N} \rightarrow$ {true, false} is an ordering relation over the natural numbers. $\langle \mathbb{N}, \geq \rangle$ is a linear structure. If $s : \mathbb{N} \rightarrow 2^{\mathbb{PV}}$, then $\langle \langle \mathbb{N}, \geq \rangle, s \rangle$ is a model.

Definition 2.3 (Semantics)
The relation "formula f holds at i in M" is denoted by $M, i \models f$, where M is a model, and $i \in \mathbb{N}$ is a time.

The operators except [] have the same semantics as usual classical propositional logic. The semantics of [] is defined as $M, i \models [A]B \Leftrightarrow \forall j.(j \geq i)\ (M, j \models B)$ *or*, $\exists k.(k \geq i)(M, k \models A$ *and* $\forall j.(k > j \geq i)(M, j \models B))$.

Informally, $\Box A$ means that A is always true, $\Diamond A$ means that A is eventually true, $[A]B$ means that B is true until A is true, and $\langle A\rangle B$ means that B is eventually true until A is true.

Definition 2.4 (Satisfiability)
If $\exists M.\exists i.M, i \models A$, then A is satisfiable(SAT). For simplicity, we mention $M, 0 \models A$ just as $M \models A$.

2.2 Tableau Method

In this section, we review a Tableau method that is well known procedure for checking satisfiability of given temporal formulae. Tableau method consists of two phases: one is *graph construction*, and the other is *eventuality checking*. Eventuality checking is a process that checks whether the constructed graph really contains a model of given formulae. We construct FTSs(Section 4) utilizing the following decomposition procedure, which is a part of the graph construction procedure. The whole satisfiability procedure is utilized for model checking(Section 5).

2.3 Decomposition procedure

First, we describe the decomposition procedure that is utilized by the graph construction procedure. The decomposition procedure takes a set of formulae S, and yields a set of sets of formulae Σ.

1. (Initialization)Put $\Sigma := \{S\}$.

2. (Decomposition)Pick a formula $f_{ij} \in S_i \in \Sigma$ which is not a literal, and apply the following rules according to the form of f_{ij}.

(a) If f_{ij} is in $\neg\neg f$ form, then replace S_i with the following set:

$$(S_i - \{f_{ij}\}) \cup \{f\}.$$

(b) If f_{ij} is in $f_1 \wedge f_2$ form, then replace S_i with the following set:

$$(S_i - \{f_{ij}\}) \cup \{f_1, f_2\}.$$

(c) If f_{ij} is in $\neg(f_1 \wedge f_2)$ form, then replace S_i with the following sets:

$$(S_i - \{f_{ij}\}) \cup \{\neg f_1\}, \quad (S_i - \{f_{ij}\}) \cup \{\neg f_2\}.$$

(d) If f_{ij} is in $[f_1]f_2$ form, then replace S_i with the following sets:

$$(S_i - \{f_{ij}\}) \cup \{f_1\}, \quad (S_i - \{f_{ij}\}) \cup \{\neg f_1, f_2, [f_1]f_2\}.$$

(e) If f_{ij} is in $\neg[f_1]f_2$ form, then replace S_i with the following sets:

$$(S_i - \{f_{ij}\}) \cup \{\neg f_1, \neg f_2\}, (S_i - \{f_{ij}\}) \cup \{\neg f_1, f_2, \neg[f_1]f_2\}.$$

3. Remove all elements which include formulae f and $\neg f$ from Σ.

Decompose(S) denote the result of application of the decomposition procedure to a set of formulae S. For instance,

$$\mathit{Decompose}(\{\Box(a \to \Diamond b)\}) = \{\{\neg a, \Box(a \to \Diamond b)\}, \{\neg b, \Diamond b, \Box(a \to \Diamond b)\}, \{b, \Box(a \to \Diamond b)\}\}.$$

2.4 Graph Construction Procedure

Now, we show the graph construction procedure. First of all, we define *temporal formula* as follows:

Definition 2.5 (Temporal formula) *Let f_1,f_2 be formulae, then formulae that are in the form of $[f_1]f_2$ or $\neg[f_1]f_2$ are called temporal formulae. Let N be a set of formulae. The set of temporal formulae in N is denoted by $TempF(N)$.*

On the other hand, a *propositional formula* denotes a formula that is not a *temporal formula.*

The tableau graph construction procedure takes a set of formulae φ. It initializes the graph $G = \langle V, E\rangle$ with $\langle\{\varphi\}, \emptyset\rangle$ and expands the graph by using the decomposition procedure.

Repeatedly apply steps (a) and (b) to nodes in V until all nodes in V have been applied. For each node $n \in V$,

(a) Apply the decomposition rule to $TempF(n)$.

(b) Let Σ_n be the result of the application in (a). Add Σ_n to V and $\{\langle n, n'\rangle | n' \in \Sigma_n\}$ to E.

2.5 Eventuality checking procedure

Next, we show the eventuality checking procedure. The graph that constructed by the above procedure called *pseudo-model* graph. To check whether the graph contains a real model of given formulae, we need to verify that all the eventuality formulae presented in a model candidate are really satisfied. Eventuality formula $\neg[a]b$ is satisfied if there exists a reachable node in which formulae $\neg a$ and $\neg b$ are true.

To do that efficiently, we utilize a *strongly-connected-components analysis*[12]. The procedure analyzes the graph into maximal strongly connected components and identifies those that are *self-fulfilling.* A strongly connected component C is self-fulfilling if each node $N \in C$ has at least one successor, and for each eventuality formula is satisfied in C. The "one successor" condition avoids judging that a strongly connected component, which has only one node with no eventuality formula and no outgoing edge, is self-fulfilling. A node has no edge means that all the successors are eliminated by contradiction. Thus, this component cannot be an evidence of satisfiability. If there exists a self-fulfilling component in the graph, the given formula is satisfiable. Otherwise, the formula is not satisfiable.

2.6 Finite Transition System

We transform a UML statechart diagram into a finite transition system(FTS, for short) to verify some properties of the diagram. An FTS is a directed state transition graph. Each node of the graph is a set of formulae that represents whether each event of a system occurs or not at the state. The formal definition of an FTS is the following:

Definition 2.6 (Finite Transition System) An FTS is a triple $\langle N_F, n_0, T_F\rangle$, where

- N_F is a set of nodes. A node is a set of formulae.

- $n_0 \in N_F$ is the initial node,
- $T_F \subseteq N_F \times N_F$ is a set of transitions.

2.7 *Primitives of UML State Diagrams for Verification*

In this paper, we will use the following syntactic format for primitive UML Statechart Diagrams, which have simple states and transitions. We will explain the other elements such as history states and composite states in the next section.

A state is shown as a rectangle with rounded corners. The name of the state is described in the rectangle. A propositional formula is attached with square brackets beside the name. This formula means it holds until a next transition fires. We call this formula *invariant* of the state.

A transition is shown as a solid line originating from the source state and terminated by an arrow on the target state. It may be labeled by a transition string that has the following format:

$$\textit{trigger}\ \text{`['}\ \textit{guard}\ \text{`]'}\ \text{`/'}\ \textit{action}$$

Trigger, guard, and action are described as propositional formulas. Both the trigger formula and the guard formula represent the conditions for firing the transition. The action formula holds when the transition fires.

A transition fires if and only if both the corresponding trigger formula and guard formula hold at once. The difference between these two attributes is dependent on the intention of the describer of the diagrams. Occurring propositional variables in each formula may be different. There is, however, no essential difference between them for verification on the diagram in this paper. Firing transition consumes the corresponding trigger event. Thus, if there exists two transitions which have the same trigger e, and if one event e occurs, then one of these two transitions will fires.

In this paper, we will use the following notation to represent statechart diagrams.

Definition 2.7 (Statechart Diagram) A statechart diagram is a tuple $\langle S, T \rangle$, where

- S is a set of states. A state s is a triple $\langle f, c, h \rangle$. f is a propositional formula, which represents the *invariant* of the state. c is the immediate container of s. A container is also a state. $\mathit{Container}(s)$ denotes the immediate container of s. $\mathit{Container}^*(s)$ denotes the set of all containers of s.

$$\begin{aligned} \mathit{Container}^*(s) &= \mathit{Container}(s) \\ &\cup\ \mathit{Container}(\mathit{Container}(s)) \\ &\cup\ \mathit{Container}(\mathit{Container}(\mathit{Container}(s))) \cup \cdots . \end{aligned}$$

  h is a set of states, which is called *history information*. We will use it to expand history indicators. Initially, h is empty.

- T is a set of transitions. A transition t is a 4-tuple $\langle s, l, p, s' \rangle$, where s is the source state, l is the label. l is a 4-tuple $\langle t, g, a, ct \rangle$, where t is the trigger, g is the guard and a is the action. Each of these is a propositional formula. ct is boolean, if the transition corresponds to *completion transition*, then ct is true. We will explain the usage of ct later. Initially, ct is false. p is an integer which means the priority

in the firing order. p is initialized to the number of elements of $Container^*(s)$, since the priority is dependent on the nest level of the source state s. s' is the target state.

3 Transform Statechart Diagrams into Flat Statechart Diagrams

To verify properties of statechart diagrams, we transform input diagrams into flat statechart diagrams. A flat statechart diagram does not have states that indicate complex structures such as composite and concurrent but simple states and transitions. Here, we present how to flatten a statechart diagram. We expand composite states, concurrent states and history states in that order. While expanding, we check the existence of conflicting transitions, inappropriate complete transitions and inappropriate history indicators.

3.1 Composite States

A composite state is a state that contains other state vertices(Figure 2). It indicates hierarchical relation between states.

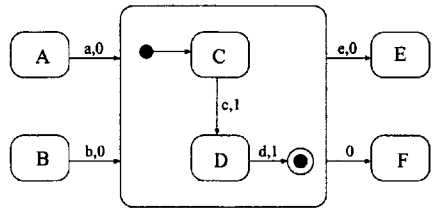


Figure 2: a, b, c, d and e are labels of corresponding transitions. The numbers beside the labels represent priorities of the transitions.

Incoming transitions of a composite state are entry points of the composite state. In the flattening process, target states of these incoming transitions are changed to the state that is indicated by the initial state of the composite state. The priority of the transition is not changed.

Outgoing transitions with explicit trigger events of a composite state (for instance, the transition labeled "e" in Figure 2) are exit points from each component state of the composite state. In the flattening process, we generate the copies of these transitions as many as the components. Each source state of these copies is changed to each component state. The priorities of the transitions are not changed.

An outgoing transition without an explicit trigger event is fired when the corresponding *completion event* occurs. Such a transition is called *completion transition* (for instance, the transition connected to the state "F" in Figure 2). A completion event of a composite state occurs if and only if the execution of the composite state is completed, in other words, the execution step reaches the final state of the composite state. Thus, in the flattening process, each target state of the transitions, whose target states are the final state, is changed to the target state of the completion transition. The completion transition flag ct of the label of the transition is set *true*. This will be used in the history indicator expansion phase. The priorities of the transitions are the same as those of the inner ones.

The flattened diagram of Figure 2 is Figure 3.

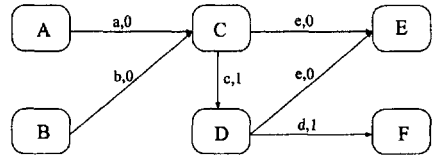


Figure 3: The flattened diagram of Figure 2

3.1.1 Inappropriate Complete Transition Use

During the flattening process, if a completion transition with no corresponding final state is detected, then the process outputs a warning "there should be a final state in the composition state". A transition like that does not violate the grammar of UML statechart diagram. It, however, will never fire since the corresponding completion event cannot occur. Thus, we consider the usage of the transition is inappropriate.

3.1.2 Conflicting Transitions

In situations where there are conflicting transitions, the selection of which transitions will fire is based in part on the priority. These priorities resolve some transition conflicts, but not all of them. If a conflict cannot be resolved by the priorities, an arbitrary transition is selected to fire. This means the behavior of the system is dependent on the way of implementation. Involuntary conflicts can be trouble spots. Thus, we think it is worth issuing a word of warning about the existence of conflicting transitions.

Not all conflicting transitions, however, can be causes of trouble. It is not necessary to check the existence of conflicts caused by concurrent composition of states, since we think this kind of nondeterminism must be intentional. Expansion of history indicator, we will explain in Section 3.3, does not generate any new conflict. Thus, it is enough to check the conflicts in this phase of the expansion process.

We explain how to detect the existence of conflicting transitions. The conflict occurs when multiple conditions of firing transitions, which have the same source state, hold at the same time. Therefore we check the following proposition on a statechart diagram $\langle S, T \rangle$:

$$\exists \langle s_1, \langle t_1, g_1, a_1, ct_1 \rangle, p_1, s_1' \rangle, \langle s_2, \langle t_2, g_2, a_2, ct_2 \rangle, p_2, s_2' \rangle \in T$$

$$\{(s_1 = s_2) \wedge (p_1 = p_2) \wedge (\mathit{Decompose}(\{t_1\} \cup \{g_1\} \cup \{t_2\} \cup \{g_2\}) \neq \emptyset)\}$$

There are conflicting transitions if the above proposition holds. $\mathit{Decompose}(p) \neq \emptyset$, where p is a set of propositional formulae, means that the conjunction of all elements of p is satisfiable. Note that it is not necessary to check eventualities for checking satisfiability of propositional formulas.

3.2 Concurrent States

If a composite state has two or more regions, the execution sequence of the composite state is the concurrent composition of each region(Figure 4). Composite transitions

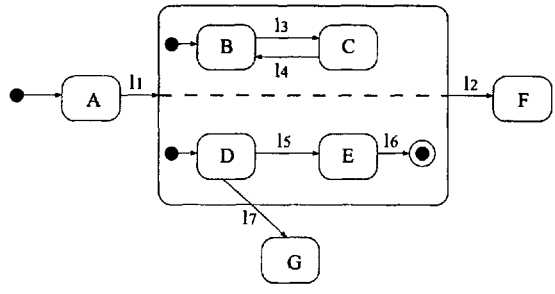


Figure 4: A concurrent composition.

described by *fork* and *join* pseudo-state can be alternative to the concurrent composite state. There is no practical difference between the semantics of these notations.

To flatten the concurrent structure, we basically construct a simple interleaving state sequence as follows. The concurrent composition of $R_1 = \langle S_1, T_1 \rangle$ and $R_2 = \langle S_2, T_2 \rangle$ is $R_c = \langle S_c, T_c \rangle$, where

$$\begin{aligned} S_c &= \{(s_1, s_2) | s_1 \in S_1, s_2 \in S_2\} \\ T_c &= \bigcup_{i=1,2} \{\langle (s_1, s_2), l, p, (s_1', s_2') \rangle | \langle s_i, l, p, s_i' \rangle \in T_i\}, \end{aligned}$$

where (s_1, s_2) is an abbreviation of $\langle f_1 \wedge f_2, c, h \rangle$, where $s_1 = \langle f_1, c, h \rangle$ and $s_2 = \langle f_2, c, h \rangle$. Actually, however, the UML statechart allows outgoing transition from concurrent regions. Thus, the expanding procedure we present in the following is rather complex. Let s_1 and s_2 be initial states of each concurrent region, and c be the immediate container of the regions. We invoke the procedure synthesize(s_1, s_2, c) to expand the concurrent regions. The return value of the procedure is the initial state of the concurrent composition. If there are more than two regions to composite, then we repeatedly apply the procedure to the state of the return value, the initial state of the remaining regions and the container c.

```
1   State synthesize(s1 ∈ S, s2 ∈ S, c ∈ S){
2       s := ⟨f1 ∧ f2, c, ∅⟩, where s1 = ⟨f1, c1, ∅⟩, and s2 = ⟨f2, c2, ∅⟩;
3       add s to S;
4       foreach({t|⟨s1, l, p, s'⟩ ∈ T or ⟨s2, l, p, s'⟩ ∈ T}){
5           if(c ∈ Container*(s1) ∧ c ∈ Container*(s2)){
6               if(t = ⟨s1, l, p, s'⟩) t' := ⟨s, l, p, synthesize(s2, s', c)⟩;
7               if(t = ⟨s2, l, p, s'⟩) t' := ⟨s, l, p, synthesize(s1, s', c)⟩;
8           } else t' := ⟨s, l, p, s'⟩;
9           add t' to T;
10      }
11      return s;
12  }
```

line 2: The invariant of a synthesized state s is the conjunction of that of s_1 and s_2. The container of s is the given container c. The history information is initialized for expansion of history indicators.

lines 5–8: If c contains both s_1 and s_2, successors of each state are synthesized in the asynchronous-interleaving way.

The flattened diagram of Figure 4 is Figure 5. Note that before flattening concurrent compositions, we must apply the procedure mentioned in Section 3.1.

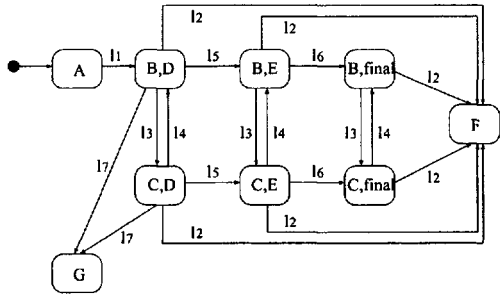


Figure 5: The flattened diagram of Figure 4

3.3 History States

A history indicator is used to restore a previous state configuration. A deep history indicator is shown as a small circle containing an 'H*'. A shallow history indicator is shown as a small circle containing an 'H'. When reached as the target of transition, a deep history indicator restores the full state configuration that was active just before the enclosing composite state was last exited. On the other hand, a shallow history restores the state within the enclosing composite state that was active just before the enclosing composite state was last exited. Note that a shallow one does not restore any substates of the last active state.

A transition may originate from the history indicator to the *default* history state. This transition is taken in case that the composite state had never been active before or the last exit point was the final state of the composite state.

To flatten a composite state containing history indicator, we add history information to each state, and distinguish states that have different history information. History information is represented by a set of states. If there is an outgoing transition from a state A which is contained by a composite state containing history state to a state C which is not contained by the same composite state, we attach the union of $\{A\}$ and A's history information set to the target state C as C's history information (Figure 6).

Consider there is an incoming transition from a state D, which has a set of states {B, E} as its history information, to a history indicator contained by a composite state F(Figure 7). Let I be the intersection of D's set of history information and the set of all substates of F. In the case in Figure7, $I = \{B\}$. The number of I's elements is at most one. If I has an element, the transition's target is the element of I with history information "(D's history information)$\backslash I$". In the case in Figure7, the target is B with history information $\{E\}$. If I is empty, then the transition's target is the

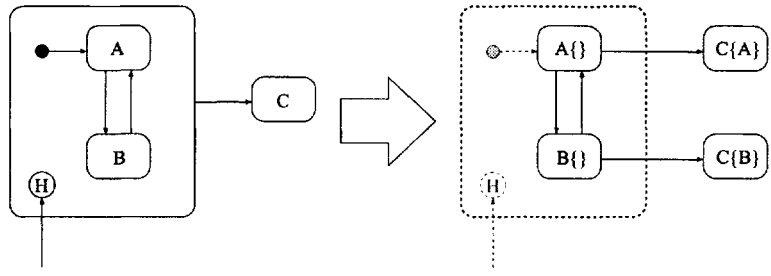


Figure 6: If the exit point of the composite state is the state B, then the target of the transition is the state C with history information $\{B\}$.

default history state of the history indicator with the same history information as C's. A default history state is the state that is the target of the transition originating from a history indicator.

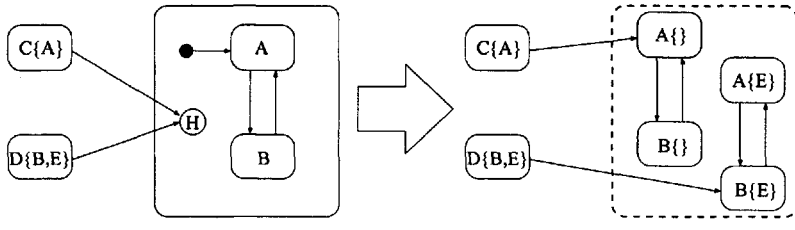


Figure 7: The state C has history information $\{A\}$. Thus, the target of C is the state A with history information $\emptyset$.

3.3.1 Inappropriate History Use

If a transition terminates on a history indicator, the target of the transition is the most recently active state prior to this entry, unless the most recently active state is the final state or this is the first entry into this state. In the latter two cases, the default history state is entered. In case no such transitions is specified, the situation is prohibited and its handling is not defined[1].

In this flattening process, if an intersection of history information and the set of states, which are contained in a composite state is empty, and if the corresponding history indicator has no outgoing transition, then the illegal situation is detected. In that case, the process outputs the warning "the history indicator must have one default history state", and terminates.

3.3.2 Expanding Procedure for History Indicator

The following is an expanding procedure for history indicators. We invoke the procedure *expand*$(s, \emptyset)$, where s is the initial state of the whole statechart diagram. Figure 9 is

the flattened diagram of Figure 8.

```
1   State expand(c ∈ S, h ∈ 2^S){
2       s := ⟨f, c, h⟩, where c = ⟨f, c, h'⟩;
3       if(s ∈ S) return s;
4       add s to S;
5       foreach({t|⟨c, l, p, s'⟩ ∈ T}){
6           if(∃cs ∈ S{cs ∈ Container*(s)
7                       ∧cs contains a history indicator hi
8                       ∧cs ∉ Container*(s')}){
9               if(ct of l is true) h'' := h\{s''|cs ∈ Container*(s'')}; else{
10                  if(hi is a deepHistory) h'' := (h\{s''|cs ∈ Container*(s'')}) ∪ {s};
11                  if(hi is a shallowHistory) h'' := (h\{s''|cs ∈ Container*(s'')})
12                              ∪{cs'|cs = Container(cs')
13                                  ∧cs' ∈ {s} ∪ Container*(s)};
14              }
15          }else h'' := h;
16          if(s' is a history indicator){
17              th := h'' ∩ {s''|s' = Container(s'')};
18              if(th = ∅){
19                  if({s'''|(s', l, s''') ∈ T} has exactly one element){
20                      r := expand(s''', h'');
21                  }else error_exit("s' must have one outgoing transition");
22              }else{
23                  nh := the element of th;
24                  remove nh from h'';
25                  r := expand(nh, h'');
26              }
27          }else r := expand(s', h'');
28          t' := (s, l, r), where t = (c, l, s'');
29          add t' to T;
30      }
31      return s;
32  }
```

lines 6–8: The condition holds when the target of the transition t is in the outside of a container that has a history indicator.

line 9: If t is a complition transition, the history information on the completed states is removed.

line 10: If the history indicator is deep, add the state s to the history information.

line 11: If the history indicator is shallow, add the state that contains s and has the same container as the indicator to the history infomation.

lines 17–21: If the target state s' is a history indicator and there is no state to restore in the history information, then if there exists one "default history transition" t', replace s' with the target state of t', otherwise output an error message.

lines 23–25: If the target state s' is a history indicator and there is a state to restore in the history information, then remove the state from the history infomation and replace s' with the state.

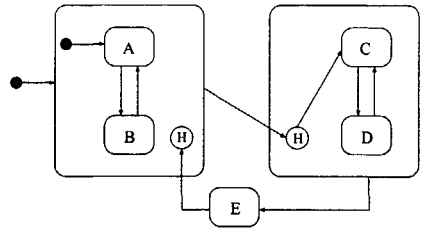


Figure 8: Two composite states have history indicators.

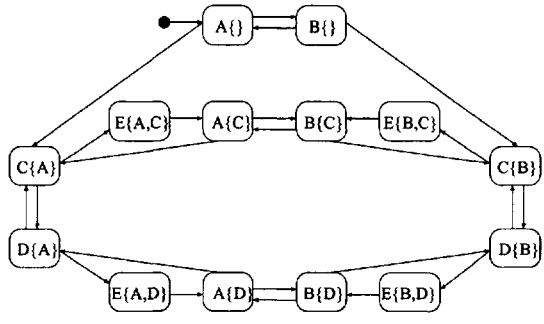


Figure 9: The flattened diagram of Figure 8.

4 Transform Flat Statechart Diagrams into FTSs

In this section, we present how to transform a flat UML statechart diagram into an FTS. To transform the diagram, we have to take these things into consideration:

- Until a transition of a state fires, the invariant of the state holds.
- When both the trigger and guard conditions of a transition hold, the transition fires.
- In situations where there are conflicting transitions,the selection of which transition will fire is based on the priorities of the transitions.
- Immediately after a transition fired, the corresponding action of the transition occur.

Figure 10(left) is a part of a flat UML statechart diagram. Here, we explain the transformation by focusing on "State1" in the Figure. First of all, we prepare the unique *ID variable* for each state. An ID variable is a propositional variable, which is used to distinguish nodes of different state which has the same guard, trigger and action. It is necessary, for instance, to detect states that are never entered during any execution of the system. Let ID_0, ID_1, ID_2, and ID_3 be ID variables for State0, State1, State2, and State3, respectively.

For State1's incoming transition, which is labeled as "$T_1[G_1]/A_1$", we create an FTS node named "Node0" in the Figure 10(right). The node is a set of formulae $\{ID_0, S_{c0}, T_1, G_1\}$. This represents the situation that the trigger T_1 occurred, and the guard G_1 is hold at the state "State0". Note that S_{c0} holds while the execution step is in the state "State0".

For State1 and its transitions, we create 2+n nodes, where n is the number of outgoing transitions of State1. Here, the corresponding nodes are Node1, Node2, Node3 and Node4. Node1 is a set of formulae $\{ID_1, S_{c1}, A_1\}$. This represents that the transition to State1 fired, and just then the action of the transition invoked.

Node3 and Node4 are the corresponding nodes for each outgoing transition of State1. To transform outgoing transitions, we sort the transitions in priority order. Suppose the transition connected to State2 has higher priority, Node3 and Node4 are $\{ID_1, S_{c1}, T_2, G_2\}$ and $\{ID_1, S_{c1}, T_3, G_3, \neg(T_2 \wedge G_2)\}$, respectively. We add negative formulae of the guard and trigger of prior transitions in the order.

Node2 is the waiting state of State1, which waits until one of the transition fires. Thus, naturally Node2 is $\{ID_1, S_{c1}, \neg(T_2 \wedge G_2), \neg(T_3 \wedge G_3)\}$. The other nodes, Node5 and Node6, represent that states just after the transitions fired same as Node0. These nodes are $\{ID_1, S_{c1}, A_2\}$ and $\{ID_1, S_{c1}, A_2\}$.

Finally, we decompose each node n in the creating FTS by the decomposition procedure *Decompose*, and replace n with the result of *Decompose*(n). In this process, we eliminate contradictory nodes from the FTS, and make the assignment for each proposition in each node explicit. As a result of this process, the FTS can be utilized for the model checking procedure we will explain in Section 5.

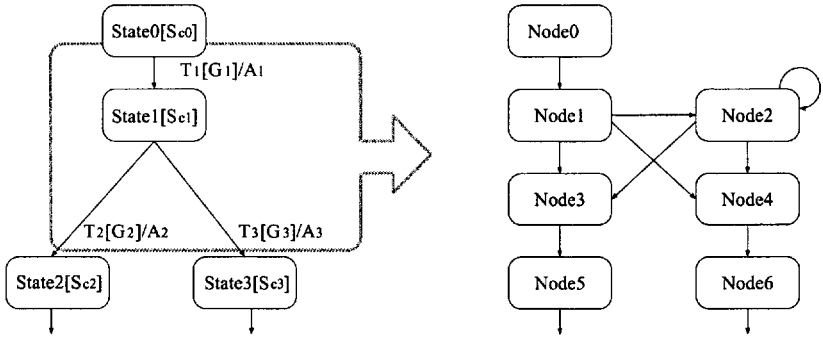


Figure 10: The left figure is a part of a flat statechart diagram. The right one is a part of a translated FTS that corresponds to the inner part of the broken line in the left diagram.

4.1 Transformation Procedure

In this section, we present a general procedure for transformation of a flat UML statechart diagram $\langle S, T\rangle$ into an FTS $\langle N_F, n_0, T_F\rangle$. The procedure is the following. Set $n_0 := \emptyset$, and add $\langle n_0, \text{constructFTS}(i, \top)\rangle$, where i is the initial state of $\langle S, T\rangle$, to T_F. Finally decompose the FTS by the procedure decompFTS(). The function *getID*(s) in the procedure constructFTS yields the ID variable of a state s.

```
1   Node constructFTS(⟨f, c, h⟩ ∈ S, a_p ∈ Formula){
2       s := ⟨f, c, h⟩;
3       n := {getID(s), f, a_p};
4       if(n ∈ N_F) return n;
5       add n to N_F;
6       Set of formulae NF := ∅;
7       Set of nodes SN := ∅;
8       foreach(⟨s, ⟨t, g, a, ct⟩, p, s'⟩ ∈ T in descending order of p){
9           n' := {getID(s), f, t, g} ∪ NF;
10          add n' to SN;
11          add n' to N_F;
12          add ⟨n, n'⟩ to T_F;
13          add ⟨n', constructFTS(s', a)⟩ to T_F;
14          NF := NF ∪ {¬t, ¬g};
15      }
16      n'' := {getID(s), f} ∪ NF;
17      add ⟨n'', n''⟩ to T_F;
18      foreach(n''' ∈ SN) add ⟨n'', n'''⟩ to T_F;
19      return n;
20  }
21
22  Void decompFTS(){
23      foreach(n ∈ N_F){
24          remove n from N_F;
25          foreach(n' ∈ Decompose(n)) add n' to N_F;
26          foreach(⟨n, n''⟩ ∈ T_F){
27              remove ⟨n, n''⟩ from T_F;
28              foreach(n' ∈ Decompose(n)) add ⟨n, n'⟩ to T_F;
29          }
30          foreach(⟨n'', n⟩ ∈ T_F){
31              remove ⟨n'', n⟩ from T_F;
32              foreach(n' ∈ Decompose(n)) add ⟨n', n⟩ to T_F;
33          }
34      }
35  }
```

5 LTL Model Checking

In this section, we explain an LTL model checking procedure on FTSs. This procedure checks whether an FTS F, which is constructed from an input UML statechart diagram,

satisfies a specification f described in LTL. If the FTS does not satisfy the specification, then the procedure outputs a counter example, which means an execution path that violates the specification.

We apply a well-known model checking procedure. First of all, we construct the product of the FTS F and the negated specification formula $\neg f$. In the construction steps, we cut nodes that contain mutually complemental formulae. Then, we apply the strongly-connected-components analysis that we explained in Section 2.5 to the product. If there exists a self-fulfilling maximal strongly connected component(MSCC) in the product, then there exists at least one counter example against the specification. A counter example in the product is a concatenation of a path from the initial node to an entry point of a self-fulfilling MSCC, and a loop path in the MSCC.

5.1 Dead State Detection

We can detect UML states that are never entered during execution. We have already eliminated all the nodes that contain mutually complemental formulae from an FTS by the procedure decompFTS. Thus, all the reachable nodes from the initial node represent all the possible states in the execution steps. If there exists a state whose ID variable does not appear in the reachable nodes, the state is not reachable in any execution.

5.2 Execution Result

We are developing a verification tool for UML statechart diagrams based on the method we explained in this paper. In this section, we present an example of model checking by the tool. Input of the checking is shown in Figure 11. This is a simple model of 2-dining-philosopher problem. Here, $\Box(\Diamond Aeats \wedge \Diamond Beats)$ is the specification to be verified. This formula means that both philosopher A and B infinitely often eats.

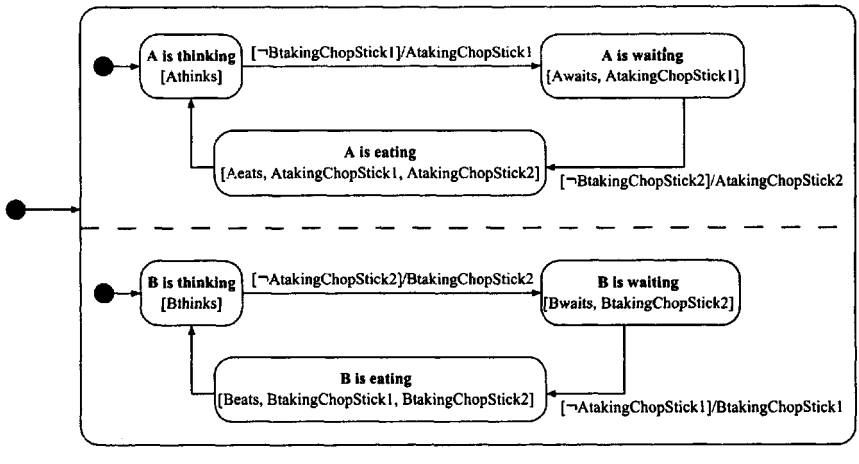


Figure 11: Input: a simple model of 2-dining-philosopher problem

The result of the model checking was "*false*", i.e. there exists at least one execution sequence that violates the specification. The tool output the following counter example.

1. $Athinks \wedge Bthinks$
2. $Athinks \wedge Bwaits \wedge BtakingChopStick2$
3. $Athinks \wedge Beats \wedge BtakingChopStick1 \wedge BtakingChopStick2$
   Back to 1.

This sequence means that only the philosopher B continuously eats and A starves.

We executed the checking on a PC, that was on a Pentium III 600 MHz CPU and 192M-byte memory. It took 6 seconds.

6 Conclusion

We presented a procedure to check the validity of UML statechart diagrams. The procedure checks the existence of conflicting transitions, inappropriate history indicators, inappropriate complete transitions and dead states while constructing model graphs of UML statechart diagrams for LTL model checking. These checks contribute to removal of both ambiguous and undefined descriptions of behaviour. After the construction of a well-defined model, the procedure executes model checking with specifications written in LTL for checking the consistency between the behaviour of the system and the describer's intention.

We are developing a tool that supports the describing processes of statechart diagrams based on the procedure. Input of the tool is statechart diagram described in XMI, which is the OMG standard XML modeling language for UML. Thus, you can utilize the tool with widely used CASE tools, such as Rational Rose and Argo/UML.

References

[1] Object Management Group(OMG). Unified modeling language specification version 1.4. http://www.uml.org/, 2001.

[2] Rational. http://www.rational.com, 2002.

[3] Microsoft. http://www.microsoft.com, 2002.

[4] Regents of the University of California. Argo/UML. http://argouml.tigris.org, 1998.

[5] Object Management Group(OMG). XML metadata interchange specification version 1.0. http://www.omg.org/technology/documents/formal/xml_metadata_interchange.htm, 2000.

[6] G. Quaroni L. Lavazza and M. Venturelli. Combining UML and formal nortations for modelling real-time systems. In *Proceedings of Joint 8th European Software Engineering Conference (ESEC) and 9th ACM SIGSOFT International Symposium on the Foundations of Software Engineering (FSE)*, pages 196–206. ACM, 2001.

[7] M. Roncato A. Morzenti, E. Ratto and L. Zoccolante. TRIO:A logic formalizm for the specification of real-time systems. In *Proceedings of Euromicro Workshop on Real Time*, pages 26–30. IEEE, 1989.

[8] G. Kwon. Rewrite rules and operational semantics for model checking UML statecharts. In *Proceedings of* $\langle\langle UML\rangle\rangle 2000$, volume 1939 of *LNCS*. Springer Verlag, 2000.

[9] K. L. McMillan. *Symbolic Model Checking.* Kluwer Academic Publishers, 1993.

[10] J. Lilius and I.P.Paltor. The semantics of UML state machines. In *Proceedings of* $\langle\langle UML\rangle\rangle 1999$, volume 1723 of *LNCS*. Springer Verlag, 1999.

[11] D. Latella S. Gnesi and M. Massink. Model checking UML statechart diagrams using JACK. In *Proceedings of the IEEE International Symposium on High Assurance Systems Engineering*, 1999.

[12] Z.Manna H.McGuire, Y.Kesten and A.Pnueli. A decision algorithm for full propositional temporal logic. volume 697 of *LNCS*, pages 97–109. Springer Verlag, 1993.

Information Modelling and Knowledge Bases XIV
H. Jaakkola et al. (Eds.)
IOS Press, 2003

Context and Uncertainty

A. T. BERZTISS
Department of Computer Science, University of Pittsburgh, Pittsburgh, PA 15260, USA
(e-mail: alpha@cs.pitt.edu; fax: +412-624-8854)

Abstract. Taking the standard definition of context as the pair $<w, t>$, where w is a slice of the world at time t, we investigate what form w should take so that it can help reduce uncertainty in information systems. We find that for some types of uncertainty the suitable w is a data set, and that for other types it is a set of rules. We also consider the significance of context in the development of product lines of software.

1. Introduction

Context-dependence arises in various situations. In text processing the interpretation of an ambiguous term requires examination of its context, i.e., of its surrounding text. In artificial intelligence the frame problem is one of context: if a book is in a box (fact A), and the box is moved (fact B), then so is the book, but this has to be inferred from facts A and B, where fact A defines a context. In a payroll system, the salary of an employee may be determined by the cost of living in the city in which the employee works. Whether distance is measured in millimeters or lightyears depends on the application. This list can be extended on and on.

The interpretation of the word *context* shows considerable variety – indeed, the definition is itself context-sensitive. Linguists may see it as a psychological construct, a subset of a hearer's assumptions about the world that is used in interpreting an utterance [1]. To an organizational theorist context is a social environment in which actions are taken [2]. To a sociologist, context is provided by macrosocial forms, such as gender, national ethos, and economic maturity of a society [3]. Context has been studied as an axiomatic system [4], and the class of context-dependent grammars provides another example of a formal approach [5].

To our knowledge there is no broad survey of context-dependence, and we shall not provide such a survey here either. But we refer to [6], where Brezillon has looked at some areas relevant to what we will be primarily concerned with, namely context in information processing.

Penco [7] distinguishes between objective and subjective context. Objective or ontological theory of context is concerned with features of the world, such as time, place, source of information, and so forth. Subjective or epistemic theory of context is concerned with a subjective cognition of the world. The latter has little relevance to information systems. Subjective decisions may be based on knowledge derived from an information system, but the system itself should be built with optimal objectivity as the goal. McCarthy [8]

formalizes objective context as the pair $<w, t>$, where w is a slice of the world at time t. We will use this definition of context in our work.

Our main concern will be the reduction of uncertainty in information by means of context. Thus, our motivation is different from that of, say, McCarthy and Buvac [9], who use a modal logic of context for reasoning in artificial intelligence systems. Consider an entity e, which may be a concept, object, or event. We have no real understanding of e unless we know how it relates to its context. A representation of e by itself in an information systems setting is a datum or data set that can be interpreted any way we like. This is equivalent to saying that it totally lacks interpretation, and that therefore it is meaningless to speak of an uncertainty of interpretation. Only when there already is some context can we begin to talk of uncertainty.

In a survey of uncertainty [10] we list 22 types of uncertainty. We shall investigate what context is needed to reduce uncertainty with reference to this list. As regards uncertainty in general, we regard an overview by Klir and Yuan [11] the best reference. Despite its title, this encyclopaedic survey with 1731 references goes well beyond just fuzziness. Dyreson [12] has generated a bibliography of uncertainty management for information systems. Measures of uncertainty in expert systems are considered in [13]. A survey by Parsons [14] on imperfect information in data and knowledge bases can also be recommended.

In Section 2 we consider the concept of context in the rather narrow domain of information processing and knowledge management. Sections 3 and 4 deal with reduction of different types of uncertainty. We note that each widening of the context should reduce uncertainty to a greater degree. However, often there is inherent uncertainty that cannot be eliminated. We recommend that more research be carried out on methods for detecting such situations.

In Section 5 we take a different view. Given a system S that is to be embedded into environments E_1 and E_2 (the contexts). There are two aspects to this situation. First, permit S to behave differently in the two environments, and determine how this difference of behavior is to be achieved. Second, make sure that S has the same behavior in the two environments. This means that the environment is to be modified in one of the cases. Our problem is to determine how this modification is to be carried out.

As is customary, we conclude with a summary of our findings. Section 6 is this summary. In it we also include suggestions for further research.

2. Components of context

As noted earlier, we shall take the pair $<w, t>$ for our definition of context. This, however, raises two questions. First, what exactly is w? Second, with what degree of certainty can we accept a given w and t? As regards the first question, Pomerol and Brezillon [15] define context as "the set of all the knowledge that could be evoked by a human being facing a situation, assuming that he has an unlimited time to think about it." Since we are considering computer systems rather than humans, we have to make a prior determination of how much knowledge is relevant, and what this knowledge is.

As regards the second question, if the degree of uncertainty affecting an entity e depends on an uncertain $<w, t>$, then, as regards a particular $v \in w$, we can use $<\{e\} \cup (w-\{v\}), t>$ to reduce the uncertainty of v, and so forth in an iterative mode. Iterative processes that hopefully ultimately converge are common in numerical computation. However, designing such numerical processes is a very difficult task, and is likely to be even more difficult in our setting.

In our study, since we have already looked at temporal uncertainty in some detail [16], we will concentrate on the w component of $<w, t>$. The w can take two forms. First, it can take the form of a data collection, which is provided with semantics by means of a concept graph [17]. This graph is directed, and has precisely one node with zero indegree [18]. Normally e would be associated with this node, but if we have an iterative reduction of uncertainty, then more than one node has to represent e. Second, w is expressed as a set of rules that e has to obey. Of course, the arcs of a concept graph can be regarded as a kind of rule set. So, the rules complement the concept graph. Coversely, a concept graph can clarify the terminology used in the rules.

In Sections 3 and 4 we consider 14 of the 22 types of uncertainty of [10], grouped into nine classes. Section 3 deals with cases in which uncertainty is to be reduced by reference to additional information, and this can be guided by a concept graph. In Section 4 we consider cases in which rules constitute the context that adds to the understanding of the items of information under consideration. There is no sharp distinction between the two classes of approaches. In every instance we are in search of meaning, and rules may be necessary to interpret a concept graph correctly, or, conversely, we may have to gather additional information to be able to interpret the terminology used in the rules.

3. Context broadening

Here we consider five types of situations in which the contextual information surrounding an entity under consideration is initially inadequate to permit the precise interpretation of the entity. Additional information is needed to improve precision, but we have to keep in mind the "principle of diminishing returns," which in this case means that at some point broadening the context no longer improves precision, i.e., that the added context components have little or no relevance.

3.1. Inconsistency. We can distinguish two cases. First, a lay person questions a prevailing opinion. With the spread of Internet usage, such questioning has become quite common. Although there can be instances in which established opinion should be modified, information overload makes it very difficult to check out all such challenges, and they are – justifiably – ignored. Second, two or more experts are in disagreement. Robinson and Pawlowski [19] make a distinction between inconsistencies, which are technical, and conflicts, which have a social basis. We adopt this distinction, and consider inconsistencies to be resolvable. Suppose that several experts have initially provided different cost estimates for the same project. However, when they become aware of what factors each expert used to arrive at an estimate, they are likely to modify their estimates, and ultimately arrive at a consensus estimate. Group decision support systems have been developed to help arrive at the consensus (see, e.g., [20]). Conflict arises most often in economics, dietary science, and medicine. For example, two physicians may disagree on the most appropriate treatment for a patient, particularly when the patient is subject to several conditions. Since experimental evidence is often inadequate to validate one or other of the competing domain models on which the experts base their opinions, attempts to resolve the conflict need not succeed. But we still have a well-defined context, namely the world of domain models in which the experts operate. In any case, an attempt should be made to arrive at consensus, i.e., at a reduction of this context to a single domain model, but cases in which the experts have irreconcilable world views should be recognized early, so that not too much futile effort is expended unnecessarily.

Related to inconsistency is *Cause/effect uncertainty*. An effect is observed that could be due to different causes, singly or jointly. This is quite common in medical diagnosis.

Analysis methods based on Bayesian networks [21] and fault trees [22] have been developed to deal with such situations.

3.2. Either/or uncertainty. An instance of this problem is a meter that has not changed its reading over an extended period of time. The reading may be steady because the state that it monitors is stable, or the lack of change may be caused by a fault in the meter. Which of these possibilities is valid can be established by perturbing the state, which is the context for the meter. A perturbation of the state should result in a change of the reading if the meter is not faulty, or in no change otherwise. We looked at this problem in some detail in [23]. Another example: the rumor "Company X will show a loss this quarter" may be true or false, but we do not know which. Additional information needs to be gathered, i.e., the context, which in this case is financial data, has to be broadened. A deeper problem with this type of uncertainty is that we first have to realize that there is in fact a problem. In the case of the meter, the problem would be discovered if successive readings were regularly compared. As regards the rumor, we would have to label items of information appropriately, thus distinguishing rumors from facts. Gathering of additional information would be initiated only in the case of rumors.

3.3. Fuzziness. To continue with the Company X example, we may determine that instead of a loss, there will be a small profit. The term "small" is fuzzy, and a fuzzy quantity is represented by a fuzzy membership function (see, e.g., [11]). Here we have the same context as under the either/or uncertainty, and we would like to use this context to arrive at a quantitative value for the profit. Using the approaches of fuzzy set theory and fuzzy logic, and various analysis techniques, such as Bayesian estimation [24, 25], we could get such a quantitative value. We must understand that the information that would enable us to arrive at a precise value of the profit cannot be complete before the end of the quarter. Hence the expenditure of resources on additional information that would not necessarily improve the accuracy of the estimate has to be carefully controlled.

The question arises whether we need a quantitative estimate at all. Zadeh has introduced the technique of "computing with words" [26], and currently there is considerable research activity in this area [27]. Let us see how this applies to our case. We are interested in the profit estimate for some purpose. The purpose may be to purchase shares in the company. If this is the case, a precise estimate of the profit is not necessarily needed – if there is a "small" profit, we are willing to buy a "few" shares, provided the price is "low". Of course, the terms "few" and "low" have to be converted into numerical values when we are ready for actual purchase of shares. This can be done by comparison to earlier share purchases in comparable situations.

Another aspect of the dependence of fuzzy terms on context is that the same term can have different interpretations in different contexts. Thus "cold" north of the Arctic Circle and close to the Tropic of Cancer would be represented by different fuzzy membership functions, the former centered on a much lower temperature value than the latter.

3.4. Vagueness. In [11] vagueness is considered a sub-category of fuzziness. We consider vagueness to be distinct from fuzziness. Under our interpretation a property is fuzzy if a measure exists for this property. Examples of measures: cold/temperature, tall/height, old/age. In contrast to fuzzy terms, we call those terms vague for which no measurement process can exist. In "People feel uncomfortable when it is hot" the term "hot" is fuzzy, but "uncomfortable" is vague – we have no dependable way of measuring discomfort. Here our first objective should be to convert a vague term into a fuzzy term. For example, "user-friendliness" is vague, but "usability" is fuzzy in that procedures exist for measuring usability (for a benchmark approach to usability measurement see [28]). A direct approach to finding a fuzzy substitute does not appear to work for the term "uncomfortable". Instead, here the context could be an ontological decomposition of this concept into a set of fuzzy concepts. Such a decomposition would show that "hot" is one aspect of

"uncomfortable", which leads to the observation that "People feel uncomfortable when it is hot" is no more than part of an ontological explication of "uncomfortable".

Note that subjectivity is not confined to vagueness. A vague expression, because it is not linked to any measurement, is necessarily subjective, but the interpretation of a fuzzy term can also be subjective. Thus, somebody at age 20 may consider age 40 as "old", but a 40-year old is unlikely to do so. This is unlike the interpretation of coldness at different geographic locations we considered earlier – there the fuzzy membership functions can be constructed objectively from differences in temperature data at different locations.

3.5. No knowledge as negation. In logic programming, if the fact X is not in the knowledge base, not(X) is assumed true. For example, if the knowledge base does not contain "Bad Schandau is close to Dresden", the query "Is Bad Schandau close to Dresden?" will result in the answer "No". Here we have a situation similar to what we faced regarding the either/or uncertainty. The first step towards resolving this problem is to determine whether there is a problem. Thus, if Bad Schandau were not close to Dresden, the given answer would be appropriate. But Bad Schandau is close to Dresden, and the misleading response is due to the closed world assumption under which any statement not supported by a "closed world" data base is false. The problem is that the closed world assumption is essential because otherwise the set of information needed to arrive at correct answers in a general question-answering system would be unbounded. Reduction of the context from the universe of total knowledge to the domain of towns and distances certainly helps. Thus, if we consider n towns, we can set up an $n \times n$ matrix of distances. Then, if we were to ask a distance question about a town not in our set of n towns, or there were no entry in the matrix for a pair of towns, we would be given a "Don't know" response. In the no-entry case, the problem has become the null-value problem of data bases, which has been thoroughly investigated [29, 30]. Note that "close to" is fuzzy, so that an interpretation of what distance range defines "close to" depends on the context.

When there is no direct knowledge, we can sometimes infer new knowledge. For example, if we know that a is a sibling of b, and b is a sibling of c, we can infer that b and c are siblings of a. Unfortunately most relations of interest are not transitive. For example, a is close to b, b is close to c, and c is close to d, do not imply that a is close to d. The "close to" relation is a kind of similarity relation. But, as discussed in detail by Rodriguez and Egenhofer [31], many relations that determine some kind of similarity are not even symmetric. Thus "a hospital is a kind of building" does not imply "a building is a kind of hospital". We have to have a context awareness which is to tell us that buildings form a broad class of which hospitals are a subclass.

4. Context defined by rules

In this section we consider information that cannot be properly interpreted unless it is augmented with rules. The rules can be of two forms: definitions and constraints. A definition gives meaning to a concept in terms of concepts that are assumed to be understood. For example, in stating that context is the pair $<w, t>$, we assume that the concept of a pair and the notation <...> are both understood. A constraint is a rule that an entity has to satisfy. An example: "An employee transferred from site a to site b must arrive at b within seven days of being notified of the transfer." Knowing that an employee was notified of the transfer on January 5, we can infer that on January 14, say, the employee is located at b. A special class of rules is expressed in deontic logic: p may happen; p must happen; p may not happen (for a survey see [32]).

4.1. Interpretation uncertainty. Often we use terms without a clear understanding of

what they mean. As a student, the author had summer employment in an insurance company in which three different interpretations of age were in use – "age-last-birthday", "age-nearest-birthday", and "age-next-birthday". It made a difference if the insurance premiums were looked up in a table for age-next-birthday, but the submitted age was age-last-birthday.

In Section 3.3 we considered fuzziness. We noted that "tall" is fuzzy because we can associate a measure, height, with this concept. However, the height range for tall buildings differs from that for tall people. We have to be aware of the context in which the term "tall" is used, and within this context a fuzzy membership function, which we regard as a graphically expressed rule, defines the degree of tallness for different height values. The form of this function will be determined by the context – it will be different for buildings and people, and even for people from different societies.

Interpretation uncertainty often arises with *rounding*, particularly when monetary values are converted. For example, when we say that the flood damage has been USD 5,000,000, we should round off the result after we have made a conversion to euro. On the other hand, we need the exact value if we convert a payment of USD 5,000,000 to euro. Here we first need to broaden context knowledge to determine what kind of situation we are dealing with, and then apply an appropriate conversion rule (with or without rounding). Note also that the t component of $<w, t>$ becomes important here because of fluctuating conversion rates.

4.2. Multiple options. People often use different addresses – the most common example is the use of work and home addresses for different purposes. Rules determine the conditions under which a particular address is to be used. Another example relates to travel. A person could travel from, say, Karlsruhe to Bad Schandau by car or by train with different options for each of these modes of travel. The selection of one particular mode of travel would be determined primarily by a combination of cost and convenience. If the determination of the travel mode is to be made by a computer program, then there has to be a rule that selects the mode. But this rule depends on context information, which has to be provided by the traveler. The traveler has to indicate the relative importance of cost and convenience. Moreover, just as we had to carry out an ontological decomposition of the term "uncomfortable" in Section 3.4, so we have to do the same here with the term "convenience". Some components are travel time, comfort, and scenic features of the route. But "comfort" and "scenic features" are still vague, and, to be able to deal appropriately with the importance of scenic features, additional information about the travel route may have to be gathered.

4.3. Temporary exceptions. We discussed earlier the rule "An employee transferred from site a to site b must arrive at b within seven days of being notified of the transfer." Suppose an employee x does not get to b within the seven days. An exceptional situation arises because inferences made on the basis of the rule may be invalid, but it is not a temporary exception under our definition. In our terminology an exception is temporary if it suspends a rule for a specific period of time, and the suspension is itself a rule that has a general nature. The lateness of x does not satisfy these criteria. Now suppose that a rule requires each employee to be localized at a particular site at all times. We can deal with this situation in several ways. One way is to have an interpretation rule that defines "localized" in such a way that the employee becomes localized at b as soon as the transfer is announced. This avoids the temporary exception problem altogether, but we can equally well define a temporary exception for the period in which the employee has left a, but has not yet arrived at b. One such temporary exception allows an employee to be associated with two sites for a limited time – this temporary exception is in force for the duration of the transfer period. A different temporary exception lifts the localization requirements for the transfer period: during this time the employee is not localized at any site. In general,

the context is a rule that defines the period of time for which a temporary exception is allowed. Whereas most exceptional situations contribute to uncertainty, a temporary exception reduces uncertainty.

4.4. Trends. The interpretation of trends is very difficult. An upward trend has been observed in global temperatures. This may be due to chance or to some underlying reason. Suppose a curve of the form $y = a + bx$ is fitted to the temperature data. Statistical analysis allows a determination to be made how significantly different the slope parameter b is from zero. This is the rule part. Next, if b is significantly different from zero, which in this case it appears to be, we have an instance of the cause/effect uncertainty we looked at in Section 3.1. It has to be kept in mind that there always remains a non-zero probability, however small, that the non-zero value of the slope is due to chance. Assuming that it is not, the determination of the cause can be seriously colored by subjectivity. Is the temperature increase indeed due to an increase of the carbon dioxide component in the atmosphere? Is the temperature increase bad or good? We do not really have answers to these questions, so that we have a case of expert conflict, also discussed in Section 3.1.

Obscurity is related to this. By this we mean that a trend is obscured by temporal variations. For example, monthly income at a ski resort is usually higher in the winter months, which may obscure a general trend in the income. Here the rule to be applied is simple: add up the monthly incomes and compare yearly incomes, but allow for the further obscuring factor that the yearly income is likely to be influenced by the amounts of snowfall in the years under consideration. In more complex situations the "rule" is time series analysis, a well-known statistical technique.

5. Systems in context

One of the approaches to reducing the cost of the development of software systems is to develop product lines. Our definition of a product line is as follows: a product line of software systems consists of a set of systems that have the same generic requirements. Lift controllers provide an example. First, generic requirements are defined. Next, a system is implemented that conforms to these generic requirements, but also takes into account the context, i.e., the architectural properties of the building into which the lifts are to go, and the needs of the users of the lifts.

Two strategies can be followed. One is to define a high-level generic design, and to instantiate this design as software systems for given contexts. We have followed this approach in defining a generic rental system [33], and have found the approach highly effective [34]. Under the second strategy, a set of implementations is consulted, and the implementation closest to the needs of the present application is selected for modification. Here, then, we have a set of contexts with corresponding software implementations. A similarity search is made for the context in this set that corresponds most closely to the present context.

The second strategy is complicated. First, there has to be a notation for representing the contexts. Second, a similarity measure has to be found that allows the "nearest" context to the present context to be identified. Third, source-to-source transformation can be difficult [35]. Some pointers on how to approach the similarity issue are given by research in case-based reasoning [36]. This knowledge engineering technique is based on a repository of cases and descriptions of the contexts in which the cases have been applied. There is context comparison, selection of the "nearest" case, and modification of the case to suit the new context. This technique has been very successful in, for example, evaluation of customer service requests. There is no fundamental reason why it should not be successful in

finding the "nearest" software system as well. However, because of the difficulties associated with source-to-source transformation of software, instantiation of generic designs appears to be more promising at the present stage of research.

Our context for this section is a user community, which is to include people who would supply relevant information on extraneous factors, such as the configuration of a building in the case of lift controllers. In the approaches outlined above the system was made to suit the users. An alternative is to leave the system unchanged, and require the context to change, i.e., have the users modify their behavior. Of course, it would be ridiculous to have just one lift controller, and thus force all buildings to be alike, but not having to change a software system can be advantageous in many other situations. Experience with a course registration system at a not-to-be-named university suggests that sometimes a context change is preferable to a system change. A course registration system was purchased, and it underwent laborious source-to-source modification lasting several years to make it conform precisely to the system already in place. Since many users were not entirely happy with the existing system in any case, changing the context instead of the system may have been wiser.

With the present trend in the direction of software constructed from commercial components (see, e.g., [37]), a hybrid approach may be most appropriate. Under such an approach a system is defined in terms of generic modules. Some modules would be purchased and used unchanged, while others would have to undergo modification. Still others would be instantiated from high level generic designs.

6. Summary and a look forward

We have indicated how context, as data or as rules, or as a combination of the two, can assist in reducing uncertainty, but one of our aims has been to show that uncertainty management remains a very rich research area. Much more work needs to be done on "diminishing returns", in particular on recognition when uncertainty has become irreducible. Since a context is essentially unbounded, we need to know which components of a context can help us reduce uncertainty, and which are irrelevant for a particular form of uncertainty. We also mentioned iterative processes as a possible means of reducing uncertainties in a set of uncertain entities that relate to each other, but it has to be realized that this would be a very difficult undertaking.

We defined context as data and rules, but did not discuss how it is to be represented, except by making reference to concept graphs. The data/rules form suggests that a logic program may be the best explicit representation of a context, but a context graph, which can give meaning to the entities represented in the logic program, should supplement the logic program. Deontic logic was briefly mentioned at the start of Section 4. To what extent it can help with context-based uncertainty reduction remains to be explored.

More research is also needed on how to convert vague terms, such as "uncomfortable", into values or at least fuzzy terms. In many cases this may not be possible. In Section 4.2 we found that cost and convenience should determine one's mode of transportation. But "convenience" is vague, and decomposing it into comfort, travel time, and scenic features does not greatly help – two of the three components are still vague.

We noted that some fuzzy relations, which may resemble similarity relations, are not transitive, or even symmetric. This emphasizes the need for ontological investigation of what types of relations can be defined on what types of entities. Ontologies are needed for a variety of areas. For example, without an ontology of money we cannot make a determination of when an amount should be exact and when it should be rounded (Section 4.1).

More research is needed on similarity itself. A similarity relation assigns entities to different classes, where the members of a class are in some sense similar to each other. Rough sets (see, e.g., [38, 39]) define such classes. But the basic problem is that we often do not have adequate measures to determine the degree of similarity between two entities. In Section 5 we pointed out the need for such a measure to determine the "distance" between similar contexts.

Our investigation has been limited to uncertainty relating to observations of an actual world. Serious uncertainty issues arrise when "what-if" situations are considered. In many cases uncertainty becomes too great for rational decisions. For an example, consider a curve fitted to time-based data. When it is extrapolated into the future, confidence ranges become so broad that they make the extrapolated values meaningless. An interesting research area is to determine what types of uncertainty remain manageable in such a possible worlds setting.

In an ideal world there would be no uncertainty. In our real world we have uncertainty, and have to cope with it. We indicated some methods of approach, but none of them can deal with vagueness. Therefore, of all the research suggestions introduced above, the most important challenge is the elimination of vagueness.

References

[1] Sperber, D., and Wilson, D., *Relevance – Communication and Cognition.* Harvard University Press, 1986.

[2] Weick, K.E., *Sensemaking in Organizations.* Sage Publications, 1995.

[3] Layder, D., *New Strategies in Social Research.* Polity Press, 1993.

[4] Buvac, S., Buvac, V., and Mason, I.A., Metamathematics of contexts. *Fundamenta Informaticae* **23** (1995), 263-301.

[5] Hopcroft, J.E., and Ullman, J.D., *Introduction to Automata Theory, Languages and Computation.* Addison-Wesley, 1979.

[6] Brezillon, P., Context in human-machine problem solving: a survey. *Knowledge Engineering Review* **14** (1999), 1-37.

[7] Penco, C., Objective and cognitive context. In *Modeling and Using Context*, Springer LNAI No.1688 (1999), 270-283.

[8] McCarthy, J., Notes on formalizing context. In *Proc. 13th Internat. Joint Conf. Artif. Intell.* (1993), 555-560.

[9] McCarthy, J., and Buvac, S., Formalizing context (expanded notes). In *Computing Natural Language*, Stanford University Center for the Study of Language and Information Lecture Notes No.81 (1998), 13-50.

[10] Berztiss, A.T., Uncertainty management. To appear in *Handbook of Software Engineering and Knowledge Engineering, Vol.2.* World Scientific, 2002.

[11] Klir, G.J., and Yuan, B., *Fuzzy Sets and Fuzzy Logic.* Prentice Hall PTR, 1995.

[12] Dyreson, C.E., A bibliography on uncertainty management in information systems. In *Uncertainty Management in Information Systems.* Kluwer, 1997, 413-458.

[13] Walley, P., Measures of uncertainty in expert systems. *Artificial Intelligence* **83** (1996), 1-58.

[14] Parsons, S., Current approaches to handling imperfect information in data and knowledge bases. *IEEE Transactions on Knowledge and Data Engineering* **8** (1996), 353-372. Addendum: *ibid* **10** (1998), 862.

[15] Pomerol, J.-Ch., and Brezillon, P., Dynamics between contextual knowledge and procedural context. In *Modeling and Using Context*, Springer LNAI No.1688 (1999), 284-295.

[16] Berztiss, A.T., Time in modeling. To appear in *Information Modelling and Knowledge Bases XIII*. IOS Press, 2002.

[17] Kangassalo, H., COMIC: A system and methodology for conceptual modelling and information construction. *Data & Knowledge Engineering* **9** (1992/93), 287-319.

[18] Berztiss, A.T., Domain analysis for business software systems. *Information Systems* **24** (1999), 555-568.

[19] Robinson, W., and Pawlowski, S., Managing requirements inconsistency with development goal monitors. *IEEE Transactions on Software Engineering* **25** (1999), 816-835.

[20] Rush, R., and Wallace, W.A., Elicitation of knowledge from multiple experts using network inference. *IEEE Transactions on Knowledge and Data Engineering* **9** (1997), 688-696.

[21] Pearl, J., *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

[22] Leveson, N.G., and Harvey, P.R., Analyzing software safety. *IEEE Transactions on Software Engineering* **SE-9** (1983), 569-579.

[23] Berztiss, A.T., Unforeseen hazard conditions and software cliches. *Journal of High Integrity Systems* **1** (1996), 415-422.

[24] French, S., and Smith, J.Q. (eds.), *The Practice of Bayesian Analysis*. Arnold with Wiley, 1997.

[25] Chulani, S., Boehm, B., and Steece, B., Bayesian analysis of empirical software engineering cost models. *IEEE Transactions on Software Engineering* **25** (1999), 573-583.

[26] Zadeh, L.A., Fuzzy logic = computing with words. *IEEE Transactions on Fuzzy Systems* **4** (1996), 103-111.

[27] Wang, P.P. (ed.), *Computing with Words*. Wiley-Interscience, 2001.

[28] De Waal, B.M.E., and Van Der Heiden, G.H., The evaluation of user-friendliness in the design process of user interfaces. In *Human Factors in Information Systems Analysis and Design*, North-Holland, 1990, 93-103.

[29] Liu, K.-C., and Sunderraman, R., Indefinite and maybe information in relational databases. *ACM Transactions on Database Systems* **15** (1990), 1-39.

[30] Kao, M., Cercone, N., and Luk, W.-S., Providing quality responses with natural language interfaces: The null value problem. *IEEE Transactions on Software Engineering* **14** (1988), 959-984.

[31] Rodriguez, M.A., and Egenhofer, M.J., Putting similarity assessments into context: matching functions with the user's intended operations. In *Modeling and Using Context*, Springer LNAI No.1688 (1999), 310-323.

[32] Meyer, J.-J., and Wieringa, R. (eds), *Deontic Logic in Computer Science*. Wiley, 1993.

[33] Berztiss, A.T., Domains and patterns in conceptual modeling. In *Information Modelling and Knowledge Bases, VIII*, IOS Press, 1997, 213-223.

[34] Berztiss, A.T., Failproof team projects in software engineering courses. In *Proc. 27th Frontiers in Education Conference*, IEEE CS Press, 1997, 1015-1019.

[35] Berztiss, A.T., Reverse engineering, reengineering, and concurrent engineering of software. *Internat. J. Software Eng. Knowledge Eng.* **5** (1995), 299-324.

[36] Leake, D.B. (ed.), *Case-Based Reasoning: Experiences, Lessons & Future Directions*. AAAI Press and MIT Press, 1996.

[37] Wallnau, K.C., Hissam, S.A., and Seacord, R.C., *Building Systems from Commercial Components*. Addison Wesley, 2002.

[38] Polkowski, L., and Skowron, A. (eds.), *Rough Sets in Knowledge Discovery 1: Methodology and Applications*. Physica-Verlag, 1998.

[39] Polkowski, L., and Skowron, A. (eds.), *Rough Sets in Knowledge Discovery 2: Applications, Case Studies and Software Systems*. Physica-Verlag, 1998.

Information Modelling and Knowledge Bases XIV
H. Jaakkola et al. (Eds.)
IOS Press, 2003

Applying Semantic Networks in Predicting User's Behaviour

Tapio Niemi[1,2] and Anne Aula[1]
[1)]Department of Computer and Information Sciences,
FIN-33014 University of Tampere, Finland
{tapio, aula}@cs.uta.fi
[2)]Helsinki Institute of Physics, CERN offices,
CH-1211 Genève, Switzerland

Abstract. Predicting the behaviour of the user is often an advantageous method for improving the performance of the system. We present a prediction method that applies both semantic information and the user's previous behaviour. Possible applications are, for example, web search engines, precaching of web pages, or database queries. As an example, we illustrate how the method can be applied to predict the user's queries in the OLAP system.

1. Introduction

In many applications, the user performs a sequence of operations, for example, browsing data by posing sequential queries or seeking information on web pages by following a sequence of links. The user often follows a train of thought but does not have a bright idea beforehand of how to perform the task. The answer of the current query can often be the starting point for a new query or a new link to follow is found after opening the next page. In this kind of information retrieval process, the performance of the system is essential for making the process as efficient and pleasant for the user as possible. An inefficient system can, in addition to wasting the user's time, make the user forget the idea he or she had in mind of how to continue with the work.

Our basic assumption is that the user does not pose the queries or click the links arbitrary but applies her/his semantic (conceptual) model in the task. When searching for certain information, the first concept that the user uses in the query activates the user's semantic network. We assume further that the conceptual description of the application is in an optimal case analogous with the conceptual model of the user. By assuming that the activation of the semantic network follows certain rules, we propose a method for predicting the user's following queries. The method will be discussed in a detailed level later in Section 3. In database applications, the conceptual model is usually constructed in the design phase so it is easily available. In other systems, like in web navigation, the

conceptual model is not always directly available but it is always possible to construct. This reasoning enables us to use the conceptual information to predict the user's future actions. The predicted information can be used to precalculate the queries or preload web pages while the user is analysing the previous result.

In the next section, we briefly review some related work. In Section 3, our prediction method is descried in more detail, and the OLAP application is presented in Section 4. Finally, the conclusions are given in Section 5.

2. Related Work

Ruge has used semantic networks in information retrieval in order to find alternative query terms [6]. The study shows that a spreading activation network can easily find alternative terms that are more than just synonyms.

There are several works on predicting the user's actions in WWW environment. For example, Zukerman et al. have studied the predicting of the user's behaviour in web page navigation [7]. They have designed a hybrid model based on individual Markov models.

Sapia has studied the predicting problem by using Markov models in the PROMISE project [5]. The idea is to predict the next query from previous queries in On-Line Analytical Processing (OLAP) [1]. In a typical OLAP session, the user first poses a query to the system, then the system processes the query and returns the answer to the user. After that, the user analyses the answer and makes further queries to specify the answer. The user needs some time to analyse the answer of the query. During this time the system is often idle. In PROMISE environment this idle time is used to predict the user's next query and precalculate the aggregations that may be needed while processing the query.

3. Prediction Method

Our prediction method applies semantic network or more precisely a spreading activation network [2]. A semantic network consists of concepts (nodes) and connections between the concepts. Each connection has a weight that indicates how the connection increases or decreases the activation. The weight reflects the semantic similarity of the nodes. The weights can have initial values or the values can be defined by teaching the network. In the teaching process, the weights are changed according to sample input/output pairs. The purpose is to find optimal weights to get the network simulate the current application as well as possible. Due to the limited space, we do not study the teaching methods in this paper. In addition to the weights of the connections, each node has an activation function that determines how the signal changes the activation level of the node. The activation function usually has a threshold value, that is, a small activation does not change the state of the node.

In practice, the semantic network works in the following way. If the task concerns a node, the activation of the node is increased. Then the increased activation spreads along

the connections to all of the node's neighbour nodes. The weights of the connections can decrease or increase the activation. The activation level of the neighbour node changes if the change is greater than the threshold value of the node. Further, the activation level of the node decreases when time progresses. Finally, the activation levels of the nodes represent the expected information, in our case the probability of the node to be involved in the user's next action.

In our prediction method, the conceptual schema representing the application is seen as the user's semantic network of the application system. Each concept is represented by a node and the relationships between the concepts correspond to the connections between the nodes. The initial weights of the connections reflect the distance of the concepts in the conceptual schema, that is, if there is a direct connection between the concepts then the weight is higher than in the case of an indirect connection. Since the initial weights reflect the semantic relationships of concepts, the system is capable of giving predictions after the user has made the first action. This is a clear benefit compared to the methods that use only information on user's previous behaviour. Further, our system is adaptive since the user's behaviour can change the weights. This ensures that the network is likely to give better predictions when the user keeps on working. In a multi-user environment, also the information about the behaviour of the other users can be applied when teaching the network

We separate two different situations. In the first case, we assume that we have only a relatively small number of objects to be represented by the nodes in a network. A typical example about this is a limited collection of web pages connected with links (see Figure 1). In these cases, it is possible to construct a network that contains all objects as nodes.

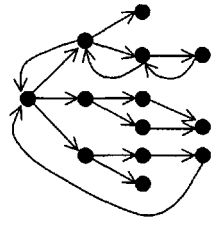


Figure 1. Nodes represent web pages and arrows the links between the pages

In the second case, the application contains so many objects that they all cannot be represented as nodes in a semantic network. This situation is typical in databases. However, the database clearly has two different layers: the conceptual layer and the instance layer. This enables us to use the two-layer network. On the conceptual layer, the aim is to predict which concepts will be used in the next query and on the instance layer which instances of the concepts will be used. In practise, we first use the conceptual level network to predict the concepts that will be used in the query. Then we need to construct instance level networks only for the instances of the predicted concepts. To work properly, the method requires that the instance networks are independent, that is, selecting instances of the concept should not affect the selection of the instance of

another concept. In OLAP, for example, the dimensions are usually independent, so the method works well. Figure 2 illustrates the two-layer method. If the two-layer method still has too much nodes or the two-layer method cannot be applied, another possibility is to group instances and construct a network for these groups. The network is now capable of predicting only groups of objects, not individual objects. For example, we do not try to predict individual days but weeks or months.

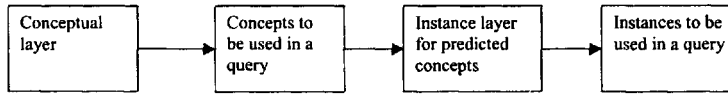


Figure 2. Two-level prediction process

The networks are fully connected but the initial weights of the connections depend on the kinds of relationships the nodes have in the conceptual schema. For example, the weight is usually high if the path between the concepts in the conceptual schema is short. In addition, the weights of the connections can change according to the user's actions. If the user uses two nodes in a query or in the sequential queries, the weight of the connection between the nodes increases, whereas if some nodes are not actively used in the queries, the weights of their connections decrease.

As a whole, our prediction method works as follows:

1. The user makes an action (e.g., poses a query or clicks a link).
2. The activations of the object(s) involved in the action are increased in the network.
3. The increased activations spread in the network. Several iterations are made.
4. The most activated nodes in the network are selected to represent the most probable objects in the next action.
5. The system performs the predicted tasks in advance (e.g., precalculates the queries or preloads web pages).
6. The user performs the next action.

4. Example Application: Predicting OLAP Queries

We illustrate our method by applying it in predicting the user's queries in OLAP (On-Line Analytical Processing) [1]. In OLAP applications, the user browses data on-line by posing queries against a multidimensional data cube in order to find causalities or abnormalities in the data. The user works on-line, but processing the queries can be time consuming. After getting the results for the query, the user has to evaluate the quality of the result. Both the processing of the query from the system's side and the evaluation of

the results from the user's side takes time. Predicting the queries and precalculating them while the user is working mentally, would make the user's work more efficient by reducing the time it takes to process the user's next query. In OLAP, we do not necessarily need to predict the exact queries but it is also useful to know a part of the result that can be used in evaluating the query [5].

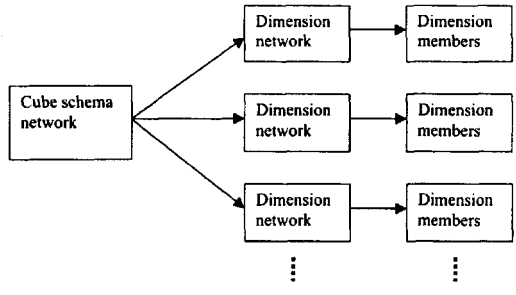


Figure 3. Prediction process in OLAP

The two-layer network method is natural for OLAP. The cube schema represents the conceptual layer and the dimension instance the instance layer. Figure 3 illustrates the method. On the cube schema layer, the aim is to predict the dimensions that will be used in the future queries, and on the dimension layer which dimension members will be needed. A network on cube schema level is called the cube schema network and it contains all attributes in the whole OLAP cube schema. The network is fully connected but initially only the nodes having a direct connection in the OLAP cube schema have non-zero weights. In this sense, the cube schema network has an analogous structure with the OLAP cube schema (see Figure 4).

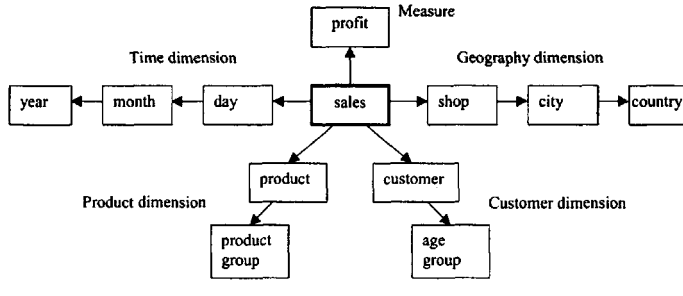


Figure 4. OLAP cube schema, an example

To predict on which dimension members the queries will operate, a semantic network is formed for each dimension. These networks, containing all instances of the concepts in

the dimension (that is, the members of the dimension), are called dimension networks. The dimension network is also fully connected but initially only instances of such concepts that have a direct connection in the dimension schema have non-zero weights. Since the dimensions should be independent [3], the dimension networks are distinct. An example of a dimension network can be seen in Figure 5 c.

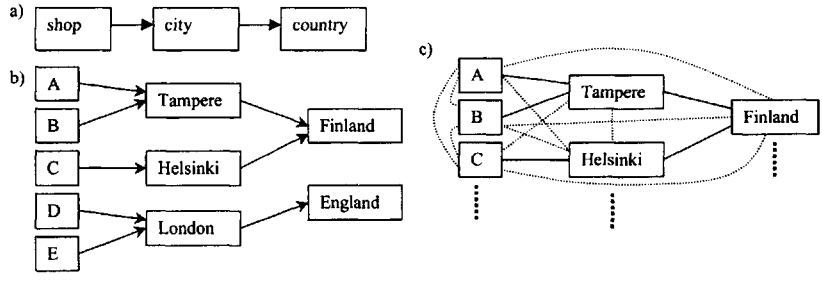


Figure 5. a) Dimension schema, b) Dimension instance, and c) Dimension network

As conclusion, the method works as follows:

1. The user formulates a query using a query language (e.g., MDX [4]) and the system parses the query and determines the concepts and constraints used in it.
2. The activations of the attributes in the query are increased in the cube schema network.
3. The increased activations spread in the network.
4. The nodes having the highest activation levels in the network are selected to represent the most probable dimensions for future queries.
5. The dimension networks of the selected dimensions are manipulated.
    a. The activation levels of the nodes presenting the members that are involved in the query are increased.
    b. The increased activations spread in the network.
    c. The most activated nodes have the highest probability to be used in the future queries.
6. The system constructs the most probable queries based on the dimension networks and precalculates the queries.
7. The user inputs the next query.

To illustrate the method, we first assume that the cube schema network has predicted the geography dimension to be involved in the next query. Now we can use the network in Figure 5 c to illustrate how dimension level network works. The dashed connections in the figure have an initial value of zero. They can change during the teaching process according to the user's behaviour. If the user poses a query concerning shop 'A', the activation of the corresponding node is increased. Then the increased activation spreads to 'Tampere' in the first round. In the second round, shop 'B' and 'Finland' are activated, too. After one round more, the increased activation is spread to 'Helsinki' and finally to shop 'C'. The activation is decreased after each connection, so the final activation of shop 'C' is much lower than the activation of shop 'B'. Now, the state of the network represents the probabilities of nodes to be involved in the next query. The activations have the following decreasing order: 'A', 'Tampere', 'B' and 'Finland', 'Helsinki', and 'C'. In this way, it is most likely that the next query still concerns shop 'A' and only values in other dimensions will be changed. If the user makes changes in the geography dimension, the most obvious change is to roll-up to town level (to 'Tampere'). After that, it is as likely to roll-up to country level or to study another shop in Tampere (shop 'B').

5. Conclusions

We have presented a method that shows how semantic networks can be applied in predicting the user's actions. This approach makes it possible to use both semantic information on the application system and the information on the user's previous actions. Therefore, the method is already effective after the user's first query. The method is useful in many applications in which the system can do preprocessing in order to decrease the response time for the user's next request. The possible applications are, for example, precaching of web pages or precalculating database queries in on-line analytical processing tasks.

6. References

[1] Codd, E. Codd, S., and Salley, C.: Providing OLAP to user-analysts: An IT mandate. Technical report, 1993. Available at http://www.hyperion.com/whitepaper/whitepaperreq.cfm

[2] Collins, A. and Loftus, E.: A spreading activation theory of semantic memory, *Psychological Review*, 82(6), 1975.

[3] Lehner, W., Albrecht, J., and Wedekind, H.: Normal forms for multidimensional databases, *Proceedings of the 10th International Conference on Scientific and Statistical Data Management (SSDBM'98)*, M. Rafanelli and M. Jarke (eds.), 1998.

[4] Microsoft OLE DB for OLAP Programmer's Reference, Microsoft, 1998.

[5] Carsten Sapia: *PROMISE: Modeling and Predicting User Behavior for Online Analytical Processing Applications*, Institut für Informatik der Technischen Universität München, Germany, 2001. (Ph.D. Thesis)

[6] Ruge, Gerda: Human Memory Models and Term Association Cognition and Association, *Proceedings of the Eighteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ACM, 1995.

[7] Zukerman, I., Albrecht, D., and Nicholson, A.: Predicting Users' Requests on the WWW, *User Modeling: Proceedings of the Seventh International Conference, UM9*, Judy Kay (ed.), Springer, 1999.

Information Modelling and Knowledge Bases XIV
H. Jaakkola et al. (Eds.)
IOS Press, 2003

The Dynamics of Children's Science Learning and Thinking in a Social Context of a Multimedia Environment

Marjatta KANGASSALO* Kristiina KUMPULAINEN**

**University of Tampere, Department of Teacher Education*
FIN-33014 University of Tampere, Finland,
Email: kamaka@uta.fi
***University of Oulu, Faculty of Education, P.O.Box 2000, FIN-90014 Oulu, Finland, Email: krkumpul@ktk.oulu.fi*

Abstract. This paper outlines an ongoing research project that investigates children's science learning and thinking in a social context of a multimedia environment in an early years classroom. The goal of the research project is to widen theoretical, methodological and pedagogical understanding of children's conceptual development in an activity context based upon tool-mediation and socially shared learning activities, child-initiation and spontaneous exploration. The research project approaches conceptual learning from the perspective of cognitive and sociocultural psychology in order to illuminate the sociocognitive processes of science learning when modern multimedia technological possibilities are in children's use. The theoretical and methodological foundations of the research project are laid by theories and set of concepts derived from cognitive and social psychology, cognitive science, studies of discourse, learning and social practice and from artificial intelligence. The empirical data of this research project is derived from one Finnish day care centre. Twenty-three children aged between six to seven years participated in the study. The data collection was conducted in a four-week period during which the children took part in a learning unit on space and time. The learning activities and tools in the unit consisted of child-initiated, exploratory activities during which children had versatile tools in their use, including a multimedia learning tool, PICCO. The data of the research project consist of video-recordings of children's interviews before and after the experiment as well as of the children's individual and social activities during the learning unit. Children's exploration paths during the use of the multimedia environment have also been recorded. Subsidiary data of the project consist of teacher interviews and parent diaries. The analyses of the empirical data of this research project are currently undergoing. In all, the research project aims at broadening and deepening the authors' existing research which has highlighted some intriguing findings concerning the development of children's conceptual understanding, exploratory learning, social interaction and their meaning in children's conceptual thinking and knowledge construction of natural phenomena when working in an open tool rich learning environment [1, 2, 3, 4, 5]. The results of the research will be important for theoretical and applied research as well as for educational practice with early years learners.

1. Introduction

The major goal of the ongoing research project is to investigate children's science learning and thinking in a social context constructed around a multimedia environment in an early years classroom. The theoretical foundation of the research project is laid by theories and set of concepts derived from cognitive and social psychology, cognitive science and -with regards to the construction and testing the formal model [5] - from artificial intelligence. The analyses of children's social interactions and activities around the multimedia environment will draw upon studies of discourse, learning and social practice.

The empirical data of this research project is derived from one Finnish day care centre. Twenty-three children aged between six to seven years participated in the study. The data collection was conducted in a four-week period during which the children took part in a learning unit on space and time. The learning activities and tools in the unit consisted of child-initiated, exploratory activities during which children had versatile tools in their use, including a multimedia learning tool, PICCO. The PICCO program is a pictorial computer simulation of particular natural phenomena, namely the variations of sunlight and the heat of the sun as experienced on earth related to the positions of the Earth and the Sun in space. The construction and further development of PICCO (e.g., [1, 2]) has been realised with the help of grants from the Academy of Finland and TEKES, the National Technology Agency in Finland.

The specific goals of the research project are:

- to examine the sociocognitive dynamics of children's social interaction in the PICCO multimedia environment and to examine their association with the development of children's conceptual models and exploration strategies within this learning context;
- to investigate the construction of conceptual models and scientific meanings in peer interactions, adult-child interactions and in children's communal exploratory activities;
- to develop further appropriate methodological tools to analyze cognitive and social interaction in learning contexts applying modern multimedia technological possibilities.

In all, the research project aims at increasing theoretical, methodological and pedagogical understanding of children's conceptual thinking and its development in a learning context based upon child-initiation, exploratory activities, social interaction and the application of multimedia technology.

The research project is realised in collaboration with Dr. Kristiina Kumpulainen of the University of Oulu, Professor Ohsuga of Waseda University, Tokyo and Dr. Marjatta Kangassalo of the University of Tampere. The project will broaden and deepen the authors' existing research particularly in relation to children's conceptual development in an activity context based upon the application of multimedia, socially shared learning activities, child-initiation and spontaneous exploration.

2. Constructing Knowledge in a Multimedia Environment: Cognitive and Social Perspectives

The processes involved in technology-based learning and instruction have been approached and explained in current research with reference to current views of learning and development, based on constructivist and sociocultural perspectives. These views define knowledge as a dynamic and complex entity which is not something objective existing outside the individual, but which is constructed through individual thinking, socially distributed processes and cultural activities [6]. Accordingly, learning is seen as a knowledge construction process which is linked to the learner's sociocognitive frameworks and to the sociocultural context of activity [7].

Constructivist views emphasise the active role of the individual as a constructor of knowledge. Consequently, meaningful learning is not regarded as taking place via knowledge transmission but rather via knowledge building mediated by socially distributed processes [6, 7]. According to the constructivist perspective, active, goal-directed information exploration and striving towards understanding, which Scardamalia and Bereiter

[8] call *intentional learning*, is segregated from procedural learning and its coincidental outcomes. According to this line of thinking, significant components regulating learning processes are found in individuals' conceptions of the nature of knowledge and learning, in the subject-related knowledge base, and in mindful and goal-directed information-seeking behaviour and reflective activity including the use of appropriate learning strategies, such as cognitive and metacognitive strategies [9, 10, 11]. Sociocultural views of development have directed attention to the social and situated aspects of knowledge as well as to the role of culturally formed semiotic tools in mediating constructive learning activity [12, 13]. From the sociocultural point of view, learning is an enculturation and meaning making process which occurs through participation in cultural, dialogic activities with peers and more knowledgeable members of the culture [14, 15]. Of particular interest from this perspective are the nature of social activities and participation structures that members of the learning community engage in during joint meaning making and knowledge construction. What has also become significant from this perspective are the cultural values, norms and rules that regulate learning activity and define what counts as knowledge and knowing under certain circumstances [16].

In the light of current views on learning, it seems crucial that instructional practice support learners' active participation and engagement in knowledge construction and communal thinking [17]. This can be achieved by creating meaningful and motivating learning activities in which learning material is presented in authentic and multiple forms and which promote formulation of representations and explanation, as well as communal sharing and knowledge building via different mediating tools [18].

3. PICCO – Computer Simulation Program as a Cognitive Tool

In this section we shall describe the PICCO computer simulation program and its theoretical basis. The simulation program has been developed for children's spontaneous exploratory activity with the goal of supporting their conceptual learning whilst interacting with the environment.

3.1. Cognitive Requirements of PICCO

PICCO has been originally constructed for research purposes, where the aim was to examine to what extent children's independent use of a pictorial computer simulation of a selected natural phenomenon could be of help in the organizing of the phenomenon and in the formation of a conceptual model of that phenomenon. The planning and construction of the pictorial computer-based simulation draws upon science education, cognitive psychology and computer science. (See [1, 2].)

When selecting the natural phenomenon for the simulation it was essential, that the phenomenon was important and significant with regard to life. The simulated phenomena have to awaken sufficient interest in the children and efficiently utilize possibilities offered by the computer. The phenomena that can be taken into consideration are the ones that can in no other way be easily and illustratively presented, such as phenomena linked with space, atmosphere, and for example, many other phenomena in elementary physics. In this research, the phenomenon chosen for simulation on the computer was a part of our solar system, namely, the interrelations between the Earth and the Sun and the phenomena which occur in nature as a result of this. The whole phenomenon is rather complex and abstract, but its consequences on the globe are very easily observable and concrete. Although the phenomenon, as a whole is abstract, it is possible to concretize it by using the computer-based simulation. As a final selection criteria for the chosen natural phenomenon can be stated, that the selected natural phenomenon with its conformities to law, forms a clear, well-organized knowledge structure and theory, and these aspects lay a strong and defined foundation for the computer simulation. (See [1, 2, 19].)

The cognitive requirements of PICCO are based on theoretical knowledge of a human's cognition (see e.g. [20]). The main aim is that a constructed computer simulation could support children in analyzing and organizing the phenomenon and in forming a conceptual model of the phenomenon. In this process, perception, remembering and thinking play an

important part. This means that in designing and modelling a computer simulation the matters that support perception, as well as the remembering and thinking processes regarding the phenomenon in question, have to be taken into account. Thus, cognitive requirements have to be considered when selecting the natural phenomenon, modelling the selected phenomenon, and simulating the selected phenomenon onto the computer, as well as displaying the simulation on the screen and using the computer simulation. (See [1, 2, 19].)

At first the requirements concerning the selection of a natural phenomenon will be examined. The natural phenomenon has to be a whole that integrates the separate and multilevel phenomena. There has to be a coherent theory of the phenomenon, which in turn has to include important, central and abstract concepts. These requirements are significant because integrated and organized information, as well as knowledge structures in human memory at a general level of the phenomenon in question, is important to effective and demanding thinking and continuous knowledge construction.

Modelling and simulating the selected phenomenon and showing the simulation on the computer screen has to fulfill the following requirements. The phenomenon has to be shown pictorially and in the form of events. This is because a child's thinking process mainly takes place by means of visual images and proceeds in the form of continuing events. Furthermore, at the earth level, the phenomenon has to be shown realistically and in a scale corresponding to reality. These requirements support a child's knowledge construction in natural situations. At the space level, the phenomenon has been represented by means of the analogue model. The phenomenon has to be modelled according to the theory and existing knowledge of the phenomenon. These requirements support a child's formation of a theory concerning the selected phenomenon.

All the pictures and views on the screen in changing situations have been constructed and represented so that they form peaceful and aesthetically valuable scenes, which is important to the user. Peaceful and harmonious scenes give the user a chance to pause, seek for something, or just look at the view very quietly. This supports a child's attention and concentration on the exploration process of the phenomenon, which again helps in imprinting things in their memory. Pictures in the program are manually drawn pictures by an artist.

The use of PICCO is based on the users' own activity. It is important, that children can proceed according to their own interests and ideas. In the program, there are no paths or rules on how to explore and go forward. Children can use as much time as they like each time. All this provides the user with possibilities to explore the phenomenon any time as long as they want and in the order as they so wish. The program is very easy to use and there is no risk of getting lost in it. When the program is under the users' control, it is possible for the user to concentrate on the phenomenon in question. A child's own activity, attention and interest, support the development and construction of a conceptual model of the phenomenon within him or her. The more complicated the phenomenon is, the more important a person's own activity and interest is in analyzing and organizing information and its storing into the memory. The requirements that support perception, memory and thought and, finally, the forming of a conceptual model of a selected phenomenon, also support at the neural level the construction and strengthening of underlying corresponding neural structures concerning the phenomenon in question (see e.g. [21, 22]).

3.2. Ontological and Epistemological Starting Points

The ontological starting points regarding the design of the computer simulation are based on a conception about the existence of the physical reality, physical objects and physical states, which exist independently of mental states. In an interaction with the physical reality, mental constructs concerning the physical reality are developing in the human mind. Further, the human mind can develop and construct material or abstract objects, for example, physical artifacts. These physical artifacts have not been constructed without the human mind. Physical artifacts are a part of products of a human culture, and they form their own realm belonging, at the same time, to the physical reality. (Cf. [23, 24].)

In this research, the purpose was to construct a pictorial computer simulation of the selected natural phenomenon, as a means of studying, whether a child's use of the simulation could support the organization of the phenomenon in a child's mind. When comparing the task with the above described entirety, the selected natural phenomenon belongs to the physical reality, the pictorial computer simulation is an artifact and the aim is

to study the formation of a child's conceptual model concerning the natural phenomenon when a child is using the simulation. In addition, in this case it is the human mind, that of the researcher, which constructs the artifact and is the modeller of the phenomenon and the designer of the simulation. (See [2, 25].)

3.3. Description of the PICCO program

The pictorial computer simulation concentrates on the variations of sunlight and heat of the sun as experienced on the earth related to the positions of the earth and the sun in space. In the simulation it is possible to explore the variations of sunlight and heat of the sun and their effects on the earth in a natural environment. It is also possible to examine the origin of these phenomena from the basis of the interconnections and positions of the earth and the sun in space. On the earth level the simulation concentrates on phenomena which are close to the everyday experiences of children, such as day and night, seasons, changes in the life of plants and birds etc. The simulation program has been implemented in such a way that the knowledge structure and theory of the phenomenon are based on events appearing together with the phenomenon in question, and these events are illustrated. In the simulation all events and necessary elements are represented as pictures and familiar symbols. At the earth level the pictorial simulation represents the surrounding world, its phenomena and objects in a very natural and realistic way. In exploring the phenomenon at the space level the interrelations of the earth and the sun are represented with the help of an analogue model. PICCO has been aimed for 5 - 8 eight years old. It is very easy to use and it does not presuppose an ability to read or write. (See [1, 2].)

3.4. Earlier Research Experiments and Main Results

Research experiments have been carried out both in day care centres and at schools. The empirical data gathered in these studies consist of altogether thirty three children aged between six to eight years old. Eleven of these children used PICCO in a day care centre for four weeks independently and spontaneously. They had no formal instruction on the chosen natural phenomenon before or during the research period. Twenty two of the children explored PICCO at a school context where they simultaneously had a teaching period concerning astronomical phenomena. The children's conceptual models were elicited before and after using PICCO. During the elicitation situations, the children modelled the phenomenon by means of various tasks, such as through action, pictorially and verbally.

The results of these studies show that after the use of PICCO, the most essential change that occurred in the children's conceptual models was that the interconnections of different aspects and phenomena began to be constructed. This construction seems to point to the direction of the currently accepted scientific knowledge. The extent of construction varied in children's conceptual models. The change that took place in the conceptual model of an individual child contained conceptual change to a different extent and depth. [2, 26, 27]

When comparing the results obtained from a group of children who explored PICCO independently and spontaneously without any teaching to those results obtained from children who received teaching about these phenomena some very interesting and challenging differences were identified. The main differences concerned the integration of the alternation of lightness and darkness, and the succession of seasons on Earth in the relationship between the Earth and Sun. The children who received teaching seemed to have more difficulties in this integration process when compared with the children who explored the phenomenon independently with PICCO. In the conceptual models of the children, who independently explored the phenomenon using PICCO, the succession of the seasons in the relation of the Earth and the Sun were discovered first. After that the children started to integrate the alternation of day and night on the earth with the relations of the Earth and the Sun in space. Whereas the children who received teaching tried to solve these questions at the same time and this appeared to cause more difficulties in the integration process. [28]

The existing studies concerning children's conceptual learning within the context of PICCO multimedia environment have mostly concentrated on the individual learning processes of children. Moreover, little attention has been directed to the social context in which children's learning takes place. This includes examination of the broader social and interindividual

contexts in which children and their learning activities are apart. In the present research project described in this paper more attention will be directed to the sociocognitive and sociocultural processes and conditions supporting children's conceptual learning and thinking in a Finnish day care centre where there are modern multimedia possibilities in children's use. In the following section, we shall briefly review other studies investigating the meaning of social interaction in learning.

4. The Sociocognitive Dynamics of Interaction and Learning

Existing research has already provided convincing evidence of the positive effects of social learning activities on students' cognitive and social development [29, 30]. The diversity in learners' prior knowledge and experience appears to provide a large base of resources for the group's knowledge construction, giving opportunities for self-reflection and joint meaning-making [51]. Although small group work activity can offer students extended opportunities for active participation, not all kinds of interactions will lead to joint meaning-making and knowledge construction. The quality of learning in small groups is strongly associated with the quality of the interactions and collaboration that learners engage in while working on academic tasks [31, 32, 33]).

Features identified as being conducive to effective learning interaction in social learning situations are a joint appreciation of the purpose and goals of the task, sharing of tools and activities, an acceptance of an educational agenda over a social one, and a willingness to speculate, make hypotheses and use valid evidence [34, 35, 36]. Instructional conditions that seem to encourage such features often include elements that foster joint task-involvement and interdependence between group members [29]. Furthermore, the task and learning material should take account of learners' prior knowledge to ensure higher level constructive activity [37].

Studies of the behaviors that promote learning during social activities demonstrate that learners who construct procedurally clear and conceptually rich explanations that clarify processes to help peers to arrive at their own solutions learn more than children who simply tell peers answers [38, 39, 40]. Other forms of interaction conducive to learning have included asking appropriate questions, exchanging ideas, justifications, speculations, inferences, hypotheses and conclusions [29]. Research has also shown that engaging in and resolving conflicts with peers encourages learning [41, 42]. Grounding for these phenomena comes from constructivist theories which hold that internal cognitive conflict arises when learners express alternative perspectives [43].

The dynamics of social interaction are complex and do not automatically lead to collaboration and joint understanding. Complex processes are present in social activity that are linked to the sociocultural context of the activity as well as to the evolving interpretations and meanings created in the immediate context [44]. The dimensions of interaction are also related to the participants' socio-cognitive and emotional processes, including their interpretations and perceptions of the aims of the activity in question [45]. Vion [46], when characterizing the complexity of interaction situations, introduces the concept of heterogenous interactive space. This refers to the social, cognitive and interactive roles and contexts which interactors have to negotiate in order to achieve a joint understanding (cf. [45]). Consequently, studies of social learning need to approach interactional phenomena from multiple viewpoints and via multiple levels of analyses that take account of the evolving and dynamic nature of interaction [47].

5. Aim, Design and Research Methods of the Research Project

5.1. Aim and Design

The research experiment of this project concentrates on studying the use of the PICCO program as a tool for children's independent and spontaneous exploration in a day care center. The experiment was conducted in the spring 2001 in the Kämmenniemi Day Care

Center of the City of Tampere. 23 children of the age of 6-7 years participated in the research. The children had the possibility to use the PICCO program for four weeks. The collected research data include video-recordings of children's interviews and their activities regarding their conceptual models of the natural phenomenon in question. The interviews were conducted both before and after the children used the PICCO program. Also video-recordings of the activities of both the children and the teachers (e.g. peer interaction, verbal and non-verbal communication) that took place during the use of PICCO were collected. Children's exploration paths when they investigated natural phenomena by using the PICCO simulation were recorded into the computer's memory click by click. Moreover, the parents of the children wrote diaries about children's questions and activities at home during the research experiment. Teacher interviews focused on teachers' conceptions of children's learning processes. The analyses of the collected empirical data have begun in 2001 and the process is still undergoing.

5.2. Research Methods

The research methods in this research project include the elicitation of children's conceptual models, in-depth analysis of children's exploration processes and social interaction.

During earlier research projects, a model was constructed for describing children's conceptual models and the changes that occur in them. This description technique is used also in this research. (See [2].) The description technique pays attention to the different stages of the formation of children's conceptual models, as well as how the conceptual model concerning a given phenomenon is expressed. This means that the operative, visual and verbal expressions of children that concern with a phenomenon or construction are included in the description and can be differentiated from each other. With the description technique, it is possible to follow the process of children's knowledge construction in its different stages, forms of expression children use, and changes that occur in the conceptual model. The basis for the development of procedures for the elicitation of a conceptual model of the given natural phenomenon has formed the theoretical framework for the concept of a conceptual model, the given natural phenomenon and the children's insufficient ability to express recalled things verbally. In the elicitation of a conceptual model, attention was paid to the order, continuity and regularity of events of the natural phenomenon on the earth, the interconnections of the Earth and the Sun in space, and interrelations of phenomena on the earth and in space.

The analysis of social interaction follows the framework developed by Kumpulainen and Mutanen [16, 47]. This method of analysis particularly focuses on the mechanisms through which the social and cognitive features of interaction operate in a socially mediated learning activity. The theoretical grounding of the analysis framework is informed by the sociocultural and socio-constructivist perspectives on interaction and learning. In this method, learning is seen to take place as a result of individuals' active participation in the practices of the social environment. Learning is viewed as an interactional process that requires an understanding of language and other semiotic tools as both personal and social resources [13, 48, 49]. Social interaction is treated as a dynamic process in which language and other semiotic tools are used as instruments of communication and learning. Interaction is seen as a complex social phenomenon which is composed of non-verbal and social properties in addition to its verbal characteristics. Discourse itself is not treated as representing a person's inner cognitive world, nor even as descriptive of an outer reality, but rather as a tool-in-action shaped by participants' culturally-based definitions of the situation [44]. In the analytical method, the dynamics of peer group interaction are approached from three analytic dimensions.

1.) The first dimension of the analysis, entitled the functional analysis, investigates the purpose and content of utterances in social interaction.

2.) The second dimension, cognitive processing, examines the ways in which children approach and process learning tasks in their social activity. This includes examining the nature of children's conceptual language constructed during the course of activity.

3.) The third dimension of the analysis, social processing, focuses on the nature of the social relationships that are developed in social activity. The latter two dimensions, cognitive and social processing, are investigated on an episode level, whereas the analysis of the communicative functions of peer interaction takes the utterance as its unit of analysis. The

data on social interaction are illustrated with the help of analytical maps which have been created for each case under investigation.

In addition, the description techniques for children's exploration processes have been developed in the earlier research. In this research those techniques will be used and developed (e.g. [2, 27]).

This research further develops the method of analysis and description that pays attention to cognitive and social elements in the development of children's conceptual thinking in activity environments that utilize information technology.

To formalise models and implement them in computers is very important both for studying cognitive processes and making computer systems intelligent. It needs a new method of modelling to include not only objects but also human being who build the model. It needs also a method of dealing with the model. These things require an advanced software technology. Ohsuga [5] is making research on such an advanced method of modelling and its manipulation. This method may be used effectively in this research project.

6. Significance of the Research

Thinking skills as well as individual and communal exploratory activities have arisen into a significant position in the rapidly changing technological environments of today and tomorrow. By the means of this research, we believe that important information will be gained about children's conceptual thinking, social construction of knowledge, and the design of powerful learning environments and support practices. These areas need urgently further investigation [50].

The significance of the present research project can be summarized at least on three dimensions. *At a theoretical dimension*, the project aims at furthering and deepening current understanding of the nature of children's conceptual thinking and its development in an early years classroom when modern multimedia possibilities are in children's use. In specific, by taking a full analysis of the learning context, i.e. ecological perspective, the research project attempts to make visible the complex nonlinear sociocognitive and sociocultural dynamics that unfold and mediate the development of children's conceptual thinking when an intentionally-driven child interacts with a multimedia-based purposefully designed learning environment within a social context. *At a methodological level*, the research project develops new analysis tools to capture the situative dynamics of social interaction and conceptual learning in adult-child and child-child interactions and social activities. *At a pedagogical level*, the research project aims at highlighting conditions and processes which have the potential to offer productive learning opportunities for children's conceptual thinking. Moreover, the project provides insights into the meaningful application of multimedia in an early years classroom.

Acknowledgements

We express our great thanks the staff of the Kämmenniemi day care centre and the children for carrying out the research in their group.

References

[1] Kangassalo, Marjatta 1992. The Pictorial Computer-Based Simulation in Natural Sciences for Children's Use. In Ohsuga, S., Kangassalo, H., Jaakkola, H., Hori, K. and Yonezaki, N. (Eds.) Information Modelling and Knowledge Bases III: Foundations, Theory and Applications. Amsterdam: IOS Press, 511-524.

[2] Kangassalo, Marjatta 1997. The Formation of Children's Conceptual Models concerning a Particular Natural Phenomenon Using PICCO, a Pictorial Computer Simulation. Acta Universitatis Tamperensis 559. University of Tampere. Tampere. 188 p.

[3] Kumpulainen, K. & Mutanen, M. 1998. Collaborative practice of science construction in a computer-based multimedia environment. Computers and Education, 30, 75-85.

[4] Kumpulainen, K., Salovaara, H. & Mutanen, M. 2001. The nature of students' sociocognitive activity in handling and processing multimedia-based science material in a small group learning task. Instructional Science, 29, 481-515.

[5] Ohsuga, Setsuo 1996. Aspects of Conceptual Modeling - As Kernel of New Information Technology. In Y. Tanaka, H. Kangassalo, H. Jaakkola and A. Yamamoto (Eds.) Information Modelling and Knowledge Bases VII. Amsterdam: IOS Press, 1-21.

[6] Salomon, G. (Ed.) 1993. Distributed cognitions. Psychological and educational considerations. Cambridge: Cambridge University Press.

[7] Resnick, L.B. 1991. Shared cognition: thinking as social practice. In L.B. Resnick, J.M. Levine & S.D. Teasley (Eds.) Perspectives on socially shared cognition, pp. 1-20. Washington, DC: American Psychological Association.

[8] Scardamalia, M. & Bereiter, C. 1993. Technologies for knowledge-building discourse. Communications of the ACM, 36, 37-41.

[9] Vosniadou, S. 1996. Towards a revised cognitive psychology for new advances in learning and instruction. Learning and Instruction, 6, 95-109.

[10] Young, A.C. 1997. Higher-order learning and thinking: What is it and how is it taught? Educational Technology, 37, 38-41.

[11] Pressley, M. & McCormick, C.B. 1995. Advanced educational psychology for educators, researchers and policymakers. New York: Harper Collins College Publishers.

[12] Vygotsky, L.S. 1978. Mind in society: The development of higher psychological processes. Cambridge, MA: Harvard University Press.

[13] Wertsch, J. V. 1991. Voices of the mind: Sociocultural approach to mediated action. Cambridge, MA: Harvard University Press.

[14] Rogoff, B. & Toma, C. 1997. Shared thinking: Community and institutional variations. Discourse Processes, 23, 471-497.

[15] Wertsch, J.V., Hagström, F., & Kikas, E. 1995. Voices of thinking and speaking. In L.M.W. Martin, K. Nelson, & E. Tobach (Eds.) Sociocultural psychology, pp. 276-290. Cambridge, MA: Cambridge University Press.

[16] Kumpulainen, K. & Mutanen, M. 1999. Interaktiotutkimus sosiokulturaalisen ja konstruktivistisen oppimisnäkemyksen viitekehyksessä. [Social interaction and learning - Interaction research in the framework of sociocultural and constructivist perspectives to learning]. Kasvatus, 1, 5-17.

[17] Goldman, S.R. 1997. Learning from text: Reflections on the past and suggestions for the future. Discourse Processes, 23, 357-398.

[18] Vosniadou, S. 1994. From cognitive theory to educational technology. In E. DeCorte & H. Mandl (Eds.) Technology-based learning environments. Psychological and educational foundations, pp. 11-17. NATO ASI Series F: Computer and Science Systems, Vol. 137. New York: Springer.

[19] Kangassalo, Marjatta 1996. PICCO as a Cognitive Tool. In Y. Tanaka, H. Kangassalo, H. Jaakkola and A. Yamamoto (Eds.) Information Modelling and Knowledge Bases VII. Amsterdam: IOS Press, 344-357.

[20] Hirschfeld, Lawrence A. and Gelman, Susan A. (Eds.) 1994. Mapping the mind. Domain specificity in cognition and culture. Cambridege University Press.

[21] Damasio, Antonio R. 1989. Concepts in the Brain. Mind & Language 4 (1 and 2), 24-27.

[22] Kohonen, Teuvo 1988. Associative Memories and Representations of Knowledge as Internal States in Distributed Systems. Proceedings of the European Seminar on Neural Computing, London, U.K., February 8-9, 1988. London: IBC Technical Services Ltd.

[23] Niiniluoto, Ilkka 1990. Maailma, Minä ja Kulttuuri. Emergentin materialismin näkökulma. (The World, I and The Culture. The Standpoint of Emergent Materialism. Author's translation.) Helsinki: Otava.

[24] Popper, Karl R. 1972. Objective Knowledge. An Evolutionary Approach. Oxford: University Press.

[25] Kangassalo, Marjatta 1998. Modelling a Natural Phenomenon for a Pictorial Computer-Based Simulation. In Hannu Kangassalo, Pierre-Jean Charrel and Hannu Jaakkola (Eds.) Information Modeling and Knowledge Bases IX. Amsterdam: IOS Press, 239-254.

[26] Kangassalo, Marjatta 1993. Changes in Children's Conceptual Models of a Natural Phenomenon Using a Pictorial Computer Simulation as a Tool. In the Proceedings of the Fifteenth Annual Conference of the Cognitive Science Society June 18-21, 1993, Boulder, Colorado. Hillsdale, NJ: Lawrence Erlbaum, 605-610.

[27] Kangassalo, Marjatta 1994. Children's Independent Exploration of a Natural Phenomenon by Using a Pictorial Computer-Based Simulation. Journal of Computing in Childhood Education 5 (3/4), 285-297.

[28] Kangassalo, Marjatta 1998b. Conceptual Change in Astronomical Phenomena Using PICCO, EARLI, Second European Symposium on Conceptual Change, Madrid, 6.-8.11.1998, 8 pages.

[29] Cohen, E. 1994. Restructuring the classroom: Conditions for productive small groups. Review of Educational Research, 64(1), 1-35.
[30] Rosenshine, B. R., & Meister, C. 1994. Reciprocal teaching: A review of research. Review of Educational Research, 64, 479-530.
[31] King, A. 1992. Facilitating elaborative learning through guided student-generated questioning. Educational Psychologist, 27, 111-126.
[32] Peterson, P.L., Wilkinson, L.C., Spinelli, F. & Swing, S. 1984. Merging the process-product and sociolinguistic paradigms. Research on small group processes. In P.L. Peterson, L.C. Wilkinson & M. Hallinan (Eds.) The social context instruction: Group organization and group processes (pp. 126-152). New York: Academic Press.
[33] Webb, N. M., Troper, J., & Fall, J.R. 1995. Constructive activity and learning in collaborative small groups. Journal of Educational Psychology, 87, 406-423.
[34] Edwards, D. & Mercer, N. 1987. Common knowledge: The development of understanding in the classroom. London: Methuen.
[35] Fisher, E. 1996. Identifying effective educational talk. Language and Education, 10, 237-253.
[36] Forman, E. & Cazden, C. 1985. The cognitive value of peer interaction. In J. Wertsch, (Ed.) Culture, Communication and Cognition: Vygotskian Perspectives. New York: Cambridge University Press.
[37] Chan, C.K.K. 2001. Peer collaboration and discourse patterns in learning from incompatible information. Instructional Science 29, 443-479.
[38] Fuchs, L.S., Fuchs, D., Karns, K., Hamlett, C.L., Dutka, S., & Katzaroff, M. 1996. The relation between student ability and the quality and effectiveness of explanations. American Educational Research Journal, 33, 631-664.
[39] Nativ, A. 1994. Helping behavior and math achievement gain of students using cooperative learning. Elementary School Journal 94, 285-297.
[40] Webb, N. M., Troper, J. and Fall, J. R. 1995. Constructive activity and learning in collaborative small groups. Journal of Educational Psychology, 87, 406-423.
[41] Doise, W. & Mugny, G. 1984. The social development of the intellect. Oxford: Pergamon Press.
[42] Nastasi, B. K., Clements, D. H., & Battista, M. T. 1990. Social-cognitive interactions, motivation, and cognitive growth in Logo programming and CAI problem-solving environments. Journal of Educational Psychology, 82, 150-158.
[43] Perret-Clermont, A-N., Perret, J-F., & Bell, N. 1991. The social construction of meaning and cognitive activity in elementary school children. In L.B. Resnick, J.M. Levine & S.D.Teasley (Eds.) Perspectives on socially shared cognition (pp. 41-62). Washington: American Psychological Association.
[44] Edwards, D. & Potter, J. 1992. Discursive psychology. Newbury Park, CA: Sage.
[45] Grossen, M. 1994. Theoretical and methodological consequences of a change in the unit of analysis for the study of peer interactions in a problem solving situation. European Journal of Psychology of Education, 11, 159-173.
[46] Vion, R. 1992. La Communication Verbale. Analyse des Interactions. Paris: Hachette.
[47] Kumpulainen, K. & Mutanen, M. 1999. The situated dynamics of peer group interaction: An introduction to an analytic framework. Learning and Instruction, 9, 449-474.
[48] Cole, M. 1996. Cultural psychology: A once and future discipline. Cambridge, MA: Harvard University Press.
[49] Halliday and Hasan 1989. Language, context, and text. London: Oxford University Press.
[50] Sinko, Matti ja Lehtinen, Erno (toim.) 1998. Bitit ja pedagogiikka. Tieto- ja viestintätekniikka opetuksessa ja oppimisessa. Atena. (See also Sinko, M. and Lehtinen, E. 1999. The Challenges of ICT in Finnish Education. Atena.)
[51] Teasley, S. 1995. The role of talk in children's peer collaborations. Developmental Psychology, 31, 207-220.

Information Modelling and Knowledge Bases XIV
H. Jaakkola et al. (Eds.)
IOS Press, 2003

Emergence of Communication and Creation of Common Vocabulary in Multi-agent Environment

Jaak HENNO
Tallinn Technical University
Ehitajate tee 5, Tallinn 19086, Estonia jaak@cc.ttu.ee

Abstract. For knowledge-based systems to be reliable and up-to date in our quickly changing world, not dependent on frozen knowledge and limited understanding of the problem at the time of system creation, systems should be able to learn. Their learning algorithms should be as natural and simple as possible, based on principles which guide learning, communication, common vocabulary and language emergence in many multi-agent systems, e.g. in human society. In human society, "...language is universal because children actually reinvent it, generation after generation"[1]. In order to achieve intelligence and flexibility of human society, computer systems should also learn the language, not relay on pre-programmed (frozen) semantic vocabularies, thesauruses etc. Here is studied emergence of common vocabulary in population of agents. Agents are guided by very simple principles, but are able to develop common vocabulary and the process is quite stable, i.e. changes in details and/or parameters do not change essentially emerging vocabulary.

1. Introduction

Classical theory of semantics, knowledge representation and generally classical AI has treated semantics, human knowledge, human intelligence, learning, communication, problem-solving etc as something very complex, but still given, ready and finished, which can be described and modelled, using the "right" algorithm, the "right" data/knowledge representation structures. The basic claim is: "the previous systems failed since they did not have the right knowledge representation, they did not have enough layers etc". The weakness of such approach is apparent - the systems are brittle, require heavy maintenance and still make elementary mistakes [2], [3]. There are many examples of "intelligent" projects, which in spite of years of improvements, new and new layers and data structures still do not work properly (or do not work at all), e.g. the Xanadu [4] or the CYC [5] system.

Human knowledge, human intelligence, problem-solving etc are not fixed, closed systems which are somehow coded in neuron connections in human's head - we are constantly re-developing, rewriting and changing the rules, meanings, principles. Computer systems, which pretend to implement intelligent capabilities should not be compared with a single human being, not with the number of neurons in one human head, but with the number of neurons in the whole mankind, taking also into account all the interconnections, the process of globalization etc. Globalization, mobile communication, development of computer networks, but also single events, such as Sept 11.2001 etc all have changed our understanding of the world, our "common-sense" knowledge. Computer systems with fixed, "built-in" are not capable to follow all these changes. The CYC developers have "put in 600 person-years of effort, and ... assembled a knowledge base containing 3 million rules of thumb that the average person knows about the world"[6] (and spent about $80 million dollars by some accounts), but how many changes had to be introduced after the Sept. 11.2001 events? Computer systems are managing more and more areas of human world and results of computer errors are becoming all the time more and more dangerous.

All "general categories" have for different people different content, i.e. syntax and semantics, and this holds even for the most common and seemingly simple ones. For instance, WorldNet[7] lists for category "name" six meanings; for human names in Western

tradition the "name" syntax includes "Middle Name", but Russians use the "Father's name" instead (and often drop the family name); in western tradition family name is the last, in Hungarian – the first etc. In much more restricted and artificially defined (compared with natural language) technical applications, where everything should have strict formal syntactical/semantic definitions we still have abundance of non-compatible forms, e.g. "filename" in Unix, Mac OS and in even in different versions of the Windows OS all have a bit different syntax and in my computer there are filenames, which some programs accept and some do not.

Computer systems, which demonstrate (more or less) intelligent or at least adequate performance, should be capable to follow, what humans do and learn from activities of humans, adapt to humans. Such is e.g. the Google search engine (the only search engine which is also commercially successful), the Amazon "Customers who bought this book also bought..." or the "Cited by...", "Similar documents..." of the NEC ResearchIndex CiteSeer [8]. Similar to CYC (but less pretentious) system MINDPIXEL [9] was launched on mid-2000, but since it is constantly learning from nearly 40,000 contributing members from more than 200 countries, it is soon supposed to take standard psychological test for humans, used by clinicians worldwide[10].

Therefore lately a different approach to human intelligence, language, human learning and knowledge, communication etc has evolved. It considers all intelligent abilities, e.g. the human language as a "complex adaptive system that is constantly constructed and reconstructed by its users" [11], which can't be understood or modelled when we do not consider it as a part of the whole and do not consider the process of its development. The human intelligence, natural language, its forms (vocabulary and grammar) and meanings (conceptualization of reality) are all tightly inter-related, emerge and develop in the process of adaptive interactions (communication) in community. But they are (can be) based on rather simple principles. Only when we understand (enough) these basic principles it becomes possible to impelent them in computer systems, which are capable of similar development, adaption and learning. The "classical" learning theories (more than 50 [12]), which are based on considering human mind as something fixed, separated from society, do not provide much help in understanding human intelligence. Every intelligent program should itself learn to become intelligent. This learning process can be computationally modelled in a multiagent (swarm[13]) environment. The multi-agent approach is very suitable to study of these phenomena (language, communication, learning, meanings) and there are already interesting results concerning language syntax [14],[15], structure[16], semantics[17], communication [18], [19], [20].

2. The setup

Here is considered one of the most basic issues of communication and language – how a community of agents, who at the beginning do not have any elements of language can develop common vocabulary, using only very simple and basic principles (algorithms). It is also considered, how quickly this system of notations will emerge and what other properties can be observed.

These questions were investigated using computational experiments, where a community of agents with similar capabilities are researching a dynamic environment. The setup is somewhat similar to one, used e.g. in [21], but simpler and takes into consideration differences in how we see the world - different people do not see the same thing in the same way.

The environment is modelled as a set of objects, which are defined by attributes, i.e. an object is some set of attributes (non-empty). Agents can perceive attributes - "blue", "big", "moving" etc and are developing common denotations (i.e. words) for these attributes.

Agents do not invent names for objects; objects are characterized only by the list of attributes. At the beginning agents do not have any words; the common vocabulary emerges in series of communication acts. The object of communication is always an outside word object, represented by a set of attributes, i.e. a list of attributes. Since we do not see things in the same way (e.g. somebody sees colour and size, but somebody – shape and/or placement), not all the attributes were used by sender (speaker) in the message; the receiver had to guess, which attributes were used.

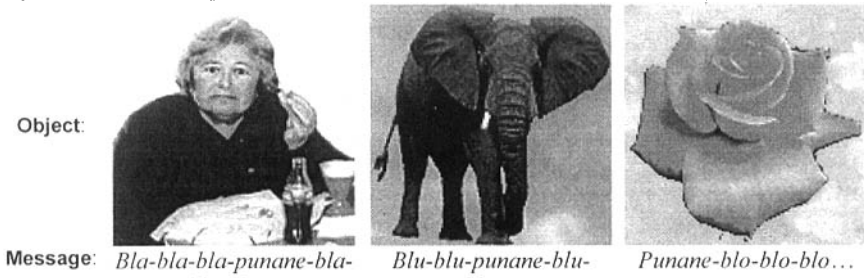
Thus the communication act consists of following steps:

- two agents are selected randomly: sender (speaker) and receiver (listener);
- an object (i.e. some set of attributes) is selected randomly; sender points (somehow) to this object to indicate the receiver, that the message contains information about this object;
- the sender randomly selects some of the object attributes (at least one, i.e. messages are not empty) and selects for them words from his vocabulary (order of words is not essential)
- the receiver tries to build correspondence between the attributes of the communication object and received words.

Every agent maintains his own language (vocabulary); at the beginning all languages (lists of attribute denotations) are empty. An agent's language contains for every attribute list of words, which denote this attribute; the same word may occur as a possible denotation for several attributes. Together with every word, agents also store its frequency count, i.e. how often this word has occurred in messages; the frequency counts are used when agent selects words for this attribute his message (there are several possibilities for this). When agent receives a message, he will add the unknown words as possible denotations for all unknown attributes of the message's object.

The process converges (agents "agree" on the same word for an attribute) when different objects have this attribute common, but all the other attributes - different.

Suppose the message's object has the following attributes: "red", "big", "old", "eating", "lady", and the corresponding message contains word "punane". When agent receives the message, he assigns the word "punane" to all observed attributes, i.e. after this communication act "punane" is added as a possible denotation to attributes "red", "lady", "old", "big", "eating". But when the same agent later receives messages containing word "punane" and the corresponding objects are red elephant ("big", "old", "fat", "eating"), and red rose (not big, not fat, not old, not eating), then the only possibility remains, that "punane" denotes object attribute "red".



All three objects are red, thus the receiver can decide, that word "*punane*" means *red*.

Thus the process is based on the following principle.

When receiver gets a message (message = object, i.e. a list of attributes and a list of words), he first separates known words, i.e. words which earlier have been used to denote

some of the attributes, which are present in the message's object. When such attributes are found, the receiver increases their use count and considers the corresponding attributes of the message's object as known. "Known" does not mean, that there is a unique word for attribute, it just means that there are some possible candidates – words, which have occurred already earlier together with this attribute. Same word may occur as possible denotation for several attributes and attributes may have several different words as possible denotations.

When "known" words and "known" attributes have been separated, all the remaining words can denote every one from the remaining attributes – agent does not have any additional information about what means what. Therefore receiver adds all the remaining words as possible denotations to all unknown attributes of the object. Since message's words can denote only attributes of the message's object, they are deleted from lists of attributes, which do not occur in the message's object (for every attribute, agents are able to check, wheather this attribute occurs in the communication object).

More formally, suppose the total set of attributes is A and the communication object is characterized by set of attributes $O = \{A_1,A_2,\ldots,A_n\}$, where $A_i \in A$.

1. Sending (speaking) agent selects some attributes $\{A_{i1},A_{i2},\ldots,A_{im}\}$ of the communication object O (all $A_{ik} \in O$, $m \leq n$), which he wants to communicate.

2. Sending (speaking) agent selects for selected attributes $\{A_{i1},A_{i2},\ldots,A_{im}\}$ words $\{w_1,w_2,\ldots,w_m\}$ from his language, e.g. the previously most often used (for this attribute) word is selected (i.e. word, which had highest frequency count); if the sender (yet) does not have a word for some attribute, he "invents" a new word and adds the word also to his language, the frequency count of this new word is set to 0.

3. Sending agent announces to receiver the communication object A (i.e. list of attributes $\{A_1,A_2,\ldots,A_n\}$) and his message $\{w_1,w_2,\ldots,w_m\}$.

4. Receiver tries to "understand" the message and modifies his language correspondingly.

4.1 First he divides the received set of words into two subsets:

- already known (i.e. listed in his language for some attribute $A_i \in \{A_1,A_2,\ldots,A_n\}$; the corresponding attribute is also marked as "known";
- all the remaining words (unknown) from the message; let us denote the set of unknown words as UW and the set of unknown attributes from the set $\{A_1,A_2,\ldots,A_n\}$ - UA.

4.2 For every word $w_i \in UW$, add the word to the list of possible denotations of every attribute from UA; set frequency count of this new word to 1.

4.3. Delete all words $\{w_1,w_2,\ldots,w_m\}$ of the message from lists of denotations of attributes from $A\backslash O$ (since words w_i denote some attribute from O, they can not denote anything from $A\backslash O$). Receiving agent may not know the whole set A - he (possibly) does not notice all the attributes of the communication object, thus he can not directly calculate the set $A\backslash O$ either. But since all agents have similar abilities, he can for every attribute A_k check, wheather it is present in the communication object or not. Thus if in the message occurs a word w_k and the word occurs in receivers dictionary as a word for A_k , then receiver is able to check, wheather A_k is present in O - if the sender noticed A_k in O, the receiver is also able to do this. If A_k is not present in O, then the word w_k should be deleted as a denotation for A_k.

In the course of experiments the steps 2 (word selection) and 4.3 were modified to allow agents to "forget" unused words (words, which have not been used for some time or which were used comparatively seldom).

Although the communication object is known, the sender (speaker) does not necessarily use all its attributes in his message, so receiver can not tell, which attributes are described in the message. At the beginning vocabularies of all agents are empty, so the first steps of this process may be the following.

Example. Suppose the object of the communication is O = (A_1,A_2,A_3,A_4). Sender decides to speak only about three of the four attributes, so that the corresponding message will be [{A_1,A_2,A_3,A_4}, {w_1, w_2, w_3}].

If receiver has earlier seen word w_3 used for objects, which also containded attributes A_1, A_3 (i.e. in his language, w_3 is in lists of words for attributes A_1, A_3) he considers A_1 and A_3 known, i.e. one of them is a possible meaning for w_3. For the other two words he does not have any cues, so he adds both w_1 and w_2 as possible denotations of attributes A_2, A_4. All words w_1, w_2, w_3 are deleted from lists of other attributes, i.e. attributes from $A \backslash O$.

3. Computational experiments

To investigate properties of the above algorithm, the described elementary communication act was repeated. The communication objects O and sending and receiving agents were in all communication acts selected randomly.

To see, what is happening, some checks were made after some number of communication acts. In a check the following parameters of agent languages were investigated:

- how many words (in average) they had for attributes;
- how quickly words become common, what (in average) was confidence of different words; if $\{[w_1, n_1],\ldots,[w_n,n_k]\}$ is the list of denotations for some attribute from a language of some agent (w_i - a word, n_i – the frequency count of the word w_i], then confidence was calculated by formula

$$c(w_i) = \frac{n_i}{\sum_i n_i} \tag{1}$$

  i.e. how often (compared to other words for this attribute) the word has been used
- a communication check - sending a message M (set of words) without the communication object O; the receiving agent had to construct an object O_1, what according to his language best corresponded to received message (i.e. for every word was found attribute, which had this word in its list of denotations with highest confidence coefficient); difference between sended object O and received object $O1$ (recognition correctness) r was calculated as

$$r = \frac{|(O \Delta O1)|}{|O|} \tag{2}$$

where $|O|$ denotes the number of elements in set O and the difference of sets is calculated as $(O \Delta O1) = (O \cup O1) \cap (O \cap O1)$.

- Confidence of receiving $O1$ was computed as the average of all coefficients $c(w_i)$ for all words in message M
- To have more reliabale check, the communication check was carried out 10 times (the object O, sending and receiving objects were in every check selected randomly); Table 1 shows dynamics of the average number of recognition correctness coefficients r. Confidence of messages was calculated as the dispersion of correctness coefficients.

The following parameters of the process were varied to investigate their influence:

- number of agents Ag
- number of attributes Atr
- total number of communication acts N
- principle of selecting a word when composing a message.

The principle of word selection was varied because of (unexpected) behaviour of the process. At the beginning a simple principle was applied – agents selected always a most frequently used word. This resulted in a rather quick convergence (i.e. all agents started to

use the same words), but nevertheless lists for attributes contained many words (comparable to the number of agents); the frequency count of one word was very high (and rising all the time), but frequencies of other words remained close to 0. This phenomenon has a simple explanation: if a word is not used, it also can't be deleted, since words are removed only in the step 4.3 of the above algorithm. Therefore the word selection was modified.

The first modification concerned making agents more "personal". Humans do not accept easily unfamiliar, strange words - if somebody has his/her own word for something, then he/she prefers to use his/her own word and not some other, strange word. Thus if the sender had in his language his "own" word (i.e. word, invented by himself for this attribute), he (with some probability) used this word, not the word, which has been used most often for this attribute. Although this made the process converge slower, the overall number of words for attributes started to decrease (now more words had chance to be deleted), so preference to use "own" words actually speed up the extinction of these word – they were used more often for "wrong" purposes, so that they had to be deleted.

Many words with very low frequency count together with some words with very high (and all the time rising) frequency count seems to be unnatural. Therefore a "forgetting" process was introduced – words with very low frequency were deleted. However, the deletion process should not start too early – in different parts of the community different word for the same attribute can become common (are used most often) and only after some time one common denotation emerges. For comparatively small sets of agents and attributes (< 50) the process converged quickly, when only words, which had use count less than $1/5^{th}$ of the most used word (for this attribute) were "forgotten", provided that they had not been used for some time (e.g. in last 500 communication acts).

4. Numerical results

Number of messages	Average number of attributes	Average number of words for an attribute	Average number of errors in 10 test messages	Average confidence of 10 test test messages
0	19.9333	9.74	4.46	0.36
500	20	12.81	3.26	0.78
1000	20	13.81	2.80	1.32
1500	20	14.00	2.70	1.39
2000	20	14.34	2.74	1.60
2500	20	14.56	2.51	1.86
3000	20	14.84	2.57	1.21
3500	20	14.78	2.34	1.54
4000	20	15.07	2.42	1.78
4500	20	14.82	2.21	2.19
5000	20	14.72	2.20	2.21

Table 1. Number of known attributes (having at least one word), average number of words for all attributes, dispersion of word frequencies (characterizes how number of words used in messages decreased), average error in 10 test messages (calculated using formula (2)) and confidence of messages changed with number of communication acts (30 agents, 20 attributes, tests after every 500 communication acts).

Attr	N = 2000	N = 10000
1	1, 1, 1, 1, 1, 1 max: 0.17	32,1 max: 0.96
2	1, 1, 1, 1, 1, 1, 4, 1, 1 max: 0.33	1,1,1,2,32 max: 0.86
3	1, 1, 1, 3, 1, 2, 1, 1, 1 max: 0.25	28 max: 1
4	1, 1, 1, 1, 1, 2, 1, 3, 1, 1, 3, 1, 4, 2, 1, 1, 1 max: 0.153	26,2,2,7,7,2 max: 0.56
5	1, 1, 1, 2, 1, 1, 5, 1, 2, 2, 2 max: 0.26	36,1 max: 0.97
6	1, 1, 2, 2, 1, 1, 2, 3 max: 0.23	36 max: 1
7	1, 1, 0, 2, 1, 1, 1, 1, 1 max: 0.222	3,3,1,1,1,36,0,2,4 max: 0.70
8	1, 1, 1, 1, 1, 1, 3, 2, 1, 1, 1 max: 0.21	39 max: 1
9	1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1 max: 0.1	37,2 max: 0.94
10	1, 1, 2, 2, 1, 1, 2, 1, 1, 1 max: 0.15	2,28,1,1,1,0,1,1 max: 0.8

Table 2. Convergence of the process to common denotations: distribution of number of words for different attributes and the confidence of most used word (formula (1)) in the language of Agent3 after 2000 (the second column) and after 10000 communication acts (20 agents, 10 attributes).

N	*#W*	*#W*
1	5	9
2	2	8
3	1	1
4	11	13
5	2	8
6	1	2
7	1	1
8	2	11
9	1	5
10	8	12
11	2	13
12	2	8
13	10	4
14	1	7
15	2	6
16	1	13
17	5	1
18	1	1
19	3	9
20	9	9
Average:	3.5	7.5

Table 3. Total number of different words for 20 attributes in languages of 30 agents after 2000 communications; in the second column probability to use "own" word was 0.9, in third column – 0.1; difference in averages is essential.

Atr No	Frequencies of different words for this attribute
1	199,1,1,1,1,1
2	208,1,1,1,1,1
3	217,1,1,1,3,2
4	1,223,1
5	1,235,1,2,7,1
6	1,240
7	1,1,5,210,1,4
8	5,1,1,1,224
9	1,213
10	226,1,4,1,0,1,1
11	1,216,2,2
12	198,2,1,1,1,5,4,1,1,1

Table 4. Frequencies of use of words for different attributes in the language of agent 7 after 10000 communication acts (10 agents, 12 attributes). Since here agents used in messages always the previously most often used word, frequency of one word is much greater than frequency of other words.

Atr	Agent 1	Agent 2	Agent 3	Agent 4	Agent 5	Agent 6
1	a1_1 (0.96)	a1_1 (0.91)	a1_1 (0.97)	a1_1 (0.96)	a1_1 (1)	a1_1 (0.85)
2	a2_4 (0.90)	a2_4 (0.78)	a2_4 (0.75)	a2_4 (0.86)	a2_4 (0.92)	a2_4 (0.81)
3	a5_4 (0.91)	a5_4 (0.92)	a5_4 (1)	a5_4 (1)	a5_4 (0.96)	a5_4 (0.92)
4	a10_1 (0.73)	a10_1 (0.77)	a10_1 (0.66)	a10_1 (0.56)	a10_1 (0.7)	a10_1 (0.8)
5	a5_3 (1)	a5_3 (0.88)	a5_3 (0.97)	a5_3 (0.97)	a5_3 (1)	a5_3 (1)
6	a6_1 (1)	a6_1 (0.96)	a6_1 (0.91)	a6_1 (1)	a6_1 (0.88)	a6_1 (0.97)
7	a7_2 (0.75)	a7_2 (0.77)	a7_2 (0.72)	a7_2 (0.70)	a7_2 (0.81)	a7_2 (0.92)
8	a2_2 (1)	a2_2 (0.83)	a2_2 (0.94)	a2_2 (1)	a2_2 (0.84)	a2_2 (0.96
9	a2_1 (0.88)	a2_1 (0.82)	a2_1 (0.84)	a2_1 (0.94)	a2_1 (0.82)	a2_1 (0.90)
10	a1_3 (0.8)	a1_3 (0.8)	a1_3 (0.862)	a1_3 (0.8)	a1_3 (0.91)	a1_3 (0.7)

Table 5. Confidence of most frequently used words (calculated using formula (1)) in languages of 6 agents for 10 attributes after 10000 communication acts (altogether 30 agents, 20 attributes). Notice, that the most frequent words for all attributes are already same in all languages and all confidences (numbers in brackets) are close to 1 (a1_1, a2_1 etc are words, e.g. a7_2 – the word, "invented" by agent 7 for attribute 2).

5. Problems

The described here setup considers only the very first phase of emergence of common lexicon. There are many problems for further studies:

- use of names, i.e. if sending agent describes in his message not only attribute values, but gives to the whole object also a name? Since this name can not occur in messages, describing other objects, they eventually will be separated, but the receiver should be prepared to use of them. It seems, that inroduction of names requires that agents also should start building an model of environment (conceptual model)
- use of multiple words to describe the same attribute;
- restrictions on memory: restricting the size of agents lexicon or number of words denoting an attribute, memory used in the coding/decoding process etc

- use of some "innate" grammar to describe environment events, i.e. generate a language etc.

References

[1] S. Pinker, The Language Instinct. Penguin Books, p. 21
[2] http://www.infoworld.com/articles/mt/xml/00/10/02/001002mtfailure.xml
[3] http://www.webreference.com/new/020103.html
[4] http://www.wired.com/wired/archive//3.06/xanadu.html?person=ted_nelson&topic_set=wiredpeople
[5] http://www.cyc.com
[6] http://www.cnn.com/2002/TECH/industry/04/11/memome.project.idg/index.html
[7] http://www.cogsci.princeton.edu/cgi-bin/webwn
[8] http://citeseer.nj.nec.com/
[9] http://www.mindpixel.com/GAC/gac.php3
[10] http://unisci.com/stories/20013/0702016.htm
[11] Luc Steels. The puzzle of language evolution. http://www.csl.sony.fr/downloads/papers/1999/steels-kogwis1999.pdf
[12] Theories of learning and instruction. http://tip.psychology.org/theories.html
[13] J. Kennedy, R. C. Eberhart. Swarm intelligence. Morgan Kaufmann Publishers, 2001
[14] S. Kirby. Spontatneous Evolution of Linguistic Structure: an iterated learning model of the emergence of regularity and irregularity. http://www.ling.ed.ac.uk/anonftp/pub/staff/kirby/skirbyieee.pdf
[15] S. Kirby. Syntax without natural Selection: How compositionality emerges from vocabulary in a population of learners. http://www.ling.ed.ac.uk/anonftp/pub/staff/kirby/evol98.pdf
[16] Christopher Allexandre and Andrei Popescu-Belis. Emergence of Simplified Tree Adjoining Grammar Conventions in an Agent Population. http://andreipb.free.fr/these/apb-these-6.ps.gz
[17] L. Steels. Perceptually grounded meaning creation. In "Proceedings of the International Conference on Multi-Agent Systems", ed. M. Tokoro, Cambridge, Ma, 1966, The MIT press
[18] M. Oliphant, J.Batali. Learning and the emergence of coordinated communication. http://citeseer.nj.nec.com/oliphant97learning.html
[19] Bruce J. MacLennan. The emergence of Communication through Synthetic Evolution. MIT Press Math6X), 1999
[20] L. Steels, F. Kaplan. AIBO's first words. The social learning of language and meaning. http://www.csl.sony.fr/downloads/papers/2002/steels-evocomm2002.pdf
[21] F. Kaplan. A new approach to class formation in multi-agent simulation of language evolution. http://citeseer.nj.nec.com/kaplan98new.html

Information Modelling and Knowledge Bases XIV
H. Jaakkola et al. (Eds.)
IOS Press, 2003

Information modelling within a Net-Learning Environment

{christian.sallaberry[(1)]|thierry.nodenot[(2)]|christophe.marquesuzaa[(2)]|marie-noelle.bessagnet[(1)]|pierre.laforcade[(1)]@univ-pau.fr}

Laboratoire d'Informatique de l'UPPA (LIUPPA)
[(2)]IUT de Bayonne Pays Basque — *FDEG et IAE[(1)]*
Château-Neuf, Place Paul-Bert — *Avenue du Doyen Poplawski*
64100 Bayonne – France — *64000 Pau - France*

Abstract. Within cooperative Information System (IS), Educational Engineering scientific community agrees that Information must be organised and structured using ontologies that the developers can manage with meta-data. So, we propose some UML based models and a specific metadata structure dedicated to Educational IS.
In this paper we present a Net-Learning case study, a conceptual model of Educational IS and we describe the main characteristics of such an IS. This case study is representative of collaborative learning notions. Model, or rather models, take into account the static and dynamic aspects necessary for the description of learning processes: we formalize them with the UML language. Finally, we detail the structural aspects of this IS.

Keywords: Educational Information System, Information modelling, MetaData, UML based Models

1 Introduction

Consistent with research that demonstrates the value of collaboration in learning, computer support for collaborative learning has become a greater interest, and various architectures for synchronous and asynchronous collaboration have been explored [17].

We propose a mode of learning centered more particularly on the *collaborative Problem Based Learning Situations* (PBLS). This approach is close to the Net-Learning [14] mode. Here are two characteristics of Net-Learning: *interaction* between teacher and learner, based on an *educational scenario.*

But beyond, our approach differs by the following points:

- interactions among learners (who, for example, collaborate for the resolution of a PBLS)
- an educational scenario based on the notion of acts and scenes in which the actor-role couples carry out specific tasks. All the learners do not play the same role so, do not learn the same matter at the same moment. Their collaboration and then, the integration of their respective works contribute to the realization of a global work. This is quite different from the Net-Learning classic approach in which the same educational scenario is usually awarded to every learner.
- the way activities and interactions among actors (learners and learners/instructors) can be controled, ...

Collaborative learning based on the notion of PBLS is an interesting approach and quite different from more classic sequences: courses, exercises, evaluation [17].

We are going to describe our approach by means of a case study that illustrates all the propositions presented further in this work.

For the description of such learning, we propose a model based on the UML language. We describe both the static and dynamic aspects of these learning situations. On the one hand, we describe PBLS imagined by the pedagogue (designer): here the notion of actor, role and scene will be detailed. We shall speak also about the notion of reusable role-component, which facilitates and accelerates the construction of new scenarios. On the other hand, we explain how to manage all the information about the real activity of every actor and, so, supply control, assistance and tutoring that is necessary for this kind of learning.

Finally, such a model can be supported only by one dedicated Information System (IS): we present its structural (meta-data class diagram) outlines.

2 A particular Net-Learning approach: case study

Our project is aimed to construct *Problem Based Learning Situations* (PBLS) and to integrate them within *collaborative* learning scenarios.

2.1 Collaboration and learning based on PBLS

Problem-Based Learning refers to a constructivist approach of Learning: a problem is proposed to the learner and the solution of this problem must be constructed by the learner thanks to the resources and tools he has at his disposal (generally there are different ways to solve the proposed PBLS).

From our point of view, it is important to note that teaching through PBLS is a didactic choice that requires a lot of abilities from the teacher. Before being able to propose a PBLS to learners, the teacher has to:

- choose the learning objectives,
- imagine a PBLS which resolution will constrain the learners to face these learning objectives,
- define the tools and resources from which the learner will construct his solution,
- imagine the guidelines and tutoring that can be proposed during the learning activity
- define the level of cooperation that can be accepted (if different learners have to cooperate to solve the problem).

As a consequence, with PBLS, one must consider [22][23]. So, before the activity, the teacher must specify what has to be learned (the learning objectives), how he can promote these learning objectives (thanks to the PBLS) and how the learning activity should be managed (collaborations, tutoring, guidance, ...).Then, during the activity, the learner has to analyse guidelines, look at the resources and then, to specify the problem before solving it.

This project consists in producing an Open Distance Learning System that enables groups of learners (helped by human or automatic instructors) to "learn by doing and interacting" while they try to solve a particular PBLS. Each group of learners has dedicated software and tools: some of them can be quite standard (spreadsheets, word-processors, programming environments, ...), others can be specific-purpose tools (e.g. a viewer

allowing a learner to know instantly the tasks that remain for him to accomplish in order to succeed), others provide facilities of communication which allow them to exchange information with other actors within the same context. For such an Information System (IS), the activity of the different actors must be regulated. So, this IS should allow:

- the teacher to specify and to design the interactions which can be established among the various actors : learners, groups of learners, instructors, ...
- the distribution, the sharing and the storage of information,
- the re-use of educational components and tools in different contexts of use (to ensure the profitability of such environments),
- the tutoring of the activities carried out by the different learners according to the specifications produced by the designer.

2.2 *General structure of a learning process*

In our educational context, the IS manipulates some notable objects (cf. italic ones). A *teaching process* aims at an *objective* and can be decomposed into a sequence of *acts*. Each *act* promotes a sequence of *scenes* in which different *actors* will appear and will do some actions in relation with the *role* they play. We make an analogy with the notion of play (theatre): a learning process (play) will be first decomposed into *acts* and then, into *scenes*,...

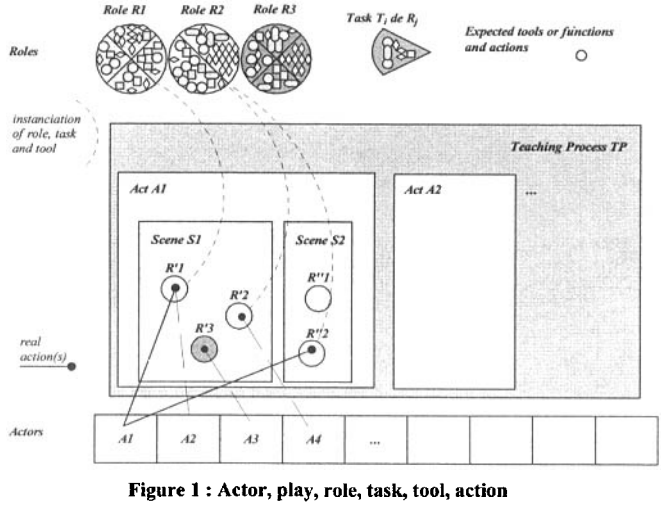


Figure 1 : Actor, play, role, task, tool, action

The various use cases bound to our context present:

- *actors*:
  - groups of learners who collaborate to solve a particular PBLS and, by the way, reach some learning objectives.
  - instructors whose job is to control and guide learners. Instructors can be human (physical person) or automatic (their behaviour is inferred by the computer).
- *roles*: bound to the learning of theoretical notions or particular techniques, to the production of works, to the control, to the help, ...

Modelling of the learning process is thus based on the role/actor duality [9], that is:

- *role* description for the *actors* (learners or instructors) and
- description of the collaboration between role/$actor_i$ and role/ $actor_j$ couples

- Whoever they are, these *actors* can interact with each other; they manipulate some *resources* and appropriate *services and tools*.

2.3 The case study

To concretise the different objects presented above, we describe now an example of cooperative situation dedicated to the learning of some advanced functionalities of Microsoft Word. This example will be treated all along the paper.

A designer (pedagogue) wants some learners to be able of "*creating a Word document (made of two chapters) using style-sheets and a table of contents* ".

The designer decides that Act 1 is that of presentation of the exercise. Act 2 concerns the production of text for chapters 1 and 2. Act 3 is dedicated to the gathering of works. Act 4 is based on the style-sheet design. Act 5 is that of the finalization of styles. During act 6 the table of contents is created and then the document is finalized. Act 7 is that of advanced learning; learners have understood the interest of styles and tables of contents. If needed, they can go on working on advanced exercises.

The designer defines different groups of physically distant actors (Fig. 2):

- Groups 1 and 2 (G1 and G2) are responsible for producing Word chapters (according to the constraints they are told to respect).
- Group 3 (G3) is responsible for production and use of style-sheets. Furthermore, G3 coordinates activities and exchanges between groups G1 and G2. G3 may also teach G1 and G2 how to build style-sheets...
- The instructor A is an automatic instructor (it is a software system) which, according to certain events, authorises and administers exchanges, executes foreseen actions from these events.

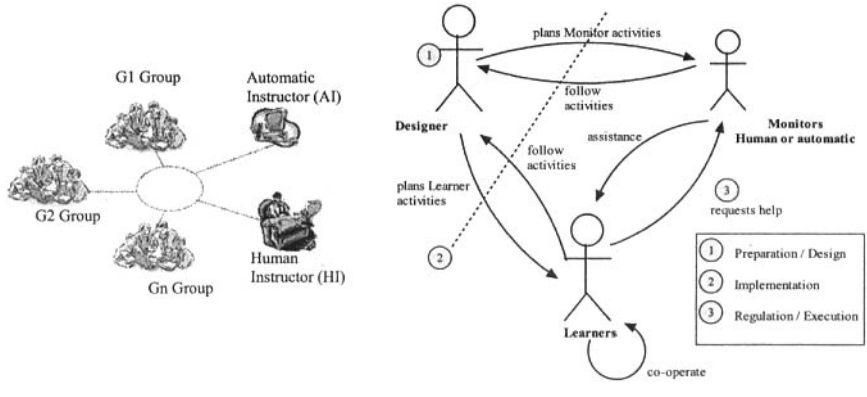


Figure 2: The actors involved in the Word case study

Naturally, these broad roles are detailed and constrained within the specification of each scene. You will find in annex 1 an example of the exchanges of information and resources that could arise among the different groups in act 4.

3 UML Models for collaborative learning scenarios description

An important axis of our work concerns the modelling [5][7][8] of the actors; their roles and their collaboration within a learning process. In this section, we present UML

specifications (collaboration, sequence and state-transition diagrams) for collaborative learning scenarios description. We first detail our approach concerning roles modelling. Further, we illustrate both the static and dynamic aspects of roles using the concrete example (cf. §3.2).

3.1 The role-component notion

We work on the specification of a library of role-component bricks (close to the notion of micro-role [10]) reusable by every designer of learning scenarios in order to build appropriate roles. This approach allows to easily build new roles when combining existing role-components.

We have to distinguish the notion of role-component with that of role which is linked to a context and which we should obviously complete by the notions of time, resources (available, created, expected) and actors.

The role-components of this library are not bound to a context (i.e. scene). However, to build specific roles into a scene, we have to initialise and/or combine one or several role-component bricks.

Activity led by a role within a scene is described with a state-transition diagram; each of the states corresponds to the initialisation of a generic role-component. Within this case study, we describe four main role-components by means of UML state-transition diagrams [6]:

1. "Consumer" corresponds to states of reading, listening, observation or taking of notes. It allows to acquire information or to interrupt a presentation in order to ask questions (fig. 3). If a question occurs to the consumer's mind, his role can temporarily change into the Explainer one
2. "Presenter" corresponds to the statement of information in front of a public. This public has previously been selected and is attentive. If an interruption occurs the presenter role can temporarily change into the explainer one.
3. "Explainer" may listen to a question (like the consumer), may also build its answer or question (like the producer) in order to present it afterward.
4. "Producer" creates resources (questions, answers, results of an exercise, ...).

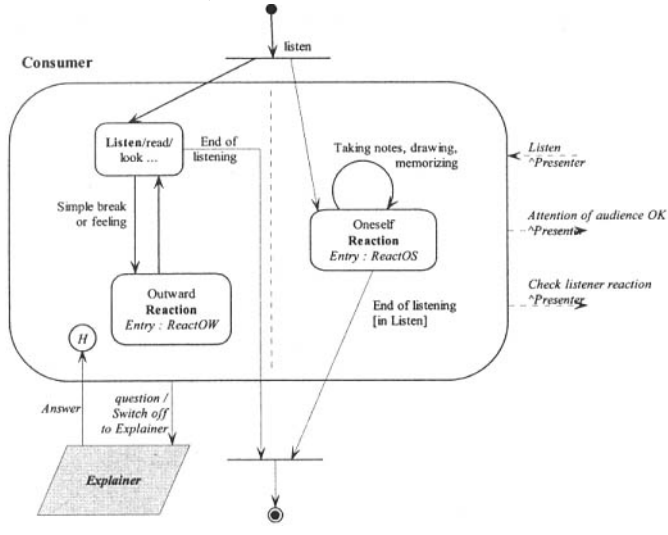


Figure 3: Consumer role-component

"Consumer" role-component is composed of two automates which can be concurrently played: the state "Oneself Reaction" (to take notes, to look for definitions, ...) can be reached and active at the same time as the "Listen" one. On the other hand, *return* transition from the state "Outward Reaction" (to say ok, I have understood; to ask questions; ...) towards the "Listen" state is automatic; as soon as these confirming or asking actions are ending.

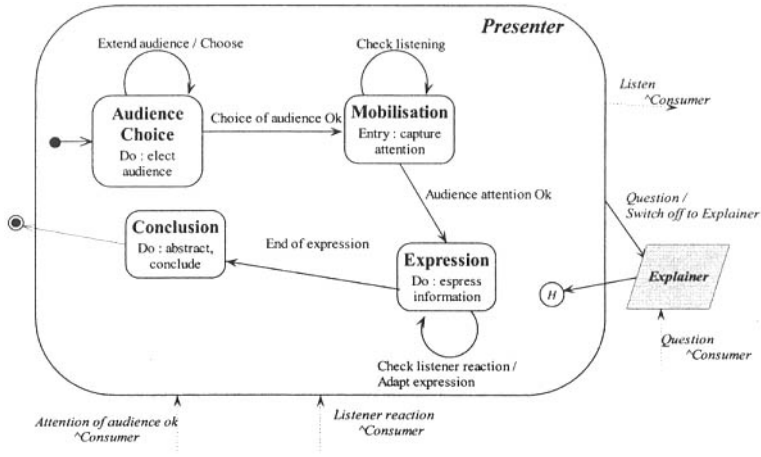


Figure 4: Presenter role-component

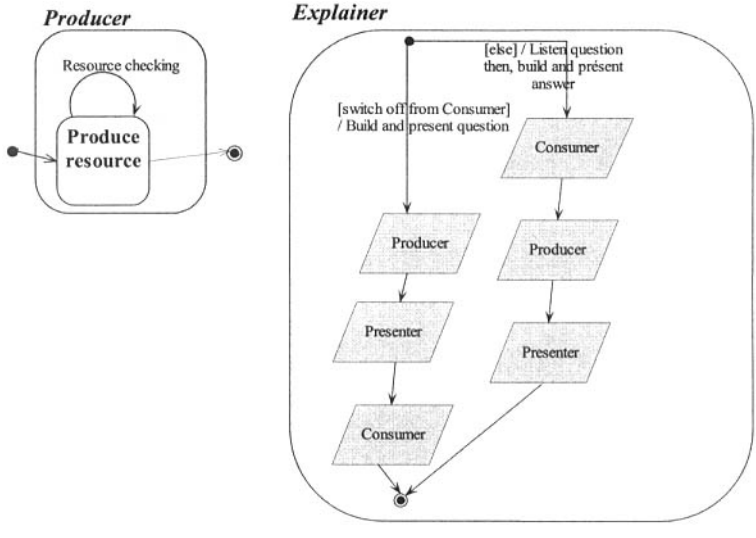


Figure 5: Producer and Explainer role-component

Within the Explainer role-component, "Consumer", "Producer" and "Presenter" respectively correspond to the role-components of the same name.

In the next section (§3.2, *cf. state-transition diagrams)* we are going to illustrate how role-component bricks are initialised and organised in order to create new roles. Role-components are thus generic and reusable.

Furthermore, compared with [9], in our approach a role belongs to a scene (a context in [9]). A role communicates with the other roles within the same scene. An actor is associated to a role through which he participates in the collaboration specified in the scene. So, each role is specified for one unique scene.

The description of roles, actors and scenes can be first modelled and implemented as follows:

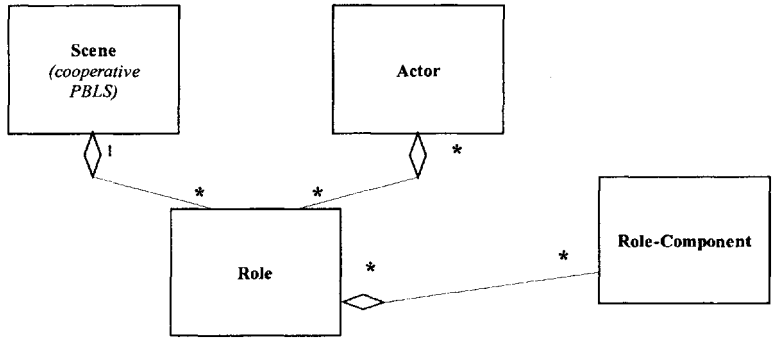


Figure 6: Modelling and implementing a role

3.2 Roles and their collaboration modelling

In order to illustrate our approach concerning roles and re-use of role-components, we are going to use the case-study presented at beginning once again.

This example is drawn out of the following context: ***Word*** teaching process, Act 4 ***Styles***, Scene 1 ***Learn, create, help***. In order to make this scene collaborative, we propose five groups of actors and three roles:

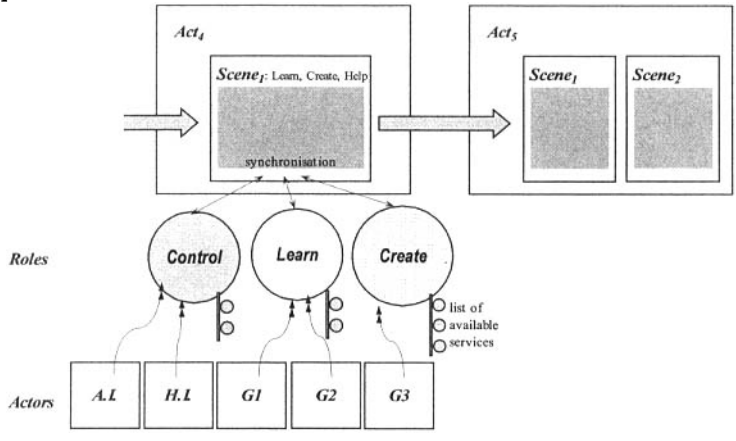


Figure 7: Three roles in the context of scene 1 in act 4

Role "**Learn style**" will be attributed to G1 and G2. This role consists in learning the design of a style-sheet and, at the same time, in asking questions to G3, when necessary.

Role "**Create style**" will be attributed to the G3. This role consists in producing a style-sheet and, at the same time, in helping G1 and G2 in their tasks of learning. When G3 has turned out his style-sheet he has also to make sure that G1 and G2 understood the various notions of style. Therefore, he makes sure of the good understanding (control) and re-explains (or makes some other tools give explanations) various significant stages (apply, create, modify, ...).

Role "**Control styles** " will be given to the Automatic and Human Instructors (AI and HI) with states "Listen to communication between groups", "Control productions", "Supply additional supports of help", ...

Collaboration and sequence diagrams

The two following diagrams show how the HI oversees the work of G1 and G3. He oversees their collaboration (using copies of communication) and can interact by completing help supplied by G3, for example.

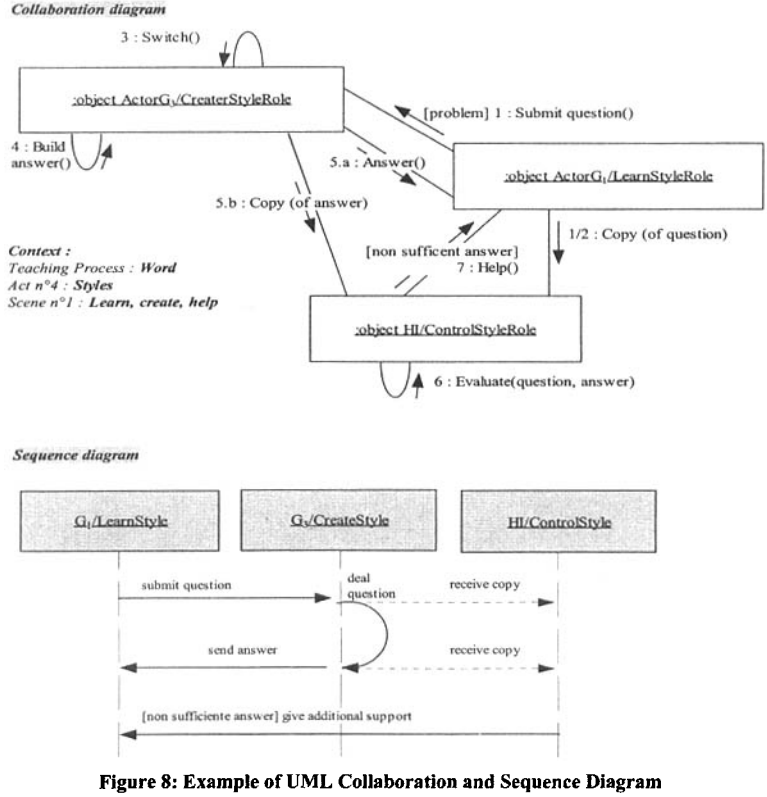


Figure 8: Example of UML Collaboration and Sequence Diagram

State-transition diagram

We present here the diagram corresponding to the description of role "Learn Style". We reuse Consumer and Explainer role-component bricks. These bricks allow us to specify quickly the various states corresponding to the description of every role.

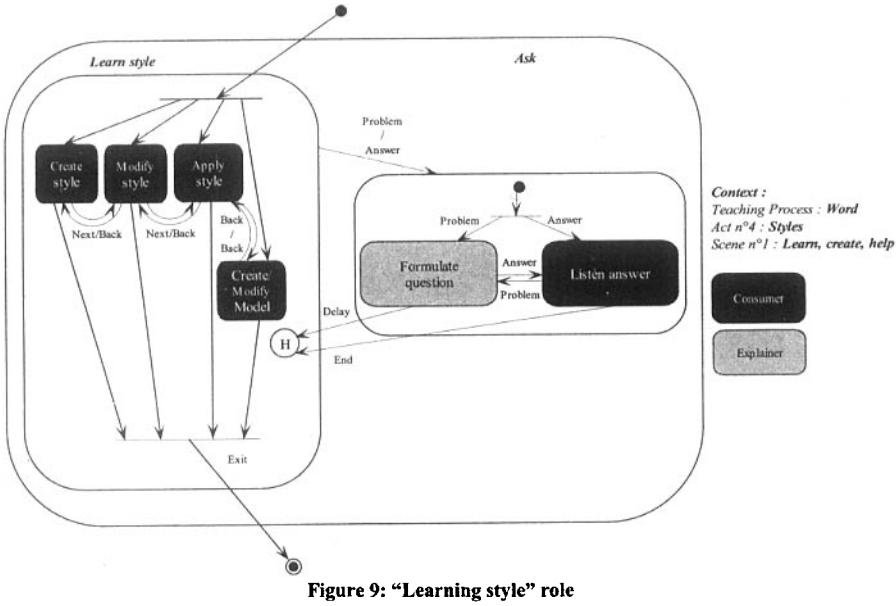


Figure 9: "Learning style" role

"Learning style" role will be given to G1 and G2. It is composed of two automates. "Learn style" is the main one and can be interrupt at any time by :

- a question to be sent to G3
- an answer returned by G3

So, "Ask" automate is played by G1 or G2 before coming back to the "Learn Style" automate at the previously left state (cf. History (H)).

Here is the role "Create style", given to G3 in Act n°4, Scene n°1.

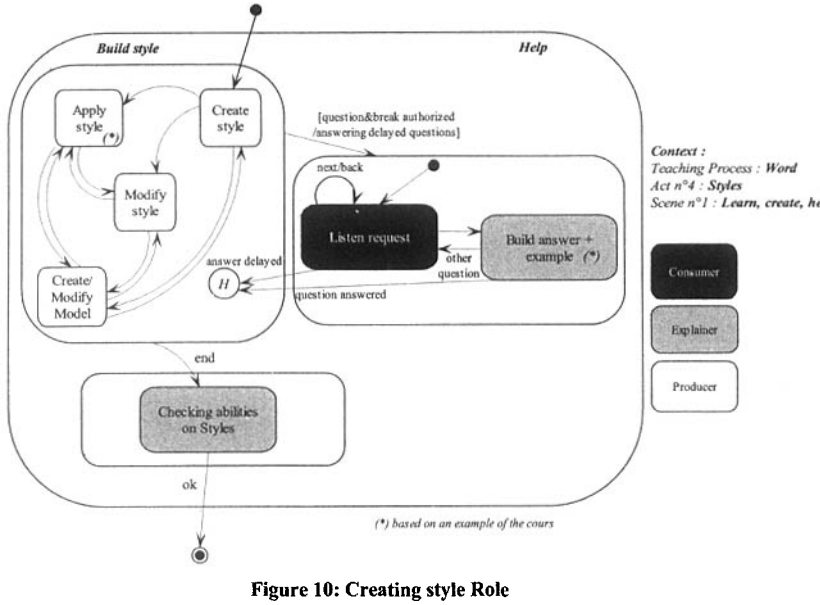


Figure 10: Creating style Role

All the static and dynamic aspects of roles and collaborations, designed using UML language in the previous models, are focused in the next dedicated IS.

4 A dedicated meta-data structure

An IS [2] [3] supporting such a model, includes notably :

- a human component: the designer of the various learning scenarios, the actors (learners and instructors)
- a technological component: hardware, software, database management and communication tools

From the technological point of view, this directs us to the centralisation of objects describing the various learning scenarios within a multimedia and Web oriented database [16]. So, we present the specification of the various data structures supporting this IS: the following UML class diagram description supports the collaborative learning scenarios, as well as guidance and tutoring activities. We must be able to store and manage information concerning :

Topic 1: The PBLS proposed by the designer,
Topic 2: The concrete activities in which the learners are engaged within the framework of the learning situation that they must solve.

In the next paragraphs, we describe topic 1 and topic 2 using a UML based class diagram. Later on, these classes instances are managed within a database schema that we shall use to produce guidance services :

- during the modelling of a PBLS, when a designer creates (or instantiates) objects describing a scenario, in the database (built according to this schema),
- during the solving of a PBLS, when learners, at their turn, create objects corresponding to the activities they are doing (resources they exploit, activated tools functionalities, ...).

So, the mapping between the objects produced by the educational designer (during the preparation phase) and the objects representing the concrete learners activities will enable an instructor, thanks to this IS, to infer some advice and tutoring (regulation phase).

4.1 The learning process description

We previously stated that learning activities occur in the framework of scenes. A PBLS is characterised by :

- its learning objectives,
- its success criteria (both functional and formal) in order to evaluate whether the objectives are satisfied or not,
- the roles suited to this situation,
- the resources that the actors are allowed to use from the role they play.

Within a pedagogical statement, an actor plays one and only one role : thus, in the following diagram, we gather the notions of role and of Pedagogical Statement.

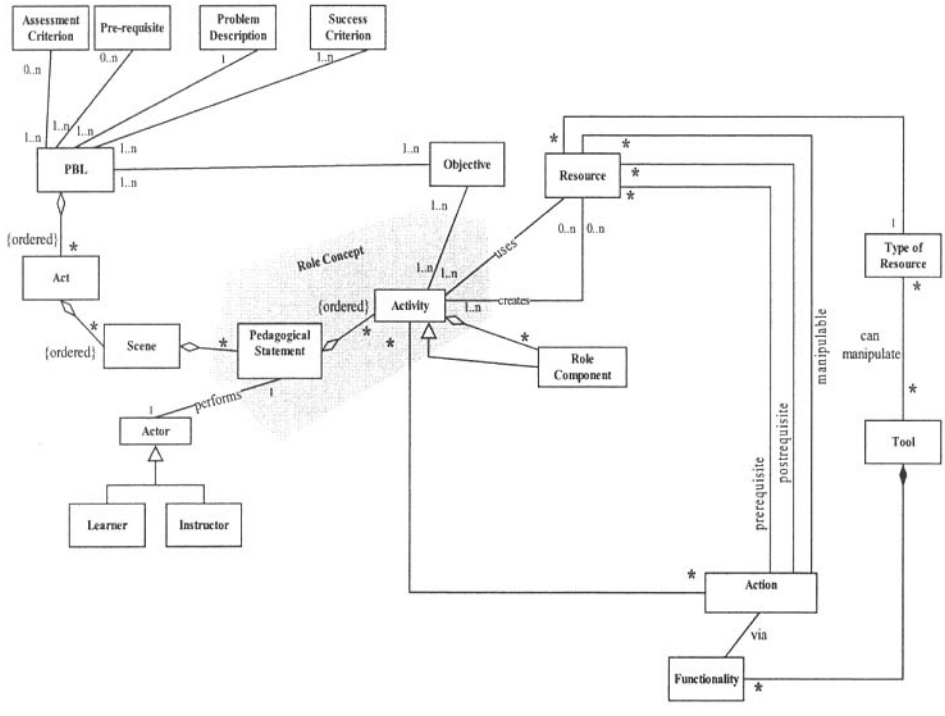


Figure 11: version 1 of the UML class diagram

This UML class diagram shows the associations between classes in our IS: cf. *Scene, Actor, Role, Resource, Action, Tool, ...* classes.

This schema shows that each actor, while playing his role, will do several activities that we can trace through his concrete actions (at the user interface level) ; these actions are engaged thanks to the tools each actor has at his disposal according to a peculiar context. As a consequence, this schema enables us to describe (during the preparation stage) and to control (during the regulation stage) the resources and tools that the actors are allowed to use in the framework of the collaborative activities.

4.2 Dynamic behaviour description

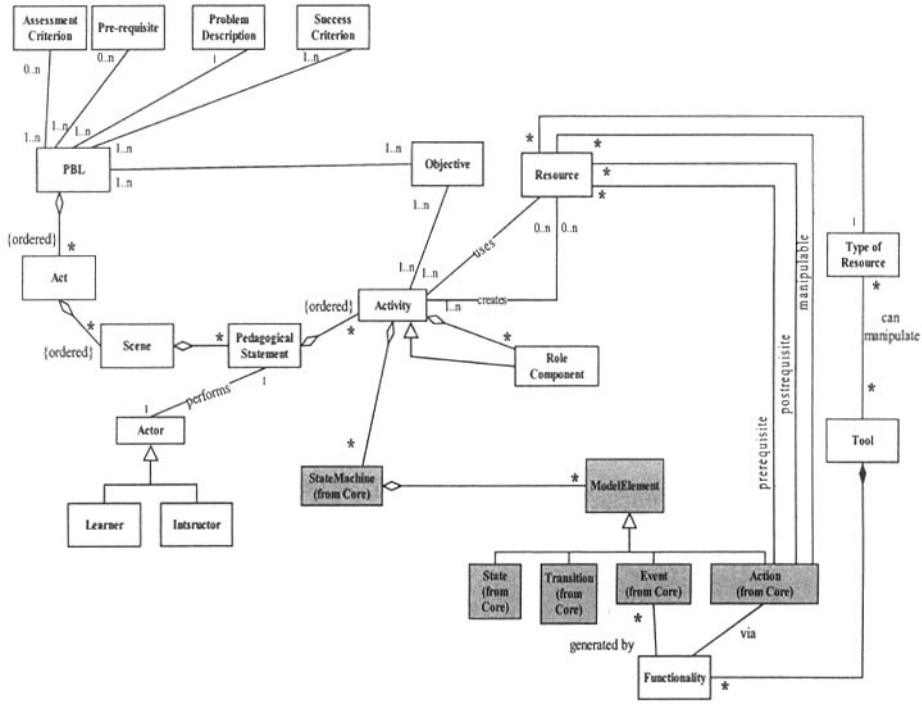


Figure 12: version 2 of the UML class diagram

This UML class diagram imports the dynamic part of the UML meta-model. So, it manages all the information needed to take into account dynamic behaviours of cooperative activities: state-transition diagrams, sequence diagrams as well as collaborative diagrams are stored thanks to this approach.

The "ModelElement" class is the super-class of the *State*, *Transition*, *Event* and *Action* classes in the UML meta-model. As a consequence, we are then able to store in this model the dynamic behaviours presented in the previous section: the *State* class has several associations with the *Transition*, *Event* and *Action* classes to enable a designer producing a state-transition diagram.

Hence, it is now possible to describe, step by step, the actors expected behaviours (actions that can be engaged by an actor in a peculiar state of the cooperative activity). This new step allows one to describe more precisely the expected activity within a Pedagogical Statement.

4.3 Actors activities control and tutoring

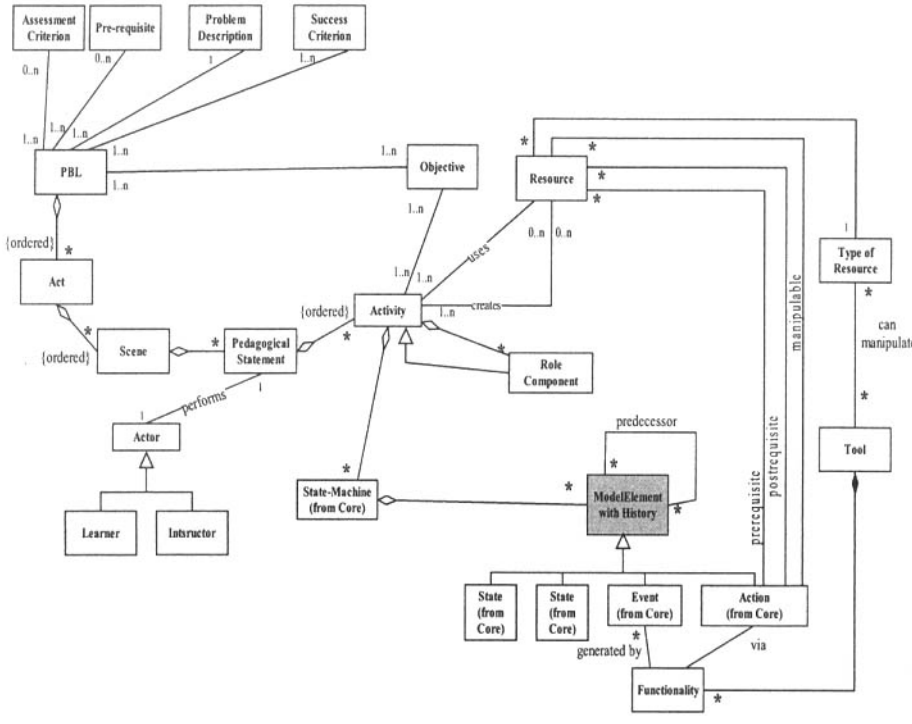


Figure 13: version 3 of the UML class diagram

From this schema the system can know the actions which are available in a given context and it can eventually prevent users from non-authorised actions and cooperations.

This diagram also enables to trace, step by step, the actual activity of an actor. That is :

- firstly, to be able to trace the history of the activities and events that an actor generates,
- then, to guide the actors thanks to this information, in order to make the actions and cooperations that they do become more efficient (lead to solve the PBLS),

So, we tried to take into account the History of the activities achieved by the different actors. To do so, we considered that an Activity should behave as a State-machine with History. This peculiar class is the entry point to access to several Historic-enhanced classes which are the action, the state, the event and the transition.

The Super-class called "ModelElement with History" incorporates an association named "predecessor" that allows to trace the successive states, events, transitions and actions reached or done by an actor. From these classes, the Information System will be able to trace the collaborations between the actors involved in the solving of a PBLS.

Tutoring activities can then be provided from the mapping between :

- the available actions defined for the role that an actor is playing, considering the current state of his activity,
- the history of the activities done by the actor while playing his role.

This mapping will be inferred from :

- the prerequisites and post-requisites of each action that can appear in a State-Transition diagram (see role-components state-transition diagrams)
- the prerequisites and post-requisites of each task that a particular actor must fulfil to play his role in the considered PBLS (see roles state-transition diagrams).

5 Conclusion

In order to provide practitioners with Problem based Learning cooperative environments [4], [3], we showed that it is necessary to develop tools which are both educational and collaborative. We address this objective globally, that is to say that we try to specify, first, the roles of the learners who are supposed to cooperate when solving a particular PBLS, and the roles of the instructors if any. Both static and dynamic characteristics of these roles, can be specified as instances of a UML class diagram. These specifications define the behaviour of the cooperative environment that will be exploited during the regulation phase by the actors (learners and instructors).

An interesting feature of our UML class diagram is that it includes part of the UML meta-model, itself : "state", "event", "transition" and "action" which are UML concepts. These topics enable us to trace the activity of the actors during the regulation phase but also to provide the actors with some tutoring functionalities. We also showed that the concepts of « role » and « role-component » are powerful enough to specify the relationships between the different groups of learners, and between the learners and the instructors.

This organization of a learning in terms of play, act, scene, ..., role, actor, tasks, is original and quite different from more classic approaches that are less centered on the cooperation among learners. The structural approach (meta-model) of the IS that we propose integrates these aspects.

In order to implement the structural approach presented here, we make functional and architectural propositions in [24]. The functional approach of this IS, drawn from [13] and [17], describes its main functions into four (Expert, Learner, Tutoring and Communication) modules. The architectural approach of the IS integrates distributed proposals: for both local and remote centralized resources and services management via client machines. These machines can be alternately client or server (cf. remote machines cooperation [19], [20], [21]). This is a first attempt to deal with the distribution of actors, data and services [11], [12] during the regulation phase of a particular PBLS. These functional and architectural approaches describe the other aspects of an Net-Learning IS.

We do know that the results presented in this paper must be considered as a first step for the development of a meta-model describing particularly services and tools enabling practitioners to produce cooperative learning environments.

Like [1], [14], [15] and [18], we think that it will be necessary to bring to the foreground standards concerning the organization of such IS, the collaborative learning interfaces, or still, the specification of dedicated educational contents.

6 References

[1] *Standardization in the field of information technologies for learning, education, and training to support individuals, groups, or organizations, and to enable interoperability and reusability of resources and tools*: http://jtc1sc36.org/terms_of_reference.html

[2] *Designing Cooperative Systems. The use of theories and models.* Proceedings of the 5th International Conference on the Design of Cooperative Systems (COOP'2000), IOS Press, 2000.

[3] De Michelis, G and all. (1998) *Cooperative Information Systems : a manifesto,* M. P. Papazoglou and G. Schlageter (editors) Cooperative Information Systems: Trends & Directions, Academic-Press, New York, pp. 315-363

[4] Y. Miao and all, *An activity-Oriented Approach to visually structured knowledge representation for problem-based learning in virtual learning environments*, Proceedings of the 5th International Conference on the Design of Cooperative Systems (COOP'2000), IOS Press, 2000, pp 303-318

[5] Rob Koper, *Modelling Units of Study from a Pedagogical Perspective : the pedagogical meta-model behind EML*, Educational Expertise Technology Centre, Open University of the Netherlands, First Draft, Version 2, June 2001

[6] www.uml.org

[7] Trygve Reenskaug, *Modelling Systems in UML2.0 A proposal for a clarified collaboration*, Version of June 22, 2001, Trygve.Reenskaug@ifi.uio.no, Mogul Norway A/S: http://www.ifi.uio.no/~trygver/documents/2001/uml20/Collaboration-011.book.htm

[8] Ralf Depke et all, *On the Integration of roles in the UML*, Technical Report No. 214, University of Paderborn, Dep. of Comp. Sci., August 2000, http://www.uni-paderborn.de/cs/ag-engels/Papers/2000/DepkeTR214.pdf

[9] T. Tamai, *Objects and roles: modelling based on the dualistic view, Information and Software Technology,* 41 (1999) 1005-1010, Elsevier Science, www.elsevier.nl/locate/infosof

[10] Grégorie Bourguin, *Un support informatique à l'activité coopérative fondé sur la Théorie de l'Activité: le projet DARE.* Thèse de Doctorat de l'Université des Sciences et Technologies de Lilles 2000 (n° ordre: 2753)

[11] Peter Muth, Dirk Wodtke, Jeannine Weissenfels, Angelika Kotz Dittrich, Gerhard Weikum, *From Centralized Workflow Specification to Distributed Workflow Execution*, Journal of Intelligent Information Systems (JIIS), Kluver Academic Publishers, vol. 10, n°2, 1998

[12] www.corba.org

[13] Méndez, Gonzalo; Antonio, Angélica de; Herrero, Pilar, *PRVIR: An Integration between an Intelligent Tutoring System and a Virtual Environment,* World Multiconference on Systemics, Cybernetics and Informatics, SCI'2001 proceedings vol. n°8

[14] Shanmugam, Ramesh, Balasubramanian, Muthukumaran, *Management of Internet Education in 21st Century - TQM Approach,* , World Multiconference on Systemics, Cybernetics and Informatics, SCI'2001 proceedings vol. n°8

[15] Robby Robson, *Report on Learning Technology Standards,* proceedings of ED-Media 2000 published by the Association for the Advancement of Computers in Education, edited by Jacqueline Bourdeau and Rachelle Heller

[16] Computer Associates, Jasmine ii e-business platform for Information Management: http://www.cai.com/products/jasmine.htm

[17] Daniel D. Suthers, *Architectures for Computer Supported Collaborative Learning,* proceedings of the IEEE International Conference on Advanced Learning Technologies (ICALT'2001), 6-8 August 2001, Madison, Wisconcin

[18] L. Anido et al., *A Component Model for Standardized Web-based Education,* WWW'10, 1-5 May 2001, Hong Kong, ACM 1-58113-348-0/01/0005 - www10.org/cdrom/papers/106

[19] http://directory.google.com/Top/Computers/Software/Online-Training/Collaborative_Learning/

[20] http://www.jxta.org/project/www/docs/OpenInnovative.pdf

[21] http://www.voelter.de/data/pub/3t.pdf

[22] Develay, M. (1993). De l'apprentissage à l'enseignement. *Collection Pédagogies*

[23] Meirieu, P. (1994). Apprendre... Oui mais comment? *Collection Pédagogies*

[24] Sallaberry, C., Nodenot, T., Marquesuzaà, C., Bessagnet, M.-N., & Laforcade, P. (July 14th-18th 2002). An attempt to design an Information System supporting Collaborative Problem-Based Learning Situations. *The 6th World MultiConference on Systemics, Sybernetics and Informatics.* Orlando (USA), http://www.iiis.org/sci2002/

7 Annex 1: a scenario of cooperation

Global objective of act 4 is the design of the style-sheet. groups G1 and G2 learn in parallel whereas group G3 designs the final style sheet; G3 also coordinates activities and exchanges between the other groups. All actors have at their disposal Microsoft Word software and a chat service to ask questions or to supply answers. Here is an example of scenario.

G1 learns how to create a new style (operational level). The Human Instructor is observing all the activities.

On the other hand, from the begining of the act, G2 asked a question to G3: "How is it possible to create a style? And is not doing profitable work.

The Automatic Instructor is noticing that G2 did never launched Word-help system (recovery of an event). According to the educational specification (cooperation rules) established by the teacher, it is deciding not to pass on question to G3 and to send back a message to G2 by advising him to launch the Word-help system with this same question.

From reception, G2 asks again the question to G3: " How is it possible to create a style?".

Automatic Instructor notices, still again, that the Word-help system was not launched and, according to the educational specification established by the teacher, it is deciding not to pass on question to G3 and to send back a message to G2 by advising him to launch the Word-help system with this same question. The Human Instructor receives also a message from the Automatic Instructor indicating him(her) the problem of non-consultation of the help by G1. Human Instructor will then decide on the assistance to supply :

- *Tele-manipulating the computer of G1 in order*
  - *To launch help or*
  - *To apply the appropriate suite of operations ("Size" menu - "Style" option and then "Create a new style") or*
- *Sending a more explicit message or*
- *Ignoring the message.*

Then, the scenario goes on.

Information Modelling and Knowledge Bases XIV
H. Jaakkola et al. (Eds.)
IOS Press, 2003

A Concept of Life-Zone Network for a High-aged Society

Jun Sasaki, Takushi Nakano, Takashi Abe and Yutaka Funyu

Iwate Prefectural University, Faculty of Software & Information Science
Sugo152-52, Takizawa-mura, Iwate-ken, Japan 020-0193
Phone: +81-19-694-2568, Fax: +81-19-694-2569, E-mail: jsasaki@soft.iwate-pu.ac.jp

Abstract. This paper proposes the concept of a Life-Zone Network (LZN) to create a new information environment in a residential area. First, we describe some problems of a high-aged society in advanced countries and the requirements of new information systems. Features of a Japanese rural area are also described. Next, we show the concept of LZN and the need for information systems in a rural area. Here, a method for the economical construction of a network is proposed. A suitable network model and capacity are estimated using the results of a traffic calculation and wireless LAN experiments.

1. Introduction

In Japan, the population rate of elderly people has been increasing more rapidly than in other advanced countries in the world. The proportion of people older than sixty-five is expected to be 22% of the entire Japanese population by 2010 [1]. A decrease in the number of working people and an increase in medical costs and welfare services will have a significant influence on industry and the government budget. Proposals to solve the issues mentioned above should be considered. However, there have not been many studies conducted on the problems of and potential solutions for the improved daily life of such a high-aged society.

In contrast, the use of Information Technology (IT) in daily life has been studied by IT vendors. Some benefits have been confirmed [2]. Now, the development of an electronic Government (E-Government) has been accelerated in Japan to improve official workflow efficiency through the introduction of IT [3]. Although this idea has been based on corporate experience, there has not been a systematic approach to applying it to government. Its successful implementation would be achieved at the individual level by using IT in the daily life of residents, the most important users of information on the country.

All people do not share the benefits of IT. Limited services, such as the use of cellular telephones, have been primarily targeting young people. This is one reason for the low usage rate of IT. Our new concept of Life-Zone Network (LZN) was created to solve the problem efficiently using IT for a high-aged society [4]. As our study location, we selected a rural area in Japan's Iwate prefecture, surrounded by mountains, which has the highest percentage of elderly people in Japan [5]. We first surveyed the information services needs in daily life. Next, we estimated the telecommunication traffic volume of potential network services. Then, we described a design method for an access network in such a rural area. A wireless local area network (LAN) was determined to be the most economical network design. We also introduced these experimental designs in a real study location to evaluate the performance of wireless LANs as part of LZN.

2. Problems of a high-aged society

In Japan, a "high-aged society" means that the proportion of people above age sixty-five is greater than 14% of the population [6]. Generally, the problems associated with a high-aged society include a decrease in the workforce and an increase in medical and welfare outlays. These could result in a decline in local industry, a reduction of the local economy, and an increase in the number people with health concerns, problems that are especially evident in an area with a large high-aged local population. The reasons mentioned above are expected to lead to the eventual breakdown in the economic structure of the local area.

Elderly people seem to have difficulty in using various information environments. We call it the "Digital Divide," and it is a significant problem [7]. Although, the Japanese government and volunteer groups have held "IT lectures" for older people in several Japanese cities to solve the Digital Divide problem, the rate of participation is even lower in rural areas than in metropolitan areas. Thus it is also a rural-area problem [8].

3. Rural area surrounded by mountains

Approximately 80% of all land in Japan is forested or mountainous, thus there is not much available living space in Japan [9]. A city, town or village that has a low population density and a little available area for agriculture (less than10%) is called "Chusankanchi" in Japanese [10]. In Japan, the Chusankanchi are expected to reach a largely high-aged society most rapidly.

We selected the rural area called Cassiopeia Lenpou in Iwate Prefecture as a research field because it has the highest proportion of elderly people (Figure 1) and it is a typical Chusankanchi [11]. We theorized that if the problem could be resolved in an area with such extreme population conditions, the solution would be effective in other high-aged areas. Cassiopeia Lenpou consists of one city, three towns and one village. The population is about 70,000, and it is 40km from east to west and 20 km from north to south. Its main industries are forestry, agriculture and small commerce.

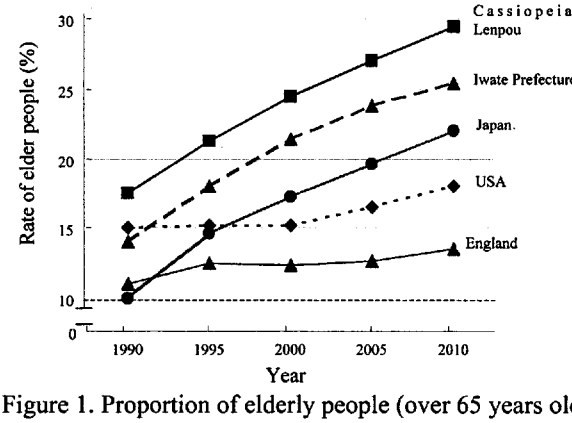


Figure 1. Proportion of elderly people (over 65 years old).

4. Concept of Life-Zone Network

Although the high-aged society problem is especially apparent in rural areas now, it will occur in metropolitan areas in the near future. There, efficient governmental services for residents are required to maintain the level of their daily life. The introduction of IT is an important key to providing these services. A metropolitan area network, a private network for an enterprise or university and an electronic village constructed by government for a special experiment already exist, but a useful network for a resident's daily life is not available.

We propose the concept of a life-zone network (LZN) as shown in Figure 2 [4]. This network is usually small in size and provides a connection between main service organizations and subscribers such as elderly people who need support in their daily life. The features are as follows:

- Provision of combined services -- users such as elderly people can be provided combined medical, welfare, and shopping services;
- Sharing of information -- main service organizations such as medical or welfare offices can share needed user information;
- Service-coordination -- combined services and information on user needs are easily determined by a service coordinator,
- Scale flexibility -- the scale of the network is flexible, adjusting to expansion of service combination or an increase in the number of users.

One of the merits of this concept is an extremely high serviceability, because users can be provided complete services by multiple organizations, for example, medical and welfare organizations, and retail stores. In a conventional situation, users have to think of and order from each individual organization themselves. We believe that a system based on this concept will be the social infrastructure to support elder people.

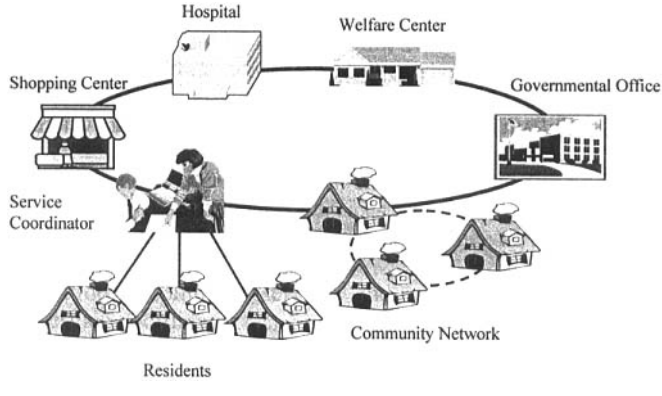


Figure 2. Concept of Life-Zone Network.

5. Survey of need for information system

We surveyed residents living in the Cassiopeia Lenpou area to determine the need for an information system as proposed by LZN.

5.1 Survey methods

The following are our survey methods:

- Selection of 996 residents at random using telephone address book;
- Distribution of three answer sheet sets per resident (a total of 2,988 answer sheets are distributed);
- Questions consist of the following six categories:
  1) Personal information;
  2) Communication services;
  3) Health, medical and welfare services;
  4) School and education services;
  5) Shopping services;
  6) Governmental services.

5.2 Completed responses

We received 410 answer sheets (41% responded). Respondent ages are as follows: 7% from persons under 20 years old; 6%, 20 – 29; 12%, 30-39; 17%, 40-49; 18%, 50-59; 21%, 60-69; and 19% over 70. The ratio of male to female is 51% to 49% of respondents.

5.3 Survey results

The study area can be considered a typical Chusankanchi because the rate of elderly people is high. The following results were obtained from this survey:

- The telephone is the most important method of communication, and 67% are local telecommunications;
- Privately owned vehicle provide the main form of transport to hospitals (62%), however for people over 60 years old, the rate of using private vehicles is comparatively lower (29%);
- The rates of people who feel inconvenienced by the far distance to shopping areas are large especially in the 10-19 year old group, and for those over 60 years old (greater than 50%).

The estimated service usage rate represents the need for a service using IT, as shown in Figure 3. The service usage rate means the rate of "use" or "potential use" of a service in all responses. Figure 3 shows that the Chusankanchi have the greatest need for communication methods such as local free telephone (66%) and e-mail. Telemedicine (48%), hospital information service (45%) and government information service (47%) are also highly desirable. These responses result from the inconvenient location of the study area. Because there are few hospitals and transportation is not easy, telemedicine, home care service and medical information service become quite important.

The need for an electronic shopping service is comparatively low (24~33%). Electronic shopping is not widely recognized in the study are because the Internet is not popular here.

We conclude that required services are as follows:

- Local communication service;
- Medical, welfare and government information service;
- Information service for female and older people (telemedicine, home care and electronic shopping).

These results correspond with the services proposed by LZN.

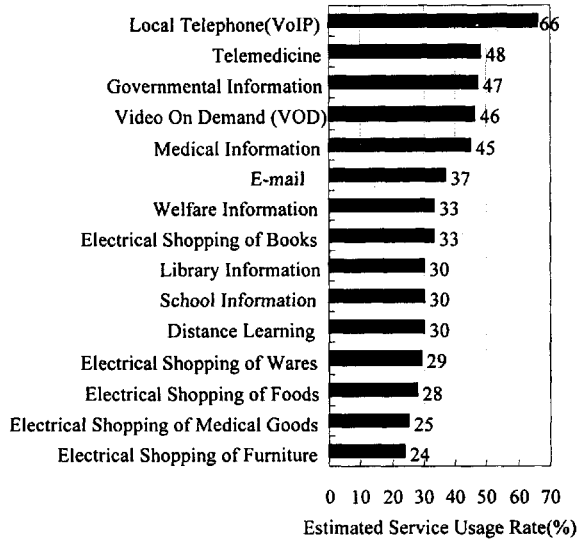


Figure 3. Survey results of Estimated Service Usage Rate.

6. Economical network design

As the Chusankanchi has a larger area and lower population density than a metropolitan area, users are distributed throughout a wider area and the organizations that provide services are located farther from users than in a metropolitan area. In these areas, the construction costs of cabling tend to be expensive because of the mountains. For this reason, we studied the possibility of introducing a wireless network in the area. At first, we surveyed the population distribution in a typical Chusankanchi and then considered some suitable network models.

Figure 4 shows an example of the population distribution in our research area, a typical Chusankanchi. Using this figure, we can find that the typical Chusankanchi consists of a concentrated area of population and several smaller groups distributed around the area. We decided to consider two types of physical network models, one for the concentrated area with a radius of several kilometers, and the other for locations distributed around the concentrated area.

To simplify the evaluation of network construction costs, we consider the use of CATV and wireless LAN in two models for the concentrated area and in three network models for the distributed area. Two models for the concentrated area are an all-cable network and a cable and wireless hybrid network model. We did not consider an all wireless network, because we think that the telecommunication capacity of wireless LAN would not be enough yet for trunk lines in the concentrated area. On the other hand, we considered three network models for the distributed area, where an all wireless network is added on above two models.

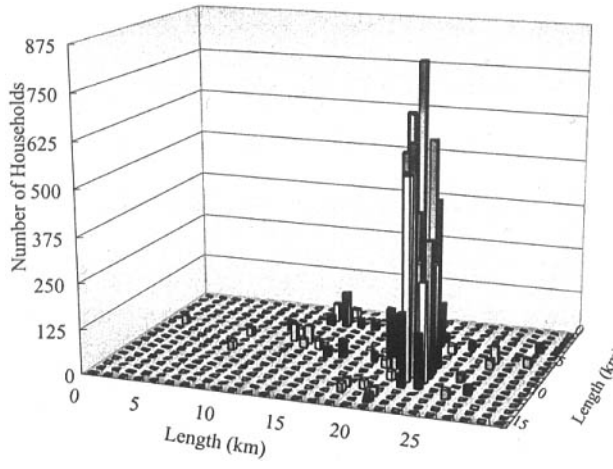


Figure 4. Population distribution example (Ichinohe town).

6.1 Concentrated area

Figure 5 shows our proposed network models for a concentrated area: (a) an all-cable network and (b) a cable and wireless hybrid access network (CW-HAN) [12]. We assumed that every cell has the same radius, R, and fewer than 50 users, to enable the comparison of calculated construction costs. Other calculation conditions are shown in Table 1.

In this paper, we mainly describe the comparison of construction cost (initial cost) for each model. Maintenance cost (running cost), reliability and traffic performance is also important issues to compare the network models. Here, as later issues will be reported another opportunity, we explain how to calculate construction cost and some calculated results as follows.

Construction costs of the all cable model (C_A) are represented by Equation (1), which shows a sum of branch cabling cost, trunk cabling cost, and connecting equipment cost (for example, switch, hub or closure),

$$C_A = S\,[2\,C_r R\,(N\,\pi R^2 / S)/3 + p + k\,R\,C_l\,]\,/\,(\pi R^2). \quad (1)$$

Construction costs of the CW-HAN (C_H) are represented by Equation (2), which shows a sum of base station cost and trunk cabling cost,

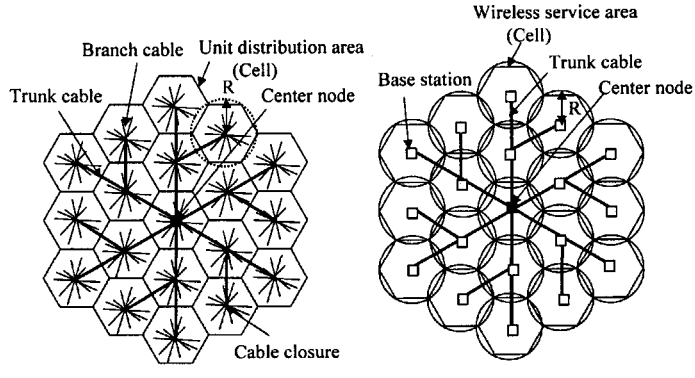
$$C_H = S\,[\,C_b + (N\,\pi R^2 / S)\,C_t + k\,C_l\,R\,]\,/\,(\pi R^2). \quad (2)$$

Where $n = (N\,\pi\,R^2 / S)$, the condition where the construction cost of CW-HAN is less expensive than that of an all cable network is shown as Equation (3),

$$N > 3\,(C_b - p)\,/\,(2\,R\,C_r - 3\,C_t). \quad (3)$$

Using the above equations, we can determine the relationships between the relative costs and the cell radius, R, for both the all cable network model and the CW-HAN model, using the applicable conditions of each model as shown in Figure 6 and Figure 7, respectively. From Figure 6, we find that the construction cost of CW-HAN is less expensive than that of an all

cable network when cell radius, R, is large. There is a crossing point where these two relative costs match each other in the same cell radius, R. From Figure 7, we find that CW-HAN is more suitable than an all cable network when there are fewer users and a large cell radius.



(a) All cable model

(b) Cable and wireless hybrid access model(CW-HAN)

Figure 5. Network topology models for a concentrated area.

Table 1. Calculation conditions.

Issues (Explanation, where cost means relative construction cost)	Values
C_b (Cost of a wireless base station)	400
C_t (Wireless terminal cost)	40
C_l (Cost of trunk cable per 1 m)	0.6
C_r (Cost of branch cable per 1 m)	0.3
P (Cost of a cable closure)	300
L_{max} (Maximum length between a center node and a base station)	2000 [m]
S (Size of a service area)	32 [Km_2]
k (Coefficient by cell shape)	1.73
N (Total number of subscribers in all service area)	2,400
n (Number of subscribers in a service area)	———

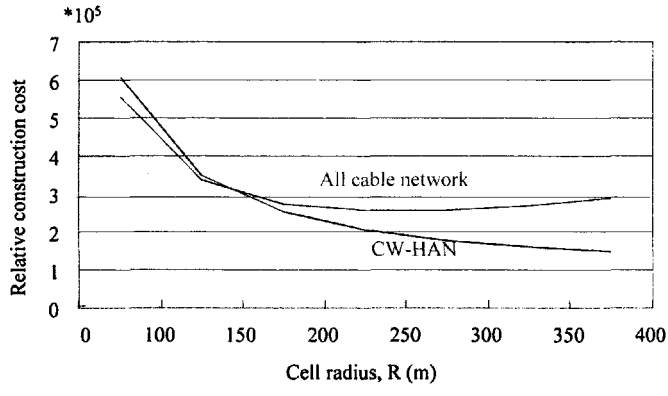


Figure 6. Construction cost of each network model in a concentrated area.

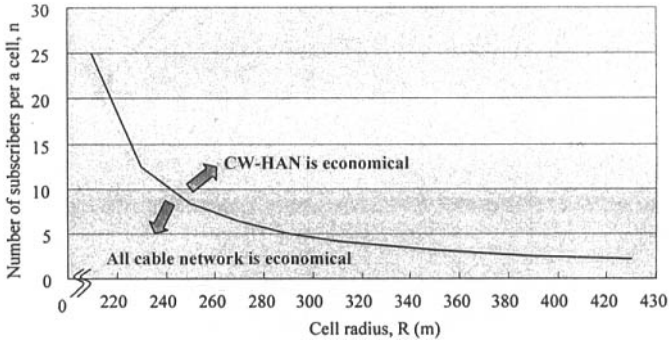


Figure 7. Optimal conditions for each network model.

6.2 Remotely distributed area

For a remotely distributed area, we considered three network models: an all cable network; an all wireless network; and a cable and wireless hybrid access network (CW-HAN'), as shown in Figure 8. We assume a circular area with radius, R, apart from a center node by distance, L, with n users of constant density. The construction cost of an all cable network is shown in Equation (4):

$$C_A' = 2\,R\,C_r\,n / 3 + L\,C_l + p. \quad (4)$$

The construction cost of an all wireless network is represented by Equation (5),

$$C_W = C_b + n\,C_t. \quad (5)$$

The construction cost of CW-HAN' is shown in Equation (6), with the directional angle of radio power, θ,

$$C_H' = C_b + n\,C_t + (L - R / \sin\theta)\,C_l. \quad (6)$$

The conditions where the relative construction costs of an all wireless network and CW-HAN' are less expensive than an all cable network are shown by Equations (7) and (8), respectively,

$$R > 3 (C_b + n C_t - L C_l - p) / (2 C_r n), \quad (7)$$

$$R > 3 (C_b + n C_t) / (2 C_r n + 3 C_l / \sin\theta). \quad (8)$$

Using the above equations, we can obtain the results shown in Figure 9 and Figure 10. They represent the relationship between the relative construction cost and the distance, L, from a center node to the center of a remote area, and the relationship between that distance, L, and the radius, R, of a remote area, respectively. From Figure 9, we can find that for a small R, an all cable network is less expensive than an all wireless network or CW-HAN' is. However, with a large R, an all cable network becomes more expensive than other networks. Although CW-HAN' is always more expensive than an all wireless
network, it can cover more distant users from the center node than all wireless network can because of the long reach of radio power. Figure 10 shows approximate optimal applicable area of each network model. When the radius, R, is over 300 m and the distance, L, is over 2 km, we determined that CW-HAN' is the most suitable. When the radius, R, is over 200 m and the distance, L, is within 2 km, an all wireless is more economical, and when the radius, R, is less than 150 m or 200 m, an all cable network is more economical.

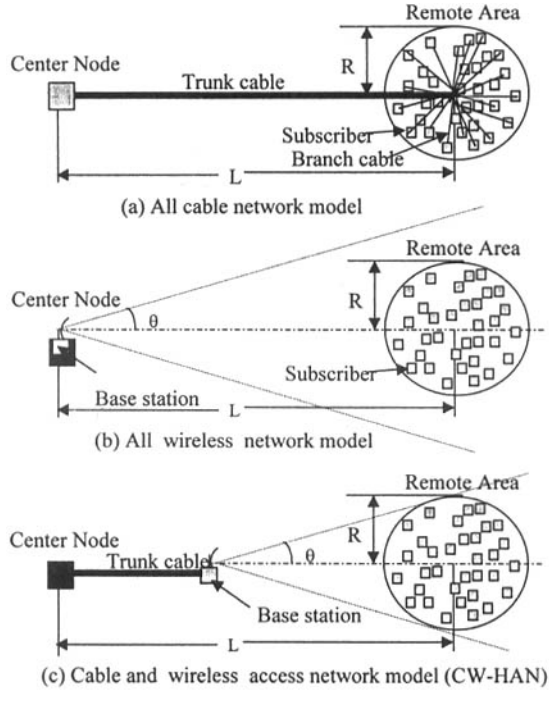


Figure 8. Network topology models for a remotely distributed area.

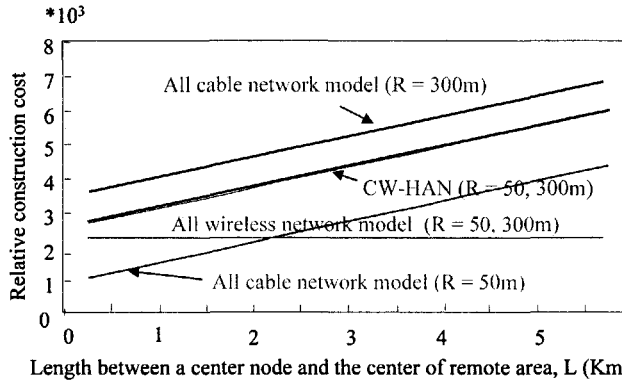


Figure 9. Construction cost of each network model for a remote area.

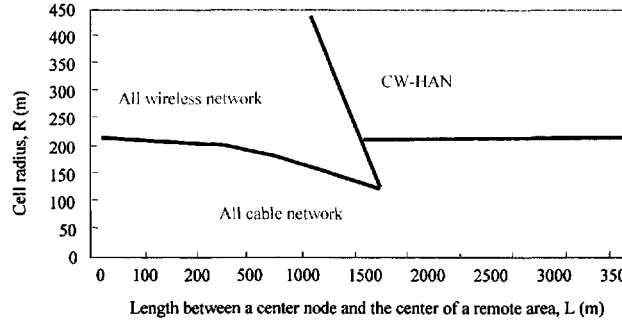


Figure 10. Optimal applicable area of each network model for a remote area.

6.3 Network design method in Chusankanchi

Using the calculated results mentioned above, we could design the best network for a Chusankanchi, as follows:

1) Identify the target area and available trunk lines on a map. Set the cell radius of service area to be R;
2) Determine whether the area is a concentrated area or a remotely distributed area by considering the number of users, n, and the topographical conditions;
3) Use the following design conditions in a concentrated area:
When $R < 200$ m and $n < 20$, all cable network is suitable.
When $R > 200$ m and $20 < n < 50$, CW-HAN is suitable.
4) Use the following design conditions in a remotely distributed area:
When $L < 2$ km and $R > 200$ m, an all wireless network is suitable.
When $L > 2$ km and $R > 200$ m, CW-HAN is suitable.
When $R < 200$ m, an all-cable network is suitable.

Figure 11 shows a network design example for a town in Cassiopeia Lenpou. The selected network was determined by applying the above design methods.

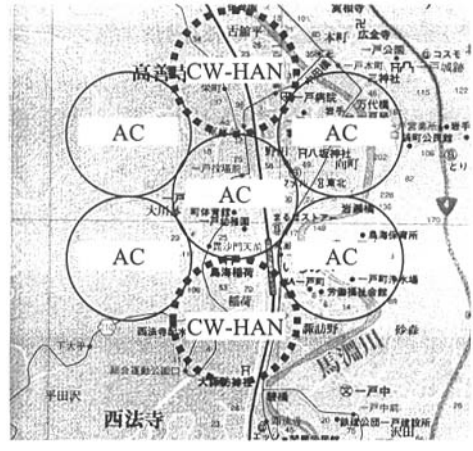


Figure 11. Example of network design.

7. Estimation of telecommunication traffic

In the previous section, we described the design method for a LZN where a wireless LAN was introduced. The volume of telecommunication traffic must now be identified to enable the design of the needed network capacity. Although a conventional study has been conducted on the regional-network design based on frequency of telephone communication [13], there has been less research completed on the design method based on residents' telecommunication-service requirements in a local area network such as the LZN. The survey results mentioned in Section 5 were used to establish a network design for a Chusankanchi, according to the LZN concept. This network model is shown in Figure 12, where there is a front-end network on the users' side, and a backyard network connecting the service organizations. The backyard network may have either a star topology or a loop topology, as shown in Figure 12.

We consider that the front end network may be wireless or cable in any area and the backyard network should be cable network in concentrated area. It is important to clarify the required bandwidth for each telecommunication line. In this paper, we describe the estimation of telecommunication traffic for the backyard network.

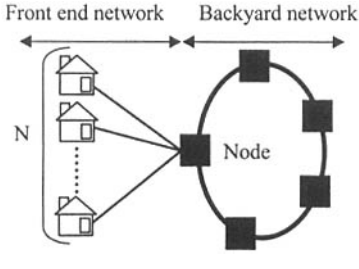


Figure 12. Network model.

Table 2. Calculation conditions of network traffic.

Services	Technology	Bandwidth	Service using time (sec)	Service usage frequency (times/week)
VOD	FTP	1.5Mbps	600	7
Telemedicine	TV conference		1800	7
Distance learning			1800	3
Local free telephone	VoIP	64Kbps	7200	2
E-mail	e-mail		1800	2
Governmental Information	HTTP		3600	7
Medical Information			1800	1
Electrical Shopping of books			1800	2
Welfare Information			3600	3
School Information			1800	3
Library Information			1800	1

7.1 Conditions for network traffic estimation

The eleven kinds of network services to be provided to residential users of the LZN are shown in Table 2, and are based upon the survey results with the highest reported use. The development of economical network services required the following assumptions: the circuit capacity to be 64kbps for normal-speed services (N-ISDN) and 1.5Mbps for high-speed services (B-ISDN), including Video-On-Demand (VOD) and teleconferencing; and that a user does not require the use of multiple services simultaneously. Assumptions concerning the duration of use and service usage frequency are summarized in Table 2.

7.2 Calculation of telecommunication traffic

Mean bandwidth for each service is shown in Equation (9),

$$V = M\ T\ W / (25{,}200\ D)\ \text{kbps}. \tag{9}$$

Here, M is usage frequency of a service per week (times/week), T is the time period for one use of the service (sec), W is bandwidth of the service (kbps), D is the available time per day (hour). As every user does not use all services, each service has a usage rate, U (%). Multiplication of V and U becomes the real mean bandwidth of the service. When n is the total number of services, the sum of the mean usage bandwidths for all services is described in Equation (10),

$$S = \sum_{i=1}^{n} V_i U_i / 100\ \text{Kbps}. \tag{10}$$

Using Equations (9) and (10), the mean traffic volume flowing into a node, $E = N S$, is shown in Equation (11),

$$E = (N/25{,}200) \sum_{i=1}^{n} M\, T_i\, W_i\, U_i / D \quad \text{Kbps}, \tag{11}$$

By using the values in Table 2 and values for U, the estimated service usage rate as described in section 5 and Equation (11), we can calculate the relationship between the mean bandwidth and the number of subscribers, N. Figure 13 shows the calculated result, with available usage time, D, as an additional parameter. From Figure 13, we find that for D=12 (hours), 50 subscribers would be available as users if an access network with 4.5 Mbps bandwidth, such as a wireless LAN is used.

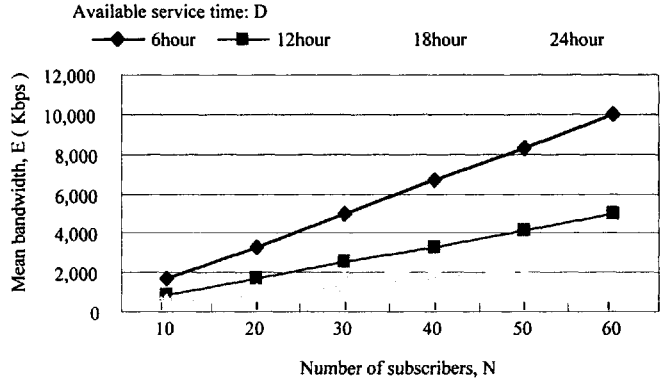


Figure 13. Relationship between mean bandwidth and number of subscribers.

8. Experiments with a wireless LAN

We mentioned that we could construct an economical network if we would use a wireless LAN in Chusankanchi. It is necessary to confirm the performance if a wireless LAN in a real rural area. In this section, we show our experimental results of using wireless LAN in a Chusankanchi area.

A wireless LAN has already been described as appropriate for the construction of an economical access network for LZN. Although some examples of 2.5 GHz frequency bandwidth wireless LAN (IEEE 802.11b) use exist in a metropolitan area [14], none are available for a Chusankanchi. This wireless LAN has been reported that it has extreme performance with 11Mbps-telecommunication capacity and 2km-transmission length. Thus, we tried to confirm the serviceability and reliability of the IEEE 802.11b LAN through experiments in a real field.

Figure 14 and Table 3 show the experimental system and specifications of experimental equipment, respectively. The field is between a governmental office and a welfare center, where the direct distance is 900 m, located in Ichinohe town in Cassiopeia Lenpou. We obtained the following experimental results:

- Ping transmission test (64Byte, 1 sec interval, 50 times): Mean response time is about 4 ms for every antenna condition. No additional delays and packet losses caused by wireless system are observed.
- Web access, IP telephone (using IP-BOX), TV conference (using PC-installed CU-

SeeME) test: There is no stress and no difference in usage feeling from 10 Mbps Ethernet.
- File transmission test (shown in Figure 15): Maximum throughput is 4.5 Mbps using two directional antennas, and 3.5 Mbps using a directional antenna and a non-directional antenna. These results show that the throughput is higher than reported in the Maeshima et al. field evaluation [14], where a similar experiment was tried in a metropolitan area. We believe that a Chusancanchi has a fine radio environment to use a wireless LAN.

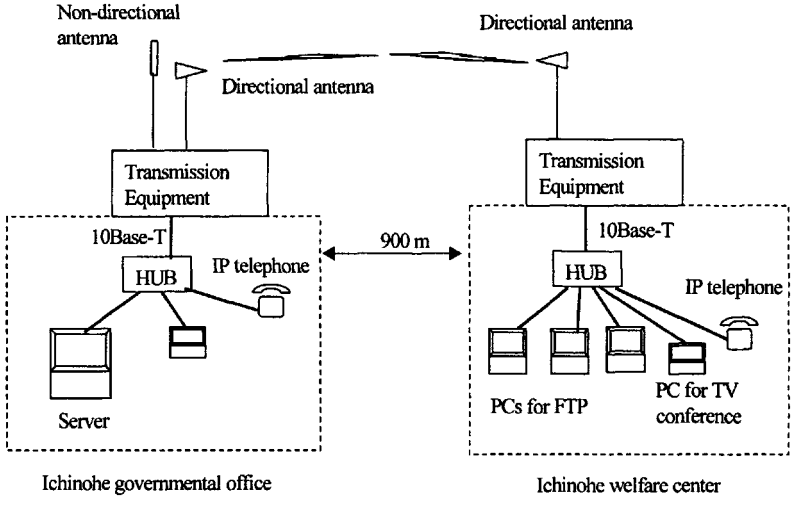


Figure 14. Experimental system for wireless LAN.

Table 3. Specification of experimental equipment.

Items	Main specification
Server	OS:Linux CPU:MMX Pentium 200MHz Memory : 32MB+32MB , VRAM2MB Inside Network : 10Base-T, duplicated
PC (clients)	OS:Win95 CPU:Pentium150MHz Memory :16MB+32MB,VRAM 2 MB NIC: PCMCIA10Base-T, half duplicated
Wireless transmission equipment	RGW2400/OD (by Root Ltd.) 2.4GHz Direct spectrum spreading Performance : 11Mbps (Official)
Antenna	Plane patch (Directional) 8 step collinear (Non-directional)

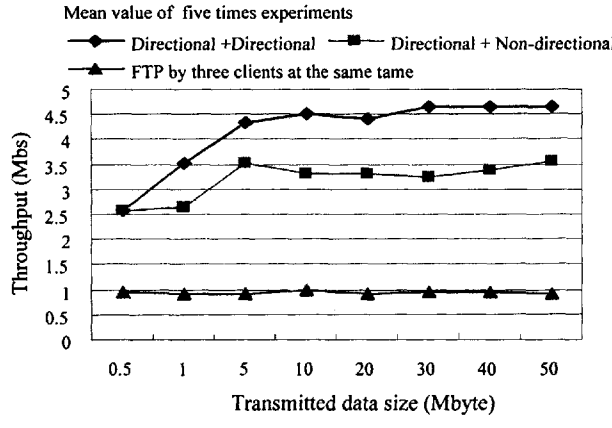


Figure 15. Experimental results using FTP.

9. Conclusion

Japan is shifting to a higher-aged society at the highest rate among advanced countries. The high-aged society will create certain problems such as a decrease in the number of working people and an increase in the medical and welfare budgets. So, it is especially important to adequately support the daily life of older residents.

1) We propose the new concept of Life-Zone Network (LZN) to resolve this issue. This involves the construction of a regional network with supporting and coordinating functions for residential life, for example to connect elderly people with volunteers, and provide information sharing venues between medical and welfare organizations. This would provide helpful services and useful information efficiently and economically, especially to elderly people.

2) We selected the Cassiopeia Lenpou Chusankanchi in Iwate prefecture as a research field because the area has the highest proportion of aged people. There, we surveyed the residents' requirements to determine necessary services to be provided by LZN.

3) To construct an economical network in a Chusankanchi, we describe a design method based on the distribution of population. This design involves five kinds of network models with cable and wireless access networks that we use to evaluate construction costs and estimate optimal conditions for the use of each network model.

4) The results of the necessary service survey were used to estimate the telecommunication traffic volume, important for selecting the proper LZN design.

5) Our field experiments confirm that a wireless LAN would be the best system for our study area conditions.

As a result we know how to select and construct a LZN system for a particular study area. Using the methods identified in our research, we can clarify the necessity, and design and construction methods for a LZN. The costs of maintaining a LZN require further study. We hope to determine needed real services, determine a budget and construct the LZN system as a basic infrastructure for the high-aged society.

References

[1] http://www.city.yokohama.jp/me/kikaku/syousi/mm/0z35.html
[2] Stan Moyer, Dave Maples, Simon Tsang, and Abhrajit Ghoshi, "Service Portability of Networked Appliances", IEEE Comm.Magazine,Vol.40, No.1, pp.116-121 (2002).
[3] http://www.e-gov.go.jp/
[4] Jun Sasaki, Takashi Mitsuishi and Yutaka Funyu, "A New Concept and Configuration Method of Life Zone Network", ICOIN-15 2001, pp.381-386 (2001.2).
[5] http://www.pref.iwate.jp/~hp0351/sakuteikeikaku/koureishishin/f2-1-1.htm
[6] http://www.zzz.or.jp/free/youhi828/aged/aged1.htm
[7] http://www.digitaldividenetwork.org/content/sections/index.cfm
[8] http://www.pref.iwate.jp/~hp0212/it/
[9] http://www.kentei.gr.jp/KIDS/LIB/syakai/data/kokudo.html
[10] http://www.kanto.maff.go.jp/nou_seibi/2/2/3/a/menu.htm
[11] http://www.nnet.ne.jp/~nikou/
[12] Jun Sasaki, Takashi Mitsuishi and Yutaka Funyu, "Cable and Wireless Hybrid Access Networks for Rural Residential Areas",TENCON2001,1922001 . 8 .
[13] Hajime Matsumura, Shigeki Yamada and Minoru Kubota, "Reliability and Cost Effectiveness of a Telecommunication Network Composed of Distributed Switching Nodes and High Performance Control Nodes" Trans. of IEICE, Vol.J83-B, No.8, pp.1135-1147 (2000).
[14] Osamu Maeshima, Naoki Fuke, Yasuyuki Watanabe, Kouichi Hirose, Masaya Nakayama and Shinichi Nakagawa, "A Field Experimentation and Evaluation of 2.4 GHz Band Wireless LAN in an Urban Area", Proceedings of the 2000 Communications Society Conference of IEICE, B-7-113, p.177(2000).

Information Modelling and Knowledge Bases XIV
H. Jaakkola et al. (Eds.)
IOS Press, 2003

Embedded Programming as a Part of Object Design Producing Program from Object Model

Setsuo Ohsuga and Takumi Aida
Graduate School of Waseda University
ohsuga@ohsuga.info.waseda.ac.jp

Abstract This paper discusses automatic programming. Programming starts from building a model at high abstract level and a program code is generated therefrom. Difficulty of making correct program depends largely on this high-level model building. But In many cases it is given from outside and there is no way to manage its building. In this paper the authors discuss a method to support programming from high-level model building to program code generation. For the purpose a new information system is necessary. What is discussed in this paper therefore is a part of the research on a very general and autonomous information system. With this system a high-level model is formed as a structure-of-activities in the application domain meeting the condition of computability. Then it is translate it into program. A simple experiment has been made. Now its generalization is proceeding.

1. Introduction

The discussion included in this paper is a part of a research project on a development of a new information system [6]. It is expected that in near future problems brought into computers include various types and domains. Here the type means a set of problems that can be dealt with by a specific method for solution. For example, design, analysis, diagnosis, control, planning and so on are the different types of problem solving. In many cases a large-scale problem is composed of sub-problems of different types and domains. In order to aid human problem solvers for solving these problems and reduce the load of problem solving new information system to solve problems of the different types and domains autonomously is required. Mutually exclusive requirements such as autonomy, generality and practicality are imposed to such information systems. The authors analyzed the condition which such a new information system should meet and induced the following conclusion. The system should be of such architecture composed of two sub-subsystems with different paradigms; a back-end sub-system based on the traditional information technology and a front-end sub-system based on the declarative representation and its processing. The major difficulties in developing this system are mostly in the front-end sub-system design.

In order to meet the conditions of autonomy, generality and practicality, various new technologies must be developed to the front-end sub-system on modeling, human interface, autonomous problem solving, problem decomposition, large-scale knowledge base management, integration of different information processing methods, human-computer co-operation, automatic programming and so on. Each of them is not easy to achieve. But most of these are closed in the front-end sub-system, and no requirement for conversion of information crossing the boarder between the procedural and declarative processing. Only exception is automatic programming. A part of information in an application domain represented in the form of a problem model and processed in the front-end sub-system must be transformed into a program in the back-end sub-system crossing the boarder. Transformation of information across the boarder of sub-systems based on the different paradigm is one of the most difficult problems that have not yet been solved. As an indispensable problem for developing new information system it is discussed in the sequel.

2. Program Generation as Transformation of Information between Different Sub-systems

Figure 1 illustrates architecture of new information system. This is double paradigm architecture composed of a back-end sub-system with traditional procedural language and its processor and a front-end sub-system with declarative language and an inference engine as its processor. In this system the real world problems are brought in the front-end sub-system and processed there in the declarative manner. This architecture is deduced necessarily from the requirement for adapting multi-type and multi-domain problem solving on one hand and at the same time for utilizing the existing program resources on the other hand. The major problems for meeting the condition of autonomy, generality and practicality arise in the front-end sub-system design. Through the analysis of this problem we list up the problems as follows as the major research objectives.

(1) New modeling method; How is a problem formalized and represented?
(2) Human interfce and model building; How is the human intent represented formally and how is it represented in the model?

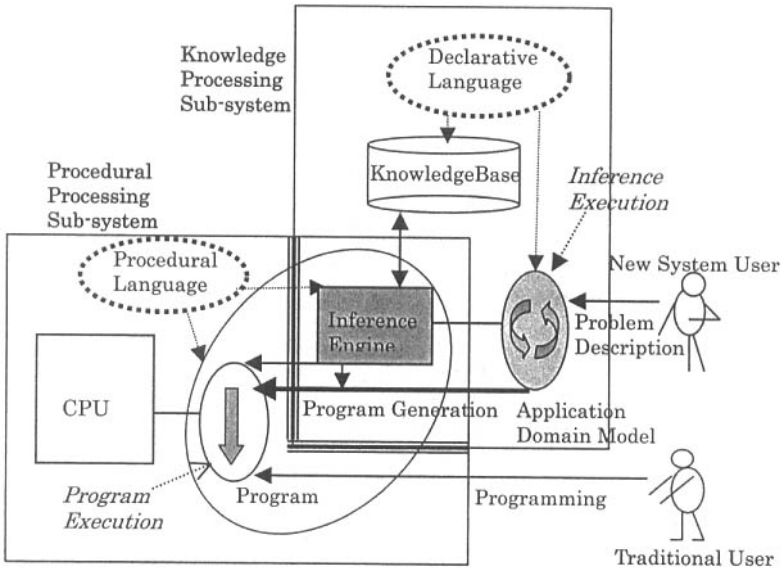


Figure 1 An illustration of the concept of new information system as a double paradigm system

(3) Large knowledge base and generation of problem solving system; How is the multi-domain knowledge managed and how is the problem solvig system suited for a specific problem generated?
(4) Autonomous problem decomposition and problem solving; How is the given problem decomposed to parts and solved autonomously?
(5)Integration of different information processing; How is a information processin method integrated with another one in the different paradigm?
(6) Automatic program generation; How is a program generated from the problem solution?
In addition to these problems knowledge acquisition through web and knowledge discovery in data are addd as the support techniques. But these topics are ignored in this paper.
Though the problem (1) through (5) are not easy to achieve these are closed in the front-end sub-system and there is no need for converting information between the sub-systems of different paradigms. An inference engine is necessary for processing declarative knowledge and it is realized as a procedural program. But it can be a black box. There is no matter how is the inference executed in view of problem representation. Moreover a part of knowledge is represented in the form of procedural program and is used during problem solving. But this kind of functional integration does not needs translation. Many researches are being made in our group.
Only exception is the program generation of (6) as is shown by a thick line in Figure 1. There is a class of problems of which the solutions are represented in the form of structure-of-activities such that user's requirement can be satisfied by executing the activities according to the order indicated by the structure. In many domains it is represented in declarative form. Therefore it must be executed by interpretation. In general it is inefficient. For achieving faster execution it is desirable to reform it into a procedural representation. If this transformation becomes possible this system becomes an automatic programming system. Problem solving in this case is to build a structure-of-activities. Very often it is easier than to design program structure in the programming domain. Thus this method achieves the easiness of program specification and high efficiency of program at execution.
In traditional programming, however, the high level model is assumed to be given from outside. Programmer can not concern it even if the model is not suited for programming. Usually a specification is made in between the model and program and programming is a process to transform the given specification into a program code [6]. Specification corresponds directly with program in the meaning but is the more comprehensive way of representation for human using expression like natural language. To make such a specification properly is often a big problem. Here each of requirement, specification and program is assumed to have a formal representation. Program is an artifact that can be executed directly by a procedural processing mechanism. It should be noted that

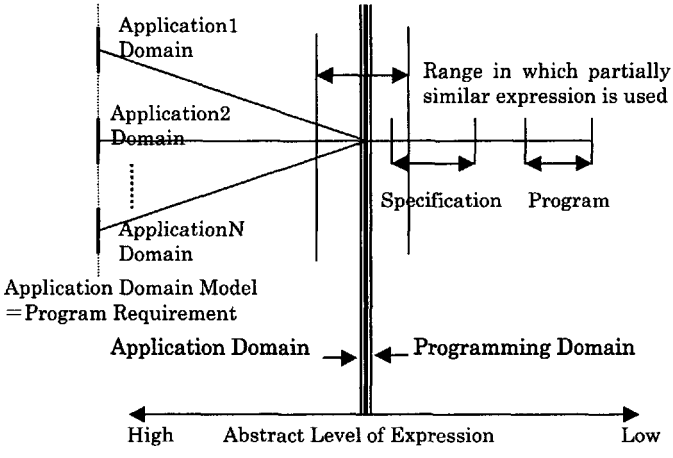


Figure 2 Relation between application domain and programming domain

the abstraction level of expression does not change continuously but abruptly during transformation from the model to program. There are the high-level and low-level zones. High-level zone belongs to an application domain and the low-level zone to the programming domain. Within a zone semantic is the same and transformation between expressions is rather easy. But transformation becomes very difficult between different zones. For example, transformation between specification at the low abstraction level and program code is not difficult but it is difficult between high level requirement and low level specification.

There are various programming paradigms with different abstraction levels like simple procedural expression and object-oriented expression. Accordingly their expressions span a certain width along the line of abstraction. Correspondingly specification spans a certain width. Specification language like UML and language Z have been developed as an attempt to express concept as close as possible to application domain. These high-level specification language introduce the modeling concepts that appear similar to model representation in the application domain. In reality however there are still difference between these models and those in application domain because of the difference in semantics. Every model in an application domain can be transformed to the other by means of domain knowledge that is available uniformly in the domain, and thereby the solution for the given problem is obtainable. Expressions of the above languages have no such characteristic. Today an automatic conversion is possible only within the same domain. Every transformation into another domain is left to human.

If however a method of automatic transforming across the boarder of zones becomes possible, then a model in application domain that appears in the process of problem solving can be the program requirement. In this case what is required for programming is to express faithfully what wants to do, that is, to specify a model in the application domain. This relates the method of modeling ((1)) in the list presented in **2**.

3. Related Researches – Crossing the Border of Zones with Different Semantics

Some researches have been made to transform information across the border of the different semantic zones. These are,

(1) reverse engineering [1]
(2) interactive method for obtaining high level model from source program [4]
(3) knowledge based software engineering [3]
(4) program generation from model in application domain [7]

These are classified into two classes; one is to obtain high abstract level model from source program in low abstract level and the other is transformation in the reverse direction. The former transformation is more difficult than the latter because some information has been lost in the process of obtaining the lower level representation. Therefore it is the role of human and the objective of information technology is limited at most to aid the persons in this work. The (1) and (2) above are of this group and (3) also relates partly to them. The latter class is easier than this and it is expected to automate the process even though it is not realized yet because of some technical difficulties. The (4) is this attempt and (3) also used for the purpose. This paper relates this problem and relates

both (3) and (4).

There is an attempt to generate embedded programs directly from application model. Since recently many embedded programs are used for controlling various engineering objects, this approach gets interest of the people working in control engineering. This approach is characterized by the closeness of program to application model. An example is the system named Matlab/Simulink [7]. It is originally a simulator of control system. An application engineer writes down a control system diagram. Then the system analyzes its behavior. The system calculates the new states of all the components of the system in a small interval between sampling times. If this calculation is achieved in real time, then this simulator can be a real control program.

MATLAB /SIMULINK is a commercial product and its application is restricted to automatic control.. But in this paper it is referred many times because it is a typical method of transforming automatically an application model into real program.

The approach of automatic programming by MATLAB /SIMULINK is not necessary exploited widely in the real application. The reasons may be in the substantial difference between this method and the traditional method of programming. The difference between this approach and ordinary programming style is analyzed as follows.

(1) Difference in the concept of programming; It is assumed that every control unit forming the application model in MATLAB /SIMULINK is based on continuous variables and represented explicitly in mathematical form. Then its calculation at every sample time is possible. If a program is provided to calculate it in real sample time, it can be a control program. This is an interpreter of application model represented in the form of control diagram. In the ordinary method of programming in the contrast, program is not made to assure the real state of object at every instance as above but designed to process sensor input and generate output to actuator so that the desired object's operation is assured as a whole. Application model in this case is not based exactly on control theory but must be made in the other way. There is not the control theory based as MATLAB /SIMULINK but based on experience.

(2) Difference of efficiencies of computations; To calculate the state of every components at every sample time is relatively inefficient comparing to program execution of ordinary programming style. If every necessary calculation is not achieved in a sample time, the properness of the control program is not assured.

(3) Difference in the characteristics of components; In the real applications it is not possible to assume that every component of control system is well formed, i.e., represented in a mathematical form with continuous variable. In the ordinary programming style, a set of standardized programs is available as a unit to form a control system. These are different in many aspects from control units used in MATLAB /SIMULINK. For example, sampled time operation cannot be applicable.

(4) Difference in representations of application models; Feed back concept in control theory cannot be used in ordinary programming style. Application model must be different from that based on control theory..

(5) Difference in the management of program complex; In case of controlling a real object like engine, a number of programs are used for controlling different parts of the object based on different sensor inputs. Their relative importance in control operation is different. In MATLAB /SIMULINK it is embodied by skipping the execution of some components in the relatively low priority programs. Its control is included in the same program. In the ordinary programming method on the other hand programs are evoked through the interrupt handling mechanism of OS in computer system. The frequency and priority of interrupts is decided based on the relative importance of program. This relative importance is adjusted independent of programming.

(6) Difference of the roles of simulators; Since MATLAB /SIMULINK is itself a simulator-based system, the necessity for the other simulator is less than the ordinary programming style. For the latter, on the other hand, simulator is mandatory because it has no solid theory for making application model in the background.

Because of these differences, adopting MATLAB /SIMULINK approach is not easy at present. It may require enterprises to reconstruct a program development system including human organization. Taking these points into consideration, this paper discusses a new method of making application model and its translation to program for traditional programming style.

4. A New Modeling Method – Model Representation of Program in Application Domain

In Figure 1 every problem is accepted and represented as a model in the front-end sub-system. This model may be incomplete including some unclear part for user. Then it represents a problem and an activity to make the unclear part clear is problem solving. Depending on the location of the unclear part the required method of problem solving becomes different. It defines different problem type. Problem type is a class of problems to which the same method for solving the problems is used. If some functionality of an object is unknown to a user in this model and the one wish to know it, an analytic problem solving is defined. On the other hand, if an object structure is not yet obtained to a given functional requirement, then a design type problem occurs. Thus depending on problem, the part of model a user pays attention is different. An application model to be used for programming

is one of them. Before to discuss it, a general framework of modeling to meet these condition is discussed.
In order to assure generality, that is, to adapt to different problems, the model must be provided with a large expressive power. Figure 3 shows a modeling scheme that we adopted in our research project. This model includes not only objects being considered in problem solving but also subjects who concern the problem solving. A subject is a one who acts on an object. Many subjects and objects that the subjects have interests in are included in the same model. A triple (subject-activity-object) is a basic unit in the model. At the same time subjects, activities and objects are organized to form their own structures respectively. It is called a multi-strata modeling scheme and a model built in this scheme is a multi-strata model. Figure 3(a) shows a framework that represents a problem solving in the real world and every subject is shown like a human subject. Figure 3(b) is its model in the information system. The subjects are either the agents of human problem solver or, when human has no concern with its activity but the activity is left to the computer, some subject can be non-human subject but an autonomous agent by computer itself. In the latter case the activity must be well defined so that computer can do processing.
In the similar way as analysis type and design type have been defined, a programming type is defined. Solution of this type problem is a structure of activities. In this case every activity to form the structure must be well defined to the detail so that its subject can be a computer. The structure-of-activities represents a high-level application model for programming. It is transformed into a program. Thus, programming is composed of two stages. In the first stage a structure-of-activities is formed as a solution of problem in an application domain and in the second stage it is transformed into a program. The first stage is very important because automatic transformation in the latter stage is possible if and only if the structure-of-activities satisfies a certain condition. What makes programming difficult today is in the fact that application model is not necessarily made convenient for programming. Or, programmer cannot say anything about making application model.

5. A proposal for Developing New Embedded Control Program

5.1 Design of structure-of-activities

Programming starts from making a model as a structure-of-activities in application domain. It is a set of connections between individual activities. A connection is to connect an output port of an activity to an input port on another activity. In order for the structure-of-activities being computable, every individual activity must be computable and a pair of activities must be connected in a computable way. At the same time as a solution of design problem in the application domain it must satisfy the given functional requirement. The functional requirement is to assure the required performance of object system. In general in order to assure it a method to simulate the total operation (HILS-Hardware In the Loop Simulation) is necessary [2]. In the sequel a method of building a model to meet the given condition is discussed by assuming that both a set of computable activities and such a simulator are given. This is not a special assumption for the research but is the common environment of embedded programming at the real field today. As mentioned above, this is a design problem in which components for making an object are the activities. Let it be called an activity design problem.
Even if every activity is computable a structure-of-activities is not always computable. For example, let an activity A has an output with a range YA and it is to be connected to an input port of another activity B with a domain XB. If the relation $YA \subset XB$ does not hold, then the connection is not allowed. This kind of range-domain condition must be checked for computability of the structure. Other than this certain logical conditions on connection exist for making the structure computable. In all possible structures that can be produced from the given set of activities a set of computable structures are rather small. Sometimes to find the computable structures is not easy. This is one of the reasons that make programming difficult because, once a computable structure-of-activities is made, its transformation to a program is not difficult.
In general, in case of object design like an aircraft design, the form of object is strictly restricted and past design experience is used effectively for making the starting model. In this case a single candidate model is made and the design proceeds as far as possible with this model. Let it be called a depth-first design. In contrast in software design there can be many different structure-of-activities as candidates. Accordingly it is necessary to generate as many computable structure-of-activities as possible and select one among them by performance evaluation by means of simulator. It is a width-first design.

5.2 A method of designing structure-of-activities – creation of application model

Design of structure-of-activities starts by selecting a set of component activities from an activity base (function base). Activities are classified into six classes for the convenience of making topological structure as follows.
(1) single-input and single-output activity (SS-type)
(2) single-input and AND multiple-output activity (SM/A-type)
(3) single-input and OR multiple-output activity (SM/O-type)

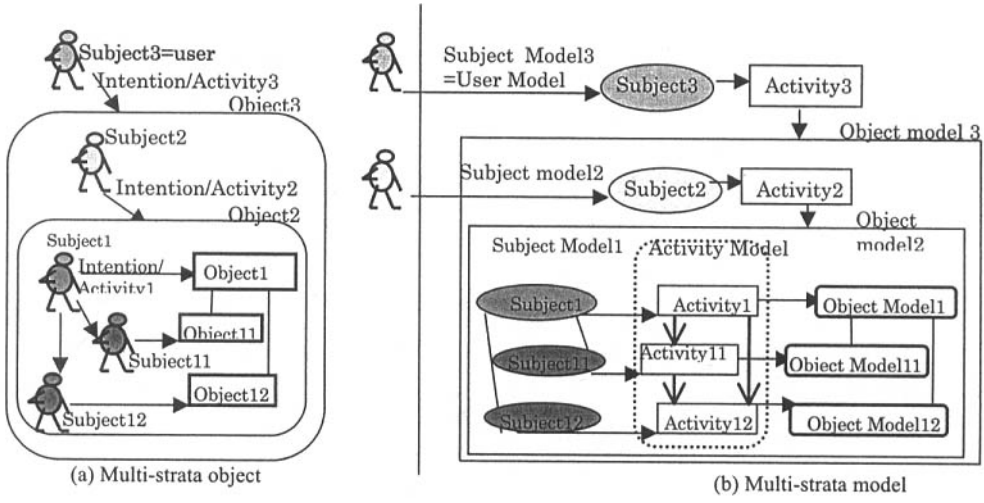


Figure 3 Multi-strata object and its model

(4) multiple-input and single-output activity (MS-type)
(5) multiple-input and AND multiple-output activity (MM/A-type)
(6) multiple-input and OR multiple-output activity (MM/O-type)

Each activity has its own characteristic that concerns meaning, e.g. its behavior, in the application domain. Every variable has also its own characteristic concerning meaning. For example, every variable is given a dimension like Mass, Length, Time etc. These characteristics are recorded in the activity (function) base with the other related information. Activities that are considered related for satisfying the given functional requirement is selected from this activity base. What is the optimum is not known in advance and the selection depends largely on the knowledge and experience of designer.

Designer is allowed to make any connection locally between every possible output-input pair that he/she thinks effective to make an optimal structure-of-activities. It is for including as many structures as possible in a consideration. This connection is called a connected pair. It results in a graph in which a number of legitimate structure-of-activities is embedded. The system extract every legitimate structure-of-activities from the graph. In the final structure-of-activity an output port is connected to one input port of another activity. In order to seek a structure to meet the condition of range-to-domain relation however a number of alternate connected pairs can be made from a single output port to multiple input ports. Afterward it is replaced by an extra XM/O-type (X is either S or M) activity in order to keep the one-to-one relation from output to input.

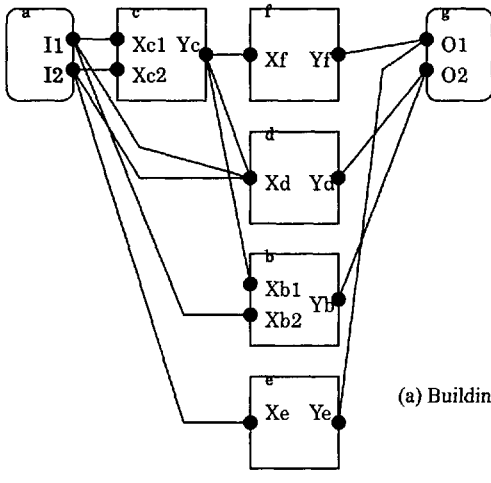
In reality an activity can be a macro activity that is itself a structure-of-activities composed of smaller activities. Thus a structure-of-activities forms a hierarchical structure. Every layer in this hierarchy is a structure-of-activities. An upper node in this hierarchy is replaced by a finer structure in the lower layer in translating into program.

After the graph is made, the structure-of-activities design proceeds as follows.

(1) The system checks the graph for finding loop. If there is a loop, then it is replaced by a macro activity and its inner structure is put in the layer below the macro.
(2) For every connected pair of activities the system checks the semantic consistency between variables. If an inconsistent connection is found it is deleted.
(3) For every connected pair of activities the system checks the range-domain relation. Such multiple connections from a single output port to a set of inputs are left such that the union set of the input domains meets the range-domain condition. In this case an extra XM/O-type (X is either S or M) activity is inserted. If in any way the range-domain condition is not met, this connection is deleted.
(4) The system checks the logical consistency for every connection. For example, OR multiple outputs of an activity cannot be connected to any multiple inputs of an activity because every multiple input must be AND input. If a connected pair of activities is logically inconsistent, then the connection(s) is deleted.

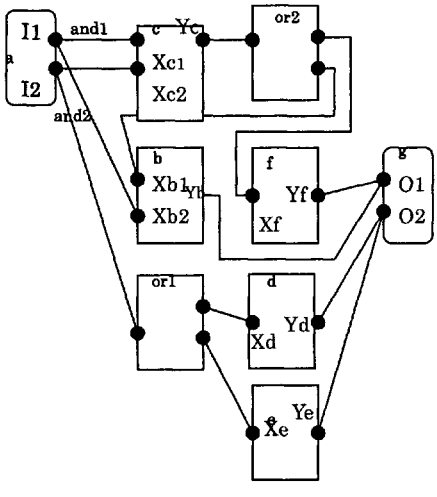
Structure-of-activities is extracted from the graph composed of the remained connections.

The structure-of-activities thus generated is computable. Among many candidates included originally in a graph,



I1 = [0,5],
I2 = [0, 20],
Xb1 = [6,10], Yb = [0,0]
Xb2 = [0,5],
Xc1 = [0,10], Yc = [0,10]
Xc2 = [0,20],
Xd = [0,10], Yd = [0,0]
Xe = [11,20], Ye = [0,0]
Xf = [0,5], Yf = [0,0]
O1 = [0,0],
O2 = [0,0]

(a) Building a model as a set of connected pairs



(b) One of the structure-of-activities obtained from the set of the connected pairs and a generated program

```
Void sol3(int a_a1, int a_a2, int *g_xg1, int
*g_xg2){
        int b_yb;
        int c_yc;
        int d_yd;
        int e_ye
        int f_yf;

        c(a_a1, a_a2, &c_yc);
        if(0<=c_yc && c_yc<=5){
                f(c_yc, &f_yf);
                *g_xg1=f_yf;
        } else{
                b(c_yc, a_a1, &b_yb);
                *g_xg1=b_yb;
        }
        if(0<= a_a2 && a_a2<=10){
                d(a_a2, &d_yd);
                *g_xg2=d_yd;
        } else{
                e(a_a2, &e_ye);
                *g_xg2=e_ye;
        }
       return;
}
```

Figure 4 An example of program generation

the real computable structure-of-activities are considerably reduced.

5.3 Logical functionality evaluation and physical performance evaluation

Programs must be made to fit the real execution environment, i.e. it must work under the control of a real-time OS and meet the condition for assuring the total system performance.

In many cases not a single program but a number of independent programs form a control system. Each program uses a specific set of inputs and has its own functionality. To each of these programs a structure-of-activities is made. In order for meeting the physical performance requirement, it is necessary to analyze a queuing problem formed by the interrupts by the events. The execution time of each event is that of corresponding program and is estimated from its structure-of-activities. Through this analysis an effective execution time is estimated for every program. In order to meet the requirement for the total system performance, the frequency of executions and the priority of interrupt are determined based on this effective execution time estimation. Though these evaluations are achieved based on the structure-of-activities before transformed into program, a total system simulation is performed using HIL after every structure-of activities is transformed into a program.

5.4 Transformation into program

Every structure-of-activities thus generated is transformed into a source program. A part of the structure including parallel connections between activities is reformed into a serial operation. Since the structure is assured to be computable and is made taking this transformation in mind, this transformation is not difficult.

6. A Simple Example of Generating Connected Pair and Program

A very simple example of generating program is shown in Figure 4. It does not include any loop but an experiment for the more general cases are being prepared. In this example a structure–of-activities is made. Then after various checking as shown in **5.2**, some structure-of-activities have been remained as computable ones. An example is shown in Figure 4(b). It is translated into a program.

7. Conclusion

Programming starts from a model at high abstract level and a program code is generated that is functionally the same as the model. Difficulty of making program depends largely on the formation of the high level model. But In many cases it is given from outside and there is no way to manage its formation. In this paper the author discussed this problem as a design problem. It is a part of the research on a very general and autonomous information system. In this system a high-level model is formed as a structure-of-activities in the application domain meeting the condition of computability. Then it is not difficult to translate it into program. That is, most of the difficulty of making program is in the formation of high-level model. A simple experiment has been made. Now its generalization is proceeding.

Acknowledgement

This research has been and is being made under the support of Ministry of Education, Culture, Sports, Science and Technology. Without this support it was difficult to continue the research. The authors wish to express sincere thanks to this support.

Reference

[1] E.J.Chikofsky and J.H. CrossII; Reverse Enjineering and Design Recovery: A Taxonomy, IEEE Software, Vol.7, No.1, 1990

[2] H. Hanselmann; Hardware-in-the-Loop Simulation Testing and its Integration into a CACSD Toolset, IEEE Int. Symp. Computer-Aided- Control System Design, 1996

[3] Kevin L. Mills and Hassen Gomaa; Knowledge-Based Automation of a Design Method for Concurrent Systems, IEEE Trans. On Software Engineering, Vol.28, No.3, 2002

[4] G.C. Murphy, D. Notkin and K.J. Sullivan; Software Reflexion Models: Bridging the Gap between Design and Implementation, IEEE Trans. on Software Engineering, Vol. 27, No.4, 2001

[5] S.Ohsuga &, S Chigusa ; Model Based Program Specification and Program Generation - In a Case of Vehicle-Engine Control System Design, International Workshop of Engineering Application of Artificial Intelligence, 2000

[6] S. Ohsuga ; How Can AI Systems Deal with Large and Complex Problems? International Journal of Pattern Recognition and Artificial Intelligence, Vol.15, No.3, 2001

[7] The MathWorks: http//www.mathworks.com

Information Modelling and Knowledge Bases XIV
H. Jaakkola et al. (Eds.)
IOS Press, 2003

Live Document Framework for Re-editing and Redistributing Contents in WWW

Yuzuru Tanaka, Daisuke Kurosaki, and Kimihito Ito
Meme Media Laboratory, Hokkaido University
Sapporo 060-8628, Japan
{tanaka, kurosaki, itok}@meme.hokudai.ac.jp

Abstract. This paper proposes a live document framework for re-editing and redistributing contents in web pages published in WWW. These contents may include live contents and web services. Our framework allows us to re-edit and to redistribute live contents and web services as well as static multimedia contents. Users need not edit HTML definitions of web documents. They can easily extract any document components through drag-and-drop operations, and paste them together on the screen to define both the layout and the functional linkages among them. Our framework is based on the 2D meme media architecture IntelligentPad. Live components extracted from web documents will become re-editable and redistributable objects when the framework wraps them with pad wrappers. You may send re-edited live documents across the Internet by attaching them to a mail. Recipients with IntelligentPad installed in their platforms can reuse, re-edit, and redistribute these live documents.

1. Introduction

Current WWW technologies provide a worldwide publication repository for people to publish their multimedia documents in HTML, to navigate through those published by other people, and to browse any of them. You may embed any services in your HTML document to publish. To define such services, you may set up servers such as database servers, file servers, and application servers. You may also define some portion of your HTML document to show the current output value of the corresponding server at the time it is accessed. Such an HTML document may change the contents of the specified portion whenever it is refreshed, or accessed again. Examples of such dynamic contents include stock prices in stock market information pages, and the space station location published in the space station home page. Such a document with an automatic periodic refreshing capability is called a live document.

Current web technologies, however, do not fully allow you to arbitrarily re-edit and redistribute published documents with embedded services. You may select any textual portion of a web page by mouse operation to make its copy, and paste this copy in your local document, for example, in the MS Word format. However, you cannot arbitrarily extract arbitrary portions of web pages and combine them together to compose a new document. If a portion to extract has dynamic contents, we would like to keep its copy alive, namely, periodically updating its contents. Such a copy is called a live copy, whereas a copy whose value is frozen at the time it is created is called a dead copy.

The re-editing of web documents requires the following capabilities:

(1) Easy extraction of an arbitrary web-document portion together with its style.
(2) To keep dynamic contents alive after arbitrary re-editing.
(3) Easy re-editing of web documents together with embedded web services by combining extracted document portions together to define both a new layout and a new functional composition.

In addition to these, the redistribution of re-edited web documents requires the following capabilities:

(4) Easy redistribution of re-edited documents across the Internet.

The publication of re-edited documents as a web document further requires the following capabilities:

(5) Easy conversion of re-edited documents to HTML format

(6) Easy registration of HTML documents to an HTTP server.

In this paper, we will propose the use of meme media technologies to achieve the first four of these capabilities. A meme media system denotes a software system of visual objects with the following capabilities:.

(1) The wrapping of an arbitrary object with a standard visual wrapper to define a media object having either a 2D or 3D representation on a display screen: A wrapped object, i.e., a meme-media content, may be a multimedia document, an application program, or any combination of them.

(2) The re-editing of meme-media objects: You can directly combine a meme-media object with another meme-media object on the display screen by mouse operations, and define a functional linkage among them. You may take out any component meme-media object from a composite meme-media object.

(3) The re-distribution of meme-media objects: Meme-media objects are persistent objects that you can send and receive for their reuse across the Internet.

IntelligentPad and IntelligentBox are respectively 2D and 3D meme-media systems. Their meme-media objects are respectively called pads and boxes.

This paper uses IntelligentPad technologies to achieve the first four of the above mentioned six capabilities. Now our goal can be paraphrased as follows:

(1) How to extract any portion of a web document, and to wrap it with a pad wrapper.

(2) How to incorporate periodic server-access capabilities in the wrapping of a live web-document portion.

Once we have solved the first problem, IntelligentPad will give solutions both to the easy re-editing of web services together with their functional linkages, and to the easy redistribution of re-edited documents across the Internet. The publication of re-edited document as a web document further needs to solve the following problem, which will require further research, and will not be discussed in this paper:

(3) How to convert a composite pad to an HTML document, and to register this to an HTTP server.

2. Related Research

Some user-customizable portal sites such as MyYahoo [1] provide another way to personalize web pages. If you have *a priori* registered your interests, the system will customize the web page only to show what you are interested in. Such a system allows you to customize only the limited portions of web documents in a restricted way. Furthermore, such a web service allows you to access only those documents it manages.

HTML4.01 [2] provides a special HTML tag <iframe>, or inline frame, for us to embed an arbitrary web document in a target web page. However, it does not allow us to directly specify either a web document portion to extract or a location in the target document to insert the extracted document. We need to edit HTML definitions.

Turquoise [3] and Internet Scrapbook [4] adopt programming-by-demonstration technologies to support the re-editing of web documents. You may demonstrate, on the screen, how to change the layout of a web page to define a customized one, and apply the same editing rule whenever the web page is accessed for refreshing. They enable us to change layouts, but not to extract any components, nor to functionally connect them together. Transpublishing [5] allows us to embed web documents in a web page. It also proposes license management and charge accounting technologies. The embedding uses a special HTML tag to embed a document.

Example tools for extracting a document component from a web document include W 4 F [6, 7] and DEByE [8]. W4F provides a GUI support tool to define an extraction. Users, however, still need to write some script programs. DEByE provides more powerful GUI support tool. However, it outputs the extracted document components in XML format. Its reuse requires some knowledge on XML.

3. Meme Media and the Outline of the IntelligentPad Architecture

Here in this section, we will give a brief introduction of meme media [13,14] and IntelligentPad as a prerequisite for the following sections. We have been conducting the research and development of 'meme media' and 'meme market' architectures since 1987. We developed 2D and 3D meme media architectures 'IntelligentPad' [9~11] and 'IntelligentBox' [12] respectively in 1989 and in 1995, and have been working on their meme-pool and meme-market architectures, as well as on their applications and revisions. 'IntelligentPad' represents each component as a pad, a sheet of paper on the screen. A pad can be pasted on another pad to define both a physical containment relationship and a functional linkage between them (Figure 1). When a pad P_2 is pasted on another pad P_1, the pad P_2 becomes a child of P_1, and P_1 becomes the parent of P_2. No pad may have more than one parent pad. Pads can be pasted together to define various multimedia documents and application tools. Unless otherwise specified, composite pads are always decomposable and re-editable.

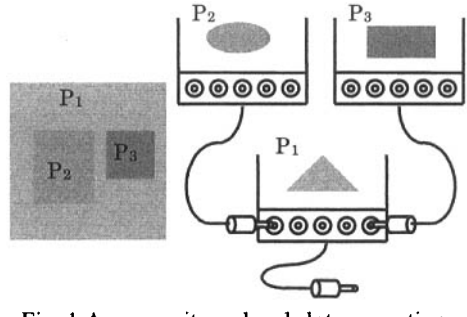
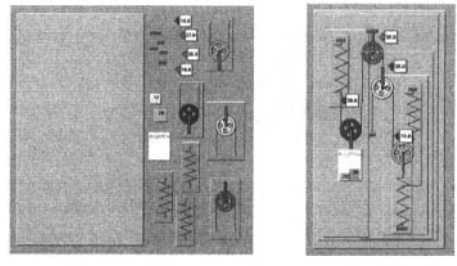


Fig. 1 A composite pad and slot connections.

In object-oriented component architectures, all types of knowledge fragments are defined as objects. IntelligentPad exploits both an object-oriented component architecture and a wrapper architecture. Instead of directly dealing with component objects, IntelligentPad wraps each object with a standard pad wrapper and treats it as a pad (Figure 1). Each pad has both a standard user interface and a standard connection interface. The user interface of a pad has a card like view on the screen and a standard set of operations like 'move', 'resize', 'copy', 'paste', and 'peel'. Users can easily replicate any pad, paste a pad onto another, and peel a pad off a composite pad. Pads are decomposable persistent objects. You can easily decompose any composite pad by simply peeling off the primitive or composite pad from its parent pad. As its connection interface, each pad provides a list of slots that work as connection jacks of an AV-system component, and a single connection to a slot of its parent pad (Figure 1). Each pad uses a standard set of messsages—'set' and 'gimme'—to access a single slot of its parent pad, and another standard message 'update' to propagate its state change to its child pads. In their default definitions, a 'set' message sends its parameter value to its recipient slot, while a 'gimme' message requests a value from its recipient slot. Figure 2 shows a set of pulleys and springs that are all represented as pads, and a composition with them. Each of these pulleys and springs is animated by a transparent pad.



(a) primitive pads (b) a composite pad

Fig. 2 An example pad composition.

4. Extraction of an Arbitrary Web-Document Portion

4.1 HTML views and their editing

Web documents are defined in HTML format. An HTML view denotes an arbitrary HTML document portion represented in the HTML document format. The pad wrapper to wrap an arbitrary portion of a web document needs to be capable of specifying an arbitrary HTML view and rendering any HTML document. We call this pad wrapper an HTMLviewPad. Its rendering function can be implemented by wrapping a legacy web browser such as Netscape or Internet Explorer. In our implementation, we wrapped Internet Explorer. The specification of an arbitrary HTML view over a given HTML document requires the capability of editing the internal representation of HTML documents, namely, DOM trees. The DOM tree representation allows you to identify any HTML-document portion, that corresponds to a DOM tree node, with its path expression. Figure 3 shows an HTML document with its DOM tree representation. The highlighted portion in the document corresponds to the highlighted node whose path expression is /HTML[0]/BODY[0]/TABLE[0]/TR[1]/TD[1]. A path expression is a concatenation of node identifiers along a path from the root to the specified node. Each node identifier consists of a node name, i.e., the tag given to this node element, and the number of its sibling nodes located to the left of this node.

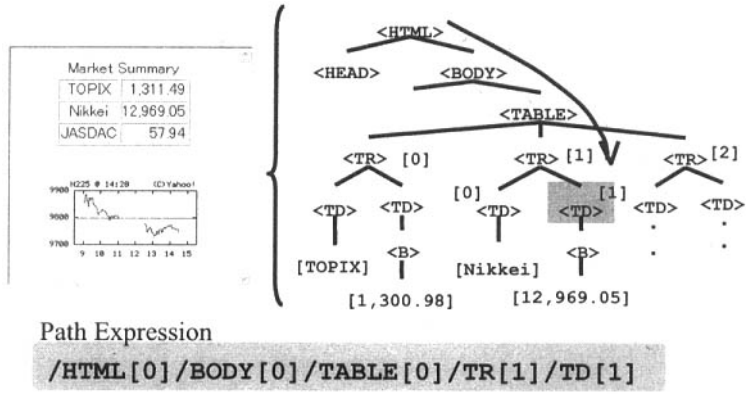


Fig. 3 An HTML document with its DOM tree, and a path expression.

Sometimes, you may need to specify, among sibling nodes, a node with a specific character string as a substring of its textual contents. You may specify such a node as

tag-name[*MatchingPattern* : *index*], where *MatchingPattern* is the specified string, and *index* selects one node among those siblings satisfying the condition.

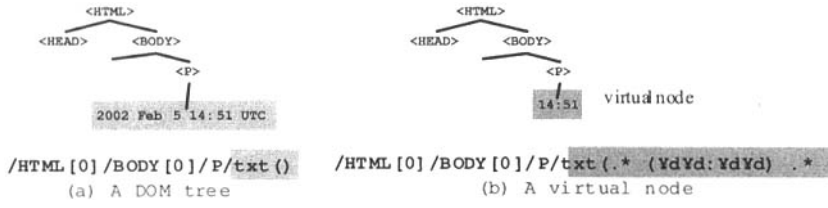


Fig. 4 A DOM tree and the path expression of a virtual node.

You may need to extract some character string in a text node. Its path expression locates this node, but does not locate such a substring. We will extend the path expression to use a regular expression for locating such a substring in a text node. For the DOM tree in Figure 4 (a), the node /HTML[0]/BODY[0]/P/txt(.* (\d\d:\d\d) .*) specifies the virtual node shown in Figure 4 (b).

The definition of an HTML view consists of the specification of the source document, and a sequence of view editing operations. The specification of a source document uses its URL. Its retrieval is performed by the function 'getHTML' in such a way as

doc = getHTML("http://www.abc.com/index.html", null).

The second parameter will be used to specify a request to the Web server at the retrieval time. Such requests include POST and GET. The retrieved document is kept in DOM format. The editing of an HTML view is a sequence of DOM tree manipulation operations selected out of the followings:

(1) EXTRACT : Delete all the nodes other than the subtree with the specified node as its root (Figure 5 (a)).

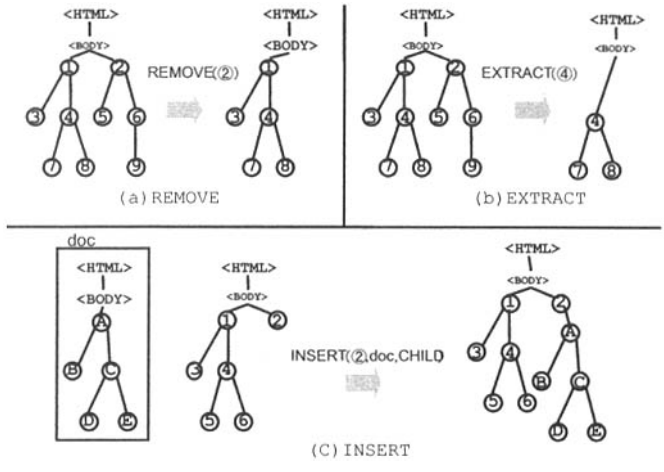


Fig. 5 REMOVE, EXTRACT, and INSERT operations on DOM trees.

(2) REMOVE : Delete the subtree with the specified node as its root (Figure 5 (b)).
(3) INSERT : Insert a given DOM tree at the specified relative location of the specified node (Figure 5 (c)). You may select the relative location out of CHILD, PARENT, BEFORE, and AFTER (Figure 6).

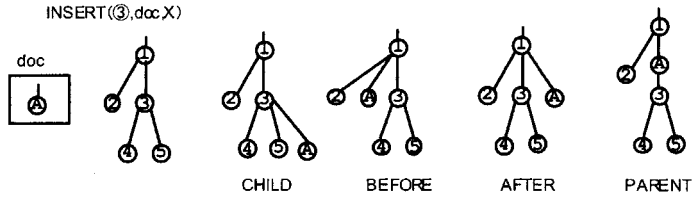


Fig. 6 Different insertion locations.

An HTML view is specified as follows:

defined-view = *source-view.DOM-tree-operation*(*node*),

where *source-view* may be a web document or another HTML document, and *node* is specified by its extended path expression. The following is an example view definition with the nested use of the above syntax.

```
    view1
=   doc
      .EXTRACT("/HTML/BODY/TABLE[0]/")
      .EXTRACT("/TABLE[0]/TR[0]/")
      .REMOVE("/TR[0]/TD[1]/");
```

You may also specify two sub trees extracted either from the same web document or from the different web documents, and combine them to define a view.

```
doc = getHTML("http://www.abc.com/index.html", null);
view2 = doc
        .EXTRACT("/HTML/BODY/TABLE[0]/")
        .EXTRACT("/TABLE[0]/TR[0]/");
view1 = doc
        .EXTRACT("/HTML/BODY/TABLE[0]/")
        .INSERT("/TABLE[0]/TR[0]/", view2, BEFORE);
```

You may create a new HTML document and insert it to an HTML document.

```
doc1 = getHTML("http://www.abc.com/index.html", null);
doc2 = createHTML("<TR>Hello World</TR>");
view1 = doc1
        .EXTRACT("/HTML/BODY/TABLE[0]/")
        .INSERT("/TABLE[0]/TR[0]/", doc2, BEFORE);
```

4.2 Direct editing of HTML views

Instead of specifying a path expression to identify a DOM tree node, we will make the HTMLviewPad frame different extractable document portions for different mouse locations so that its user may change the mouse location to see every extractable document portion (Figure 7). This method, however, cannot distinguish different HTML objects with the same display area. To identify such an object, we use an additional consol panel with

two buttons and the node specification box. The node specification box changes its value while you move the mouse to select different document portions. The first button is used to move to the parent node in the corresponding DOM tree, while the second to move to the first child node.

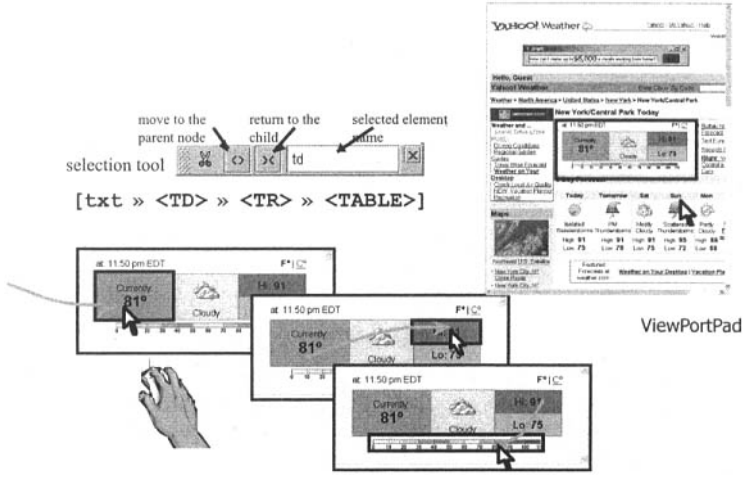


Fig. 7 The mouse movement changes the extractable element marked with a frame.

When the HTMLviewPad frames what you want to extract, you can drag the mouse to create another HTMLviewPad with this extracted document portion. The new HTMLviewPad renders the extracted DOM tree on itself. Figure 8 shows an example extraction using such a mouse-drag operation, which internally generates the following edit code.

```
doc = getHTML("http://www.abc.com/index.html", null);
view = doc
        .EXTRACT("/HTML/BODY/…/TABLE[0]/");
```

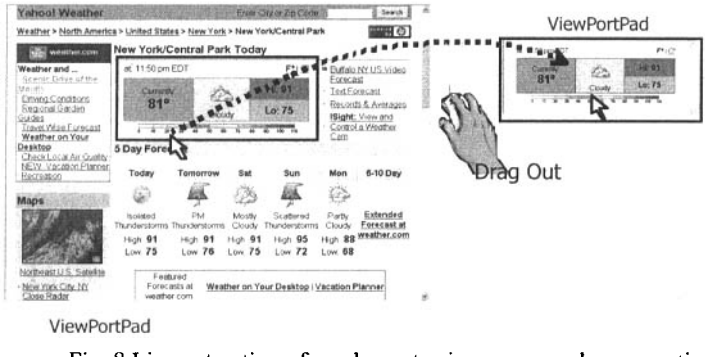


Fig. 8 Live extraction of an element using a mouse-drag operation

The HTMLviewPad provides a pop-up menu of view-edit operations including EXTRACT, REMOVE and INSERT. After you select an arbitrary portion, you may select either EXTRACT or REMOVE. Figure 9 shows an example remove operation, which generates the following code.

```
doc = getHTML("http://www.abc.com/index.html", null);
view = doc
        .EXTRACT(("/HTML/BODY/TABLE[0]/")
        .REMOVE("/TABLE[0]/TR[1]/");
```

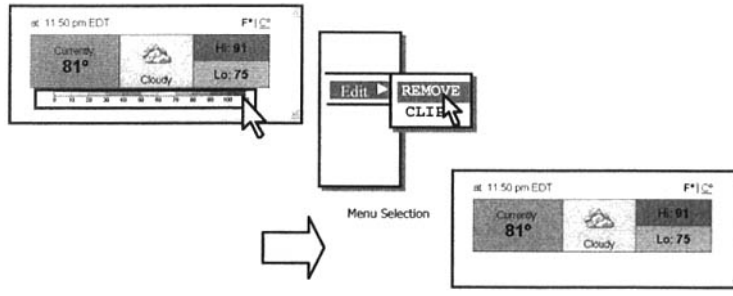


Fig. 9 Direct manipulation for removing an element from a view.

The INSERT operation uses two HTMLviewPads showing a source HTML document and a target one. You may first specify INSERT operation from the menu, and specify the insertion location on the target document by directly specifying a document portion and then specifying relative location from a menu including CHILD, PARENT, BEFORE, and AFTER. Then, you may directly select a document portion on the source document, and drag and drop this portion on the target document. Figure 10 shows an example insert operation, which generates the following code, where the target HTMLviewPad uses a different name space to merge the edit code of the dragged-out HTMLviewPad to its own edit code:

```
A::view = A::doc
        .EXTRACT("/HTML/BODY/.../TD[1]/.../TABLE[0]")
        .REMOVE("/TABLE[0]/TR[1]/");

view = doc
        .EXTRACT("/HTML/BODY/.../TD[0]/.../TABLE[0]/")
        .REMOVE("/TABLE[0]/TR[1]/")
        .INSERT("/TABLE[0]", A::view, AFTER);
```

The dropped HTMLviewPad is deleted after the insertion.

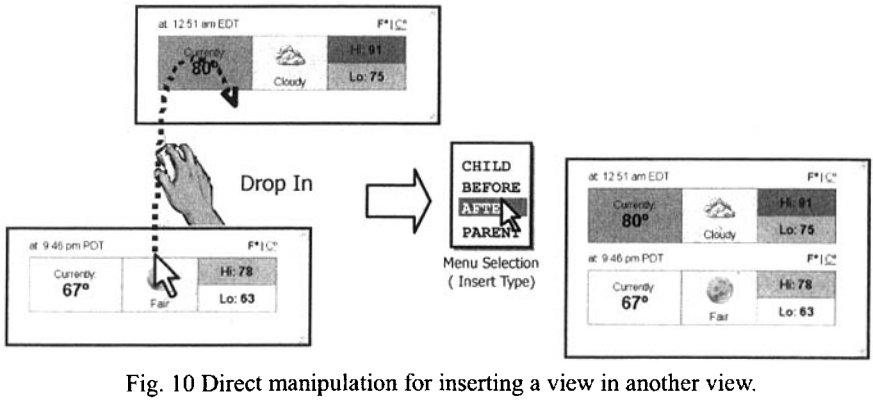


Fig. 10 Direct manipulation for inserting a view in another view.

4.3 Data mapping to define slots

The HTMLviewPad allows you to map any node values of its view and any events on its view to its newly defined slots. The definition of such a mapping takes the following form:

MAP(<*node*>, *NameSpace*)

An example of such a mapping is as follows:

MAP("/HTML/BODY/P/txt( )", "#value")

Depending on the node type, the HTMLviewPad changes the node value evaluation to map the most appropriate value of a selected node to a newly defined slot. These evaluation rules are called node-mapping rules. Each node-mapping rule has the following syntax:

target-object => *naming-rule*(*data-type*)<*MappingType*>

naming-rule	:	naming rule for the new slot
data-type	:	data type of the slot
MappingType	:	<IN| OUT| EventListener| EventFire>

Slots defined with the OUT type are read-only ones. The IN-type mapping defines a rewritable slot. The rewriting of such a slot may change the display of the HTML view document. The EventListener-type mapping defines a slot that changes its value whenever an event occurs in the node selected on the screen. The EventFire-type mapping, on the other hand, defines a slot whose update triggers a specified event in the node selected on the screen.

For a general node such as </HTML/.../txt( )>, </HTML/.../attr( )>, or </HTML/.../P/>, the HTMLviewPad defines a slot, and sets the text in the selected node to this slot. If the text is a numerical string, it converts this string to a numerical value, and sets this value to the slot (Figure 11).

a text in the selected node (character string)
=> NameSpace::#Text(string)<OUT>
a text in the selected node (numerical string)
=> NameSpace::#Text(number)<OUT>

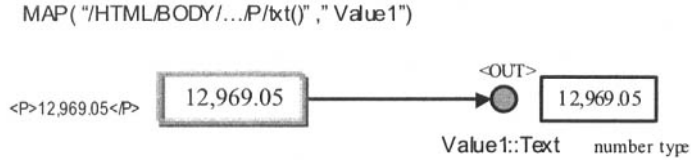


Fig. 11 The mapping of a text string node to define a slot.

For a table node such as </HTML/.../TABLE/>, the HTMLviewPad converts the table value to its CSV (Comma-Separated Value) representation, and maps it to a newly defined slot of text type (Figure 12).

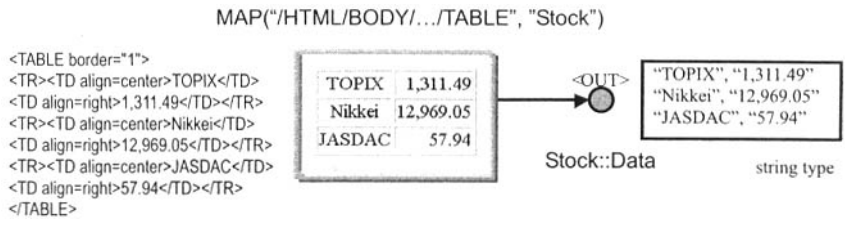


Fig. 12 The mapping of a table node to define a slot.

For an anchor node such as </HTML/.../A/>, the HTMLviewPad performs the following three mappings (Figure 13):

a text in the selected node
=> NameSpace::#Text(string, number)<OUT>
href attribute of the selected node
=> NameSpace::#refURL(string)<OUT>
URL of the target object
=> NameSpace::#jumpURL(string)<EventListener>

The third mapping has the EventListener type. Whenever the anchor is clicked, the target URL is set to a string-type slot.

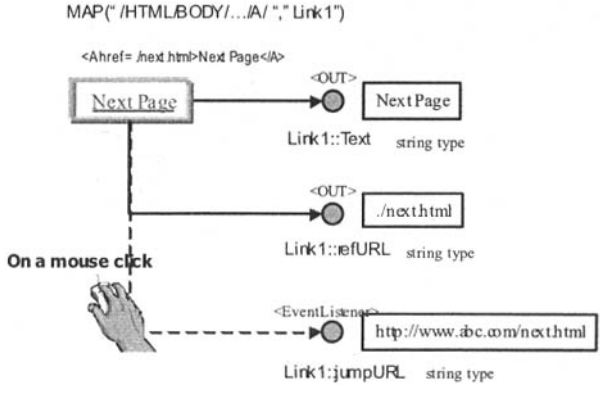


Fig. 13 The mapping of an anchor element to define three slots.

For a form node such as </HTML/.../FORM/>, the HTMLviewPad performs the following three mappings (Figure 14):

the value attribute of the INPUT node with the name attribute in the selected node
=> NameSpace::#Input_type_name(string, number)<IN, OUT>
Submit action
=> NameSpace::#FORM_Submit(boolean)<EventFire>
the value obtained from the server
=> NameSpace::#FORM_Request(string)<EventListener>

type = <text|pasword|file|checkbox|radio|hidden|submit|reset|button|image>
name = <name> attribute in the INPUT node

The third mapping has the EventListener type. Whenever an event to send a form request occurs, the HTMLviewPad sets the corresponding query to a newly defined slot. The second one is a EventFire-type mapping. Whenever a TRUE is set to the slot, the HTMLviewPad triggers a form-request event.

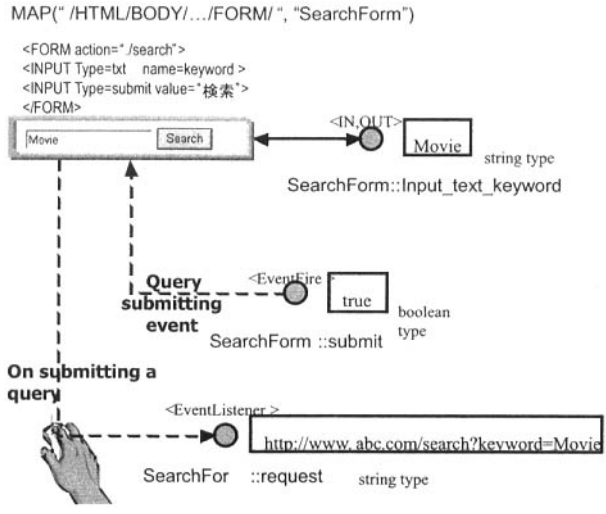


Fig. 14 The mapping of a form element to define three slots.

Each HTMLviewPad has the following four default slots. The #UpdateInterval slot specifies the time interval for the periodical polling of referenced HTTP servers. A view defined over a web document refresh its contents by periodically retrieving this web document in an HTTP server. The #RetrievalCode slot stores the code to retrieve the source document. The #ViewEditingCode slot stores the view definition code. The #MappingCode slot stores the mapping definition code. The HTMLviewPad updates itself by accessing the source document, whenever either the #RetrievalCode slot or the #ViewEditingCod slot is accessed with a set message, the interval timer invokes the polling, a user specifies its update, or it becomes active after its loading from a file. In addition to these four slots, the HTMLviewPad automatically creates slots defined by the mapping code that is set to the #MappingCode slot.

4.4 Visual definition of slots for extracted Web contents

Our HTMLviewPad also allows users to visually specify any HTML-node to work as a slot. In its node specification mode, an HTMLviewPad frames different extractable document portions of its content document for different mouse locations so that its user may change the mouse location to see every selectable document portion. This method, however, cannot distinguish. To identify one out of different HTML objects with the same display area, we use the same consol panel used to extract Web contents. When the HTMLviewPad frames what you want to work as a slot, you can click the mouse to pop up a dialog box to name this slot. Since each extracted Web component uses an HTMLviewPad to render its contents, it also allows users to specify any of its portions to work as its slot. We call such a slot thus defined an HTML-node slot. The value of an HTML-node slot is the HTML view of the selected portion. The HTMLviewPad converts ill-formed HTML into well-formed HTML to construct its DOM-tree. Therefore, you may connect an HTMLviewPad to an HTML-node slot to view the corresponding HTML view.

If the HTML-node slot holds an anchor node, the HTMLviewPad connected to this slot shows the target web page.

Figure 15 shows an HTMLviewPad showing a Yahoo Japan's Web page with an embeded Web application to convert US dollar to Japanese yen based on the current exchange rate. On this pad, you can visually specify the input form for inputting dollar amount and the output text portion showing the equivalent yen amount to work as slots. The HTML-path of the input-form is represented as

HTML[0]/BODY[0]/DIV[0]/FORM[0]/INPUT[0]/text[(.*)],

while the HTML-path of the selected output text portion is represented as

HTML[0]/BODY[0]/TABLE[0]/TR[0]/TD[1]/A[0]/attr[href].

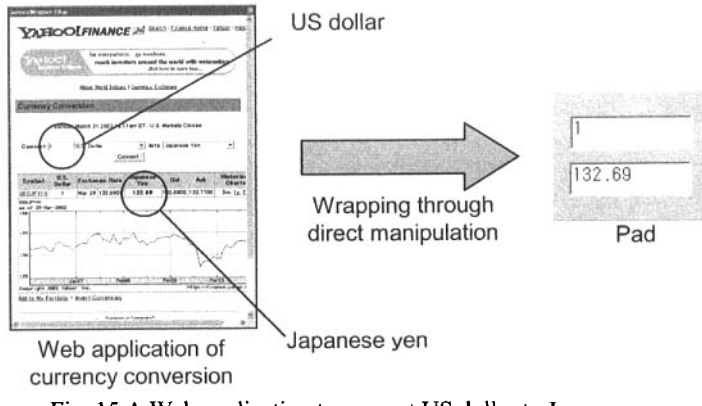


Fig. 15 A Web application to convert US dollar to Japanese yen and its wrapping with two slot definitions.

You may name the corresponding HTML-node slots as #dollarAmount and #yenAmount respectively. The HTMLviewPad allows you to suspend the rendering of its contents. In this mode, you may use an HTMLviewPad with an HTML view as a blank pad with an arbitrary size. Figure 15 shows, on its right hand side, a currency rate converter pad. We have defined this pad from the above mentioned web page just by defining two slots, resizing the HTMLviewPad, and pasting two text IO pads with their connections to #dollarAmount and #yenAmount slots.

Such a pad wraps a Web application, providing slots for the original's input forms and output text strings. We call such a pad a wrapped Web application. Since a warapped web application is a pad that allows you to change its primary slot assignment, you may specify any one of its slots to work as a primary slot.

Figure 16 shows an HTMLviewPad showing a Lycos' Web page for a realtime stock-price browsing service. We have wrapped this page defining a slot for the current stock price. Then we paste the wrapped currency conversion Web application with its #dollarAmount specified as its primary slot on this wrapped Lycos stock-price page. We connect the conversion pad to the newly defined current stock price slot. The right hand side in this figure shows a composite pad combining these two wrapped Web applications. For the input of different company names, we use the input form of the original Web page. Since this Web application uses the same page layout for different companies, the same path expression correctly identifies the current stock-price information part for every different company.

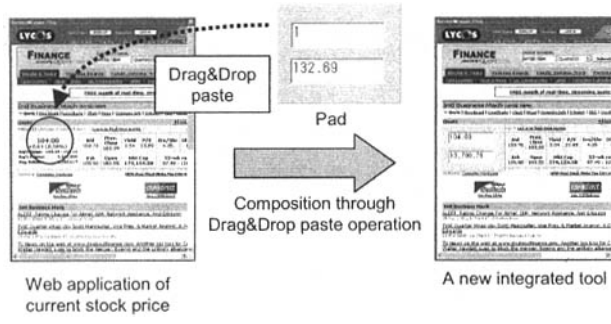


Fig. 16 The wrapping of a stock-price information service and the pasting of the wrapped curency conversion service in Figure 15 on this wrapped service.

5. Example Applications

5.1 Live copies of numerical data

The HTMLviewPad allows us to extract an arbitrary HTML element from the web document it displays. The direct dragging-out of this portion creates another HTMLviewPad showing the extracted portion. The periodic polling capability of the latter HTMLviewPad keeps the extracted document portion alive. We call such a copy of a document portion a live copy. You may paste a live copy in a pad form on another pad with a slot connection for functional composition. You may also paste a pad on a live copy in a pad form, and connect the former pad to one of the slots of the latter. Using such operations, you may compose an application pad integrated with live copies of document portions extracted from different web pages.

Figure 17 shows the plotting of the NASA Space Station's orbit and Yohkoh Satellite's orbit. We used a world map with a plotting function. This map has a pair of #longitude[1] slot and #latitude[1] slot, and creates, on user's demand, more pairs of the same type slots with different indices. First, you need to access the home pages of the space station and the satellite. These pages show the longitude and the latitude of the current locations of these space vehicles. Then, you may make live copies of the longitude and the latitude in each web page, and paste them on the world map with their connection respectively to #longitude[i] and #latitude[i] slots. The live copies from the space station web page uses the first slot pair, while those from the satellite web page uses the second slot pair. These live copies updates their values every 10 seconds by polling the source web pages. The independent two sequences of plotted locations show the orbits of the two space vehicles.

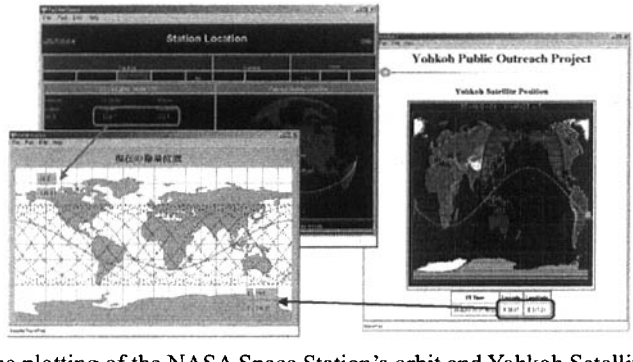


Fig. 17 The plotting of the NASA Space Station's orbit and Yohkoh Satellite's orbit.

Figure 18 shows an application to the real-time visualization of stock prices change. First, you need to access Yahoo Finance web page showing the current Nikkei average stock price in real time. Then, you may make a live copy of Nikkei average index, and paste it to a DataBufferPad with its connection to #input slot. A DataBufferPad associates each #input slot input with its input time, and outputs this pair in CSV format. We pasted this composite pad on a TablePad with its connection to #data slot. A TablePad adds every #data slot input at the end of the list stored in CSV format. You need to change the primary slot of the TablePad to #data slot to paste this pad on a GraphPad with its connection to #input slot. A GraphPad adds a new vertical bar proportional to the input value whenever it receives a new #input slot value.

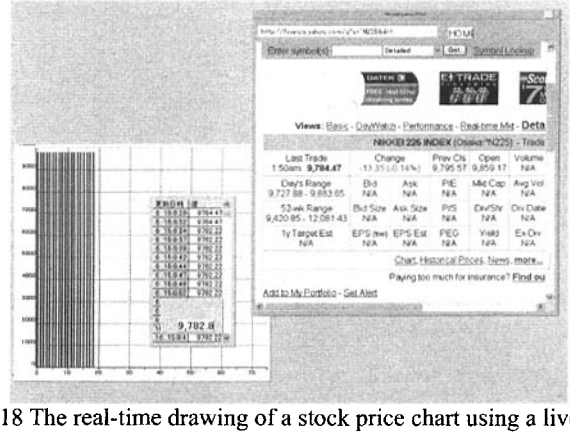


Fig. 18 The real-time drawing of a stock price chart using a live copy.

5.2 Live copies of table data

Figure 19 shows another page of Yahoo Finace service. This page shows, for a specified company, the time series of its stock prices over a specified period. You may make a live copy of this table, and paste it on a TablePad with its connection to #input slot. The contents of the extracted table is sent to the TablePad in CSV format. You may paste the same live copy on a GraphPad with its connection to #list slot, which produces a chart shown in this figure.

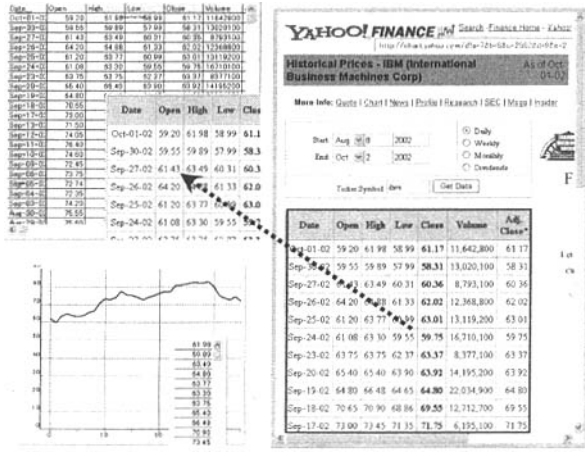


Fig. 19 The real-time drawing of a stock price chart using a live copy of a table element.

5.3 Live copies of anchors

Figure 20 shows a Yahoo Maps web page. You can obtain a map around a location you specify. You may make live copies of its map display portion, its zooming control panel, and its shift control panel, and paste the two control panels on the map display with their connections to #RetrievalCode slot of the map display. Whenever you click some button on either of these control panels, the control panel sets the URL of the requested page, and sends this URL to #RetrievalCode slot of the map display. The map display, then, accesses the requested page with a new map, and extracts the map portion to display.

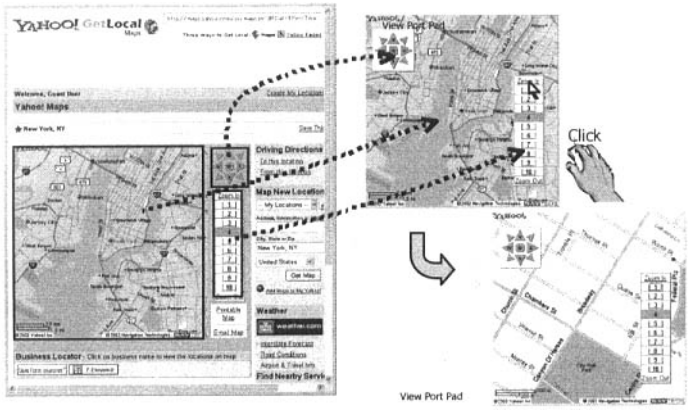


Fig. 20 Composition of a map tool using a map service and its control panels.

5.4 Redistribution of live copies

Whenever you save a live copy extracted from some web document, the system saves only the pad type, namely 'HTMLviewPad', and the values of the two slots, #RetrievalCode slot and #ViewEditingCode slot. A copy of a live copy shares only these with the original. The redistribution of a live copy across the Internet needs to send only its save format representation. When a live copy is activated on the destination platform, it invokes the retrieval code stored in #RetrievalCode slot, and executes the view editing code in #ViewEditingCode slot to display only the defined portion of the retrieved web document. You can further extract any of its portions as a live copy.

6. Concluding Remarks

This paper has proposed a live document framework for re-editing and redistributing contents in web pages. These contents include live contents and web services. Our framework allows us to re-edit and to redistribute live contents and web services as well as static multimedia contents. Users need not edit HTML definitions of web documents. Users can easily extract any document components through drag-and-drop operations, and paste them together on the screen to define both the layout and the functional linkages among them.

Our framework is based on the 2D meme media architecture IntelligentPad. Live components extracted from web documents will become re-editable and redistributable objects when we wrap them with pad wrappers.

You may send re-edited live documents across the Internet by attaching them to a mail. Recipients with IntelligentPad installed in their platforms can reuse, further re-edit, and redistribute these live documents.

When applied to the over-the-counter services in e-banking, the framework enables financial planners to dynamically collect appropriate live information from WWW and local web pages that access internal databases, to dynamically combine them together to

compose customized portfolios of live stock-market information, and send it to their clients. Financial planners can re-edit the live portfolios according to their clients' demands. Clients can also re-edit the proposed live portfolios to define summaries or focused information. They can also combine more than one live portfolio obtained from different financial planners to define a cross-comparison view.

Live document framework also opens a new vista in the circulation and reuse of scientific knowledge. In bioinformatics for example, there are already many different kinds of database services, analysis services, and related reference information services; most of them are available as web services. However, they are serviced by independent groups, and hard to interoperate with each other. Therefore, even gene annotation with information available from some public service requires either manual operations or a program code to automate them. Our framework enables researchers in this field to dynamically extract and combine public web services, such as homology search of gene databases and other related information services, and to make them interoperate with each other, using an advanced software media technology.

References

[1] MyYahoo, http://my.yahoo.co.jp/
[2] HTML 4.01 Specification, http://www.w3.org/TR/html4/.
[3] R.C. Miller, B.A. Myers, Creating Dynamic World Wide Web Pages By Demonstration. Carnegie Mellon University School of Computer Science Tech. Report, CMU-CS-97-131, 1997.
[4] A. Sugiura, Y. Koseki, Internet Scrapbook: Automating Web Browsing Tasks by Demonstration. Proc. of the ACM Symposium on User Interface Software and Technology (UIST), pp.0-18, 1998.
[5] T.H. Nelson, transpublishing for Today's web: Our Overall Design and Why It is Simple. http;//www.sfc.keio.ac.jp/ted/TPUB/Tqdesign99.html, 1999.
[6] A.Sahuguet, F. Azavant, Building Intelligent Web Applications Using Lightweight wrappers. Data and knowledge Engineering, 36 (3), pp.283-316, 2001.
[7] A. sahuguet, F. Azavant, Wysiwyg Web wrapper Factory (W4F). http://db.cis.upenn.edu/DL/www8.pdf, 1999.
[8] B. A. Ribeiro-Neto, A.H.F. Laender, A.S. Da Silva. Extracting Semistructured Data Through Examples. Proc. of the 8th ACM int'l Conf. On Informtion and knowledge Management (CIKM'99), pp.91-101, 1999.
[9] Y. Tanaka, and T. Imataki. IntelligentPad: A Hypermedia System allowing Functional Composition of Active Media Objects through Direct Manipulations. In Proc. of IFIP'89, pp.541-546, 1989.
[10] Y. Tanaka, A. Nagasaki, M. Akaishi, and T. Noguchi. Synthetic media architecture for an object-oriented open platform. In Personal Computers and Intelligent Systems, Information Processing 92, Vol III, North Holland, pp.104-110, 1992.
[11] Y. Tanaka. From augmentation media to meme media: IntelligentPad and the world-wide repository of pads. In Information Modelling and Knowledge Bases, VI (ed. H. Kangassalo et al.), IOS Press, pp.91-107, 1995.
[12] Y. Okada and Y. Tanaka. IntelligentBox:a constructive visual software development system for interactive 3D graphic applications. Proc. of the Computer Animation 1995 Conference, pp.114-125, 1995.
[13]Y. Tanaka. Meme media and a world-wide meme pool. In Proc. ACM Multimedia 96, , pp.175-186, 1996.
[14] Y. Tanaka. Memes: New Knowledge Media for Intellectual resources. Modern Simulation and Training, 1, pp.22-25, 2000.

Information Modelling and Knowledge Bases XIV
H. Jaakkola et al. (Eds.)
IOS Press, 2003

A Family of Web Diagrams Approach to the Design, Construction and Evaluation of Web applications

Takehiro Tokuda, Tetsuya Suzuki,
Kornkamol Jamroendararasame and Sadanobu Hayakawa
{tokuda, tetsuya, konkamol, hayakawa}@tt.cs.titech.ac.jp
Department of Computer Science
Tokyo Institute of Technology
Meguro, Tokyo 152-8552, Japan

Abstract. Our daily life is heavily dependent on Web applications such as reservation systems, shopping systems and banking systems. Unfortunately, because of sudden growth of the Web application technology, confusing semantics, alternative implementation methods, and overall usability of the system are not examined thoroughly yet. This paper is one of our attempts to improve this unfortunate situation. We propose a family of diagrams which may be helpful for our design, construction and evaluation processes of Web applications. These diagrams are simple but powerful enough to represent design processes for many types of implementations of Web applications.

1 Introduction

Our daily life is heavily dependent on Web applications such as reservation systems, shopping systems and banking systems. Unfortunately, because of sudden growth of the Web application technology, confusing semantics, alternative implementation methods, and overall usability of the system are not examined thoroughly yet [1, 2] . This paper is one of our attempts to improve this unfortunate situation. We propose a family of diagrams, called Web diagrams, which may be helpful for our design, construction and evaluation of Web applications.

Traditionally, for each type of programming activities, we usually have graphical diagrams to help processes for design, construction, and evaluation of the application. For procedural programming activities, we have diagrams such as flow charts, HIPO charts, and NS charts. For object-oriented programming activities, we have diagrams such as use case diagrams, class diagrams and activity diagrams. Unfortunately, for Web programming activities, except some attempts such as Web site planning diagrams [3] and UML diagrams for Web applications [4], we do not yet have diagrams to represent the overall structure and behavior of Web applications [5] . Web site planning diagrams emphasizes graphical designs and hyperlink structures. UML diagrams for Web applications describe deep level program structures, which are a little bit far away from surface level structures of Web applications, because most Web applications are not object-oriented systems at least in the surface syntax level.

Unlike procedural programming and object-oriented programming, diagram representations of Web applications have following difficulties.

1. Web applications use different implementation methods such as CGI, Servlet, JSP, ASP, JavaScript, and Applet [6, 7, 8, 9, 10, 11, 12, 13, 14] . Web applications

use programs which are running on the sever side or on the client side or on both sides.

2. Web applications use two different types of computations such as data-flow computations and side-effect computations. For example, data-flow computations write one entire Web page at a time. Side-effect computations write many results on the same part of one Web page.

3. Programs running on the sever side are usually independent each other and they even do not know whether they are forming a part of a session. In addition, the behavior of Web client browsers may not be controllable. At any moment Web client browsers may return to previously visited Web pages or abandon requested computations or restart computations after a pause. (From this viewpoint Web applications are uniquely different from ordinary standalone systems.)

Here we present a family of Web diagrams which can represent the overall structures and behaviors of Web applications based on many types of implementation methods having read-only computations and side-effect computations. Our diagrams are simple but powerful enough to represent design processes of various Web applications.

The organization of the rest of this paper is as follows. In Section 2, we explain five types of Web diagrams. In Section 3, we illustrate one design process based on Web diagrams. In Section 4, we explain one example of construction of Web applications from Web diagrams. In Section 5, we explain possible use of Web diagrams in usability evaluations. In Section 6 we give comparisons of our diagrams with other diagrams. In Section 7 we give concluding remarks.

2 A Family of Web Diagrams

We introduce five types of Web diagrams. Snapshot diagrams are for external considerations of use cases. Page diagrams are for considerations of appearance of each page. Data-flow diagrams and Hyperlink diagrams are for considerations of data-flow relations and hyperlink relations respectively. Source diagrams are for considerations of source files and codes. We refer to the composite diagram of a data-flow diagram and a hyperlink diagram as a Web transition diagram.

2.1 Snapshot Diagrams

This diagram is a directed graph which externally explains transition relations of Web pages or blocks of Web pages for one use case. The node of the graph is one Web page represented by a box, or one block of Web pages represented by a dotted line box. Each box has a verbal summary. An arrow from one box to another box shows a transition relation. Each transition has its verbal summary of events such as "select" or "activate". The topology of the graphs may be linear chains or trees or shared trees. Fig.1 shows a skeleton of one snapshot diagram.

2.2 Page Diagrams

This diagram is the graphical explanation of one Web page with its immediate relations with other Web pages and programs. The Web page to be graphically explained is represented by a big box. Web pages and programs, which are immediately related to the

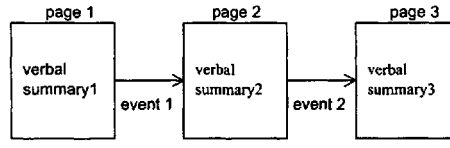


Figure 1: Snapshot diagram

Web page, are represented by small squares and small circles respectively. Undirected lines connect one Web page with pages or programs. Fig.2 shows a skeleton of one page diagram.

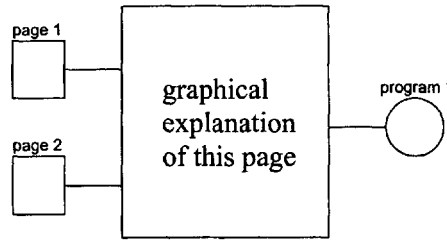


Figure 2: Page diagram

2.3 Data-flow Diagram

This diagram is internal explanation of data-flow relations of Web pages, programs and databases. Data-flow relations are represented by an arrow having a triangle with a blocking line. The source of an arrow is a form or an actuator or a program or a database. The destination of an arrow is a page or a part of a page or a program or a database. CGI programs and Servlets produce data-flow arrows going to entire Web pages. Programs used in JSP and ASP produce data-flow arrows going to parts of Web pages or entire Web pages.

If we need to distinguish programs running on the server side and programs running on the client side, we may use the following conventions. The top page will appear in the leftmost position. Programs running on the server side produce data-flow arrows from left to right. Programs running on the client side produce data-flow arrows from right to left. Fig.3 shows an example of data-flow diagrams. Program1 is a server side program and program2 is a client side program.

2.4 Hyperlink Diagram

This diagram is internal explanation of hyperlink relations. Hyperlink relations are represented by an arrow having a triangle. The source of an arrow is a link and destination of an arrow is a page. Fig.4 shows an example of hyperlink diagrams.

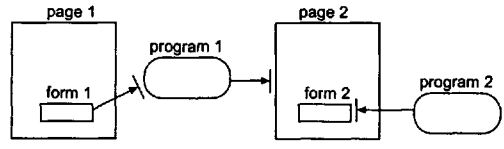


Figure 3: Data-flow diagram

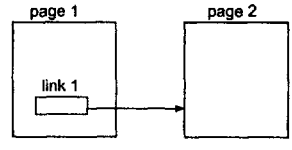


Figure 4: Hyperlink diagram

2.5 Source Diagrams

This diagram is an undirected graph which internally explains relations of source files and codes. Nodes may be Web pages, Web page templates, parts of Web pages, data files, databases and programs. One node may be inside of another node. An outermost node corresponds to one source file. Web pages may be inside of a program when the program produces these Web pages as output. Undirected lines connect forms or actuators with corresponding programs. Fig.5 shows an example of source diagrams. There are three source files "page1", "program1" and "program2".

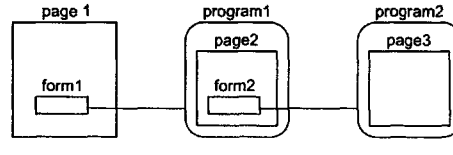


Figure 5: Source diagram

3 Design of Web Applications

We show one design process of Web applications using our family of Web diagrams in Table1. In step 1 and 2 we construct page block level snapshot diagrams and page level snapshot diagrams respectively for a number of use cases. In step 3 and 4 we construct page diagrams in page-program level and in program-database level respectively. In step 5 we construct one Web transition diagram which is a composite diagram of a data-flow diagram and a hyperlink diagram. In step 6 we construct one source diagram. And finally in step 7 we construct source files corresponding to the source diagrams.

Table 1: Our design process

Step 1. Page block level snapshot diagram We give verbal summary description of page blocks and their relations of transitions for a number of use cases.

Step 2. Page level snapshot diagram We perform division of page blocks into one or more Web pages. We give verbal summary description of Web pages and their relations of transitions for a number of use cases.

Step 3. Page-program level page diagram We give distinction of transitions by hyperlinks and transitions by programs. For each page we give graphical description of one Web page and its relations with Web pages and programs of relational distance one.

Step 4. Program-database level page diagram We give description of relations of programs with databases. Databases may be ordinary databases, client storage and server storage of relational distance one.

Step 5. Data-flow & hyperlink diagram We give distinction of programs producing entire pages and programs producing a part of pages. We give distinction of programs on the server side (which will be shown on the left-hand side of the page) and programs on the client side (which will be shown on the right-hand side of the page). We compose data-flow lines and hyperlink lines.

Step 6. Source diagram We give relations of source files and codes for Web pages, Web page templates, data files, databases, server side programs and client side programs.

Step 7. Source files We give description of Web pages, Web page templates, data files, databases, server side programs and client side programs.

As an example of this design process, we show an example of Web shopping system for books and CDs. This shopping system allows Web users to view the list of books and CDs or the details of each item, add their items and quantities into the shopping cart, remove them from the shopping cart, and eventually register as a new member or login to the system to confirm their purchase.

Fig.6 shows two parts of one snapshot diagram for one use case. Here a user visits the top page, a list of items page, one item page and purchases the item in Fig.6(1). In Fig.6(2) a user logins the member page, gives purchase information, and gives purchase confirmation. Fig.7 shows one page diagram for the top page. The top page has immediate relations with two programs "book", "cd" and one page "complete". Fig.8 shows one part of the data-flow diagram and the hyperlink diagram separately. For example, in Fig.8 a) a program "book" produces page "Books" by the request from page "Top" using contents of database "book". Fig.9 shows the source diagram. Here we have one HTML page "Top" and four programs "book", "book details", "add1" and "total". First three programs produce entire pages and the last program produces a part of a page. Fig.10, Fig.11 and Fig.12 show the part of corresponding source codes. The first two files are JSP files and the last file is a Servlet file.

4 Construction of Web Applications

Web diagrams are useful for construction of Web applications not only manually but also automatically. We may associate actual program templates with program nodes on a composite diagram of a data-flow diagram and a hyperlink diagram, and we can directly generate Web applications from Web transition diagrams. Fig.13 shows such a Web transition diagram for a Web application generator called T-Web system [15, 16, 17].

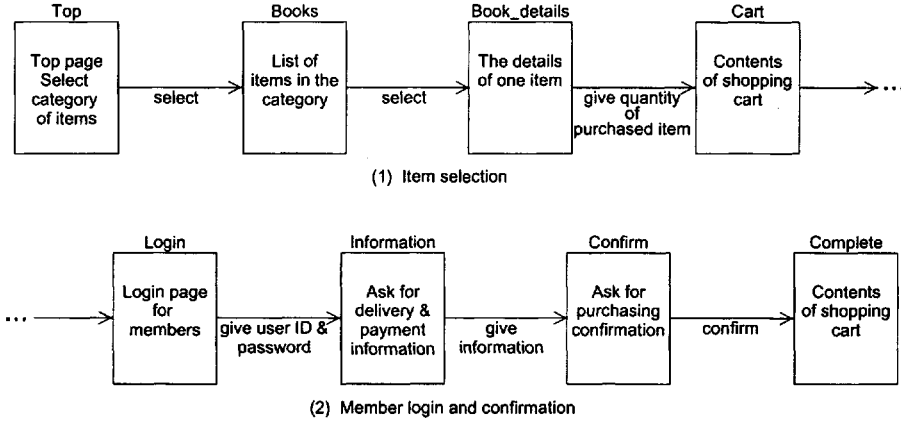


Figure 6: Snapshot diagram for a shopping system

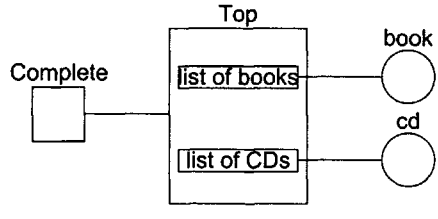


Figure 7: Page diagram for the top page

Here all programs are running on the server side. (We do not need to use conventions for distinction of programs on the server side and programs on the client side.) The page "Top" is located on the top-center. From the top page a user may check books, CDs, and the shopping cart. In Fig.13, 12 Servlet programs and 2 Bean programs are generated using 7 Servlet templates and 2 Bean templates respectively. Fig.14 shows screen shots of generated Web applications by T-Web system from the diagram of Fig.13. One example of Servlet template is shown in Fig.15.

5 Evaluation of Web Applications

Web diagrams may be also useful in evaluating poorness of usability of existing Web applications. Some of the sources of poor usability of Web applications are use of too many possible actions, use of too many links, use of two many colors, and use of very high chromaticness colors. If we measure the number of possible actions, the number of links, the number of colors, the number of high chromaticness colors, and attach measured values to Web transition diagrams, then we may graphically show the states of one aspect of usability of Web applications. Another application would be analyzing human complexity of Web pages. Fig.16 shows potential use of Web transition diagrams

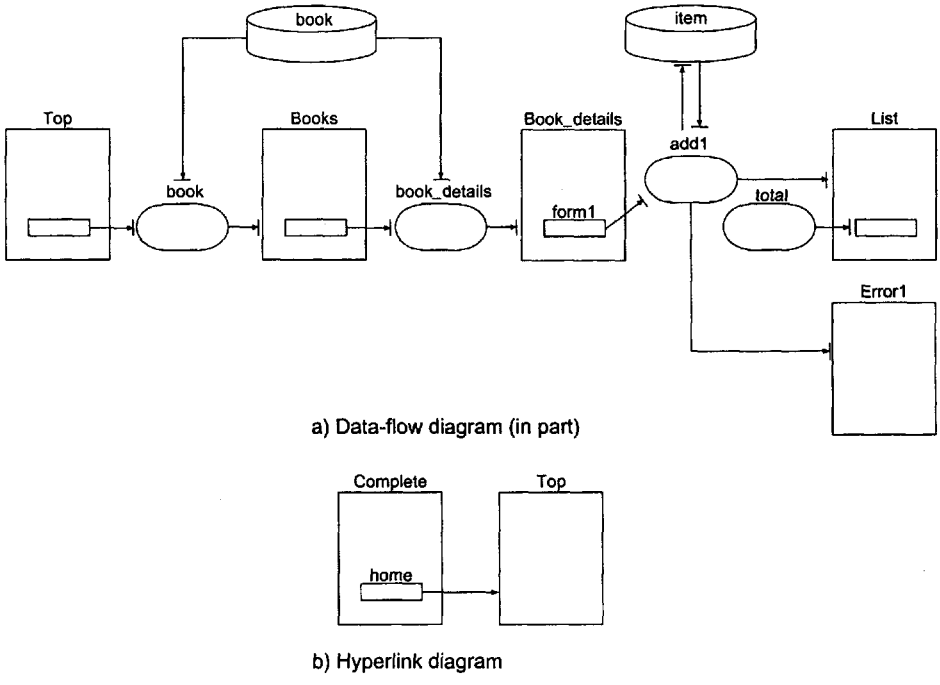


Figure 8: Data-flow & hyperlink diagram for the shopping system

in expressing human complexity of searching for Web pages. We can compute expected cost for reaching Web pages using these diagrams.

6 Comparisons

We give comparisons of our diagrams with other existing diagrams. In practice we may observe that many unnamed diagrams are widely used and that some named diagrams appear in the literature. Unnamed diagrams usually represent hyperlink structures of Web sites. Some named diagrams are Web site planning diagrams and UML diagrams for Web applications. Web site planning diagrams are 3D diagrams emphasizing graphical design of pages and hyperlink relations. UML diagrams for Web applications describe deep level program structures which are a little bit far away from surface level structures of Web applications. On the surface structure level Web applications may be non-object-oriented programs or HTML pages attached with scattered programs. Fig.17 shows one example of UML diagrams for a glossary system Web application. We may notice that the description level is very low and microscopic. Fig.18 shows our Web transition diagram for the same glossary system Web application. Our diagram can describe the overall structure and behavior more clearly. Also for free compositions of templates of automatic Web application generation, data-flow diagrams are much easier to handle than UML diagrams.

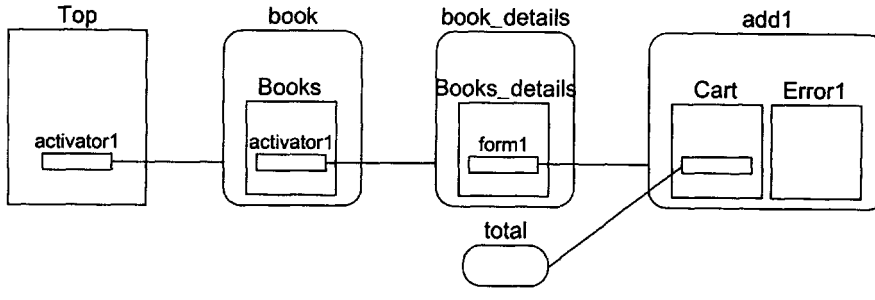


Figure 9: Source diagram for the shopping system

```
<HTML>
<TITLE>Book&CD online shopping system</TITLE>
<BODY>
<%= request.getAttribute("result") %><BR>
</BODY>
</HTML>
```

Figure 10: Source code 'Books.jsp' for the shopping system

```
<HTML>
<TITLE>Book&CD online shopping system</TITLE>
<BODY>
<A HREF="http://ttlab-vaio:8100/servlet/shopping.book">list of books</A><BR>
<A HREF="http://ttlab-vaio:8100/servlet/shopping.cd">list of CDs</A><BR>
<A HREF="http://ttlab-vaio:8100/servlet/shopping.order">your cart</A><BR>
</BODY>
</HTML>
```

Figure 11: Source code 'Top.jsp' for the shopping system

```
package shopping;
// template name:show-all-info
import java.sql.*;
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class book extends HttpServlet {
 public void doGet(HttpServletRequest request, HttpServletResponse response)
   throws ServletException, IOException {
  String dbName = "shopping";
  Connection con = TWeb.doConnect(dbName);
  String id = "";
  String name = "";
  String price = "";
  String isbn = "";
  String author = "";
  String pin = "";
  pin = request.getParameter("pin");
  String[] fNames = {"<A HREF=shopping.detail1?pin="+pin+"&id=","">",
   "</A><BR>"};
  int[] colNumbers = {0,1};
  if (TWeb.cookieAuthenticate(request,response,"","")) {
   String tbName = "book";
   String query = "SELECT * FROM "+tbName;
   request.setAttribute("result",TWeb.doSelect(con,query).toHTML("mydefine",
    "CYAN",fNames,colNumbers).toString());
   gotoPage("/shopping/Books.jsp",request,response);
  } else {
   gotoPage("/shopping/cookieError.jsp",request,response);
  }
  TWeb.doDisconnect(con);
 }
 public void doPost(HttpServletRequest request, HttpServletResponse
response)
   throws ServletException, IOException {
  doGet(request,response);
 }
 private void gotoPage(String address, HttpServletRequest request,
   HttpServletResponse response) throws ServletException, IOException {
  RequestDispatcher dispatcher = getServletContext().getRequestDispatcher(
   address);
  dispatcher.forward(request,response);
 }
}
```

Figure 12: Source code 'book.java' for the shopping system

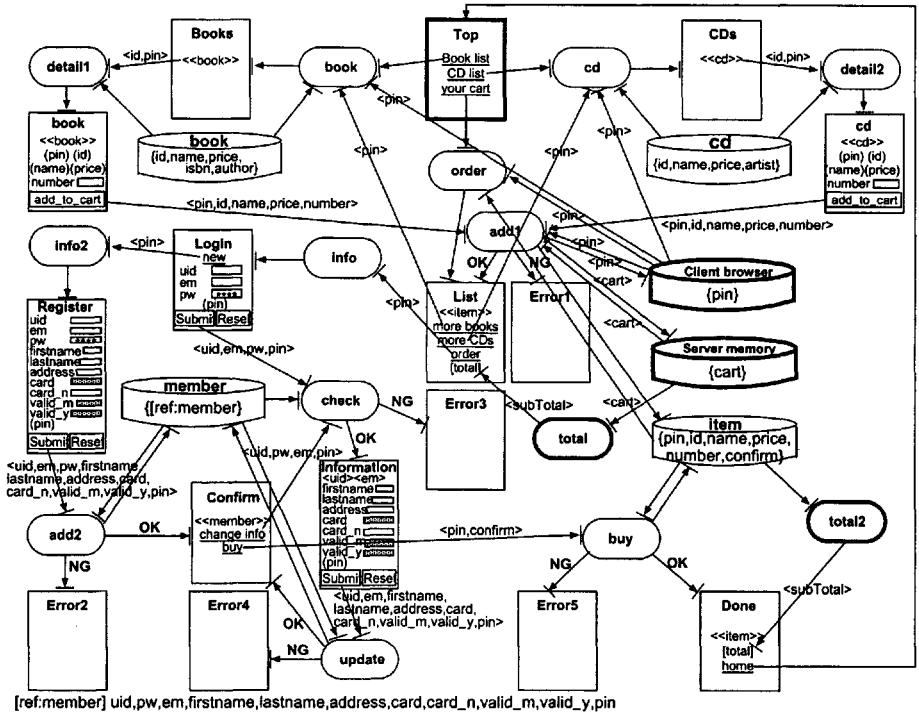


Figure 13: Data-flow & hyperlink diagram for the shopping system

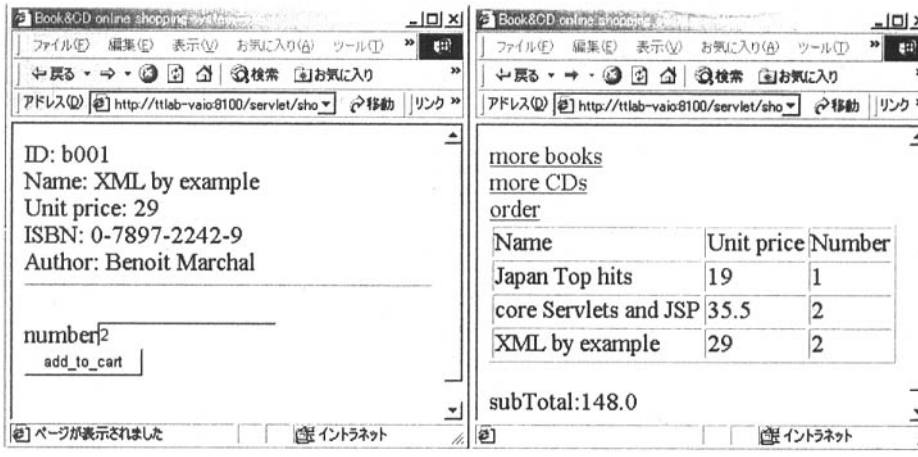


Figure 14: Screen shots of the generated shopping system

```
/*package PACKAGE;*/
// template name:show-all-info
import java.sql.*;
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class /*CLASSNAME*/ extends HttpServlet {
 public void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
  String dbName = "/*DBNAME*/";
  Connection con = TWeb.doConnect(dbName);
  /**String PARAMETER = "";
  **/
  /**INPUTPARAM = request.getParameter("INPUTPARAM");
  **/
  String[] fNames = {/*"COLUMNLABEL"*/};
  int[] colNumbers = {/*COLUMNNUMBER*/};
  if
(TWeb.cookieAuthenticate(request,response,/*"AUTHCOOKIE"*/,/*AUTHCOOKIE*/)){
   /*String tbName = "TABLE";
   String query = "SELECT * FROM "+tbName;
   request.setAttribute("result",TWeb.doSelect(con,query).toHTML("STYLE",
"CYAN",fNames,colNumbers).toString());*/
   /**GETCOOKIE = TWeb.cookieGet(request,"GETCOOKIE");
   **/
   /**TWeb.cookieDel(request,response,"DELCOOKIE");
   request.getSession(true).removeAttribute("DELCOOKIE");**/
   gotoPage("/*/JSPDIRECTORY/*//*OKPAGE.jsp*/",request,response);
  } else {
   gotoPage("/*/JSPDIRECTORY/*/cookieError.jsp",request,response);
  }
  TWeb.doDisconnect(con);
 }
 public void doPost(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
  doGet(request,response);
 }
 private void gotoPage(String address, HttpServletRequest request,
HttpServletResponse response) throws ServletException, IOException {
  RequestDispatcher dispatcher =
getServletContext().getRequestDispatcher(address);
  dispatcher.forward(request,response);
 }
}
```

Figure 15: A Servlet template

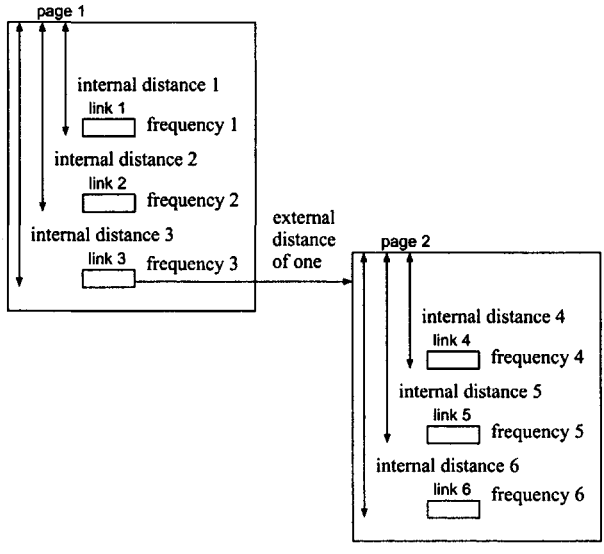


Figure 16: Human complexity of reaching Web pages

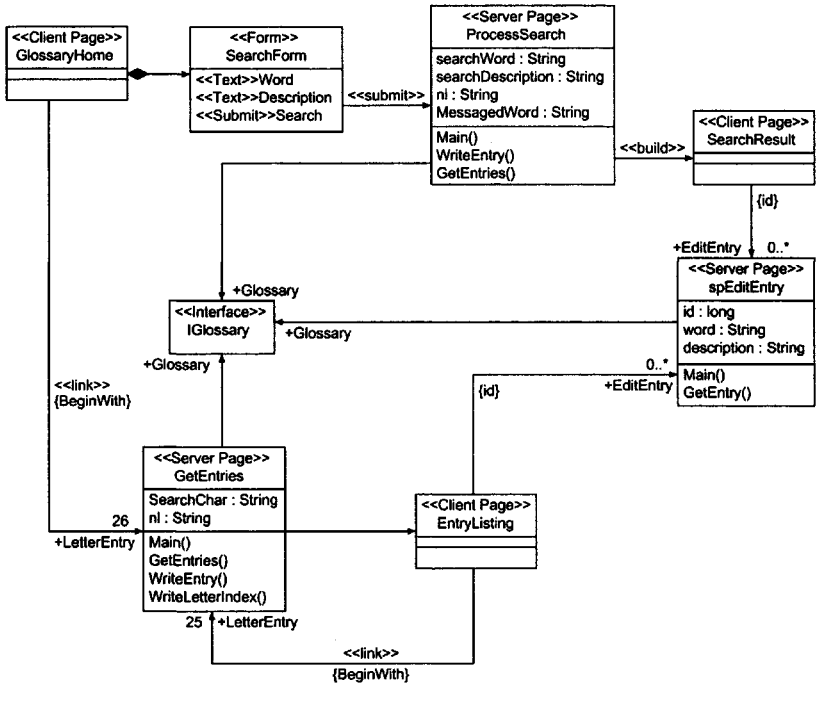


Figure 17: UML diagram for the glossary system (in part)

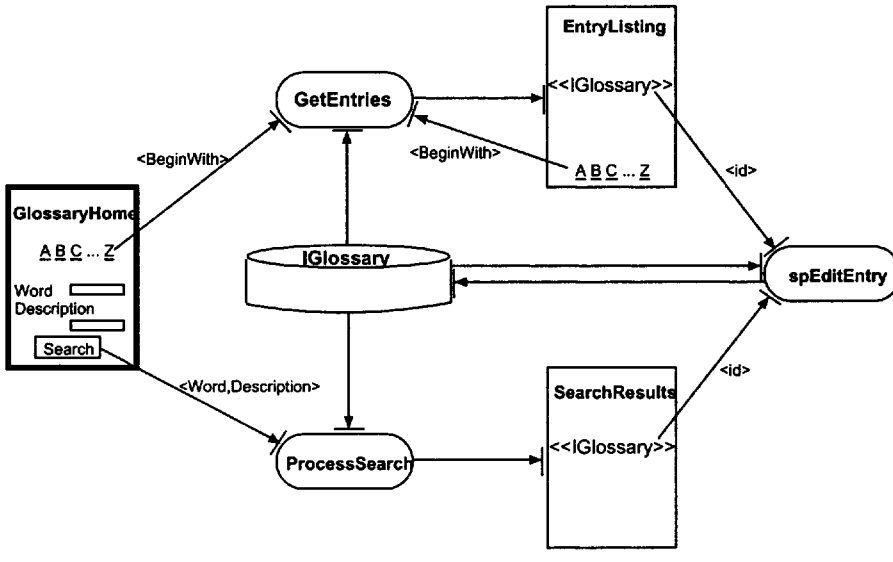


Figure 18: Web transition diagram for the glossary system

7 Conclusion

We have presented a family of Web diagrams which may be helpful for us to design, construct, and evaluate Web applications. Our Web diagrams are simple but powerful enough to represent the overall structure and behavior of many types of Web applications. Some of the future work may be as follows. Currently our hyperlink diagram cannot represent multiple frames. Our data-flow diagram cannot show the ordering of data-flow arrows in time sequence. We would like our representation method to cover those aspects of design processes. Finally we may use UML diagrams for the analysis of deep level program structures and our diagrams for the presentation and generation of surface level structures of Web applications.

References

[1] Piero Fraternali. Tools and Approaches for Developing Data-Intensive Web Applications: A Survey. ACM Computing Surveys, Vol. 31, No. 3, pages 227-263, 1999.

[2] Mark Pearrow. Web Site Usability Handbook. Charles River Media, 2001.

[3] Paul Kahn, Krzysztof Lenk. Mapping Websites: Digital Media Design. Rotovision, 2000.

[4] Jim Conallen. Building Web Application with UML. Addison-Wesly, 2000.

[5] An Atlas of Cyberspaces. http://www.cybergeography.org/atlas/atlas.htm

[6] Shinshir Gundavaram. CGI Programming on the World Wide Web. O'Reilly & Associates, 2000.

[7] Marty Hall. Core Servlets and JavaServer Pages. Prentice Hall, 2000.

[8] Alex Homer. Dave Sussman and Brian Francis and others. Professional Active Server Pages 3.0. Wrox Press, 1998.

[9] David Flanagan. JavaScript: The Definitive Guide. O'Reilly & Associates, 2001.

[10] Sun Microsystems, Inc. Applets. http://java.sun.com/applets/

[11] James Gosling, Bill Joy, Guy Steele, Gilad Bracha. The Java Language Specification. Addison-Wesley, Massachusetts, 2000

[12] Dave Raggett, Arnauld Le Hors, Ian Jacobs. HTML 4.01 Specification. http://www.w3.org/TR/html4/, 1999.

[13] Hypertext Transfer Protocol. http://www.w3.org/Protocols

[14] Chetan Sharma, Jeff Kunins. VoiceXML: Professional Developer's Guide with CDROM. John Wiley & Sons, 2001.

[15] K.Jamroendararasame, T. Matsuzaki, T.Suzuki, T.Tokuda. Generation of Secure Web Applications from Web Transition Diagrams. Proceedings of the IASTED International Symposia Applied Informatics, pages 496-501, 2001.

[16] K.Jamroendararasame, T. Matsuzaki, T.Suzuki, T.Tokuda. Two Generators of Secure Web-Based Transacion Systems. Proceedings of the 11th European-Japanese Conference on Information Modelling and Knowledge Bases, pages 348-362, 2001.

[17] K.Jamroendararasame, T.Suzuki, T.Tokuda. JSP/Servlet-Based Web Application Generator. 18th Conference Proceedings Japan Society for Software Science and Technology, 2C-1, 2001.

Information Modelling and Knowledge Bases XIV
H. Jaakkola et al. (Eds.)
IOS Press, 2003

A Model for Defining and Composing Interaction Patterns

Thomas Feyer and Bernhard Thalheim

Computer Science Institute
Brandenburg Technical University at Cottbus, FRG
{feyer, thalheim}@informatik.tu-cottbus.de

Abstract

Design of adaptive and ergonomic user interfaces has been recognized as a complex and expensive task. In particular, in information services available to a large user variety, the acceptance of a system strongly depends on the adequateness of its user interface. In fact, there emerged structures of user interaction that occur in many information systems, for example, dialog structures that support searching. As a consequence, the notion of pattern was applied to the area of user interaction to improve the design task in terms of time and quality [10, 15, 4]. Roughly, an interaction pattern comprises dialog structures and corresponding actions that are typically applied to solve a specific, interactive task.

A basic problem of interaction patterns concerns composability. In contrast to closed subsystems, interaction patterns may execute in an interleaved way. Thus, the dependency between patterns commonly is not a sequential invocation, but rather parallel or intertwined. Therefore we propose that besides the internal structure, interaction patterns additionally specify legal interfaces that may be used for their composition. For this purpose, we introduce a model to define (i) the behavior and interfaces of interaction patterns, and (ii) the composition of patterns. We employ place/transition systems which offer suitable abstraction and synchronization methods necessary for interaction design.

1 Introduction

It has been recognized that specific dialog structures occur in many information systems. To support the design process, the notion of patterns [9] was carried over to the area of user interaction [10, 4, 15, 8]. Commonly, these models comprise a pattern name, problem description, solution, related patterns, and examples. Simple patterns are scrolling back and forth within a list, navigational history — offering the opportunity to go backwards some interaction steps, or guided tour through a domain [11]. More complex patterns are searching a catalog, or managing a shopping cart.

An essential and not yet satisfactorily solved problem arises when interaction patterns must be composed in a dependent way. For example, to provide navigational history during scenarios of searching catalogs, both patterns must be composed in a way that allows parallel execution. Furthermore, both patterns must be able to exchange their

internal context to realize the dependencies between them. For example, if the user steps backwards by using the navigational history pattern, the view on the catalog must be adapted subsequently.

To enable interleaved compositions, we propose the following approach. An interaction pattern describes a function that consumes an (user) input, processes it according to its current state/context, and produces an output. This definition divides the overall behavior of a pattern into atomic interactive steps. Figure 1 illustrates an interleaved execution of an interaction sequence (or user scenario) $S_1 \times S_2 := i_1, i'_1, i'_2, i_2, i_3, i'_3$ that affects two different patterns. Each box represents an atomic interactive step which is initiated by an (user) input i. Besides parallel execution, patterns may propagate contexts to resolve dependencies between one another.

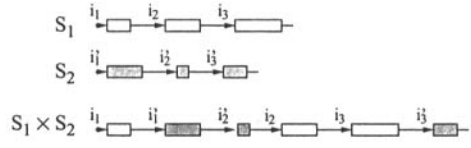


Figure 1: Example of an interleaved user scenario $S_1 \times S_2$

Figure 2 presents a big picture how interaction patterns are composed and integrated into an interactive environment. Events initiated at an input channel are transfered to a pattern that reacts on this event. After processing, output is generated onto output channels. In addition, interaction patterns may interact with each other, for example, to publish their contexts.

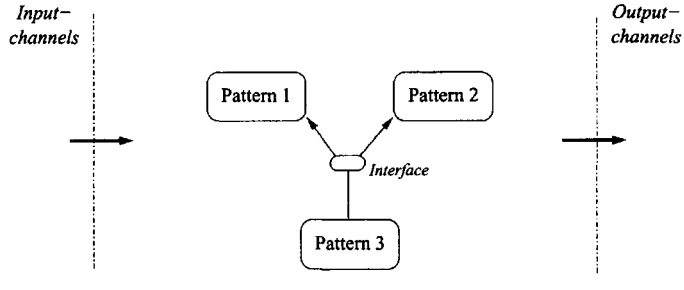


Figure 2: Interplay of interaction patterns and their environment

Throughout the paper, we will concentrate on the formal model of interaction patterns, but neglect their rather descriptive part. However, we consider a pattern definition as comprising both a descriptive *and* a formal part. Furthermore, we mainly target at user interaction instead of interaction in its general sense. Nonetheless, the proposed model for interaction pattern does not restrict its application to user interaction.

Place/transition nets (p/t nets) are successfully applied for interaction design, for example [1, 6, 13]. We employ p/t nets as a basis for defining interaction patterns because of the following reasons:

- The granularity of input and output elements can be varied by the designer. This allows to model interaction at different levels of abstraction.

- According composition, p/t nets offer a natural framework to model parallelism and synchronization.
- Functional abstraction/specialization is achieved by defining/refining black-box transitions (or macro transitions).
- P/T nets possess a well-defined semantics.
- Specifications may be independent from particular user interface paradigms. In fact, depending on the granularity of input and output elements, specifications can be completely independent from, dependent on a class of, or dependent on specific user interface paradigms (as, for example, HTML). We are rather concerned with independent specifications since they address a larger class of applications, in particular, future user interface paradigms.
- There exist graphical representations.

The reminder of the paper is organized as follows. In section 2, we describe the application of p/t nets for interaction specification and introduce interaction nets and their composition as a basis for interaction patterns. Section 3 defines interaction patterns and composition of patterns. Their application according user interaction is illustrated in section 4. Sections 5 and 6 discuss open problems and conclude the paper.

2 Utilizing Place/Transition Nets

Definition 1 *A* place/transition net $\mathcal{N} = (P, T, W, v, m_0)$ *is defined by a finite set of places* P*, a finite set of transitions* T *with* $P \cap T = \{\}$*, a relation* $W \subseteq P \times T \cup T \times P$*, a function* $v : W \rightarrow I\!N^+$ *(called* arc labeling*), and a function* $m_0 : P \rightarrow I\!N$ *(called* initial marking*).*

Thereby, $I\!N$ and $I\!N^+$ denote the set of natural numbers and the set of positive natural numbers respectively. We further define a countable infinite base set $\mathbf{P}$ of possible places, and a countable infinite set $\mathbf{N}$ of possible names.

For simplicity, we will refer to the basic definition of p/t nets in the formalization of interaction patterns and adopt a standard semantics for the net execution [3]. However, to design readable net specifications, extensions to the basic definition are essential. Coloured petri nets [12] represent a commonly applied extension. They basically provide a framework to include typed places and function labeled transitions. Instead of unstructured tokens, they enable to include arbitrary complex data elements which are processed by functions that are associated with transitions. In contrast to basic p/t nets, place markings are multi-sets of data elements and arc labels are typed variables.

To distinguish places which are used for interaction, we define

Definition 2 *An* interaction net $\mathcal{N}^i = (P, T, W, v, m_0, P^i)$ *is defined by a place/transition net* (P, T, W, v, m_0) *and a subset* $P^i \subseteq P$ *called* i/o places.

I/O places are places which can be used by the environment of the net. I.e., external processes may produce/consume tokens onto/from i/o-places. Thus, i/o places are used

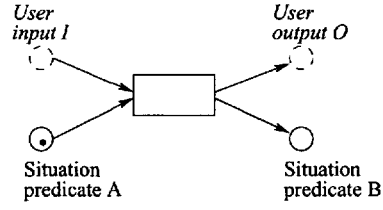


Figure 3: A simple example of an interaction net

as an interface of the interaction net to its environment. Figure 3 illustrates a simple interaction net where i/o places are distinguished by dashed circles.

At a pragmatic point of view, we distinguish input places, output places, context places, and transitions:

Input places Input places are i/o places where external components may write into. Examples are input places which correspond to (i) events initiated by a user – for example, logging in, (ii) parameters published by another net – for example, the task history of the current session, or (iii) controlling information used for synchronization – for example, an activation signal.

Output places Output places are i/o places where external components may read from. The interaction net conveys information via output places. If the external component connected to an output place represents a subsystem or driver, output places represent an abstraction of external actions. For example, if an output place is externally consumed by a user interface driver or database manipulation driver, producing elements into an output place corresponds to outputting information to the user or initiating a manipulation request to a database.

Context places Context places are non-i/o places. They represent the internal context of the net during the execution of interactive scenarios. They can be interpreted as situation predicates since the elements in each context place characterize a single facet of the current situation. For example, elements in context places may represent which subtasks a user already accomplished within an interactive task as well as preferences of the current user.

Transitions Transitions define context changes. In particular in user interaction, the context of interacting parties changes after each utterance (input) [17, 14]. To act/react properly, these changes must be reflected in the system. They are realized by transitions that compute the new context by altering the situation predicates depending on the utterance.

To enable associativity and commutativity of the composition, we introduce a rename operator. It renames places to prepare nets for their composition.

Definition 3 *A* renaming $\rhd_f$ *of a place/transition net* (P, T, W, v, m_0) *is defined by a total, injective function* $f : P \to \mathbf{P}$. *The renamed place/transition net* $(P', T', W', v', m_0') := \rhd_f((P, T, W, v, m_0))$ *is defined as*

$$P' := f(P), \quad T' := T,$$
$$W' := \{(p', t) \mid (f^{-1}(p'), t) \in W\} \cup \{(t, p') \mid (t, f^{-1}(p')) \in W\},$$
$$v'(p', t') := v(f^{-1}(p'), t'), \quad v'(t', p') := v(t', f^{-1}(p')) \quad \textit{for } p' \in P', t' \in T',$$
$$m_0'(p') := m_0(f^{-1}(p')),$$

where $f(P) := \bigcup_{p \in P}\{f(p)\}$, *and* f^{-1} *denotes the reverse function of* f.

If function f is not total on P, we can generally apply the comprehension of f as $f \cup \{p \to p \mid p \in P \wedge f(p)$ is undefined$\}$. The renaming $\rhd_f$ of an interaction net (P, T, W, v, m_0, P^i) is defined accordingly. There, all i/o places in P^i must be renamed by function f in addition.

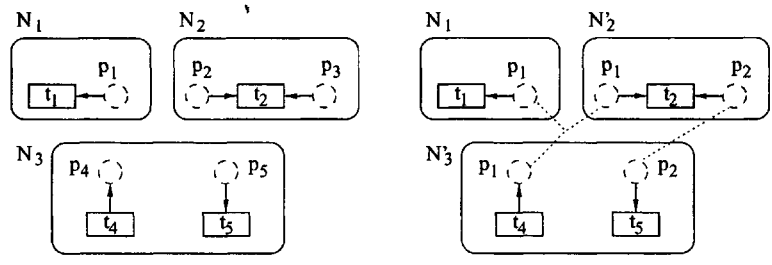


Figure 4: *Left:* Nets $\mathcal{N}_1, \mathcal{N}_2, \mathcal{N}_3$; *Right:* Renamed nets $\mathcal{N}_2' := \rhd_{f_2}(\mathcal{N}_2)$, $\mathcal{N}_3' := \rhd_{f_3}(\mathcal{N}_3)$

Figure 4 demonstrates the renaming. The nets $\mathcal{N}_2'$ and $\mathcal{N}_3'$ result from rename operations $\rhd_{f_2}(\mathcal{N}_2)$ and $\rhd_{f_3}(\mathcal{N}_3)$ with functions $f_2 := \{p_2 \to p_1, p_3 \to p_2\}$ and $f_3 := \{p_4 \to p_1, p_5 \to p_2\}$.

The composition of nets is defined by merging shared places:

Definition 4 *The* composition $\mathcal{N}_1 \bowtie \mathcal{N}_2$ *of place/transition nets* $\mathcal{N}_1 = (P_1, T_1, W_1, v_1, m_{0_1})$, $\mathcal{N}_2 = (P_2, T_2, W_2, v_2, m_{0_2})$ *defines a place/transition net* $\mathcal{N} = (P, T, W, v, m_0)$ *as*

$$P := P_1 \cup P_2,\ T := T_1 \cup T_2,\ W := W_1 \cup W_2,\ v := v_1 \cup v_2, \textit{ and}$$
$$m_0(p) := \begin{cases} m_{0_1}(p) & \textit{if } p \in P_1 - P_2, \\ m_{0_2}(p) & \textit{if } p \in P_2 - P_1, \\ m_{0_1}(p) + m_{0_2}(p) & \textit{if } p \in P_1 \cap P_2. \end{cases}$$

assumed that $T_1 \cap T_2 = \{\}$.

For coloured nets, the composition must additionally assume that merged places are type-compatible. Then, the place markings result from the *union* of the markings of the argument nets.

The composition $(P_1, T_1, W_1, v_1, m_{0_1}) \bowtie (P_2, T_2, W_2, v_2, m_{0_2})$ of interaction nets is defined accordingly. There, the set of i/o places P^i of the composed net computes as the union of the i/o places of the argument nets. Since interaction nets may be connected at i/o places only, it must be assumed that the non-i/o places of $(P_1, T_1, W_1, v_1, m_{0_1})$

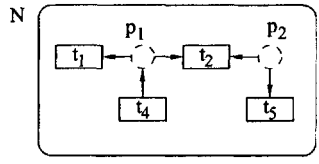


Figure 5: Composed net: $\mathcal{N} := \mathcal{N}_1 \bowtie \mathcal{N}_2' \bowtie \mathcal{N}_3'$

and $(P_2, T_2, W_2, v_2, m_{0_2})$ are disjunctive. The composition $(\mathcal{N}_1 \bowtie \mathcal{N}_2') \bowtie \mathcal{N}_3'$ of the nets in Figure 4 is shown in Figure 5.

Because of the following proposition, we can generally omit parenthesis in multiple compositions, and may define $\bowtie \{\mathcal{N}_1, \ldots, \mathcal{N}_k\} := \mathcal{N}_1 \bowtie \ldots \bowtie \mathcal{N}_k$.

Proposition 1 *The composition operator* $\bowtie$ *is associative and commutative.*

Proof sketch By the definition, associativity and commutativity can be reduced to the associativity and commutativity of the set operators union ($\cup$) and intersection ($\cap$) and the summation operator ($+$).

3 Interaction Patterns

In [5], the behavior of interactive systems is specified by a relation between input streams and output streams. To enable the division of the overall behavior into atomic (context dependent) interactive steps, we adapt the definition by incorporating the internal context (system state). Thereby, the interactive behavior can be determined by a relation between input, internal context, and output. Concerning user interaction, it describes the interplay between user events, context changes, and initiated actions.

In the following, we define interaction patterns as a generalization of interaction nets. The generalization mainly concerns two aspects:

(i) To enable a meaningful and consistent composition of interaction patterns, we extend the notion of interface by integrity constraints. The rational behind is that through unrestricted composition, we cannot derive statements about the behavior of the composed net without knowing the exact structure of the net. Constrained interfaces offer the opportunity for generic compositions with a specified behavior. Examples are sequentialized, parallelized, or mutually synchronized compositions.

(ii) Due to flexibility, place/transitions net may be "under-specified". When a pattern is employed for the design of a specific application, unspecified elements can be adapted to this application. For example, unspecified transition functions or initial data elements are defined as desired.

Definition 5 *A* (constrained) interface $\mathcal{I} = (P^i, S_\mathcal{R})$ *is defined by a set of places* P^i *and a set of interface roles* $S_\mathcal{R}$. *An* (interface) role $\mathcal{R} = (r, \Sigma)$ *wrt. a set of places* P^i *is defined by a name* r *and a set of dynamic integrity constraints* Σ *over* P^i.

Figure 6, illustrates a constrained interface $\mathcal{I}_1$ with two roles: a *sender* and a *receiver*. The interface comprises two places *data* and *ack* and enables a synchronized parameter transfer. Net $\mathcal{P}_1$ acting the role of the sender generates a data element into the place *data*. Net $\mathcal{P}_2$ acting the role of the receiver consumes the data element and generates an acknowledgment. The corresponding integrity constraints of the sender and the receiver are:

Σ_{sender}: If one of the places *ack* or *data* is changed, then this change obeys the rule: consume an element from place *ack* and produce an element into place *data*.

$$\forall m_1, m_2.\ m_1 \rightarrow m_2 \wedge (m_1(ack) \neq m_2(ack) \vee m_1(data) \neq m_2(data)) \Rightarrow m_2(ack) = m_1(ack) - 1 \wedge m_2(data) = m_1(data) + 1$$

Thereby, $m_1 \rightarrow m_2$ represents the relation between two subsequent net states.

Σ_{receiver}: If one of the places *ack* or *data* is changed, then this change obeys the rule: consume an element from place *data* and produce an element into place *ack* for confirmation.

$$\forall m_1, m_2.\ m_1 \rightarrow m_2 \wedge (m_1(ack) \neq m_2(ack) \vee m_1(data) \neq m_2(data)) \Rightarrow m_2(ack) = m_1(ack) + 1 \wedge m_2(data) = m_1(data) - 1$$

In case of coloured nets, '+'/'−' operators and '≠' relation must be replaced by according union/difference operators and inequality relation over multi-sets.

Constrained interfaces provide the basis to define complex interfaces (called *active interfaces*) with internal states and multiple roles. They are discussed in section 4.

Interaction patterns can be connected via different interfaces. The association of a net with an interface, we call interface binding (for convenience, we apply the "." notation to reference subcomponents):

Definition 6 *An* interface binding $\mathcal{B} = (b, \mathcal{I}, S_{\mathcal{R}}, a)$ *wrt. a set of places* P *is defined by a name* b*, an interface* $\mathcal{I}$*, a set of roles* $S_{\mathcal{R}} \subseteq \mathcal{I}.S_{\mathcal{R}}$*, and a total, injective function* $a : \mathcal{I}.P^i \rightarrow P$ *called* binding function.

Definition 7 *An* interaction pattern $(P, T, W, m_0, v, S_{\mathcal{B}})$ *is defined by an under-specified place/transition net* (P, T, W, v, m_0) *together with a set of interface bindings* $S_{\mathcal{B}}$ *wrt.* P *with pairwise different names and pairwise disjunctive ranges of the binding functions. It is required that the integrity constraints in* $S_{\mathcal{B}}$ *are locally enforced by the net.*

Unique names are necessary for the identification of interfaces. The condition of disjunctive ranges of the binding functions ensures that interface bindings do not overlap, i.e., places are bound to a single interface exclusively. Under-specification includes open arrow semantics (for example, "or"-semantic instead of "and"-semantic), and missing initial tokens. According coloured nets, under-specification also comprises unspecified transition functions, untyped places, missing arc inscriptions, and missing guards.

Figure 6 represents two interaction patterns $\mathcal{P}_1$, $\mathcal{P}_2$. There, places of interface bindings are visually grouped by dotted frames. The binding function is indicated by dashed lines for the interface binding b_1 of $\mathcal{P}_1$ and $\mathcal{P}_2$.

The condition of local integrity enforcement is defined more precisely as:

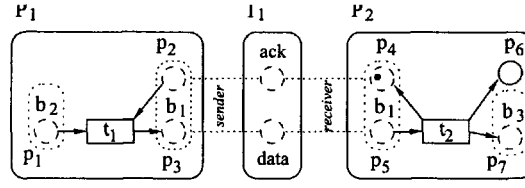


Figure 6: Two interaction patterns $\mathcal{P}_1$, $\mathcal{P}_2$ bound to a common interface $\mathcal{I}_1$

Definition 8 *An interaction pattern* $(P, T, W, m_0, v, S_\mathcal{B})$ (locally) enforces *the integrity constraints of an interface binding* $(b, \mathcal{I}, S_\mathcal{R}, a) \in S_\mathcal{B}$ *with* $S_\mathcal{R} = \{(r_1, \{\sigma_{1,1}, \ldots, \sigma_{1,n_1}\}), \ldots, (r_k, \{\sigma_{k,1}, \ldots, \sigma_{1,n_k}\})\}$, *if it fulfills the integrity constraint*

$$(\sigma_{1,1} \wedge \cdots \wedge \sigma_{1,n_1}) \vee \cdots \vee (\sigma_{k,1} \wedge \cdots \wedge \sigma_{k,n_k})$$

where all places p *are exchanged by* $a(p)$.

In other words, if an interface binding provides several roles, their integrity constraints are combined by the logical "or". Intuitively, it means that everything is allowed what is allowed in any of the roles.

To enable associativity and commutativity of the composition, we introduce two rename operators. They rename places as well as interface bindings to prepare nets for their composition.

Definition 9 *A* renaming $\rhd_f$ *of an interaction pattern* $(P, T, W, m_0, v, S_\mathcal{B})$ *is defined by a total, injective function* $f : P \to \mathbf{P}$. *The renamed pattern* $(P', T', W', m_0', v', S_\mathcal{B}') := \rhd_f((P, T, W, m_0, v, S_\mathcal{B}))$ *is defined as*

$$(P', T', W', v', m_0') := \rhd_f((P, T, W, v, m_0)),$$
$$S_\mathcal{B}' := \bigcup_{(b,\mathcal{I},S_\mathcal{R},a)\in S_\mathcal{B}} \{(b, \mathcal{I}, S_\mathcal{R}, f \circ a\},$$

where "$\circ$" denotes function concatenation.

In other words, a pattern is renamed by renaming its underlying p/t net (as defined in section 2 and renaming all binding functions.

Definition 10 *An* interface renaming $\rhd_f^i$ *of an interaction pattern* $(P, T, W, m_0, v, S_\mathcal{B})$ *is defined by a total, injective function* $f : \bigcup_{\mathcal{B}\in S_\mathcal{B}} \{\mathcal{B}.b\} \to \mathbf{N}$. *The renamed interaction pattern* $(P', T', W', m_0', v', S_\mathcal{B}') := \rhd_f((P, T, W, m_0, v, S_\mathcal{B}))$ *is defined as*

$$(P', T', W', v', m_0') := (P, T, W, v, m_0),$$
$$S_\mathcal{B}' := \bigcup_{(b,\mathcal{I},S_\mathcal{R},a)\in S_\mathcal{B}} \{(f(b), \mathcal{I}, S_\mathcal{R}, a\}$$

The composition of nets is defined by merging compatible interface bindings. The bindings b_1 of the patterns $\mathcal{P}_1$ and $\mathcal{P}_2$ in Figure 6 represent a compatible binding.

Definition 11 *A pair* $(\mathcal{B}_1, \mathcal{B}_2)$ *of interface bindings is* compatible, *iff* $\mathcal{B}_1.b = \mathcal{B}_2.b$ *and* $\mathcal{B}_1.\mathcal{I} = \mathcal{B}_2.\mathcal{I}$

Definition 12 *The* composition $\mathcal{P}_1 \bowtie \mathcal{P}_2$ *of interaction patterns* $\mathcal{P}_1 = (P_1, T_1, W_1, m_{0_1}, v_1, S_{\mathcal{B}_1})$, $\mathcal{P}_2 = (P_2, T_2, W_2, m_{0_2}, v_2, S_{\mathcal{B}_2})$ *defines an interaction pattern* $\mathcal{P} = (P, T, W, m_0, v, S_{\mathcal{B}})$, *as*

$$(P, T, W, v, m_0) := (P_1, T_1, W_1, v_1, m_{0_1}) \bowtie \rhd_{f_1}(\rhd_{f_2}(\ldots \rhd_{f_k}((P_2, T_2, W_2, v_2, m_{0_2}))\ldots)),$$

$$\begin{aligned} S_{\mathcal{B}} \;:=\; & \{(\mathcal{B}_1.b, \mathcal{B}_1.\mathcal{I}, \mathcal{B}_1.S_{\mathcal{R}} \cup \mathcal{B}_2.S_{\mathcal{R}}, \mathcal{B}_1.a) \mid (\mathcal{B}_1, \mathcal{B}_2) \in S_{\mathcal{B}_1\mathcal{B}_2}\} \cup \\ & \{\mathcal{B}_1 \in S_{\mathcal{B}_1} \mid \nexists \mathcal{B}_2.(\mathcal{B}_1, \mathcal{B}_2) \in S_{\mathcal{B}_1\mathcal{B}_2}\} \cup \\ & \{\mathcal{B}_2 \in S_{\mathcal{B}_2} \mid \nexists \mathcal{B}_1.(\mathcal{B}_1, \mathcal{B}_2) \in S_{\mathcal{B}_1\mathcal{B}_2}\} \end{aligned}$$

where $S_{\mathcal{B}_1\mathcal{B}_2}$ *denotes the set of all compatible interface bindings:* $S_{\mathcal{B}_1\mathcal{B}_2} := \{(\mathcal{B}_1, \mathcal{B}_2) \in S_{\mathcal{B}_1} \times S_{\mathcal{B}_2} \mid (\mathcal{B}_1, \mathcal{B}_2)$ *is compatible.*$\}$, $|S| = k$, *and each compatible pair* $(\mathcal{B}_1, \mathcal{B}_2) \in S_{\mathcal{B}_1\mathcal{B}_2}$ *defines an* $f_i := \mathcal{B}_1.a \circ \mathcal{B}_2.a^{-1}$ $(i = 1, \ldots, k)$;

assumed that $T_1 \cap T_2 = \{\}$, *and* $\forall p.\, p \in P_1 \cap P_2 \Rightarrow p$ *occurs in* $S_{\mathcal{B}_1\mathcal{B}_2}$.

The definition is sound, because the sequence of renaming operators is not significant, since they do not affect each other. The resulting interface bindings originate from: (i) for each pair of compatible interface bindings, one interface binding is chosen together with the union of the roles, (ii) each interface bindings of pattern $\mathcal{P}_1$ and pattern $\mathcal{P}_2$ that do not occur in a compatible interface binding are chosen unmodified.

The composition $\mathcal{P}_1 \bowtie \mathcal{P}_2$ of the patterns in Figure 6 is shown in Figure 7. The bindings b_1 of patterns $\mathcal{P}_1, \mathcal{P}_2$ merge into a single binding b_1 of pattern $\mathcal{P}$ with two roles $\{(sender, \Sigma_{sender}), (receiver, \Sigma_{receiver})\}$.

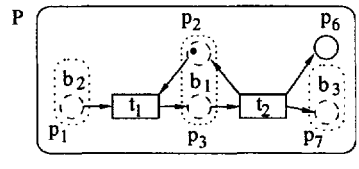


Figure 7: Composed pattern: $\mathcal{P} := \mathcal{P}_1 \bowtie \mathcal{P}_2$

Proposition 2 *The composition operator* $\bowtie$ *wrt. interaction patterns is associative and commutative modulo renaming interface places.*

Proof sketch By the definition, associativity and commutativity can be reduced to the associativity and commutativity of the set operators union ($\cup$) and intersection ($\cap$) and the summation operator ($+$). Furthermore, place names of compatible interface bindings are taken from the left pattern. Therefore, these names depend on the composition sequence. However, these names do not affect the composition, since merging is defined on the basis of binding names and binding functions. In addition, the composed pattern enforces the integrity constraints of its argument patterns due to the "or" semantics in the enforcement condition of Definition 8.

4 Applying Patterns for User Interaction

With respect to user interaction, we divide patterns into two classes: patterns of user and non-user interaction respectively. We employ patterns of non-user interaction to realize complex generic compositions between user interaction pattern. Thus, they play the role of *active interfaces*, i.e., interfaces with internal states. In the following, we will exclusively use the term interaction pattern for user interaction, and use the term active interface for patterns of non-user interaction. Figure 8 outlines the use of active interfaces by an example that indicates a producer/consumer buffer. One party acting the role of the producer may generate data elements. Another party acting the role of the consumer may asynchronously process them.

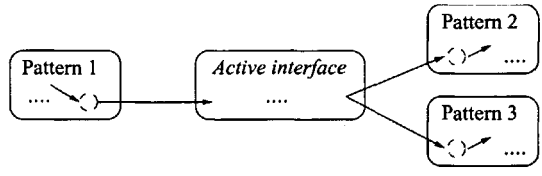


Figure 8: Patterns composed through an active interface

Our aim is to compose patterns in a generic way that allows to achieve (and guarantee) a specific behavior. We therefore employ active interfaces to specify the behavior of compositions. In particular, we intend to provide a set of active interfaces that cover characteristic compositions of interaction patterns. Essential interfaces are:

Sequential composition: Two or more patterns are executed in a predefined sequence. After all interactive steps of one pattern are accomplished by the user, the next pattern becomes active.

Parallel composition: Two or more patterns are executed in parallel. If two user events occur at the same time, according patterns are invoked concurrently.

Synchronized composition: Two or more patterns are active in parallel, i.e., they are concurrently listening for user events. If a user event occurs, then the according pattern is invoked while the other patterns are passive during event processing. (This mode is also known as pseudo-parallel composition.)

Context publication: A pattern publishes parts of its current context to other patterns.

The example of composing navigation history with a search pattern can be realized through two active interfaces (i) a synchronized composition together with (ii) a bi-directional context publication. In each dialog situation, the user either initiates a 'jump' event through the navigation history or a 'choice' event through the search pattern. The affected pattern processes the event and publishes its altered context. For example, if the user jumps back, the navigation history adapts its context. After the context is published, the search pattern is invoked and adapts its output to the former dialog situation.

Figure 9 illustrates an active interface that realizes context publication. It is drawn in a coloured petri net style. It transfers data elements produced by the publisher to the subscriber. In contrast to a queued publication, it overwrites data elements, if formerly published elements were not consumed by the subscriber. Thus, the interface can be interpreted as a view onto the current context of the publisher. Thereby, the publisher and the subscriber patterns are connected to the active interface by the constrained interface $\mathcal{I}_1$ discussed in Figure 6.

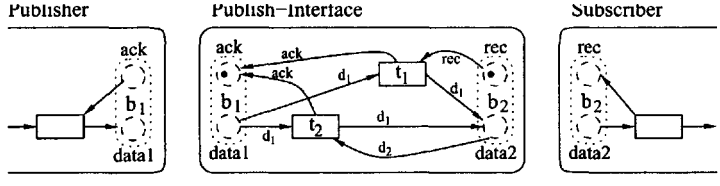


Figure 9: An active interface used for (non-buffered) context publication

Figure 10 motivates the application of active interfaces, i.e., the composition of complex patterns by elementary ones. There are three (non-specified) interaction patterns: 'Search' representing a pattern for keyword search, 'Categories' representing a pattern for navigating along categories, and 'Selection' representing a pattern for selecting elements from small-sized lists. The labeled arrows in the picture represent a shortcut for active interfaces. Thus, the composed pattern illustrates a common structure for catalog search.

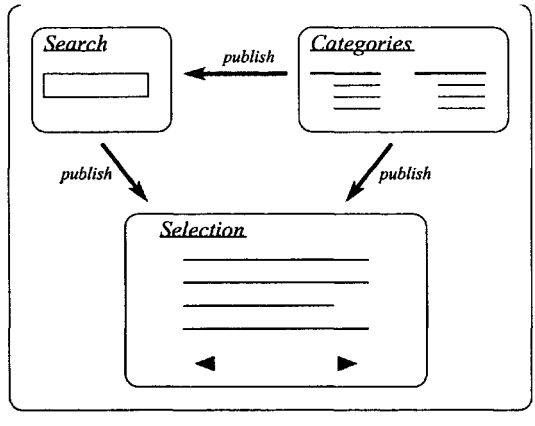


Figure 10: Composing patterns: An application example

5 Open Problems and Future Work

Patterns are commonly characterized by relations to other patterns [18]. Typical relations are specialization, part-of relation, and association. These relations must be

determined by the designer. If new patterns are developed, it becomes a major maintenance problem. To a certain extent, formal specifications pattern behavior might permit to derive pattern relations automatically. More specifically, the part-of relation can be derived from the composition itself, i.e., a pattern $\mathcal{P}_1$ is part of a pattern $\mathcal{P}$, if $\mathcal{P} := \cdots \bowtie \mathcal{P}_1 \bowtie \cdots$. Furthermore, the specialization relation among patterns can be derived from net refinements [3].

As future work, we also consider the formalization of interaction patterns frequently used in information services [7, 16] as, for example, search and learning patterns. Our intention is to provide patterns which are preferably independent of the user interface and which can be formally verified. The aspect of verification is in particular useful for adaptable user interaction. Our hope is that formal models of human's interactive behavior as, for example, analyzed in the logic of question and answer [2, 14], can be represented and verified through interaction patterns at least for certain application domains.

To compose patterns in a semantic way, active interfaces for certain modes of composition must be provided. To verify the composition semantics, we intend to apply the notion of streams used in [5]. It will enable to consider compositions as black boxes which provide a defined behavior.

6 Conclusion

The paper introduced a model that can be used for the formal specification of interaction patterns. It is based on place/transition nets and permits an user interface independent specification and verification. Through the extension of interface definitions, it enables to compose patterns. The composition of patterns possesses the essential closure property, i.e., composing patterns results in a pattern itself. Furthermore, we extended the notion of interface to active interfaces. These interfaces enable the composition of patterns at different semantic abstractions.

References

[1] R. Bastide and P. A. Palanque. A petri net based environment for the design of event-driven interfaces. In *16th Int. Conf. on Application and Theory of Petri Nets*, pages 66 – 83, Turin, Italy, June 1995.

[2] N. Belnap and T. Steel. *The logic of questions and answer.* D. Reidel Publishing Co., Dordrecht, 1976.

[3] E. Best, R. Devillers, and M. Koutny. *Petri Net Algebra.* Springer, 2001.

[4] J. Borchers. *A Pattern Approach to Interaction Design.* John Wiley & Sons, New York, 2001.

[5] M. Broy and K. Stølen. *Specification and Development of Interactive System.* Springer, New York, 2001.

[6] W. Clauss and J. Lewerenz. Abstract interaction specification for information services. In *Int. Conf. On Managing Information Technology Resources in Organizations*, Hershey, Pennsylvania, May 1999.

[7] T. Feyer, K.-D. Schewe, and B. Thalheim. Conceptual design and development of information services. In *17th Int. Conf. on Conceptual Modeling*, LNCS 1507. Springer, Berlin, Nov. 1998.

[8] M. Gaedke and G. Gräf. Development and evolution of web-applications using the webcomposition process model. In *Int. Workshop on Web Engineering at the 9th Int. World-Wide Web Conf.*, Amsterdam, The Netherlands, May 2000. Springer.

[9] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns — Elements of Reusable Object-Oriented Software.* Addison-Wesley, 1995.

[10] F. Garzotto, P. Paolini, D. Bolchini, and S. Valenti. "Modeling-by-patterns" of Web applications. In *Int. Workshop on the World-Wide Web and Conceptual Modeling*, LNCS 1727. Springer, Berlin, Nov. 1999.

[11] HPR. Hypermedia design patterns repository. *http://www.designpattern.lu.unisi.ch/index.htm.*

[12] K. Jensen. *Coloured Petri Nets. Basic Concepts, Analysis Methods and Practical Use, Volume I.* Springer, 1997.

[13] J. Lewerenz. *Human-Computer Interaction in Heterogeneous and Dynamic Environments: A Framework for its Conceptual and Automatic Customisation.* Dissertation, Brandenburg Technical Universtity at Cottbus, Germany, 2000.

[14] J. McCarthy. Notes on formalizing context. In *International Joint Conference on Artificial Intelligence*, Chambéry, France, Jan. 1993.

[15] D. Schwabe, G. Rossi, Luiselena, and F. Lyardet. Web design frameworks: An approach to improve reuse in web applications. In *Int. Workshop on Web Engineering at the 9th Int. World-Wide Web Conf.*, Amsterdam, The Netherlands, May 2000. Springer.

[16] B. Thalheim. *Entity-Relationship Modeling - Foundations of Database Technology.* Springer, Heidelberg, 2000.

[17] J. van Benthem. Shifting contexts and changing assertions. In *Computing Natural Language*, pages 51 – 65. C S L I Publications, Apr. 1998.

[18] C. Zimmermann. Relationships between design patterns. In *Pattern Languages in Program Design*, pages 345–364. Addison-Wesley, 1995.

Information Modelling and Knowledge Bases XIV
H. Jaakkola et al. (Eds.)
IOS Press, 2003

Reconstructing Propositional Calculus in Database Semantics

Roland Hausser

Universität Erlangen-Nürnberg
Abteilung Computerlinguistik (CLUE)
rrh@linguistik.uni-erlangen.de

Abstract

Propositional calculus is reconstructed as a simplified version of natural language used for communication. This requires a number of changes with respect to the traditional approach,

First, the method must be changed from the meta-language definitions of old logic to a declarative specification for computer software. Second, the 'realist ontology' of old logic, treating meaning as a direct relation between sentences and the 'world' (defined as a set-theoretic model) must be replaced by a functional system comprising an agent's cognitive operations.[1] Third, the syntax of propositional calculus must be adapted to the time-linear nature of language. Fourth, the truth conditional semantics of propositional calculus must be integrated into the time-linear process of communication.

The latter task requires solutions to the following problems: First, whether an expression like **p** & $\tilde{\mathbf{p}}$... is contradictory cannot be decided in a time-linear interpretation until the continuation, e.g., **p** & $\tilde{\mathbf{p}}$ ∨ **q**..., is known; second, using truth conditions for checking consistency must avoid the SAT problem, which is known to be computationally intractable.

The reconstruction is presented as the declarative specification of a computer program, comprising (i) a data structure for storing formulas of propositional calculus as concatenated propositions, and (ii) LA-grammars for conceptualization, production, and interpretation. Written at a high level of abstraction, the reconstruction provides a comparison between the metalanguage-based handling of truth in traditional logic and the procedural model of communication in Database Semantics.

1 Introduction

In Database Semantics (DBS), communication is modeled as a time-linear procedure for (i) mapping content in the speaker's database into language and (ii) mapping language into equivalent content in the hearer's database. Technically, DBS consists of (i) a data structure called a word bank, and (ii) a time-linear algorithm called LA-grammar for navigating through the content of the data structure and for reading content into and out of the database. Navigation is also used in query-answering and inferencing (Hausser 2001b).

Compared with DBS, propositional calculus is a model of simplicity, and reconstructing it in the powerful DBS system may seem like an academic exercise. Nevertheless, the reconstructing is interesting for the following reasons:

[1] This is partially in concord with Frege's 1879 original design of propositional calculus as a 'formula language for pure thought' (op. cit, p. 6). Our reconstruction differs from Frege, however, in that it includes the procedures of language production and interpretation inside the cognitive agent.

First, it is easier to understand the new (i.e., DBS) after enabling a view from the old (i.e., propositional calculus). Second, old logic guards a treasure trove of important results, and reconstructing its most basic component in DBS is an effective way of importing these results into the new system. Third, the novel use of propositional calculus as a simplified language for communication makes it abundantly clear that basic assumptions of traditional propositional calculus, such as its realist ontology and meta-language-based methodology,[2] must be sacrificed for the success of the reconstruction, thus shedding new light on old logic.

Furthermore, the reconstruction is non-trivial because it requires (i) a time-linear treatment of propositional calculus and (ii) storage in a database. The problem is that logical formulas treat the relations between propositions 'graphically' by depending on the order of their signs – which is unsuitable for a database.[3]

DBS solves this problem by defining the semantic atoms, called proplets,[4] as *feature structures* with attributes which specify the relations to other atoms (bidirectional pointering). By coding conceptual relations between proplets[5] by means of attributes located inside their feature structure, atoms related to each other semantically may be stored arbitrarily far from each other in the database.

In addition, this new data structure is suitable for a time-linear model of communication: First, syntactic-semantic language analysis may be easily defined to code all inter-proplet relations in terms of the attribute-value pairs of individual proplets (hearer mode). Second, once the proplets have been read into the database, they are suitable for time-linear navigation along the proplet connections, providing a model of thought and thus conceptualization (*what to say* in language production). Third, the proplets traversed by the navigation may be copied into a sequence which may be easily mapped into suitable language surfaces (speaker mode).

When DBS is used for communication in natural language, there arise a number of descriptive tasks, such as controlling the navigation, pragmatic interpretation in the speaker mode, pragmatic interpretation in the hearer mode, etc. Propositional calculus as a simplified language of communication, however, is much too basic to require any work in these departments. All that is needed for the task of this paper is a time-linear interpretation, conceptualization, and production of this simple formal language.

Our task is enriched, however, by the main achievement of propositional calculus, namely the truth conditional semantics of the connectives &, ∨, negation, etc. At first, these truth conditions may seem of rather minor interest to DBS with its procedurally defined semantic primitives.[6] What could be the purpose of truth conditions in a functional model of communication between cognitive agents?

The natural answer is: *for checking consistency* by integrating truth conditions into the time-linear navigation through the content of the data structure, here concatenated propositions. But what about SAT (Boolean SATisfiability)? This well-known problem is a textbook example of exponential complexity. If integrating the truth conditions of propositional calculus into a consistency-checking navigation were equivalent to Boolean Satisfiability, then the intended reconstruction would be computationally intractable – and therefore useless.

[2]For a more detailed discussion see Hausser 2001a.

[3]The problem of graphical presentation is shared by theoretical linguistics' use of tree structures, which are likewise unsuitable for databases.

[4]In the case of propositional calculus, the proplets represent the propositional constants.

[5]In natural language, there are proplets for verbs, nouns, and adjectives. Furthermore, there are (i) *intrapropositional relations* between proplets belonging to the same proposition, and *extrapropositional relations* (ii) between the verbs of different propositions, related by conjunctions, and (iii) between the nouns of different propositions, related in terms of identity.

[6]For example, *red* defined as an electromagnetic measurement.

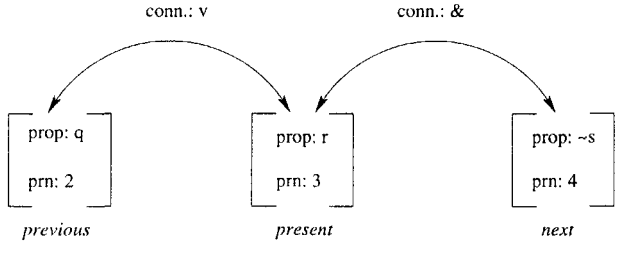
Thus, equivalence with SAT must be avoided and low (preferably linear) complexity must be assured. This is achieved by replacing the arbitrary truth value assignment underlying SAT with a more restricted value assignment based on treating propositions as *database assertions*. In this sense, our reconstruction of predicate calculus in DBS is yet another example where a resounding *no* of old logic is turned into a simple *yes* by adapting old logic's outmoded 'realist' ontology to that of cognitive agents.[7]

2 Data structure of the reconstruction

For simplicity, formulas of propositional calculus are written in conjunctive normal form (CNF).[8] The first step of the reconstruction consists in parsing CNF formulas and translating them automatically into the data structure of a word bank. Thereby, the information represented graphically in traditional logical formulas, e.g., **p** ∨ **q̃** & **r** must be recoded in such a way that the elements of the formula, e.g., **p**, **q̃**, and **r**, as well as the connectives can be stored in arbitrary places in the database.

This is achieved by analyzing propositional constants in the database not as primitives, but as feature structures designed to represent elementary propositions as well as their extrapropositional relations. As an illustration of what needs to be coded into the propositional feature structures consider the following analysis of the propositional constant **r** occurring as part of the formula **q** ∨ **r** & **s̃**:

2.1 BASIC UNITS OF PROPOSITIONAL CONCATENATION



The feature structure representing **r** is the item in the middle, called *present*. It is characterized by two attributes: the name of the proposition (here [name: **r**]) and its proposition number (here [prn: 3]).

Within the formula, **r** has two kinds of external relations, one to the previous **q**, the other to the next **s̃**. The relations to previous and next are each characterized by two attributes: The previous by the connective ∨ and the propositional constant **q**; the next by the connective & and the propositional constant **s̃**.[9]

[7] Other examples of overcoming the limitations of old logic by adopting an adequate ontology are propositional attitudes (Hausser 1999/2001, pp. 395 f.), vagueness and three-valued logic (Hausser 1999/2001, pp. 402 f.), and the Epimenides paradox (Hausser 1999/2001, pp. 383 f., 413 f.).

[8] CNF consists of constants with or without negation, connected by '&' (and) and 'V' (or). The negation of, for example, **p** is written as **p̃**, called *external negation* and paraphrased as *It is not the case that* **p**. CNF formulas are written without parentheses whereby, for example, **p** ∨ **q** ∨ **r** & **s** ∨ **t** ∨ **u** is treated as equivalent to (**p** ∨ **q** ∨ **r**) & (**s** ∨ **t** ∨ **u**). Cf. Hopcroft & Ullman 1979, p. 325.

[9] Extrapropositional relations have the form [n-1 c] for the previous and [c n+1] for the next, where n is the number of the current proposition and c is the connective.

Consider example 2.1: the proposition **r** has the number 3 (n = 3); consequently, the backward connection is [2 ∨], while the forward connection is [& 4].

According to this analysis, the propositional constant **r** within the formula of 2.1 may be characterized by the following feature structure:

2.2 FEATURE STRUCTURE OF A PROPOSITIONAL CONSTANT

prop: **r**	prop $=_{def}$ name of propositional constant
ctn: & $\tilde{\mathbf{s}}$	ctn $=_{def}$ connective to next
ctp: **q** $\vee$	ctp $=_{def}$ connective to previous
prn: **3**	prn $=_{def}$ proposition number

In a lexical *type*, most attributes of the feature structures have the value NIL (represented by space), except for the attribute naming the item, as illustrated below:

2.3 PROPOSITIONAL CONSTANTS AND CONNECTIVES AS LEXICAL ITEMS

propositional constants *connectives*

prop: **q**	prop: **r**	prop: $\tilde{\mathbf{s}}$		
ctn:	ctn:	ctn:	[conn: $\vee$]	[conn: &]
ctp:	ctp:	ctp:		
prn:	prn:	prn:		

During parsing of traditional CNF formulas by a semantically interpreted LA-grammar, these lexical types are (i) assigned to the propositional constants and connectives of the input and (ii) their attributes are filled with proper values by means of copying. For example, the formula **q** $\vee$ **r** & $\tilde{\mathbf{s}}$ represented graphically in 2.1 translates equivalently into the following set of proplets (tokens):

2.4 RECODING **q** $\vee$ **r** & $\tilde{\mathbf{s}}$ AS A SET OF PROPLETS

prop: **q**	prop: **r**	prop: $\tilde{\mathbf{s}}$
ctn: $\vee$ r	ctn: & $\tilde{\mathbf{s}}$	ctn:
ctp:	ctp: q $\vee$	ctp: r &
prn: 2	prn: 3	prn: 4

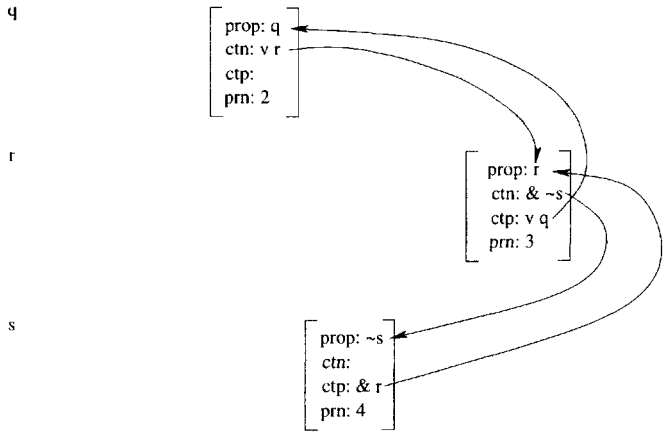
These proplet tokens are suitable for storage in the following data structure:

2.5 DATA STRUCTURE OF A WORD BANK

$type_a$	$proplet_{a1}$ $proplet_{a2}$ $proplet_{a3}$ $proplet_{a4}$, etc.
$type_b$	$proplet_{b1}$ $proplet_{b2}$ $proplet_{b3}$ $proplet_{b4}$, etc.
$type_c$	$proplet_{c1}$ $proplet_{c2}$ $proplet_{c3}$ $proplet_{c4}$, etc.
etc.	

The types are ordered alphabetically, and proplets of the same type are sorted in a line behind their type, as in a network database. Proplets are related to each other in terms of attributes contained in their feature structures.

2.6 BIDIRECTIONAL POINTERING BETWEEN PROPLETS

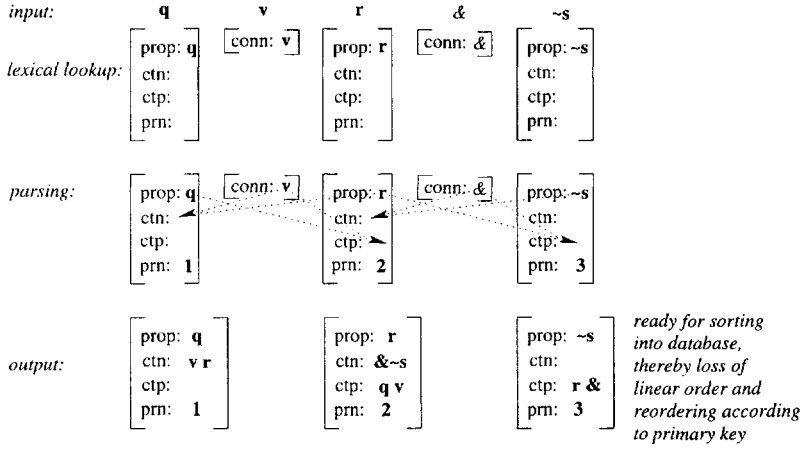


Navigation along the concatenations, depicted here by arrows, is realized technically by means of the retrieval mechanism of the database employed. Take for example proplet **r**: specification of its proposition number, the name of the previous proplet, and the type of conjunction are sufficient not only to characterize the relation between **r** and **q**, but also to retrieve the previous proplet (backward navigation) – and similarly for forward navigation from **r** to **s̃**.

3 Syntactic-semantic interpretation

The semantically interpreted LA-grammar for parsing CNF formulas and translating them into equivalent sets of proplets is called LA-propositional_calculus_interpretation, or *LA-pci* for short. The semantic interpretation of *LA-pci* consists in copying values between feature structures, as illustrated intuitively below:

3.1 TRANSLATING CNF FORMULA INTO PROPLETS



LA-pci consists of a lexicon LX, a variable definition, a set of start states ST_S, a set of rules (here r-1 and r-2), and a set of final states ST_F.

3.2 DEFINITION OF *LA-pci*

$$LX =_{def} \begin{bmatrix}\text{prop: p}\\ \text{ctn:}\\ \text{ctp:}\\ \text{prn:}\end{bmatrix} \begin{bmatrix}\text{prop: } \tilde{\text{p}}\\ \text{ctn:}\\ \text{ctp:}\\ \text{prn:}\end{bmatrix} \begin{bmatrix}\text{prop: q}\\ \text{ctn:}\\ \text{ctp:}\\ \text{prn:}\end{bmatrix} \begin{bmatrix}\text{prop: } \tilde{\text{q}}\\ \text{ctn:}\\ \text{ctp:}\\ \text{prn:}\end{bmatrix} \begin{bmatrix}\text{prop: r}\\ \text{ctn:}\\ \text{ctp:}\\ \text{prn:}\end{bmatrix} \text{, etc. ,}$$

$$[\text{conn: } \vee], [\text{conn: } \&]$$

Variable definition: $\alpha, \beta \in \{p, \tilde{p}, q, \tilde{q}, r, \tilde{r}...\}$, $c \in \{\vee, \&\}$,
$x, y, z =_{def}$ optional values (can be NIL)

$$ST_S =_{def} \{[(\begin{bmatrix}\text{prop: } \alpha\\ \text{ctn:}\\ \text{ctp:}\\ \text{prn:}\end{bmatrix}) \{\text{r-1}\}]\}$$

$$\text{r-1:} \quad \begin{bmatrix}\text{prop: } \alpha\\ \text{ctn:}\\ \text{ctp: z}\\ \text{prn: x}\end{bmatrix} [\text{conn: c}] \Rightarrow \begin{bmatrix}\text{prop: } \alpha\\ \text{ctn: c}\\ \text{ctp: z}\\ \text{prn: x}\end{bmatrix} \{\text{r-2}\}$$

nw-conn $\xrightarrow{c}$ ss-ctn
$copy_{ss}$

$$\text{r-2:} \quad \begin{bmatrix}\text{prop: } \alpha\\ \text{ctn: c}\\ \text{ctp: z}\\ \text{prn: x}\end{bmatrix} \begin{bmatrix}\text{prop: } \beta\\ \text{ctn:}\\ \text{ctp:}\\ \text{prn:}\end{bmatrix} \Rightarrow \begin{bmatrix}\text{prop: } \alpha\\ \text{ctn: c } \beta\\ \text{ctp: z}\\ \text{prn: x}\end{bmatrix} \begin{bmatrix}\text{prop: } \beta\\ \text{ctn:}\\ \text{ctp: } \alpha \text{ c}\\ \text{prn: y}\end{bmatrix} \{\text{r-1}\}$$

ss-prop: $\xrightarrow{p}$ nw-ctp
ss-ctn: $\xrightarrow{a}$ nw-ctp
nw-prop $\xrightarrow{a}$ ss-ctn
$copy_{ss}$, $copy_{nw}$

$$ST_F =_{def} \{[\begin{bmatrix}\text{prop: } \alpha\\ \text{ctn:}\\ \text{ctp: z}\\ \text{prn: x}\end{bmatrix} \text{rp-2}]\}$$

The lexicon LX contains *types* of feature structures (cf. 2.3). They comprise the propositional constants **p**, $\tilde{\mathbf{p}}$, **q**, $\tilde{\mathbf{q}}$, **r**, $\tilde{\mathbf{r}}$, etc., and the connectives $\vee$ and &.

The variable definition specifies variables for defining the rule patterns and the operations on these patterns by the rules of *LA-pci*. The variables α and β are restricted to propositional constants, while the variable c is restricted to connectives.

The set ST_S contains a single state. It indicates that any derivation of *LA-pci* must begin with a propositional constant and the rule r-1.

The rule r-1 attaches a connective to a proposition sequence, e.g., **p q r +** &, by copying the value of the connective into the ctp attribute of the last proposition, e.g., **r**. The rule r-2

attaches a next propositional constant to a proposition sequence, e.g., **p q r + s** by supplying proposition numbers to the ctn and ctp attributes of the last proplet of the current sequence, e.g., **r**, and the next proplet, e.g., **s**.[10]

The working of *LA-pci* is illustrated below with the parsing of **p** ∨ **q** ∨ **r** & **p̃**. The start state of *LA-pci* matches the name of the first proposition **p** and activates rule r-1, which reads ∨ as the next. The control structure of the parser assigns the proposition number 1 to the feature structure of **p**.

3.3 APPLYING r-1 TO '**p** + ∨'

r-1: $\begin{bmatrix}\text{prop: } \alpha\\ \text{ctn:}\\ \text{ctp: z}\\ \text{prn: x}\end{bmatrix}$ [conn: c] ⇒ $\begin{bmatrix}\text{prop: } \alpha\\ \text{ctn: c}\\ \text{ctp: z}\\ \text{prn: x}\end{bmatrix}$ {r-2}

nw-conn —[c]→ ss-ctn

copy_{ss}

$\begin{bmatrix}\text{prop: } \mathbf{p}\\ \text{ctn:}\\ \text{ctp:}\\ \text{prn:}\end{bmatrix}$ [conn: ∨] ⇒ $\begin{bmatrix}\text{prop: } \mathbf{p}\\ \text{ctn: } \vee\\ \text{ctp:}\\ \text{prn: 1}\end{bmatrix}$

The operations of r-1 copy the connective into the ctn slot of the last sentence start proposition, called ss (nw-conn —[e]→ ss-ctn). In the result, the proplet of the connective is discarded by copying only the ss (copy_{ss}). The proposition number 1 is assigned to the resulting ss proplet by the control structure of the system.

The successful application of rule r-1 activates its rule package. Its only rule, r-2, is applied to the new start and the next word **q**, read from the input formula.

3.4 APPLYING r-2 TO '**p** ∨ + **q**'

r-2: $\begin{bmatrix}\text{prop: } \alpha\\ \text{ctn: c}\\ \text{ctp: z}\\ \text{prn: x}\end{bmatrix}$ $\begin{bmatrix}\text{prop: } \beta\\ \text{ctn:}\\ \text{ctp:}\\ \text{prn:}\end{bmatrix}$ ⇒ $\begin{bmatrix}\text{prop: } \alpha\\ \text{ctn: c } \beta\\ \text{ctp: z}\\ \text{prn: x}\end{bmatrix}$ $\begin{bmatrix}\text{prop: } \beta\\ \text{ctn:}\\ \text{ctp: } \alpha\text{ c}\\ \text{prn: y}\end{bmatrix}$ {r-1}

ss-prop: —[c]→ nw-ctp

ss-ctn: —[a]→ nw-ctp

nw-prop —[a]→ ss-ctn

copy_{ss}, copy_{nw}

$\begin{bmatrix}\text{prop: } \mathbf{p}\\ \text{ctn: } \vee\\ \text{ctp:}\\ \text{prn: 1}\end{bmatrix}$ $\begin{bmatrix}\text{prop: } \mathbf{q}\\ \text{ctn:}\\ \text{ctp:}\\ \text{prn:}\end{bmatrix}$ ⇒ $\begin{bmatrix}\text{prop: } \mathbf{p}\\ \text{ctn: } \vee\text{ q}\\ \text{ctp:}\\ \text{prn: 1}\end{bmatrix}$ $\begin{bmatrix}\text{prop: } \mathbf{q}\\ \text{ctn:}\\ \text{ctp: p } \vee\\ \text{prn: 2}\end{bmatrix}$

The operations of r-2 copy the name of ss proplet into the ctp slot of the next proplet, called nw, (ss-prop —[e]→ nw-ctp), the value of the ctn slot of the ss into the ctp slot of the nw (ss-ctn —[a]→ nw-ctp), and the name of the nw into the ctn slot of the ss (nw-prop —[a]→

[10] Syntactically, this LA-grammar is finite state, and its rules do not need to represent the sentence start (cf. Hausser 1989, pp. 167 f.). The current formulation was chosen to resemble the standard format of more powerful systems, based on matching rule patterns with input/output expressions.

ss-ctn). The ss and the nw are both retained in the result ($copy_{ss}$ $copy_{nw}$). The proposition number 2 is assigned to the nw by the control structure.

The successful application of rule r-2 activates its rule package. Its only rule, r-1, is applied to the new start and the next word ∨, read from the input formula.

3.5 APPLYING r-1 TO '**p** ∨ **q** + ∨'

r-1:
$$\begin{bmatrix}\text{prop: } \alpha\\ \text{ctn:}\\ \text{ctp: z}\\ \text{prn: x}\end{bmatrix}\ [\text{conn: c}]\ \Rightarrow\ \begin{bmatrix}\text{prop: } \alpha\\ \text{ctn: c}\\ \text{ctp: z}\\ \text{prn: x}\end{bmatrix}\ \{\text{r-2}\}$$

nw-conn —[c]→ ss-ctn

$copy_{ss}$

$$\begin{bmatrix}\text{prop: } \mathbf{q}\\ \text{ctn:}\\ \text{ctp: p} \vee\\ \text{prn: 2}\end{bmatrix}\ [\text{conn: } \vee]\ \Rightarrow\ \begin{bmatrix}\text{prop: } \mathbf{q}\\ \text{ctn: } \vee\\ \text{ctp: p} \vee\\ \text{prn: 2}\end{bmatrix}$$

Again, the feature structure of the connective is discarded in the output. First, however, the name of the connective is copied into the ctn attribute of **q**.

The successful application of rule r-1 activates its rule package. Its only rule, r-2, is applied to the new start **q** and the next word **r**, read from the input formula.

3.6 APPLYING r-2 TO '**p** ∨ **q** ∨ + **r**'

r-2:
$$\begin{bmatrix}\text{prop: } \alpha\\ \text{ctn: c}\\ \text{ctp: z}\\ \text{prn: x}\end{bmatrix}\ \begin{bmatrix}\text{prop: } \beta\\ \text{ctn:}\\ \text{ctp:}\\ \text{prn:}\end{bmatrix}\ \Rightarrow\ \begin{bmatrix}\text{prop: } \alpha\\ \text{ctn: c } \beta\\ \text{ctp: z}\\ \text{prn: x}\end{bmatrix}\ \begin{bmatrix}\text{prop: } \beta\\ \text{ctn:}\\ \text{ctp: } \alpha\text{ c}\\ \text{prn: y}\end{bmatrix}\ \{\text{r-1}\}$$

ss-prop: —[c]→ nw-ctp

ss-ctn: —[a]→ nw-ctp

nw-prop —[a]→ ss-ctn

$copy_{ss}$, $copy_{nw}$

$$\begin{bmatrix}\text{prop: } \mathbf{q}\\ \text{ctn: } \vee\\ \text{ctp: p} \vee\\ \text{prn: 2}\end{bmatrix}\ \begin{bmatrix}\text{prop: } \mathbf{r}\\ \text{ctn:}\\ \text{ctp:}\\ \text{prn:}\end{bmatrix}\ \Rightarrow\ \begin{bmatrix}\text{prop: } \mathbf{q}\\ \text{ctn: } \vee\text{ r}\\ \text{ctp: p} \vee\\ \text{prn: 2}\end{bmatrix}\ \begin{bmatrix}\text{prop: } \mathbf{r}\\ \text{ctn:}\\ \text{ctp: q} \vee\\ \text{prn: 3}\end{bmatrix}$$

The control structure of the parser assigns the proposition number 3 to the lexical feature structure of **r**. The copying operations of r-2 write the name and ctn value of **q** into the ctp slot of **r**, and add the name of **r** to the ctn slot of **q**. This parsing procedure may be continued indefinitely, turning arbitrarily long CNF sequences of propositional calculus into corresponding sets of proplets.

4 Storage and basic navigation in the database

Parsing formulas of propositional calculus with *LA-pci* results in *sets* of proplets. The parsing is strictly time-linear, analyzing the input from left to right, but the output of the parser is independent of any graphical constraints:

4.1 SEMANTIC REPRESENTATION OF '**p** ∨ **q** ∨ **r** & $\tilde{\mathbf{p}}$ ∨ **s** ∨ **q**'

prop: p	prop: q	prop: r	prop: $\tilde{p}$	prop: s	prop: q
ctn: ∨ q	ctn: ∨ r	ctn: & $\tilde{p}$	ctn: ∨ s	ctn: ∨ q	ctn:
ctp:	ctp: p ∨	ctp: q ∨	ctp: r &	ctp: $\tilde{p}$ ∨	ctp: s ∨
prn: 1	prn: 2	prn: 3	prn: 4	prn: 5	prn: 6

The storage of such a set in the data structure of a word bank consists in adding the completed proplets at the end of their respective token lines (reordering).

4.2 STORING '**p** ∨ **q** ∨ **r** & $\tilde{\mathbf{p}}$ ∨ **s** ∨ **q**' IN A WORD BANK

types	*proplets*		
prop: p	prop: p	ctn: ∨ q	prop: $\tilde{p}$
ctn:	ctp:	prn: 1	ctn: ∨ s
ctp:			ctp: r &
prn:			prn: 4
prop: q	prop: q	prop: q	
ctn:	ctn: ∨ r	ctn:	
ctp:	ctp: p ∨	ctp: s ∨	
prn:	prn: 2	prn: 6	
prop: r	prop: r		
ctn:	ctn: & $\tilde{p}$		
ctp:	ctp: q ∨		
prn:	prn: 3		
prop: s	prop: s		
ctn:	ctn: ∨ q		
ctp:	ctp: $\tilde{p}$ ∨		
prn:	prn: 5		

The purpose of storing proplets in this manner is (i) easy retrieval based on (ii) easy storage plus (iii) easy navigation.

Storage of a proplet is based on the name of the proplet and the temporal order of its arrival. Retrieval is based on the name of the proplet searched for and its proposition number. Navigation from the current proplet to the next is based on the retrieval of the next as specified in the current one.

Simple navigation (e.g., without consistency checking) serves to *activate* the content traversed. In our reconstruction, it is powered by an LA-grammar called LA-propositional_calculus_navigation or *LA-pcn* for short, and defined as follows:

4.3 DEFINITION OF *LA-pcn*

LX $=_{def}$ proplets in a word bank

Variable definition: α, β ε {p, $\tilde{p}$, q, $\tilde{q}$, r, $\tilde{r}$...}; c ε {∨, &}; n, n' ε ℕ;
w, x, y, z $=_{def}$ optional values.

$ST_S =_{def}$ {[prop: α / ctn: c β / ctp: / prn: n] {r-1}], [prop: β / ctn: / ctp: α c / prn: n'] {r-2}]}

$$\text{r-1:}\quad \begin{bmatrix}\text{prop: } \alpha\\ \text{ctn: c } \beta\\ \text{ctp: x}\\ \text{prn: n}\end{bmatrix}\begin{bmatrix}\text{prop: } \beta\\ \text{ctn: y}\\ \text{ctp: } \alpha\text{ c}\\ \text{prn: n'}\end{bmatrix} \Rightarrow \begin{bmatrix}\text{prop: } \beta\\ \text{ctn: y}\\ \text{ctp: } \alpha\text{ c}\\ \text{prn: n'}\end{bmatrix} \{\text{r-1}\} \qquad \textit{(forward navigation)}$$

$$\text{r-2:}\quad \begin{bmatrix}\text{prop: } \beta\\ \text{ctn: y}\\ \text{ctp: } \alpha\text{ c}\\ \text{prn: n'}\end{bmatrix}\begin{bmatrix}\text{prop: } \alpha\\ \text{ctn: c } \beta\\ \text{ctp: x}\\ \text{prn: n}\end{bmatrix} \Rightarrow \begin{bmatrix}\text{prop: } \alpha\\ \text{ctn: c } \beta\\ \text{ctp: x}\\ \text{prn: n}\end{bmatrix} \{\text{r-2}\} \qquad \textit{(backward navigation)}$$

$$ST_F =_{def} \{[\begin{bmatrix}\text{prop: } \beta\\ \text{ctn:}\\ \text{ctp: } \alpha\text{ c}\\ \text{prn: n'}\end{bmatrix} rp_1\ ],\ [\begin{bmatrix}\text{prop: } \alpha\\ \text{ctn: c } \beta\\ \text{ctp:}\\ \text{prn: n}\end{bmatrix} rp_2]\}$$

The operations of *LA-pcn* are so simple that they can be specified solely in terms of patterns – without any copying operations. Consider the following example of forward navigation:

4.4 EXAMPLE OF *LA-pcn* FORWARD NAVIGATION

level of rule pattern

$$\text{r-1:}\quad \begin{bmatrix}\text{prop: } \alpha\\ \text{ctn: c } \beta\\ \text{ctp: x}\\ \text{prn: n}\end{bmatrix}\begin{bmatrix}\text{prop: } \beta\\ \text{ctn: y}\\ \text{ctp: } \alpha\text{ c}\\ \text{prn: n'}\end{bmatrix} \Rightarrow \begin{bmatrix}\text{prop: } \beta\\ \text{ctn: y}\\ \text{ctp: } \alpha\text{ c}\\ \text{prn: n'}\end{bmatrix} \{\text{r-1}\}$$

match *search* *next proplet*

level of dbs proplets

$$\begin{bmatrix}\text{prop: q}\\ \text{ctn: } \vee\text{ r}\\ \text{ctp: p } \vee\\ \text{prn: 2}\end{bmatrix}\begin{bmatrix}\text{prop: r}\\ \text{ctn: } \&\ \tilde{\text{p}}\\ \text{ctp: q } \vee\\ \text{prn: 3}\end{bmatrix} \Rightarrow \begin{bmatrix}\text{prop: r}\\ \text{ctn: } \&\ \tilde{\text{p}}\\ \text{ctp: q } \vee\\ \text{prn: 3}\end{bmatrix}$$

The upper level shows rule r-1 with its pattern for ss, nw, and new ss, while the lower level shows matching proplets in the word bank 4.2.

The ss pattern ([prop: α]) of the rule is matched with the first proplet of the **q** token line in the word bank, thus binding the variables α, c, β, and n to the values **q**, ∨, **r**, and 2, respectively. These values are also assigned to the corresponding variables in the nw pattern (whereby n' = 3, cf. footnote 7 in Section 2), thus enabling retrieval of proplet **r**, which is returned as the new ss (output).

Backward navigation works the same way, except that the continuation attribute is the ctp of the start, instead of the ctn.

4.5 EXAMPLE OF *LA-pcn* BACKWARD NAVIGATION

level of rule pattern

$$\text{r-2:}\quad \begin{bmatrix}\text{prop: } \beta\\ \text{ctn: y}\\ \text{ctp: } \alpha\text{ c}\\ \text{prn: n'}\end{bmatrix}\begin{bmatrix}\text{prop: } \alpha\\ \text{ctn: c } \beta\\ \text{ctp: x}\\ \text{prn: n}\end{bmatrix} \Rightarrow \begin{bmatrix}\text{prop: } \alpha\\ \text{ctn: c } \beta\\ \text{ctp: x}\\ \text{prn: n}\end{bmatrix} \{\text{r-2}\}$$

match *search* *next proplet*

level of dbs proplets

$$\begin{bmatrix}\text{prop: } \tilde{\text{p}}\\ \text{ctn: } \vee\text{ s}\\ \text{ctp: r } \&\\ \text{prn: 4}\end{bmatrix}\begin{bmatrix}\text{prop: r}\\ \text{ctn: } \&\ \tilde{\text{p}}\\ \text{ctp: q } \vee\\ \text{prn: 3}\end{bmatrix} \Rightarrow \begin{bmatrix}\text{prop: r}\\ \text{ctn: } \&\ \tilde{\text{p}}\\ \text{ctp: q } \vee\\ \text{prn: 3}\end{bmatrix}$$

The operations of *LA-pcn* do not modify the input to the rules, – in contradistinction to the copying operations of *LA-pci* as defined in 3.2.

5 Propositional calculus consistency navigation

The next step of the reconstruction consists in integrating the truth conditions of propositional calculus into the navigation algorithm of *LA-pcn*. For this, the different ontologies of predicate calculus and its reconstruction must be taken into account: propositional calculus is a metalanguage-based theory relating propositions and the world, while the reconstruction is a procedural theory treating propositions as database assertions.

This difference is crucial for avoiding equivalence between consistency navigation and the problem of SAT mentioned at the end of the Introduction. The SAT problem is to determine for arbitrary CNF formulas whether there exists a value assignment which makes it true. SAT is based on the following assumptions:

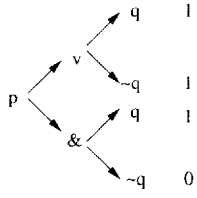
1. *Arbitrary values*: for checking consistency, propositional constants may be assigned arbitrary values, e.g., **p** may be 1 or 0.
2. *Coreference*: in a formula, e.g., ...**p**....**p**..., different occurrences of **p** must have the same truth value.

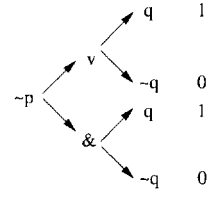
As a consequence, the SAT algorithm[11] must keep track of previous truth value assignments in a formula, which is why determining satisfiability for arbitrary formulas is exponential in the worst case and $\mathcal{NP}$-complete.

For a database, however, the above assumptions are not realistic because one would not randomly switch truth values to see whether there exists a value assignment satisfying a sequence of concatenated propositions. Instead, some propositions in the database are asserted to be 1 (true), e.g., **p**, while others are asserted to be 0 (false), e.g., $\tilde{\mathbf{q}}$.

As a result there is no need to remember which value was assigned to earlier occurrences of, for example, **p** in a formula, and consistency checking may be described in terms of the following continuations:

5.1 Branching structure of '&' and 'V' with truth values





On the left-hand side, the navigation starts with **p**. The first continuation alternative is whether the connective is V or &. For each connective, the second continuation alternative is whether the next proposition is unnegated (**q**) or negated ($\tilde{\mathbf{q}}$).

On the right-hand side, the navigation starts with $\tilde{\mathbf{p}}$. The subsequent continuation alternatives are the same as those on the left hand side. Because of the different starts, however, the truth values resulting on the left differ from those resulting on the right-hand side.

[11] An LA-grammar for SAT is defined in Hausser 1992, footnote 19. See also Hausser 1989, pp. 157 f., where SAT is treated as a lexically ambiguous LA-grammar, and Hausser 1999/2001, p. 218.

The continuation patterns 5.1 may be interpreted in the traditional way by focusing on the connectives: the upper half of 5.1 may be reassembled into the traditional truth table for ∨, the lower half into that for &. But what about tautologies and contradictions?

Consider the following comparison of the traditional value assignments in predicate calculus and its reconstruction in Database Semantics:

5.2 TAUTOLOGIES, CONTRADICTIONS, AND CONTINGENCIES IN DBS

	tautology		*contradiction*		*contingent complex propositions*							
	p ∨ $\tilde{\mathbf{p}}$		**p** & $\tilde{\mathbf{p}}$		**p** ∨ $\tilde{\mathbf{q}}$		**p** & $\tilde{\mathbf{q}}$		**p** ∨ **q**		**p** & **q**	
traditional logic:	1	0	1	0	1	1	1	1	1	1	1	1
	0	1	0	1	1	0	1	0	1	0	1	0
					0	1	0	1	0	1	0	1
					0	0	0	0	0	0	0	0
DBS *reconstruction*:	1	0	1	0	1	0	1	0	1	1	1	1

In traditional logic, a tautology like **p** ∨ $\tilde{\mathbf{p}}$ and a contradiction like **p** & $\tilde{\mathbf{p}}$ each has two value assignments, while a contingent proposition like **p** ∨ $\tilde{\mathbf{q}}$ has four.[12]

From the purpose of checking the consistency of a DBS navigation, these multiple value assignments are redundant. Furthermore, all possible truth conditional constellations of propositional calculus may be expressed equivalently by assigning truth values in accordance with whether or not a propositional constant carries external negation. For example, instead of choosing between the four possible values (1 0), (1 1), (0 1), and (0 0) as assignments to (**p** ∨ $\tilde{\mathbf{q}}$), the same truth conditional constellations may be expressed by choosing between the DBS assertions (**p** ∨ $\tilde{\mathbf{q}}$), (**p** ∨ **q**), ($\tilde{\mathbf{p}}$ ∨ **q**), and ($\tilde{\mathbf{p}}$ ∨ $\tilde{\mathbf{q}}$).

The only aspect of traditional logic which is somewhat reduced in the DBS reconstruction are the truth conditions of tautologies and contradictions as compared to contingent propositions. For example, **p** ∨ $\tilde{\mathbf{p}}$ (tautology) and **p** ∨ $\tilde{\mathbf{q}}$ (contingent) are evaluated the same, i.e., 1, because DBS does not assign 0 to **p**. Similarly, **p** & $\tilde{\mathbf{p}}$ (contradiction) and **p** & $\tilde{\mathbf{q}}$ (contingent) are evaluated the same, i.e., 0, because DBS does not assign 1 to $\tilde{\mathbf{q}}$.[13]

With the complexity issue raised by SAT out of the way, let us consider how to integrate the truth conditions of propositional calculus into a consistency navigation. When traversing concatenated propositions like **p** ∨ $\tilde{\mathbf{q}}$ ∨ **r** & **s** ∨ **t**, etc., the navigation continues as long as (i) the concatenated assertions are consistent and (ii) possible continuations are available. When the end of the concatenation is reached, the navigation terminates in a legal final state. It terminates in an error, however, as soon as the concatenated propositions traversed turn out to be inconsistent.

The tricky question is how to handle transitions in which the truth value is undetermined (#) because the remainder of the formula has not yet been read. Consider, for example, **p** ∨ **q** ∨ **r** & $\tilde{\mathbf{p}}$ ∨ $\tilde{\mathbf{q}}$ ∨ **r**. The navigation traverses the concatenated propositions from left to right. The initial propositions **p** ∨ **q** ∨ **r** are all 1. Then the navigation reaches & $\tilde{\mathbf{p}}$. At this point, no bivalent truth value can be assigned, for which reason the navigation goes into the state #

[12] The truth values of the complex propositions, derived via the standard truth tables from the values assigned to the elementary propositions, are omitted here to avoid cluttering of the presentation.

[13] In DBS semantics, tautologies and contradictions do not deserve special treatment because they are neither particularly interesting nor desirable as content in a database. Old logic's quasi-alchemist dream of deriving all philosophical truth from the tautologies is not considered viable in DBS.

(undetermined) and continues until the following propositions decide whether the navigation returns from state # to state 1, or terminates in state 0.

The first possibility is illustrated by the following example:

5.3 CONSISTENCY NAVIGATION IN UNDEFINED TERRITORY (1)

1 **p** ∨ **q** ∨ **r** & $\tilde{\mathbf{p}}$ ∨ $\tilde{\mathbf{q}}$ ∨ **r** ...

2 1

3 1

4 #

5 #

6 1

The algorithm can only decide at the very end (line 6) whether the value # (line 4 and 5) will revert to 1, or turn to 0, bringing the navigation to a halt. If the last constant is unnegated, as the **r** shown above, the resulting state is 1. If **r** were to be replaced by $\tilde{\mathbf{r}}$, the resulting state would be 0. Consider the following example:

5.4 CONSISTENCY NAVIGATION IN UNDEFINED TERRITORY (0)

1 **p** & $\tilde{\mathbf{q}}$ ∨ $\tilde{\mathbf{r}}$ ∨ $\tilde{\mathbf{s}}$ & $\tilde{\mathbf{t}}$ ...

2 #

3 #

4 #

5 #

6 0

Here, the truth value of the whole sequence is decided by the last shown connective and propositional constant: if we changed the last connective from & to ∨ and the last proposition from $\tilde{t}$ to t, the state # resulting in line 6 would be 1 rather than 0.

6 Navigation algorithm of the reconstruction

A CNF formula may begin in eight different ways, corresponding to the start states of LA-propositional_calculus_consistency_navigation, or *LA-pccn* for short, to be defined in 6.3 below. These eight states may be characterized intuitively as follows:

6.1 INTUITIVE CHARACTERIZATION OF THE START STATES OF *LA-pccn*

	start	*value*		*start*	*value*
1.	**p** ∨ **q**	1	5.	$\tilde{\mathbf{p}}$ ∨ $\tilde{\mathbf{q}}$	#
2.	**p** ∨ $\tilde{\mathbf{q}}$	1	6.	**p** & $\tilde{\mathbf{q}}$	#
3.	$\tilde{\mathbf{p}}$ ∨ **q**	1	7.	$\tilde{\mathbf{p}}$ & **q**	0
4.	**p** & **q**	1	8.	$\tilde{\mathbf{p}}$ & $\tilde{\mathbf{q}}$	0

Start states 1–6 result in a continuing navigation, while states 7 and 8 terminate the navigation before it even starts.

From the start states 1–6, the navigation may be continued by means of five LA-grammar rules, which may be characterized intuitively as follows:

6.2 INTUITIVE CHARACTERIZATION OF *LA-pccn*'S CONTINUATION RULES

rule name	*connective*	*next*	*rule package*
r1-(11)	∨	**q**	{r1-(11), r2-(1#)}
	∨	$\tilde{\mathbf{q}}$	
	&	**q**	
r2-(1#)	&	$\tilde{\mathbf{q}}$	{r3-(#1), r4-(##), r5-(#0)}
r3-(#1)	∨	**q**	{r1-(11), r2-(1#)}
r4-(##)	∨	$\tilde{\mathbf{q}}$	{r3-(#1), r4-(##), r5-(#0)}
r5-(#0)	&	**q**	{ }
	&	$\tilde{\mathbf{q}}$	

Rule r1-(11) combines three continuations which maintain state 1. Rule r2-(1#) changes from state 1 into #. Rule r3-(#1) changes from state # into 1. Rule r4-(##) maintains state #. Rule r5-(#0) combines two continuations which change from state # into termination with an error (empty rule package).

To express the patterns of the states and rules, the variable definition of *LA-pccn* specifies the following restricted variables: α for unnegated start proplets, β for unnegated next proplets, γ for negated start proplets, and δ for negated next proplets. Also, for a parsimonious formulation of the rules, the variables κ and μ are defined for proplets which may be unnegated or negated.

6.3 DEFINITION OF *LA-pccn* (PART 1, FORWARD NAVIGATION)

LX $=_{def}$ proplets in a word bank.
Variable definition: α, β ε {p, q, r,...}; γ, δ ε {$\tilde{p}$, $\tilde{q}$, $\tilde{r}$...}; κ, μ ε {p, $\tilde{p}$, q, $\tilde{q}$, r, $\tilde{r}$...};
c ε {∨, &}; n, n' ε ℕ; *w, x, y, z* = optional values (may be NIL).

$$ST_S =_{def} \{ [\begin{bmatrix} \text{prop: } \kappa \\ \text{ctn: c } \mu \\ \text{ctp:} \\ \text{prn: n} \end{bmatrix}, \{\text{r1-(11), r2-(1\#), r3-(\#1), r4-(\#\#), r5-(\#0)}\}] \}$$

r1-(11): $\begin{bmatrix}\text{prop: } \kappa\\ \text{ctn: } c\ \mu\\ \text{ctp: } w\\ \text{prn: } n\end{bmatrix}\begin{bmatrix}\text{prop: } \mu\\ \text{ctn: } x\\ \text{ctp: } \kappa\ c\\ n'\end{bmatrix} \Rightarrow \begin{bmatrix}\text{prop: } \mu\\ \text{ctn: } x\\ \text{ctp: } \kappa\ c\\ n'\end{bmatrix}$ {r1-(11), r2-(1#)}
where c = & and κ and $\mu \in$ {**p, q, r, s, t,** etc.},
or c = ∨ and κ or $\mu \in$ {**p, q, r, s, t,** etc.}

r2-(1#): $\begin{bmatrix}\text{prop: } \kappa\\ \text{ctn: } \&\ \delta\\ \text{ctp: } w\\ \text{prn: } n\end{bmatrix}\begin{bmatrix}\text{prop: } \delta\\ \text{ctn: } x\\ \text{ctp: } \kappa\ \&\\ \text{prn: } n'\end{bmatrix} \Rightarrow \begin{bmatrix}\text{prop: } \delta\\ \text{ctn: } x\\ \text{ctp: } \kappa\ \&\\ \text{prn: } n'\end{bmatrix}$ {r3-(#1), r4-(##), r5-(#0)}

r3-(#1): $\begin{bmatrix}\text{prop: } \kappa\\ \text{ctn: } \vee\ \beta\\ \text{ctp: } w\\ \text{prn: } n\end{bmatrix}\begin{bmatrix}\text{prop: } \beta\\ \text{ctn: } y\\ \text{ctp: } \kappa\ \vee\\ \text{prn: } n'\end{bmatrix} \Rightarrow \begin{bmatrix}\text{prop: } \beta\\ \text{ctn: } y\\ \text{ctp: } \kappa\ \vee\\ \text{prn: } n'\end{bmatrix}$ {r1-(11), r2-(1#)}

r4-(##): $\begin{bmatrix}\text{prop: } \kappa\\ \text{ctn: } \vee\ \delta\\ \text{ctp: } w\\ \text{prn: } n\end{bmatrix}\begin{bmatrix}\text{prop: } \delta\\ \text{ctn: } z\\ \text{ctp: } \kappa\ \vee\\ \text{prn: } n'\end{bmatrix} \Rightarrow \begin{bmatrix}\text{prop: } \delta\\ \text{ctn: } z\\ \text{ctp: } \kappa\ \vee\\ \text{prn: } n'\end{bmatrix}$ {r3-(#1), r4-(##), r5-(#0)}

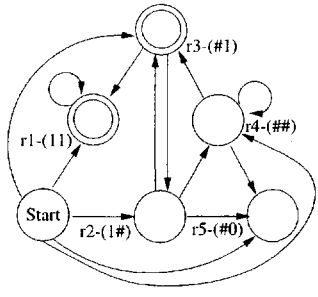
r5-(#0): $\begin{bmatrix}\text{prop: } \kappa\\ \text{ctn: } \&\ \mu\\ \text{ctp: } w\\ \text{prn: } n\end{bmatrix}\begin{bmatrix}\text{prop: } \mu\\ \text{ctn: } z\\ \text{ctp: } \kappa\ \&\\ \text{prn: } n'\end{bmatrix} \Rightarrow \begin{bmatrix}\text{prop: } \mu\\ \text{ctn: } z\\ \text{ctp: } \kappa\ \&\\ \text{prn: } n'\end{bmatrix}$ { }

$ST_F =_{def}$ { [$\begin{bmatrix}\text{prop: } \kappa\\ \text{ctn:}\\ \text{ctp: } \mu\ c\\ \text{prn: } n\end{bmatrix}$, rp1-(11)], [$\begin{bmatrix}\text{prop: } \kappa\\ \text{ctn:}\\ \text{ctp: } \mu\ \vee\\ \text{prn: } n\end{bmatrix}$, rp3-(#1)]

Final states showing the navigation to be consistent are defined by the application of rules r1-(11) or r3-(#1) in conjunction with suitable final proplets characterized by their empty continuation attributes. For backward navigation, the start states, rules, and final states of *LA-pccn* have to be to complemented by a corresponding set of definitions.

The rules of *LA-pccn* calling rule packages constitute a finite state transition network, as in all LA-grammars:[14]

6.4 FINITE STATE TRANSITION NETWORK OF *LA-pccn*



[14]Other examples of finite state transition networks, characterizing LA-grammars for $a^k b^k c^k$ and for English and German, may be found in Hausser 1999/2001 on pp. 189, 365, 335, 333, and 338.

Arrows going into a state correspond to applications of the same rule, though from different rule packages. Arrows going out of a state correspond to different rules in the same rule package. Final states are indicated by double circles.

Next consider a schematic derivation, followed by three rule applications:

6.5 SCHEMATIC DERIVATION

rules:		r1-(11)		r2-(1#)		r4-(##)	r3-(#1)		r2-(1#)		r5-(#0)	
continuations:	**p**	∨	**q**	&	**r̃**	∨	**s̃**	∨	**t**	&	**g̃**	& **h**
values:		1		#		#	1		#		0	

6.6 APPLICATION OF R1-(11) IN THE FIRST COMBINATION

r1-(11):

prop: κ ctn: c μ ctp: w prn: n	prop: μ ctn: x ctp: κ c n'	⇒	prop: μ ctn: x ctp: κ c n'	{r1-(11), r2-(1#)}
prop: p ctn: ∨ q ctp: prn: 1	prop: q ctn: & r̃ ctp: p ∨ prn: 2	⇒	prop: q ctn: & r̃ ctp: p ∨ prn: 2	

6.7 APPLICATION OF R2-(1#) IN THE SECOND COMBINATION

r2-(1#):

prop: κ ctn: & δ ctp: w prn: n	prop: δ ctn: x ctp: κ & prn: n'	⇒	prop: δ ctn: x ctp: κ & prn: n'	{r3-(#1), r4-(##), r5-(#0)}
prop: q ctn: & r̃ ctp: p ∨ prn: 2	prop: r̃ ctn: ∨ s̃ ctp: q & prn: 3	⇒	prop: r̃ ctn: ∨ s̃ ctp: q & prn: 3	

6.8 APPLICATION OF R4-(##) IN THE THIRD COMBINATION

r4-(##):

prop: κ ctn: ∨ δ ctp: w prn: n	prop: δ ctn: z ctp: κ ∨ prn: n'	⇒	prop: δ ctn: z ctp: κ ∨ prn: n'	{r3-(#1), r4-(##), r5-(#0)}
prop: r̃ ctn: ∨ s̃ ctp: q & prn: 3	prop: s̃ ctn: ∨ t ctp: ∨ r̃ prn: 4	⇒	prop: s̃ ctn: ∨ t ctp: ∨ r̃ prn: 4	

etc.

The upper level of 6.6 – 6.8 represents the rules, while the lower level shows the matching proplets of the database. The remaining rule applications are omitted.

7 Communicating with propositional calculus

During language production, the navigation driven by *LA-pccn* serves as the speaker's conceptualization. The conceptualization is mapped into language by (i) copying the proplets traversed into a buffer and (ii) mapping the buffer sequence of proplets into an equivalent CNF formula. Procedure (ii) is based on the following LA-grammar, called LA-propositional_calculus_output, or *LA-pco* for short:

7.1 DEFINITION OF *LA-pco*

LX $=_{def}$ proplets in a word bank
Variable definition: $\alpha, \beta \in \{p, \tilde{p}, q, \tilde{q}, r, \tilde{r}...\}$; $c \in \{\vee, \&\}$; $n, n' \in \mathbb{N}$;
w, x, y, z = optional values (may be NIL).

$ST_S =_{def}$ {[[prop: α / ctn: cβ / ctp: / prn: n] {r-1}], [[prop: α / ctn: c β / ctp: / prn: n] {r-2}]}

r-1: [prop: α / ctn: c β / ctp: w / prn: n] [prop: β / ctn: y / ctp: α c / n'] ⇒ [sur: α] [sur: c] [prop: β / ctn: y / ctp: α c / n'] {r-1, r-2}

r-2: [prop: α / ctn: c β / ctp: w / prn: n] [prop: β / ctn: / ctp: α c / prn: n'] ⇒ [sur: α] [sur: c] [sur: β] { }

$ST_F =_{def}$ {[[sur: α] [sur: c] [sur: β], rp-2]}

Language production may be illustrated by treating 4.1 as a buffer sequence of proplets resulting from an *LA-pccn* navigation through 4.2. The application of rule r-1 of *LA-pco* to the first two proplets of the sequence has the following form:

7.2 ILLUSTRATING APPLICATION OF *LA-pco* RULE R-1

r-1: [prop: α / ctn: c β / ctp: w / prn: n] [prop: β / ctn: y / ctp: α c / n'] ⇒ [sur: α] [sur: c] [prop: β / ctn: y / ctp: α c / n'] {r-1, r-2}

[prop: p / ctn: ∨ q / ctp: / prn: 1] [prop: q / ctn: ∨ r / ctp: p ∨ / prn: 2] ⇒ [sur: p] [sur: ∨] [prop: q / ctn: ∨ r / ctp: p ∨ / prn: 2]

This rule application (i) realizes the name of the first proplet and (ii) its first ctn value, thus reversing the process of connective absorption shown in 3.3 and 3.5.

A similar effect results from the following application of rule r-2 of *LA-pco* to the last two proplets of 4.1:

7.3 ILLUSTRATING APPLICATION OF *LA-pco* RULE R-2

r-2:

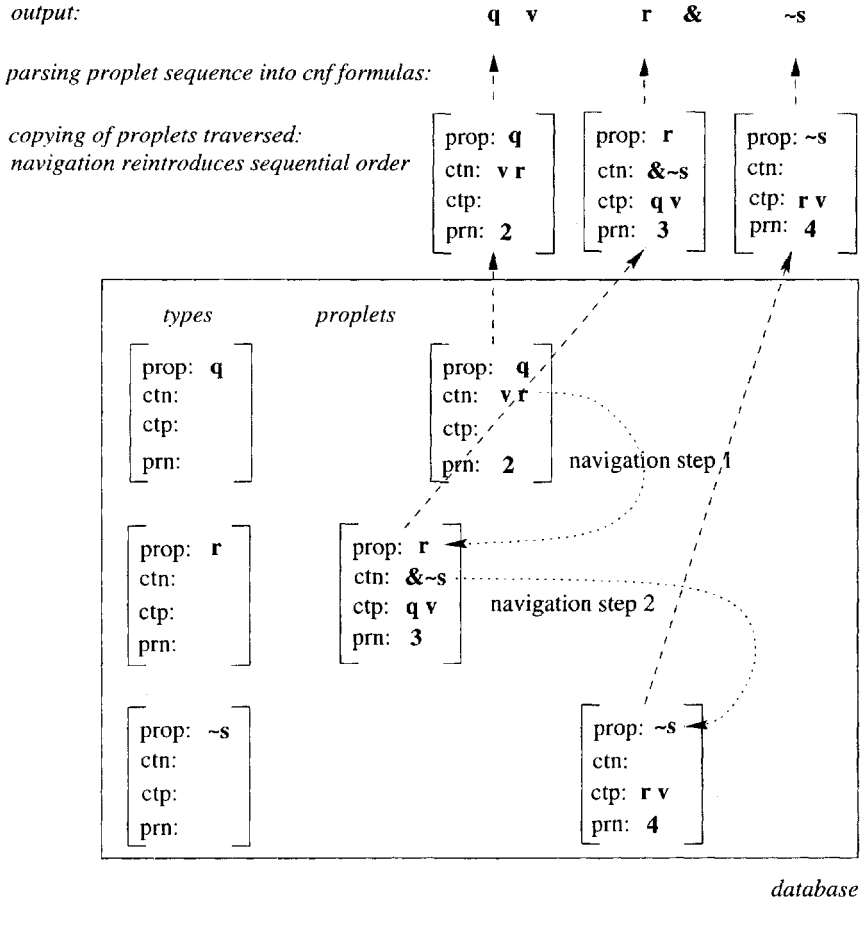
prop: α	prop: β	⇒	sur: α	sur: c	sur: β	{ }
ctn: c β	ctn:					
ctp: w	ctp: α c					
prn: n	prn: n'					

prop: s	prop: q	⇒	sur: s	sur: ∨	sur: q
ctn: ∨ q	ctn:				
ctp: p̃ ∨	ctp: s ∨				
prn: 5	prn: 6				

Rule r-2 realizes the name of the first proplet, its ctn connective, and the name of the second proplet, ending the derivation with an empty rule package.

The process of navigating with LA-pccn, copying the proplets traversed into a buffer, and realizing the buffer sequence as equivalent cnf surfaces is shown schematically in 7.4:

7.4 STORAGE OF PROPLETS AS WELL AS NAVIGATION AND OUTPUT

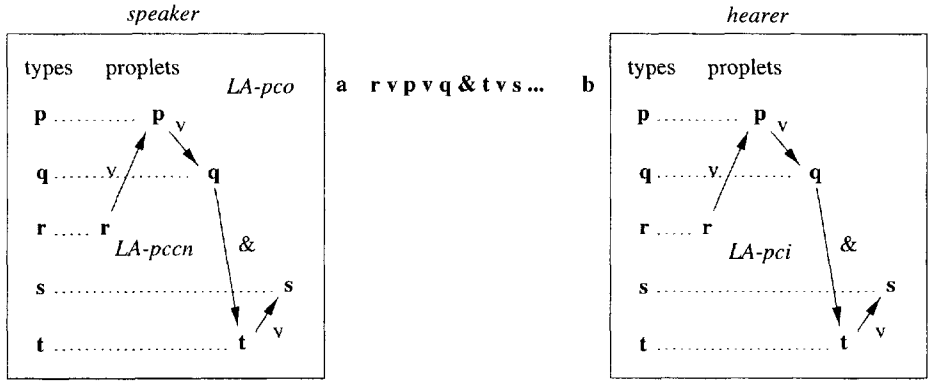


8 Summary

In DBS, communication between a speaker and a hearer is successful if a database content mapped into language by the speaker is reconstructed equivalently in the database of the hearer. Accordingly, the reconstruction of propositional calculus in DBS requires (i) a data structure for storing CNF formulas as concatenated propositions (common to the speaker and hearer), (ii) a procedure for mapping concatenated propositions into CNF formulas (speaker mode) and (iii) a procedure for mapping CNF formulas into concatenated propositions (hearer mode).

This interaction between speaker and hearer has been reconstructed using propositional calculus as a simplified form of natural language.

8.1 PROPOSITIONAL CALCULUS AS A COMMUNICATION LANGUAGE



The process of communication depicted in 8.1 is based on the data structure of a word bank, consisting of alphabetically ordered token lines (cf. 2.5 and 4.2), and three LA-grammars, called *LA-pccn* (defined in 6.3), *LA-pco* (defined in 7.1), and *LA-pci* (defined in 3.2).

In the speaker's database, CNF formulas are stored as concatenated propositions, each represented as a feature structure called proplet. This content is activated by navigating along the concatenations, using *LA-pccn* as the motor algorithm. During the navigation, consistency of the concatenated propositions traversed is checked. This process models a speaker's monitoring whether or not what he or she is currently thinking is consistent.

In addition, the navigation provides the conceptualization for language production. Thereby, the navigation is mapped into a corresponding CNF expression by (i) copying the proplets traversed into a buffer and (ii) mapping the buffer sequence into suitable surfaces. Procedure (ii) is provided by *LA-pco*.

The hearer parses the incoming surfaces using *LA-pci*. The semantic interpretation of *LA-pci* results in a set of proplets which are stored in the hearer's database. The interaction of *LA-pccn*, *LA-pco*, and *LA-pci* ensures that the content mapped by the speaker into CNF formulas is reconstructed by the hearer into an equivalent content, resulting in successful communication.

9 Outlook

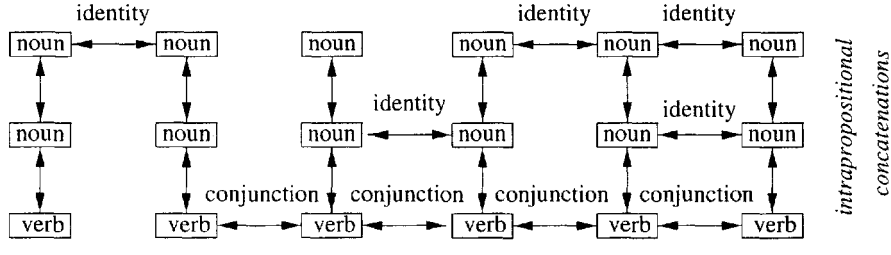
As a language, propositional calculus is especially simple in that it uses only one type of extrapropositional relations, namely the logical connectives. As a consequence, there are

only two types of navigation, namely forward and backward.[15]

This situation changes significantly, however, when we move from propositional calculus to predicate calculus with its distinction between functors and arguments. These are represented as proplets for verbs and nouns, whereby the logical connectives are reconstructed as conjunctions between the proplets of verbs. In addition, a new kind of extrapropositional relation is defined, namely the identity relation between the proplets of nouns.

Consider the following schematic presentation of the railroad system of predicate calculus, assuming only elementary nouns and two-place verbs:

9.1 SCHEMATIC RAILROAD SYSTEM OF PREDICATE CALCULUS



extrapropositional concatenations

The nouns and two-place verbs forming propositions are distributed all over the data structure. In line with the coding technique of DBS, all intra- and extrapropositional concatenations are established solely in terms of attributes, i.e., proplet name, proposition number, identity, and conjunction.

Extrapropositional relations are shown in 9.1 as horizontal double arrows, while intrapropositional relations are shown as vertical double arrows. Each proposition of predicate calculus may have up to three extrapropositional relations, one based on verbal conjunction and two based on nominal identity.

As a consequence, the navigation through a data structure containing proplets of predicate calculus is confronted at each point with a choice between several possible continuations – in contradistinction to propositional calculus. This choice may be made either at random or by constructing a suitable control structure. The latter requires a kind of component which is indispensable in DBS for a procedural definition of semantic primitives, but missing in old logic, namely contextual (or non-verbal) recognition and action.

10 Conclusion

The reconstruction of propositional calculus and the prospective reconstruction of predicate calculus may be regarded conceptually as a step by step upscaling from well-known formal languages to a model of natural language communication as defined in DBS. This approach is well-suited to integrate the important results of traditional logic into DBS. It also serves to highlight the differences between the metalanguage-based approach to natural language meaning in terms of truth conditions (Montague 1974) and the procedural approach of Database Semantics, designed as a functional model of communication based on the cognitive operations of agents which use language as speakers and hearers.

[15]For simplicity, language production based on *LA-pco* is defined here only for forward navigation.

References

Frege, G. (1879) "Begriffsschrift, a formula language, modeled upon that of arithmetic, for pure thought," in J. van Heijenoort (ed.) *Frege and Gödel, Two Fundamental Texts in Mathematical Logic*, translated into English by S. Bauer-Mengelberg, Harvard University Press, Cambridge, Mass.

Hausser, R. (1989) *Computation of Language, An Essay on Syntax, Semantics and Pragmatics in Natural Man-Machine Communication*, Springer-Verlag, Berlin-New York.

Hausser, R. (1992) "Complexity in Left-Associative Grammar," *Theoretical Computer Science*, Vol. 106.2:283-308, Elsevier, Dordrecht.

Hausser, R. (1999/2001) *Foundations of Computational Linguistics: Human-Computer Communication in Natural Language*, Springer-Verlag, Berlin-New York.

Hausser, R. (2001a) "The Four Basic Ontologies of Semantic Interpretation," in H. Kangassalo et al. (eds) *Information Modeling and Knowledge Bases XII*, IOS Press Ohmsha, Amsterdam.

Hausser, R. (2001b) "Database Semantics for Natural Language," *Artificial Intelligence*, Vol. 130.1:27–74, Elsevier, Dordrecht.

Hopcroft, J.E. & Ullman, J.D. (1979) *Introduction to Automata Theory, Languages, and Computation*, Addison-Wesley, Reading, Mass.

Montague, R. (1974) *Formal Philosophy*, Yale Univerity Press, New Haven.

Author Index