data and knowledge (DS-2)

edited by

r. a. meersman and a. c. sernadas

IFIP

north-holland

DATA AND KNOWLEDGE (DS-2)

Proceedings of the Second IFIP 2.6 Working Conference on Database Semantics, 'Data and Knowledge' (DS-2)
Albufeira, Portugal, 3-7 November, 1986

edited by

R. A. MEERSMAN
INFOLAB, Tilburg University
Tilburg, The Netherlands

A. C. SERNADAS
INESC
Lisbon, Portugal

Dedicated to the Memory of
John G. Gaschnig

1988

NORTH HOLLAND
AMSTERDAM • NEW YORK • OXFORD • TOKYO

ISBN: 0 444 70528 7

Published by:

ELSEVIER SCIENCE PUBLISHERS B.V.
P.O. Box 103
1000 AC Amsterdam
The Netherlands

Sole distributors for the U.S.A. and Canada:

ELSEVIER SCIENCE PUBLISHING COMPANY, INC.
655 Avenue of the Americas
New York, N.Y. 10010
U.S.A.

Library of Congress Cataloging-in-Publication Data

IFIP 2.6 Working Conference on Database Semantics "Data and Knowledge"
(2nd : 1986 : Albufeira, Portugal)
Data and knowledge (DS-2) : proceedings of the Second IFIP 2.6
Working Conference on Database Semantics, "Data and Knowledge"
(DS-2), Albufeira, Portugal, 3-7 November, 1986 / edited by R.A.
Meersman and A.C. Sernadas.
p. cm.
"Dedicated to the memory of John G. Gashnig."
Bibliography: p.
ISBN 0-444-70528-7
1. Data base management--Congresses. 2. Artificial intelligence-
-Congresses. 3. Gaschnig, John G., 1950-1982. I. Meersman, R.
II. Sernadas, A. C.. 1952- . III. Gaschnig, John G., 1950-1982.
IV. Title.
QA76.9.D3I16 1986
005.74--dc19 88-25213
CIP

PRINTED IN THE NETHERLANDS

DEDICATION

These proceedings are dedicated to the memory of Dr. John G. Gaschnig (June 24, 1950 - March 4, 1982), who helped pioneer research into important aspects of the field covered by these proceedings.

R.A.M.

PREFACE

"Knowledge and Data" was the theme of the second IFIP Working Conference on Database Semantics (DS-2), held in Albufeira, Portugal, 3-7 November 1986. It was organized by the IFIP Working Group WG 2.6 (Databases) and is one of several conferences held over the last years exploring links between the database (DB) and artificial intelligence (AI) fields. It was a successor conference to DS-1, "Database Semantics", also organized by WG2.6 in Hasselt, Belgium (1985) [1].
It is probably too early at this point in time to make definitive statements on how techniques, methods and/or tools from one field can be used to advance the other, for instance, to extend the class of problems that may be solved by either "approach".
Therefore, we shall attempt rather to do a practical survey of the issues, techniques etc. and more specifically those relevant to knowledge representation.

It has indeed become "fashionable" to state that the application domains of database theory (and practice) and AI have become overlapped. However, database people would be so-say primarily concerned with efficiency and the practical solution of large but essentially trivial problems in an environment where things like concurrency, recovery and large volumes of uniformly structured data are dominant. AI people of course originally were only concerned with intelligence, "hard" problems of toy size and couldn't care less about use of their solutions in a practical production environment until recently, "when AI reached the marketplace" - or so it seems at least to many people.

Needless to say, this represents a gross over-simplification of the state of affairs. Nevertheless the tendency to trivialize database issues by AI people and to reduce applicability of AI techniques to toy worlds by database researchers is present and there was some concern that this might lead to fundamental gaps of understanding on a conference such as this one in Albufeira.

This fear most certainly proved unjustified afterwards: scientific and social contacts proved excellent, also no doubt helped by the sublime weather and scenery of the Algarve region, even in November. (It certainly was an improvement over the infamous January 1985 "Hasselt Blizzard" [1]!).

There occurred genuine cross-pollination not in the least caused by many provocative lectures by invited and contributing speakers. It was however noted during to some present at the discussions in the margin of the conference, that quite frequently problems <u>common</u> to both domains indeed are classified "easy" by one and "difficult" by the other and vice versa; and problems <u>specific</u> to either domain are often
claimed to have been solved by the other "a long time ago".
For example, the problem of transforming a semantic net structure of one kind or other to a well-engineered database design (normalized etc.) is perceived as non-trivial by many DB people but often of "negligible" interest to AI practitioners. Another area would be security, recovery and concurrency -dominant issues in database theory but minor ones in intelligent systems- until

recently. (As somebody once said, "The idea of security in a Lisp Machine is a baroque notion"). Conversely, problems related to taxonomy and inheritance are treated in depth in AI everywhere but only recently started to receive deserved attention by the database community with the advent of object-oriented database systems.
Also, many techniques, principles and algorithms that have come forth from AI research (e.g. backtracking, inferencing, new abstraction mechanisms) have sometimes been dismissed by the database community as being inefficient, impractical or not yet applicable in a database environment; yet databases sorely need the ability to incorporate much more semantics from present-day application domains into the represented information structures.

Overlaps between AI and Databases
The two principal overlaps between the DB and AI fields are the area of knowledge representation and the application of DB technology to large AI problems: The latter one received very little attention at this Conference, since this issue belongs almost wholly within the implementation of new database technologies.

In knowledge representation, we can distinguish the roles of
- Abstraction mechanisms;
- Introduction of logic into database theory;
- Semantic networks and graphical representation;
- The use of natural language in knowledge representation;
- New or extended methodologies;
- Formal theories of knowledge representations.

This list is not exhaustive, but pretty much sums up what the principal areas of focus for current research are. Each of these domains is reflected in one or more papers to be found in these proceedings.

Abstraction mechanisms
Databases for a very long time have mainly been concerned with one single (conceptual) abstraction mechanism, namely the abstraction from instance by introducing entity or object types and relation types. Already this fairly elementary abstraction mechanism (also called classification) has caused many a debate about what are entities, types, relations etc. and whether in fact the type construct is relevant, useful or even necessary. Particularly the AI community seemed recalcitrant to make object types part of their axiomatic foundations in knowledge representation - or at least until recent times; the introduction of (very) large populations has made instance abstraction a fundamental necessity in most
of such systems. The issue of "types" is closely related to the distinction between explicitly intensional databases and purely extensional databases. ("logic databases", treated in several papers in this book have contributed enormously in increasing our deeper understanding of the modeling process in this respect; we shall mention some of these developments further on).
Other abstraction mechanisms include abstraction from detail or components. This is linked (but not the same as) to the ability to structure data objects in a top-down (hierarchical) fashion; especially in the early phases of a knowledge representation exercise (or database design if one prefers) we need to express facts about aggregated object (-types) without going into too much detail on how such object (-types) are built up. Many "classical"

information system design methodologies have addressed this abstraction phase: note however that this form of abstraction occurs then almost exclusively in the manual design process itself and only the final <u>result</u> is effectively translatable into a database design: no current database management system (DBMS) retains in its conceptual schema the system of aggregated objects at the same level as the more detailed objects related to them. (It is of course possible to <u>represent</u> this link in a given conceptual schema, but the DBMS itself does not "know" about the meaning of this link: this semantics is completely pushed into the surrounding application environment).
Examples of this type of abstraction could be abstraction from time-dependency of relations and objects, from "attributes" of objects and from internal structural complexity: the fact that e.g. companies make profits in certain years can be a coarse but meaningful piece of knowledge without knowing how that profit is structured and related to that company's internal operation, for instance. This last example however superficial illustrates how a solution to this abstraction problem will involve (among many other things) handling of <u>derived knowledge</u>. Finally, as a third important abstraction mechanism we have <u>generalization</u> or abstraction from specific "use" or "context". This allows to describe more specific objects (<u>subtypes</u>), relations and behaviour in terms of more generic ones from which they inherit all (or some) properties. Object-oriented databases are a rapidly growing research topic in database theory and practice which treats this and the previous aggregation problem as central issue; also in AI this approach is reasonably well established since the introduction of object-oriented programming environments such as SmallTalk and others.

Note however, that in such environments often the first form of abstraction -from instance- is often "confused" with (receives a similar treatment as) generalization. This is sometimes apparent through the ambiguous use of the "is-a" relation both as an instance definition and as a subtype ("is_a_kind_of") relation. This confusion - which otherwise is not a major problem - perhaps originated from the property that a true "is_a" relation introduces a new meta-level while
a "is_a_kind_of" relation does not; most object-oriented systems collapse meta levels after the first one, i.e. they have only one meta level.

<u>Logic and Databases</u>

One of the most profound developments in database theory has been the introduction of logic and logic systems as enhanced database description formalisms. We shall not build up a complete discussion here; for an excellent and comprehensive survey see H. Gallaire's contribution to these proceedings. Most of the following terminology can be found explained there as well.
The logic approach views a database as a set of <u>clauses</u> (e.g. individual facts/records/tuples become predicate instances) and by allowing or disallowing certain clause types and/or limiting or expanding the inference mechanisms (the rules by which clauses may be combined and used together) one can derive many interesting formal properties of such <u>logical databases</u>. Important concepts are for instance the Closed World Assumption and Negation. By Failure which essentially allow to precisely handle negation (opposites) of facts stored in a database.

Another concept is the distinction between the so-called "model" and "theory" views of a database which involve whether or not to distinguish explicitly the intension ("conceptual schema") of the database from the rest of its population or extension, and the consequences of choosing either view. For a rigorous treatment, again see Gallaire's paper. The link between logic and databases is the focus of a large research effort today.

Semantic networks and graphical representation

Quite a number of knowledge representation techniques are supported by some kind of graphical formalism, usually called a "semantic network" of sorts. Examples are E-R, The Binary Relationship Model, the Functional Data Model, etc.. Most accommodate some kinds of semantical information other than pure information structure through constraints or rules to varying degrees of sophistication. Of course, not all semantics is representable in such a graphical representation; the general philosophy usually is to allow representation of the structural content of information. Especially for database design where usually the population (of instances) is large compared to the number if entity types, structuring of information at the type level is very important. Furthermore, a two-dimensional net structure has clear advantages over, say, a set of relation types which first of all pre-supposes a given valid grouping of entity and/or attribute types, and secondly, does not usually explicitly represent links among relation types.
Therefore, semantic nets allow to construct an explicit connection between on the one hand "AI-style" knowledge representation and on the other hand "classical" database design. For this latter purpose, an explicit mapping
[algorithm] from the net to a correct database design (i.e. a set of relation types) is required. Several such algorithms have been reported in the literature. Much less study has been devoted to the corresponding transformation of the semantics to procedures on the relational model obtained in this way; see however De Troyer's paper in this book, which also contains references to related work.
This area promises to become a very important one in practice and already has popped up in situations where e.g. existing expert system shells had to be adapted to existing large databases. Also, the semantic network as a graphical representation is particularly suited to graphical implementation of front-ends to future database design tools usable by database engineers and data administrators.

The use of natural language in knowledge representation

Understanding natural language has proven to be one of the toughest problems to be solved by computer programs; no satisfactory methodology covering the whole plethora of natural language constructs seems to emerge at present. For many researchers this problem essentially constitutes most of the AI field today since it directly or indirectly touches upon practically all of the issues identified within AI. Database researchers typically have not occupied themselves very much with the problem; some query languages have been defined that "resemble" natural English, but this of course actually turns the issue around. Quite interesting developments are taking place though especially for limited domains ("universes of discourse"). See for instance Schank's text in this book. The main technical difficulty lies with the strong resistance that natural language exhibits to application of "standard" abstraction mechanisms or classification in general. Shank's contribution may shed additional light on this matter.

<u>New methodologies and formal theories for representing knowledge</u>
A methodology is a set of methods with a procedure for applying them in some specific sequence to a problem statement (usually expressed in natural language) in order to construct a solution expressed in some suitable formal representation of the underlying knowledge domain. Typically the database and information system field have progressed much more in this area than AI has. AI people tend to underestimate the importance of this aspect of problem solving - or to overestimate the natural intelligence of the practitioner community at large when having to arrive at a knowledge representation in one of "their" formalisms. Of course, most methodologies developed for information system design such as [E-R] [NIAM] [IDEF1-X] and others are not universally or even effectively applicable in general but at least they try to prescribe a framework of steps to be taken from problem statement to solution (implementation). A point in case is for instance Prolog which althrough a powerful language has proven difficult to use effectively by even
experienced database engineers (at least on non-trivial problems). For classical programming languages many methodologies (e.g. top-down, structured programming, SASD,...) exist; there does not seem currently to exist equivalent comprehensive ones for any of the "AI languages" -and Prolog in particular. Usually the user of such a language has to induce from toy examples a more general and hopefully useful way of applying the language formalism to larger problems. Techniques such as "frames" and "Object-Oriented Programming" can be seen as embryonic developments towards more general methodologies for modelling a universe of discourse with the help of AI formalisms. Conversely, one observes a distinct evolution in the development of existing methodologies towards the inclusion of more semantics. Essentially three types of evolution prevail:

i. <u>extension</u> of existing methods with new, "semantic" concepts such as rules, constraints, subtypes etc. This has to be done carefully to maintain conceptual integrity;
ii. <u>merging</u> of two or more distinct methods, usually so that each method is primarily concerned with a particular phase of analysis - process analysis, data analysis, constraint analysis and so on;
iii. <u>development</u> of <u>new methods</u> more suited to be representation of richer semantics associated with a data model; for example object-oriented databases seem to be quite promising in this respect.

Much of this work is made possible by a strongly increased understanding of the important methodological "primitives" underlying all system design methods. Formalisation and the development of mathematical theories covering this area play a crucial role here and will certainly continue to do so. For specific illustrations of this importance, the reader is referred to some papers in these proceedings, such as the papers of H.-D. Ehrich, K. Drosten and M. Gogolla: "Towards an Algebraic Semantics for Database Specification" and R. Carapuca and J. Fiadeiro: "Varying Representation Schemata vs Fact Updating in KB Management".

Formalisation seems, by the way, to be a general trend and undoubtedly is related to the maturing of the field of knowledge representation and database modelling as a science, as is hopefully reflected by the contents of this book.

On the Conference event itself
About 120 experts attended this IFIP Working Conference in Portugal. Such Working Conferences are set up as discussion-oriented events where the authors and invited speakers get ample time (respectively 1 and 2 hours) to treat their subject matter in reasonable depth, and allow plenty of time for audience interaction. We planned long afternoon breaks of 4 hours from luncheon time to promote hallway (actually, terrace) discussions among participants. The unusually splendid Algarve coastal scenery and weather early November
contributed to an excellent and stimulating environment, superbly arranged by Amilcar and Christina Sernadas and their team from INESC, Lisbon.

There were five invited lectures (by Hervé Gallaire, Robert Meersman, Alan Robinson, Roger Schank and Dana Scott) and 17 refereed contributions, selected from about 70 submissions. All papers were presented by (one of) their authors with the exception of Van Nguyen's paper, which was read by John Sowa. At the time of this writing, the next conference (DS-3) is planned in Guangzhou, People's Republic of China in July 1988 to be organized by WG2.6 (Databases) together with IFIP's WG8.1 (Information Systems). Its title will be "The Role of Artificial Intelligence in Databases and Information Sytems".
The proceedings will be available from North-Holland as well.

Acknowledgements
Amilcar Sernadas compiled and edited the preliminary participant's edition for the conference itself. Special thanks from the editors are due to Tineke Kleine-Vromans at the INFOLAB in Tilburg for the excellent (and, alas, frequent) re-typing of some manuscripts and her skillful administration of the editing process.

Robert Meersman.

Reference
[1] "Database Semantics", Proceedings of the IFIP TC-2 Working Conference (DS-1), Eds. T.B. Steel Jr. and R. Meersman, North-Holland, 1986.

PROGRAM COMMITTEE AND INVITED SPEAKERS

OFFICERS

General Chairman	A. Sernadas	Portugal
Program Co-chairmen	R. Meersman	Belgium
	J. Sowa	USA

PROGRAM COMMITTEE MEMBERS

G. Bracchi	Italy
J. Bubenko Jr.	Sweden
P. Chen	USA
C. Esculier	France
H.-D. Ehrich	West Germany
J. van Griethuysen	The Netherlands
M. King	United Kingdom
E. Knuth	Hungary
C. Kung	PR China
L. Kerschberg	USA
J. Mylopoulos	Canada
E. Ruspini	USA
C. Sernadas	Portugal
T. Steel, Jr.	USA
Y. Tanaka	Japan
G. Wiederhold	USA

INVITED SPEAKERS

H. Gallaire	West Germany
R. Meersman	Belgium
J. Robinson	U.S.A.
R. Schank	U.S.A.
D. Scott	U.S.A.

TABLE OF CONTENTS

Data and Knowledge (DS-2)
R.A. Meersman and A.C. Sernadas (Editors)
Elsevier Science Publishers B.V. (North-Holland)
© IFIP, 1988

KL-DB: Towards a Unified Approach to Knowledge Representation

Sanjaya Addanki
Anil Nigam

IBM T.J. Watson Research Center
P.O. Box 218
Yorktown Heights., NY 10598

Abstract

This paper presents a new knowledge representation scheme, KL-DB, that is useful for Database as well as AI systems. Such a scheme is targeted at semantically augmented database systems. Two diverse approaches to knowledge representation are reviewed and compared - SDM from the extension-oriented DB world and KL-ONE from the intension-oriented AI world. KL-DB retains the semantic cleanliness and expressiveness of KL-ONE and provides constructs needed to handle extensions stored in a relational database.

Introduction

Knowledge representation is an active area in Artificial Intelligence that studies mechanisms for the representation and use of domain knowledge for AI systems [BL 85]. Several issues raised recently by database efforts in semantic modelling [BMS 84] [AM 84] [BM 86] are very similar to some of the issues in knowledge representation. In particular, the thrust of much work in semantically augmenting a database to provide greater flexibility in databases is much the same as that of work on frame and semantic net formalisms in AI.

Semantically augmented databases are those that allow the database to capture and maintain a complex view of data [K 82] [MNB 83] [LKMPM 85]. For example, a database for engineering design or CAD will not only have to maintain records of available parts and sub-designs but will also have to represent the complex ways in which these parts interact; otherwise the database is not of much help to the designer. Such semantically augmented databases also provide users with more flexibility in retrieving information. If the database represents the structure of the domain in a meaningful manner, a user that is familiar with the domain but not with the exact schema or the organizational structure of the database will still be able to explore the database. Interaction languages can be made higher-level in that they can deal with the language of the domain; the translation to the actual query language can take place through the semantic model.

AI has looked long and hard at two of the three issues in semantic modeling, viz. representing complex relationships and ease of interaction with knowledge bases. Much research in AI is aimed at developing knowledge representation formalisms in which complex descriptions of objects may be manipulated by programs. It is difficult for programs to interpret, and plan actions on, formalisms based on large numbers of primitives and operations that are not too different from each other. Hence the thrust in AI research has been towards semantic cleanliness of representation structures; such cleanliness being easily achieved by formalisms that incorporate few, very well-defined primitives.

However, knowledge representation research in AI is usually interested in describing all the complex relationships in an object *type* while database research is interested in efficient access to a few features of a large number of *instances* of the object. For example, an AI approach to representing an oscilloscope would try to encompass our knowledge of the oscilloscope from I/O behavior through the design history, and perhaps even down to the physics. A DB approach is typically more interested in representing the supplier, model number, location, owner, and serial number of each oscilloscope in the building. From the AI point of view it is said that AI representations are interested in capturing relatively complete intensional descriptions of objects while database representations are driven by a need to efficiently manipulate many instances in the extension of the object.

The thesis laid out in this paper is that coupling a relational database management scheme to a semantically clean knowledge representation scheme will give rise to a powerful, and usable, semantic modelling framework. We believe it is possible to merge the best of both worlds into a semantically augmented database system that is both powerful in its representational capabilities and efficient and robust in its implementation.

The knowledge representation scheme that we use as the basis of our formalism is KL-ONE, a much implemented and used semantic net formalism that is known for its clean semantics. The first part of the paper compares KL-ONE with SDM, a semantic data model from the database world. SDM is a detailed, but unimplemented, formalism that is similar to frames and semantic nets. The comparison is interesting in that it highlights the differences between AI and database approaches to similar problems. SDM is clearly aimed at providing the user with various hooks to manipulate extensional data in different ways. However, SDM uses many primitives and operators to do this. KL-ONE is aimed at providing a clean semantics for representing intensions such that they can be interpreted by programs. KL-ONE does not provide all the facilities required to manipulate extensional data.

The second part of the paper presents KL-DB, a modified and augmented version of KL-ONE that provides many of the representation and access mechanisms needed in a semantically augmented database. Even though some of the modifications are significant, KL-DB does retain the semantic cleanliness and integrity of KL-ONE. The description of KL-DB also includes details of the mapping between KL-DB structures and relational databases. Details of query languages and optimization are deferred to a later paper.

We see the significance of this work as two-fold, a) the approach taken to motivate a unified knowledge representation scheme and b) the evolution of the KL-DB scheme itself. The comparison of SDM with KL-ONE not only furthers an understanding of the AI and DB approaches to knowledge representation but also motivates the features required of a unified scheme. As our work progresses, we see KL-DB as an effective vehicle for cross-fertilization of knowledge representation ideas from DB and AI fields.

Comparing SDM and KL-ONE

The SDM model was proposed by Hammer and McLeod [HM 81] as a structuring paradigm for semantically augmented databases. SDM attempts to capture the many ways in which people might want to model their data by providing a large number of different constructs that may be used to describe aggregations.

KL-ONE [B 79] [SB 81], on the other hand, is a semantic net formalism that is based on the belief that the road to clean semantics lies in using very few types of nodes and edges. Further, KL-ONE insists that the meanings of these nodes and edges be tightly constrained to dealing with the structural aspects of defining concepts; that is, nodes and edges have no domain-level or implementation-level semantics.

Class vs. Concept

A database consists of extensional entities that correspond to objects in the application domain. SDM provides a structure called a Class that is used to aggregate these entities; similar entities belong to the same class. For example, the SDM network of Fig. 1 shows the classes SHIPS, OFFICERS, and OIL-TANKERS among others. Classes have Attributes that describe their properties and characteristics. Class Attributes describe properties of the class as a whole, e.g. average-age, cardinality, etc., and member attributes describe properties of the individuals in a class, e.g. Captain of SHIPS, Rank of OFFICER, etc..

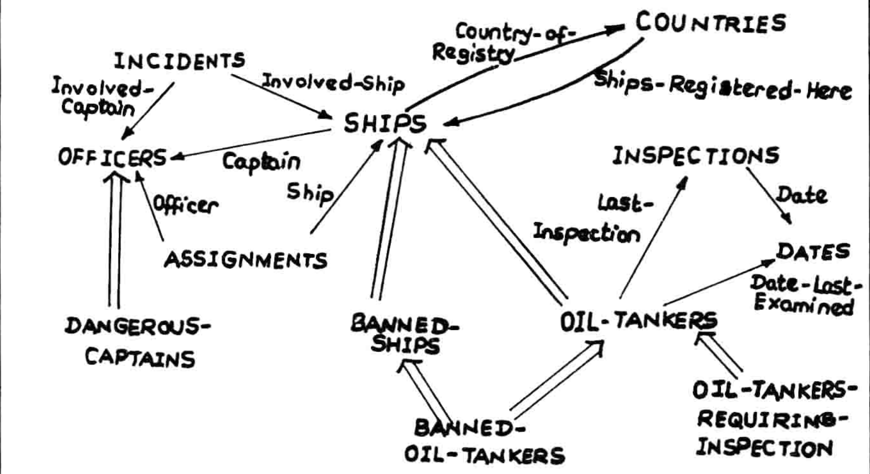

Figure 1. Graphic Adaptation of a sample SDM Schema (from [HM 81])

A Concept in KL-ONE is the definition of an entity in the world. A Generic concept is like a class in that it may have many instances in the world. An Individual concept may have at most one instance in the world. However, a concept is not a class in that it is not a collection of extensional objects. A concept is strictly an intensional description of an entity. KL-ONE provides Roles to describe the properties of concepts. Roles are like attributes except that class attributes have no equivalent because concepts are intensional descriptions. All roles are like member attributes and RoleSets are like multi-valued attributes. BLOCK-OBJECT, ARCH, and SQUARE-ARCH are examples of concepts in Fig. 2. Lintel is a Role and Part is a RoleSet.

Both roles and attributes have further structure; it is possible to specify the concept/class of the fillers of roles/attributes. For example, the filler of the attribute Captain of the class SHIP can be restricted to the class OFFICER, and the filler of the role Upright in the concept ARCH can be restricted to the concept BLOCK. Such restrictions are called the ***value class*** of an attribute in SDM and the ***value restriction*** (v/r) of a role in KL-ONE. KL-ONE roles can also specify restrictions on the number of fillers, e.g. 2 uprights for an ARCH, Fig. 2. Attributes in SDM may also be derived through more complex relationships and these are described later.

Sub-Class vs. IS-A

SDM provides interclass connections to describe relations between classes. The simplest of these is the sub-class connection and is shown as a double arrow in Fig. 1. To say that the class OIL-TANKERS is a sub-class of SHIPS is to say that all members of OIL-TANKERS are also members of SHIPS. SDM provides inheritance in that all the attributes of the super-class are also attributes of the sub-class.

KL-ONE provides the SuperC or IS-A link that has a similar purpose. Also shown as a double arrow, Fig. 2, the SuperC link states that the sub-concept inherits all the roles of the super-concept. For example, SQUARE-ARCH inherits all the roles of ARCH. Inheritance plays a crucial role in KL-ONE because it is the principal mechanism for defining concepts. All concepts are defined in terms of SuperC links to other concepts and all concepts are derived from a root con-

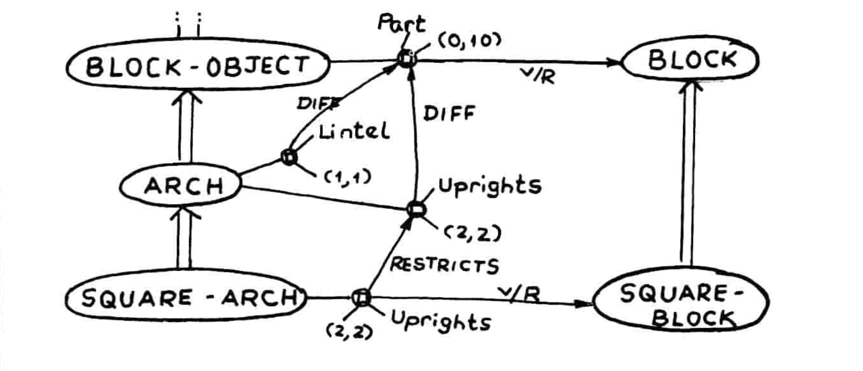

Figure 2. A Simple KL-ONE Hierarchy

cept such as THING. Since it is often impractical to define all concepts as derivatives of a common root, KL-ONE provides Starred Concepts that derive their meaning from user-written code.

The extensional and intensional characteristics of SDM and KL-ONE are emphasized in their treatment of inheritance. SDM relegates inheritance to a minor role because classes may be defined as collections of extensional objects. KL-ONE, however, being an essentially intensional language, is forced into deriving its semantics from inheritance; given that a KL-ONE network is a collection of intensional descriptions, the only way to define a new concept is in terms of other concepts! SDM and KL-ONE have some similarities in their treatment of inheritance. For example, inheritance is pure and exceptions are not allowed in either formalism. SDM allows further restriction of the value class of an attribute while defining a sub-class. A sub-class so defined is called an Attribute Defined Sub-Class. Value Restriction in KL-ONE performs exactly the same function.

Set Operations in SDM

SDM allows the definition of classes as set intersections, unions, and differences of other classes. For example, BANNED-OIL-TANKERS is defined as the intersection of BANNED-SHIPS and OIL-TANKERS, and SHIPS-TO-BE-MONITORED is defined as the union of BANNED-OIL-TANKERS and OIL-TANKERS-REQUIRING-INSPECTION. KL-ONE supports intersection through the SuperC link. Any concept defined to be a sub-concept of two concepts is, by virtue of the properties of inheritance, the intersection of the two concepts; i.e. any instance of the sub-concept is also an instance of both the super-concepts. Union and difference are difficult to support because they require defining logical predicates on the roles, and this is not permitted in KL-ONE.

Procedural Attribute Derivations in SDM

SDM also allows procedural derivation of attribute fillers. For example, the attribute Seniority of the class OFFICERS is derived as "order by CommissionDate". Such derivation is impossible in KL-ONE because the computation requires iterating over elements of the class/concept and is hence an extensional notion. KL-ONE only represents the intensional definition of objects. Other SDM derivations that are not supported in KL-ONE include transitive closures and functions such as MIN, MAX etc.. Note that these are all extensional notions.