37, 1994.

[SwG95] B. Swartout and Y. Gil: EXPECT: Explicit Representation for Flexible Acquisition. In *Proceedings of the 9th Knowledge Acquisition for Knowledge-based Systems Workshop (KAW-95)*, Banff, Alberta, Canada, 1995.

[ToA94] J. Top and H. Akkermans: Tasks and Ontologies in Engineering Modeling, *International Journal of Human-Computer Studies (IJHCS)*, 41:585—617, 1994.

[vHB92] F. van Harmelen and J. Balder: $(ML)^2$: A Formal Language for KADS Conceptual Models, *Knowledge Acquisition*, 4(1), 1992.

[vLPT93] I. van Langevelde, A. Philipsen, and J. Treur: A Compositional Architecture for Simple Design Formally Specified in DESIRE. In J. Treur et al. (eds.), *Formal Specification of Complex Reasoning Systems*, Ellis Horwood, New York, 1993.

[Wil92] R. Wille: Concept Lattices and Conceptual Knowledge Systems, *Computers Mathematical Applications*, 23(6-9):493-515, 1992.

[DGL95] G. De Giacomo and M. Lenzerini: What's in an Aggregate: Foundations for Description Logics with Tupels and Sets. In *Proceedings of the 14th International Conference on Artificial Intelligence (IJCAI-95)*, Montreal, Canada, 1995.

[Fen95] D. Fensel: Formal Specification Languages in Knowledge and Software Engineering, *The Knowledge Engineering Review*, 10(4), 1995.

[FAS] D. Fensel, J. Angele, R. Studer: The Knowledge Acquistion and Representation Language KARL, to apear in *IEEE Transactions on Knowledge and Data Engineering*.

[FvH94] D. Fensel and F. van Harmelen: A Comparison of Languages which Operationalize and Formalize KADS Models of Expertise, *The Knowledge Engineering Review*, 9(2), 1994.

[FFR97] A. Farquhar, R. Fickas, and J. Rice: The Ontolingua Server: a Tool for Collaborative Ontology Construction, *International Journal of Human-Computer Interaction (IJHCI)*, 46(6), 1997.

[FLORID] http://www.informatik.uni-freiburg.de:80/~dbis/flogic-project.html.

[FMD+97] D. Fensel, E. Motta, S. Decker, Z. Zdrahal: The Use of Ontologies For Specifying Tasks and Problem-Solving Methods: A Case Study. In *Proceedings of European Knowledge Acquisition Workshop (EKAW-97), Lecture Notes in Artificial Intelligence (LNAI)*, Springer-Verlag, 1997.

[FS] D. Fensel and R. Straatman: The Essence of Problem-Solving Methods: Making Assumptions to Gain Efficiency, to appear in *International Journal of Human-Computer Studies* (*IJHCS*).

[GaW96] B. Ganter and R. Wille: *Formale Begriffsanalyse*, Springer Verlag, Berlin, 1996.

[Gua97] N. Guarino: Understanding, Building, and Using Ontologies, *International Journal of Human-Computer Interaction (IJHCI)*, 46(6), 1997.

[Har84] D. Harel: Dynamic Logic. In D. Gabby et al. (eds.), *Handook of Philosophical Logic*, vol. II, *Extensions of Classical Logic*, Publishing Company, Dordrecht (NL), 1984.

[HSW97] G. van Heijst, A. T. Schreiber, and B. J. Wielinga: Using Explicit Ontologies in Knowledge-Based System Development, *International Journal of Human-Computer Interaction (IJHCI)*, 46(6), 1997.

[KLW95] M. Kifer, G. Lausen, and J. Wu: Logical Foundations of Object-Oriented and Frame-Based Languages, *Journal of the ACM*, vol 42, 1995.

[LeR96a] A. Y. Levy and M.-C. Rousset : CARIN: A Representation Language Combining Horn Rules and Description Logics. In *Proceedings of the 12th European Conference on AI (ECAI-96)*, Budapest, Hungary, August 11-16, 1996.

[LeR96b] A. Y. Levy and M.-Ch. Rousset: The Limits on Combining Recursive Horn Rules with Description Logics. In *Proceedings of the 13th National Conference on Artificial Intelligence, AAAI-96*, Portland, Oregon, August, 1996.

[LMM+95] D. Lukose, G. Mineau, M.-L. Mugnier, J.-U. Möller, P. Martin, R. Kremer, G. P. Zarri: Conceptual Structures for Knowledge Engineering and Knowledge Modelling. In *Proceedings of the International Conference on Conceptual Structures (ICCS-95)*, USA, 1995.

[Mac90] MacGregor: *LOOM Users Manual*, ISI/WP-22, USC/Information Sciences Institute, 1990. http://www.isi.edu:80/isd/LOOM/.

[Neb96] B. Nebel: Artificial Intelligence: A Computational Perspective. In G. Brewka (ed.), *Principles of Knowledge Representation*, CSLI publications, Studies in Logic, Language and Information, Stanford, 1996.

[Prz88] T. C. Przymusinski: On the Declarative Semantics of Deductive Databases and Logic Programs. In J. Minker (ed.), *Foundations of Deductive Databases and Logic Programming*, Morgan Kaufmann Publisher, Los Altos, CA, 1988.

[ScB96] A. TH. Schreiber and W. P. Birmingham (eds.): Special issue on Sisyphus-VT, *International Journal on Human-Computer Studies (IJHCS)*, 44, 1996.

[Sow84] J. F. Sowa: *Conceptual Structures: Information Processing in Mind and Machine*, Addison Wesley, Reading, Mass., USE, 1984.

[SWA+94] A. TH. Schreiber, B. Wielinga, J. M. Akkermans, W. Van De Velde, and R. de Hoog: CommonKADS. A Comprehensive Methodology for KBS Development, *IEEE Expert*, 9(6):28—

algorithm. This is different for DLs as they do not differentiate between a specification and an implementation. In their case there is one *representation* language that has to serve for both (possibly conflicting) purposes. However it is interesting to note that recent approaches in the bandwagon of DLs try to increase to expressive power of these languages so as to cover structured objects and meta-reasoning (i.e., *CATS* [CDGL95], [DGL95]).

We can distinguish three different approaches in regard to the representation of control knowledge. Nearly all specification languages for KBSs allow the explicit representation of control knowledge that guides the inference process of a KBS. They provide this service for the following simple reason: Most problems tackled with KBSs are computational hard problems ([Neb96], [FS]). However, experts often can provide heuristic strategies that significantly improve average-case behaviour. In most cases, this knowledge exists and is necessary to achieve usable systems. The early literature on expert systems is full of cases where experts encoded control knowledge implicitly in production rule systems ([Cla83], [Cla86]). This implementation formalism was assumed to be declarative and to abstract from all control. However, knowing its inference algorithm one can encode control knowledge in the sequence of rule conditions and rules and one can introduce rule conditions that act as control flow variables. Very often such knowledge was used to improve the performance of the implemented systems. Therefore, specification languages for KBSs. like $(ML)^2$ and KARL, use a distinct logical language to represent control knowledge. Dynamic logic [Har84] is used to express procedural control on top of declaratively specified inference steps and knowledge units. DL-systems take precisely the opposite point of view. All control is hidden from the user. He should only be able to specify declarative knowledge. The inference service is realized by the deductive engine of the DL system. Approaches to logic programming take more of an intermediate position. In principle, they provide declarative approaches where all control is encapsulated by the general inference engine for Horn rules. However, every PROLOG programmer uses of knowledge of the inference strategy of the rule interpreter, and some of the language constructs, like the *cut* , explicitly allow improvement of the efficiency of a program.

Acknowledgement. We thank all participants for their productive discussions and the firendly atmosphere they created.

References

[Ang93] J. Angele: *Operationalisierung des Models der Expertise mit KARL*, Infix, St. Augustin, 1993.

[Bal95] M. Balaban: The F-Logic Approach **for** Description Languages, *Annals of Mathematics and Artificial Intelligence*, 15:19-60, 1995.

[Bal94] M. Balaban: *The (Frames)-Logic Approach* **for** *Description Languages II: A Hybrid Integration of Rules and Descriptions*, Technical Report FC-94-10, Dep. of Mathematics and Computer Science, Ben-Gurion University of the Negev, Israel, 1994.

[CDGL95] D. Calvanese, G. De Giacomo, and M. Lenzerini: Structured objects: Modeling and resoning. In P*roceedings of the Fourth International Conference on Deductive and Object-Oriented Databases (DOOD-95)*, Lecture notes in Computer Science, Springer, 1995.

[Cla83] W. J. Clancey: The Epistemology of a Rule-Based Expert System - a Framework for Explanation, *Artificial Intelligence*, 20, 1983.

[Cla86] W. J. Clancey: From GUIDON to NEOMYCIN and HERACLES in Twenty Short Lessons: ORN Final Report 1979-1985, *The AI Magazine*, August 1986.

Dieter Fensel and Stefan Decker discussed requirements on formal specification languages for ontologies in knowledge engineering [FMD+97]. A task ontology for *parametric design* [ScB96], a problem-solving method ontology for *propose and revise* [ScB96], and the mapping of both ontologies were sketched in Sloppy-logic (S-logic)[4]. The main purpose was to illustrate the requirements on expressive power necessary to model such ontologies. Complex data models, meta-relationships, and axiomatic knowledge must be expressed. Usual DLs clearly fail to fulfil the requirements for these kinds of tasks. The mapping to FL could be performed but requires some modifications of the original models.

Mira Balaban proposes FL as a unifying framework for the different variants of DL ([Bal94],[Bal95]). Currently, semantics is assigned to DLs by defining their counterparts in predicate logic. However, FL provides concepts and attributes as language elements that can be used directly to attribute semantics to concepts and roles in DL. Different constructors in DL can be axiomized. Because FL contains arbitrary first-order formulas or rules in its Horn fragment it can directly be used to express extended DLs and hybrid languages that combine descriptors and rule languages. In a nutshell, FL is better suited than predicate logic as a semantical framework for the different DLs because the distance between the language primitives of DL and FL is much smaller than the difference between DL and predicate logic.

Critical comments on the usefulness of FL for this purpose were raised by Enrico Franconi. First, DLs rely on the ontological distinction between concepts and elements. FL blows away this distinction. Semantically, both are modeled by individuals in FL. This is actually the way in which FL reifies meta reasoning in a first-order framework. However, different sorts could be defined that represent the distinction in FL. Second, the perfect model semantics of the Horn fragment of FL, with its implicit closed-world and domain-closure assumptions, conflict with the open-world and open-domain assumptions of DL. This difference already appears for class definitions that are sufficient and necessary because this requires negation to expressed in Horn logic. However this does not hold for FL in general but only for the Horn version of FL which has been defined to model logic programming and deductive databases. Third, DL uses the unique name assumptions whereas FL includes term equality.

Finally, a kind of cultural difference between people working on specification or representation becomes apparent. DLs are *representation formalisms*. That means, decidability and efficiency of inference support are key issues in language design. So two main research goals for DLs can be identified: (1) Extending first-order logic syntactically to improve the modeling support of the language and (2) restricting the expressive power of first-order logic to enable sound, complete and efficient inference service. The first goal is also shared by *specification languages*. In fact, it is even more important for them. The purpose of these languages is to narrow the gap between an informal model of a system and an implementation. Therefore, they need to combine a formal semantics with as much expressive power as possible. As a consequence, the second goal is less important for specifications. Some of them, like KARL, are executable to provide testing as a means to evaluate specifications. In this case, stronger restrictions on the expressive power have to be introduced. However, the requirements on efficiency and completeness of the inference service are less significant because it is an executable prototype but *not* the efficient and robust implementation of the system. The application can be efficiently solved by a special purpose

4. *S-logic* is a semiformal specification formalism with a logical flavor. However no formal syntax nor semantics are provided.

the existing approaches. Two further relevant approaches that were not covered by the workshop are the work on *conceptual graphs* [Sow84], [LMM+95] and on *formal concept analysis* [Wil92], [GaW96].

Marie-Christine Rousset gave a survey on DLs and presented her language CARIN [LeR96a], that combines DL with a datalog like rule language. DLs are a family of languages that were developed for modeling complex hierarchical structures. They make it possible to define complex classes of objects (called concepts) and their properties (called roles) in a formal and declarative way. A DL terminology includes a set of primitive concepts and roles as well as definitions of complex concepts and roles. The definitions of complex concepts and roles are given via descriptions that are built using a set of constructors, which vary from one DL to another. The strength of DLs is that they are associated with algorithms for subsumption checking, automatic classification and instance recognition. However, a strong limitation of DLs for modelling purpose is their restricted representation formalism that allows only the specification of terminological knowledge. Therefore, CARIN enriches DL with Horn rules. However, these rules are non-recursive and do not allow the derivation of new terminological knowledge (i.e., the head of a clause may not be an element of the Tbox). Based on these restrictions a complete and sound inference engine is provided for existential queries.

Michael Kifer presented a survey on FL. FL is developed to present a declarative framework for object-oriented and frame-based languages. It provides object identity, complex objects, inheritance, polymorphic types, methods, encapsulation and integrates these features in a logic-based framework. The syntax of FL is higher-order, which, among other things, allows the user to explore data and schemes using the same declarative language. FL integrates this higher-order syntax in a model-theoretic semantics and a sound and complete resolution-based proof procedure. For the Horn fragment of FL, perfect Herbrand model semantics is defined.

Two inference engines for FL dialects were demonstrated. First, [FLORID] (F-LOgic Reasoning In Databases) is a deductive object-oriented database prototype employing FL for data definition and as a query language. Second, the “old“ interpreter [Ang93] and the “new“ interpreter[3] for KARL were demonstrated. Both approaches share the problem of stratification necessary for perfect model semantics [Prz88]. Because local stratification is undecidable, stronger syntactical versions of stratification are necessary. However, the very flexible syntax of FL makes this task much more difficult than for usual datalog-like approaches.

Christian Schlepphorst provided a case study on the use of FL in the domain of computational linguistics. In his contribution he described an attempt to remodel an existing DL application from a major project to build a German language knowledge acquisition system at the Freiburg University Computational Linguistics Lab (CLIF). This application seems to be very appropriate for modeling in F-Logic for a number of reasons: The core problem of the application is addressed in a purely logical, rule-based way (called Qualification Calculus); these rules rely on A-box inference, not on subsumption between concepts; to apply those rules to statements about domain items, a huge reification framework was necessary in the DL version and exploiting the meta-features of F-Logic avoids reification. As a consequence, this allows the calculus rules and the domain facts to be expressed in a much more concise and readable manner.

---

3. Integrated in Netscape and applicable to a richer logical languages, see http://www.aifb.uni-karlsruhe.de/WBS/sde/KARL.

Workshop on Comparing Description and Frame Logics [1]

D. Fensel*, M.-C. Rousset+, and S. Decker*

* Institut AIFB, University of Karlsruhe, D-76128 Karlsruhe,
{dieter.fensel, stefan.decker}@aifb.uni-karlsruhe.de

+ LRI, University of Paris-Sud, 91405 Orsay Cedex, France, mcr@lri.lri.fr

Abstract. The specification of reusable terminological knowledge is one of the key issues in today's knowledge engineering. Providing formal languages with precise semantics and inference support can significantly support this activity. The aim of the workshop was to understand and to compare existing approaches developed in other research communities. We investigated research on description languages and research on object-oriented databases. Both provide the combination of rich terminological modelling primitives with well studied semantics and inference support. To better understand and compare them as well as to highlight common aspects and differences were the goals of the workshop.

Knowledge-based systems (KBSs) consist of large amount of domain knowledge and problem-solving methods that describes the inference process of the system [SWA+94]. The domain knowledge defines concepts, properties, relationships, heuristic rules, instances etc. that are necessary to define the application problem and its solution process. Recent work on *ontologies* aim at developing reusable terminological knowledge which improves knowledge sharing and prevents a development from scratch for each new system ([ToA94], [FFR97], [Gua97], [HSW97]). Therefore, the specification of terminological knowledge is one of the key issues in today's knowledge engineering.

Given this fact, there is clear need to improve existing approaches to specification languages for KBSs. Some of them are rather weak in supporting terminological representation because they focus on other aspects (for example, $(ML)^2$ [vHB92] or DESIRE [vLPT93]); others use different techniques for representing terminologies and provide little understanding of the advantages and disadvantages of the different choices. Specification approaches that aim at rich terminological specification formalisms are either inspired by Description Logics or by approaches of object-oriented databases and logic programming.[2] The language KARL [FAS] uses a customization of *Frame logic* [KLW95] for this purpose. Frame logic (FL) accounts in a clean, declarative fashion for most of the structural aspects of object-oriented and frame-based languages. The EXPECT approach [SwG95] for specifying and implementing knowledge based systems is based on LOOM [Mac90], a *Description Logic* (DL) with high expressive power. The representation language CARIN [LeR96a], [LeR96b] combines DL with a Datalog like rule language. Therefore it was quite natural to organize a workshop that aims at a better understanding of the different possibilities by comparing the existing approaches. We focused our attention on DLs and FL, which are the technical core of many of

1. March 26-27, 1997 at the Institut AIFB, University of Karlsruhe, Karlsruhe, Germany. For more information see http://www.aifb.uni-karlsruhe.de/WBS/dfe/dlfl.

2. See [FvH94] and [Fen95] for surveys of specification languages in knowledge engineering.