DOM: An Active Assistance System for Architectural and Engineering Design

Shirin Bakhtari
BSR Consulting GmbH
Wirtstrasse 38
HD - 81539 Muenchen
GERMANY

Wolfgang Oertel
Technical University of Dresden
Department of Artificial Intelligence
D - 01062 Dresden
GERMANY

This article delineates an active design assistance system for conceptual design, called DOM which is the abbreviation for Domain Ontology Modelling. The intention of our work is to endorse the role of modelling a common and shared platform of design knowledge as well as to address the crucial task of representing design decisions and engineering judgements in order to evaluate design layouts and to support layout construction from scratch. The prerequisites and assumptions for an appropriate role of an active design assistance system are explained. The presented paper contains both a conceptual and a technical exploration of the DOM system.

Keywords: design ontology, decision making, analysis, synthesis

1 Introduction

We present an active design assistance system for architectural and engineering design [1] that we called DOM which is the abbreviation of Domain Ontology Modelling [2]. The identification and formalization of the domain ontology is for our purpose twofold. First, we represent a solid model of the deep design knowledge that establishes the meaning and permissible use of the design objects as well as the relations between the objects. Secondly, we consider the represented deep knowledge and manifest design decisions and engineering judgements that shape a design. This provides the ability to determine the plausibility, reliability and workability of the design layouts.

A solid model of deep design knowledge and a set of design decisions and engineering judgements comprise the domain ontology in the DOM system. According to this kind of design ontology modelling, we achieve a qualitative leap towards active design assistance that can make a meaningful contribution to the current design practice.

The deep design knowledge is represented in terms of a structured model of design objects with a set of their miscellaneous features. All design objects are classified within useful levels of abstraction. We also keep a set of design layouts that have been turned out to be meaningful and useful and we refer to them as cases. Cases are organized and stored in a case base. They can be recalled and reused by architects and engineers whenever they are needed in order to solve the current problem of interest. This model of deep knowledge constitutes a common and shared platform of design knowledge that supports designers being involved in designing an artefact within a collaborative framework.

The design decisions and engineering judgements are represented as maxims and quality assessment functions which are used for

(a) the assessment of reliability and permissible use of design objects and their arrangement in a proposed layout (analysis) and

(b) for assistance during the construction of a new design layout (synthesis).

This decision making knowledge enables the DOM system to assist designers by advice and suggestions for improvement and construction due to the specified analysis and synthesis rules and functions.

The DOM system provides the ability to consider the current state of the design elaboration and invokes the operations that are relevant for that particular elaboration stage. This is also valid for proposing layout fragments that are still under exploration.

2 Prerequisites and assumptions for design assistance

The central issue of this section is to figure out the three significant characteristics of the design domain that have been decisive while we were spelling out an appropriate concept of development for our active design assistant system.

There is general acceptance that design applications, such as building design, are to be classified as ill-structured and open world problems [Beheshti et al 93; Pu 93] where the tasks incorporate rather vague and evolving requirements as well as underspecified goals. The process of designing an artefact can be characterized as a non-linear stepwise refinement process. At each stage of exploration there may occur new territories of tasks with new patterns of demands.

The most significant implication of this assumption is that the conceptualization and realization of the DOM design assistance system has to ensure the ability for dealing with incomplete and incrementally expanding knowledge [Bakhtari et al 94]. Hence, the evaluation of design layouts has to proceed in terms of plausibility and workability [3] within the realm of engineering judgements, design decisions, and a set of agreements. For the purpose of a formal representation of this kind of decision making knowledge, we considered the idea behind expert critiquing techniques as proposed in [Silverman 93]. Our quality assessment rules and functions can be successfully applied when partial knowledge in the domain of application is given, or more general, when only certain aspects of a problem are to be handled by the system. The DOM decision making knowledge helps designers by advice and suggestions for improvement to evaluate their own approach with regard to the considered knowledge and the current state of design exploration.

Due to the complex nature of architectural and engineering design, design activities usually proceed in multiple steps of elaboration. Design activities usually start with a conceptual design and proceed down to detailed design (the realization of the conceptual design layout using for instance precast building elements). At each stage of the design process a more or less elaborated layout is designed.

Since one fundamental issue in the development of the DOM is to permit higher order solutions at each abstraction level of the design process we have to adjust the decision making rules and functions due to the current level of interest. Our ultimate goal within the development of DOM is to give a strive for the ability to support designers within different stages of elaboration at conceptual as well as detailed design activities.

An essential characteristic of the most complex real-world applications, such as design, is that the activities are almost carried out in a multiple discipline collaborative framework using a common and shared knowledge platform. Within a collaborative framework the domain knowledge shared among different designers and engineers having different foci of interest on the domain knowledge is referred to as an ontology. There are notable research and development investigations in this field, such as the ARPA funded Knowledge Sharing and Reuse Effort at Stanford [Neches et al, 91], the YMIR ontology modelling [Alberts et al, 94] that is proposed by the group at Twente University for design applications, and the ROOMMOVE system proposed in [Branki et al., 94].

The DOM ontology incorporates a shared and reusable deep knowledge model as well as knowledge about engineering judgements and design decisions. The represented deep knowledge [Bakhtari et al, 95] establishes the meaning and permissible use of the design objects and the relation between these objects. The decision making knowledge takes a holistic view on the proposed layout and supports designers through a continuous loop of synthesis and analysis activities for designing an artefact.

3 DOM: a conceptual view

The central issue of the DOM system is to assist designers by providing reusable shared design knowledge and decision making knowledge. The provided functionality of the DOM can be summarized as follows:

(a) review a design layout in its current state in terms of the permissible use of the involved design objects,

(b) assess the quality of a proposed layout in terms of coherence and conformance,

(c) provide suggestions for improvement,

(d) draw the users attention to discrepancies, errors of omission, and conflicts,

(e) assist the user by providing reusable design layouts (cases) and their adaptation to the current problem of interest, and

(f) support rectification of proposed layouts and construction of a new layout from scratch.

The current specification and implementation of DOM determines the meaning and the permissible use of the design objects, the relations between the objects as well as the plausibility and reliability of whole arrangements relative to ARMILLA [Haller 851. ARMILLA is a spatial ordering and coordination methodology for the design of the heating, ventilating and air conditioning systems (HVAC) for highly complex buildings, such as office buildings or schools. Figure 1 illustrates the CAD-environment in which the DOM system is situated [4]. It shows an integration concept of the DOM within the casebased design support system of the FABEL project [Walther et al, 94].

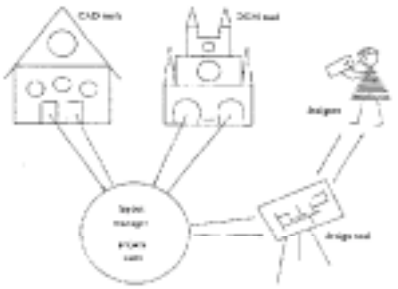

Figure 1: The integration of DOM within a LAD environment

In the following we give a conceptual explanation of the DOM system in term of its components. It is characterized by an external layout platform, a design board, a domain ontology, and a design assistance.

The DOM system is accessible for layout input through its external layout Flatform that serves the interaction between the DOM and an external drawing system or a layoutmanagement system. A proposed layout goes through the following transformation and examination steps. All design objects are first represented on the layout platform. One examination is to check whether the incorporated design objects belong to one subsystem (return air, supply air, heating etc.). If this is not the case, the involved design objects will be classified due to the subsystems. The syntax check of the involved design objects includes a plausibility examination of the given positions of objects (coordinates and dimensions), e.g. the value of coordinates is not supposed to extend a predefined value (constraint). All design objects which pass this examination are represented in terms of their geometry and their additional descriptive data.

The design board is specified for a certain project in which the proposed layout has to be embedded [2]. The initialization of the design board incorporates e.g the overall building geometry, the internal division of the building, the geometry of service cores, like elevator, exit stairs, internal stairs, and the geometry of technical service space. On the design board, the design objects go through a qualification and aptitude test. First, all design objects which have to be viewed as a whole will get aggregated and made explicit as aggregate objects. Design objects which represent relations between other objects are identified in terms of their corresponding geometric-topological relations, e.g. contact, overlap, etc.

At this stage each design layout is specified through the geometry of its design objects, their descriptive data, and the topological relations between the objects. If only small changes on design objects are needed to get the proposed layout accepted, there are also a set of rectification operations specified that can be applied in order to let the proposed layout pass the aptitude test and go for evaluation.

The ontology is the heart of the DOM system. It provides semantic views on the prepared design objects. An essential part of the ontology is the definition of a set of rules that identify semantic concepts out of the given syntactic design objects or that generate new syntactic design objects according to the semantic concepts. Due to the identification rules, the proposed design objects get identified through their corresponding semantic and are represented as instances of generic semantic structures that we call concepts. Concepts describe classes of design objects and are organized in a taxonomy. The interconnection between the concepts is also specified in a concept partonomy.

The term concept stands for a variety of generic domain knowledge chunks, such as semantic concepts that give the meaning of design objects, e.g. outlets, pipes, walls, or aggregated objects, e.g. pipeline system fragments as well as topological relations, e.g. contact, overlap etc. The functionality of the concepts is determined through the subsystem in which the layout objects have been specified. For example the functionality of a ductwork within the subsystem supply air is to connect the in- and outlets in order get the concerned area supported by fresh air. The following components comprise our deep knowledge model:

(a) a taxonomy of the concepts,
(b) a partonomy of concepts,
(c) a set of associations of the concepts with characteristic features,
(d) concept identification and generation rules, and
(e) feature determination algorithms.

Since the fundamental issue of DOM is to permit higher order solutions, we specified a set of functions and rules for analysis (quality assessment) and synthesis (completion and construction) that can be activated through the assistance component. The knowledge for decision making support in DOM is, as mentioned above, formalized on the basis of a spatial ordering and coordination design methodology called ARMILLA. The criterion of success in quality assessment is that of being conform with the underlying methodology of designing an artefact.

The design process is viewed as a continuous loop of analysis and synthesis. Hence, the quality assessment functions are defined to assess the quality of the layouts in terms of the ultimate goal that has been under the designers consideration while elaborating the proposed layout. Each quality assessment function constitutes a semantic view on the current case of interest with the focus on one or more particular features and evaluates it in terms of the required operations.

The current state of the specification and implementation of the DOM serves, for example, the following quality assessment examination:

(a) Coherence assessment:

In general, the examination can be viewed as a check for a closed loop of well arranged ductsystems that connect all well situated in- and outlets together in order to satisfy a certain functionality, where the functionality is given through the subsystem for which the layout has been designed.

(b) Spatial ordering and organization maxims:

The spatial ordering and organizational maxims for the service space divisions indicate which layers of the technical service space are to be used for which sorts of ducts laying in which directions. The maxims for the spatial ordering of outlets define which configurations of outlets are desirable. The coordination maxims give regulations and priorities for the spatial organization of different subsystems within the layer structure of the technical service space.

Over the period of our ontological engineering, we found out that the knowledge for decision making support - which we specified in quality assessment functions - has much in common with the knowledge for constructing new layouts from scratch. We also got an insight into the fact that adaptation of former cases that we store in a case base relies also to a large extent on the same knowledge. Thus we decided to extend the support services on the basis of the domain ontology modelling for multiple purposes. Our current

state of the DOM system supports assessment and construction. Adaptation of former cases is subject of our ongoing work.

The layout construction rules are applicable for generating a goal oriented solution from scratch. They may also be applied for the completion of a partially specified layout. For instance adequate rules can be used to design a proper connection between the specified outlets and the main support trunk. All of the system suggestions for construction proceed in a way that is conform with the underlying organizational maxims and quality requirements. The construction can proceed in several steps and may serve as a coach for the designer while designing a specific layout for a certain subsystem. It indicates synthesis alternatives in order to give the user the opportunity to get informed about possible constructions that are conform with the underlying ontology. Our implementation activities within the DOM service support incorporate the assistance services: a) the completion of a partially specified case and b) assistance for case construction from scratch. Figure 2 gives a conceptual view of the DOM system in terms of its components.

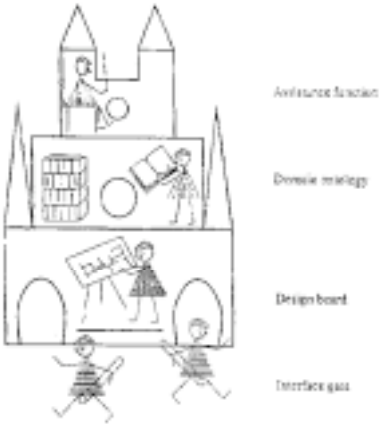

Figure 2: The DOM concept

3.1 A scenario

With the articulation of a requirement the cue is given to the executive manager sitting in the top storey of the. DOM system, as shown in figure 2. The DOM executive manager expects as necessary inputs: an assistance call, a design layout, the subsystem(s) for which the layout has been designed, and the design grid. Optional inputs are: a specific project in which the designed subsystem has to be embedded in, a particular structure of the technical service space. The assistance support services offered by DOM are classified as follows:

(a) Concept and Topological Information,
(b) Quality Assessment,
(c) Rectification,
(d) Completion, and
(e) Construction from Scratch.

Consider a proposed layout as shown in figure 3, which gives a top view of a conceptual design of a return-air subsystem. The assistance call is formulated as Concept and Topological Information and Quality Assessment. Due to this requirement the DOM executive manager undertakes the role of a critic and causes a process of analytic examination.

On the interface platform all syntactic design objects involved in the proposed layout will be specified in terms of a DOM internal representation and classified due to one subsystem. Each object is represented in term of its geometrical data, descriptive data, and presentation data (visualization).

The next step is the design board. The knowledge needed for qualification and aptitude test is formalized as syntax examination rules, consistency check, and, since each

layout is designed syntactically, a set of identification rules that classify the syntactic design objects into classes of semantic structures.

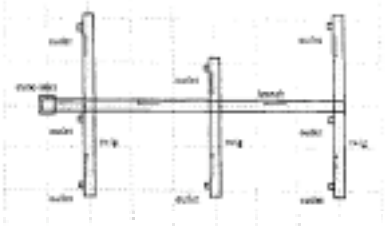

Figure 3: A return-air subsystem layout

The domain ontology provides a semantic view on the syntactically qualified design objects and identifies these objects as instances of generic semantic structures that we call concepts. Since all concepts are allocated within a deep knowledge network, the characteristic features of the involved objects are now accessible and can be determined through the classes. Having the deep knowledge determined for a proposed layout, we are now able to establish a set of useful information about the meaning and permissible use of the involved concepts and topological arrangements in the proposed case. The result of this stage is a set of topological relations between a set of semantically interpreted design objects in terms of their permissible use and relations. We make all these information transparent for the user and serve the purpose for the call Concepts and Topological Information.

To meet the requirement of evaluation, that is to determine whether the obtained solution is assumable and workable due to quality assessment rules, we give a rather detailed elaboration of an examination process for another return-air layout as shown in figure 4.

Some of the quality assessment rules for the examination of coherence are the following:

(a) All involved concepts in the considered case have to be connected together in order to let the return air from all get into the trunk-inlet. Result: O.K.

(b) There is no connecting duct which does not end in an in- or outlet. Result: violation, there exists a duct that does not end in an in- or outlet.

(c) There must be an include relation for the connection of a branch to the trunkinlet. Result: violation, there exists only a contact relation.

Another set of examination rules are classified as Spatial Ordering and Organization Maxims. For this kind of assessment, we need some more information. The necessary geometric-topological properties of the concepts, e.g. size, capacity etc., get calculated and determined on the design board and are put at assessments disposal. The results of this examination attest whether the involved concepts and their topological relations are conform relative to a specific agreements. Examples of this examination due to the example shown in the figure 4 a-e:

(a) All outlets have to be situated in the lowest or highest layer of the technical service space division within a certain distance (determined on the basis of the design grid) and orderly due to a certain topological configuration. Result: O.K.

(b) The branch is to be placed in middle layer of the installation service space only if it is a big branch (determined through its radius). Result: O.K.

(c) The laying direction of the ducts in each layer should be orthogonal to the ducts of its neighbor layers. Result: O.K.

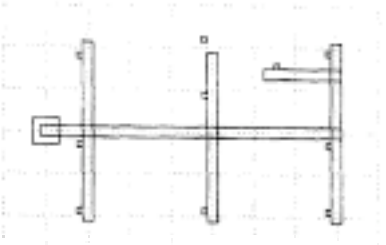

Figure 4: A spatial and organizational well arranged, but not coherent return-air layout

4 Technical underpinning

As we have outlined in the previous sections, the process of designing an artefact is characterized as a non-linear stepwise refinement process. The consequence Is that we have to deal with incomplete knowledge and take precautions for a stepwise extension as well as goal oriented modification of the stored knowledge without incurring' the whole cost of re-representation and reorganization of the whole system. Each step of knowledge extension has to proceed in a guided way which ensures that we are not going to sink in a mass of unorganized and inconsistent knowledge. The scope of completion or modification of some knowledge chunks should be kept local and is not allowed to cause a global modification throughout the whole program.

Therefore, the DOM conceptualization, to put it concisely, has a throughout object-oriented realization.

The object-oriented paradigm allows software developers to create units of generic functionality that are specific for the considered domain but highly reusable in future tasks and are shareable units of information within a collaborative framework for that particular domain. According to the object-oriented technology a considered domain is to be represented as structured model consisting of objects. The behavior of the objects -represented by a set of methods - is attached and the communication and cooperation between the objects is carried out by sending messages to each other. The common features and behaviors of a set of object instances are glued together and are referred to as classes. Classes are usually organized in a hierarchy. The lineage from class to object instance may carry the capabilities of all parental types. This is called inheritance.

4.1 Knowledge representation

Following the object-oriented knowledge modelling, the domain knowledge is represented as a hierarchy of classes. Classes are generic semantic structures and we refer to them as concepts.

```
(defclass knowledge () ((concepts) (specials) (partials)))
```

The extension of the scope of knowledge is easily done through addition of classes. The scope of knowledge in the DOM system is supposed to include also cases and schemes as classes. Schemes are a set of design patterns for certain subsystems.

```
(defclass knowledge () ((concepts) (schemes) (cases) (partials) ((specials)))
```

The concept model can be viewed as a network due to various associations. The most important class associations are specialization, partialization. The specialization part represents the current state of the involved classes in terms of a taxonomy, the part-ofrelation between the classes is represented by the partialization. Concepts include the following features:

```
(defstruct (concept (:type list)) name crspace crtime analyse synthesize)
```

As mentioned before, the subject of our ongoing work is to support adaptation of former layouts that are stored in a case base. Each concept name has a corresponding classification term in the real world, e.g. trunk, duct, in- and outlets, etc. The creation-space (crspace) comprises the geometrical data.

Corresponding to the methods of a class, the procedural knowledge is represented in the parts analyse and synthesize. The methods are formalized in rules and constraints and cause an examination process on the objects of a layout of current interest as outlined in the previous section.

(defstruct (concept (:type list)) name crspace crtime analyse synthesize adapt)

Since a proposed layout includes arrangements of physical and abstract domain objects, the main issue is to determine and establish links between the objects that comprise a proposed layout and the specified classes in the formal representation. In other words, the objects that are involved in the proposed layout are subjects of classification due to a set of identification rules. The specific arrangement of the objects is also kept in terms of aggregate relations and topological relations. Examples for topological relations are e.g. contact, overlap, etc.

(defclass data () ((objects (concepts) (aggregates) (contact) ... (includes)))

4.2 System organization

The organization and interaction between the system components is also organized in the object-oriented paradigm. The main system classes are: Transfer, Data, Knowledge, and Behavior and each of which include static components (slots) and dynamic components (methods). A class specification together with its set instances build a so called base. So we get a transfer base, a data base, a knowledge base, and a behavior base. As mentioned above, each of these classes are specified by a set of slots and a set of methods. The slots are containers for sets, for instance sets of files, of building objects, of connections, of concepts, or of assistance operations. The methods are defined to operate on the whole instances or on single objects of the sets that are stored in the slots of instances. Examples for this kind of methods are create, delete, get, eval, draw.

We reckon with the merit that it is possible to work simultaneously with a set of transfer bases, data bases, knowledge bases, and behavior bases and to connect them with each other in several not pre-determined ways. Our ultimate goal is to make an isomorphic map between the structure of the system kernel and the structure of the user interface. Having achieved this goal it would guarantee a clear, transparent organization of the whole system and would allow the user to influence internal processes conscious and goal driven. The following list shows how the internal system units are currently mapped to external user interface units (widget).

(a) class <-> frame as container
(b) class name <-> menubutton in a frame
(c) method of a class <-> menu-entry in a frame
(d) instance of a class <-> listbox-element in a frame

Each class of the kernel has a frame with a corresponding part on the external level. The first part is the class name that is realized as menu-button. Via the bound menu, all methods of the class can be activated. In general, the methods work on instances of the class. The instances are contained as elements in a listbox belonging to the frame, too. All these widgets are the contents of a permanent control window. Figure 5 shows the objectoriented system organization of DOM.

4.3 Implementation

The DOM application system is the result of a cascade of system development, shown in figure 6, starting with the generic knowledge-based development system, FAENSY [Oertel 94] which has been implemented in Allegro Common Lisp and runs on UNIX workstation with a user interface based on Tcl/Tk [Ousterhout 93]. The next system was the DOM development system using FAENSY as development tool. The DOM development system is especially designed for development issues in the course of ontology formalization. It is implemented in Allegro Common Lisp under UNIX.

Figure 5: System organization hierarchy

The system architecture is analogous to the conceptual description, namely:
(a) interface gate <-> transfer base,
(b) design board <-> short-time data base,
(c) domain ontology <-> knowledge base, and
(d) assistance function <-> behavior base.
The main parts of the system, mentioned above, and an interaction framework comprises our system architecture. Currently, the DOM system contains about 100 concepts and layouts including up to 100 design objects have been analysed successfully.

5 Summary and outlook

Figure 6: The cascade of the system development

We have shown -- conceptually, implementationally and practically -- that it is not only possible and desirable, but absolutely necessary to extent traditional design support systems with techniques of a powerful domain ontology modelling. Our ongoing work will proceed in terms of the following tasks: The implementation of schematic knowledge as a connection between cases and concepts, the realization of case adaptation. We use the potential of the specified quality assessment rules as well as the analysis results as a basis for the realization of the case adaptation task. Following the ultimate goal of supporting the adaptation of a retrieved case to the case at hand, the rectification support services are viewed as the first necessary step towards a case adaptation framework. For this purpose, we have specified and are now about to implement an extension of the realm of knowledge in the design board component.

6 Endnotes

[1] This research was supported by the German Ministry for Research and Technology (BMFT) within the joint project FABEL under contract no. 011W104. Project partners in FABEL are German National Research Center for Computer Science (GMD), Sankt Augustin, BSR Consulting GmbH, Munich, Technical University of Dresden, HTWK Leipzig, University of Freiburg, and University of Karlsruhe.

[2] A shared and reusable knowledge base within a collaborative framework is referred to as an ontology.

[3] Plausibility and workability are to taken as evaluation principle that have to be distinguished from the logical criteria of correctness.

[4] DOM in the german language means cathedral. The given translation may help for a better understanding of the figures in this paper.

7 Bibliography

Alberts, L.K.; & Dikker, F.: Integrating Standards and Synthesis Knowledge Using the YMIR Ontology. In: Gero, J. S.; Sudweeks, F. (eds): Artificial Intelligence in Design 94. Kluwer Academic Publishers, Dordrecht 1994, pp. 517- 534.

Bakhtari, S.; & Bartsch-Spoerl, B.: Bridging the Gap between Al Technology and Design Requirements. In: Gero, J. S.; Sudweeks, F. (eds): Artificial Intelligence in Design 94. Kluwer Academic Publishers, Dordrecht 1994.

Bakhtari, S.; Bartsch-Spoerl, B.; Oertel, W.; & Eltz, U.: DOM: Domain Ontology Modelling for Architectural and Engineering Design. Fabel-Report Nr. 33, GMD, Sankt Augustin 1995.

Beheshti, M.R.; & Zreik, K. (eds): Advanced Technologies: Architecture, Planning, Civil Engineering. Elsevier 1993.

Branki, C; Douglas, J.; & Bailey, D.: Agent Communications Server for Shared Ontologies in Planning and Design. Third international Conference on Artificial Intelligence in Design. Workshop Notes: A Semantic Basis for Sharing Knowledge and Data in Design, 1994.

Fabel Consortium: A Survey of FABEL. Fabel-Report Nr. 2, Sankt Augustin, GMD 1993.

Haller, F.: ARMILLA - ein Installationsmodell. Institut fuer Baugestaltung, Baukonstruktion und Entwerfen, University of Karlsruhe,1985.

Hovestadt, L.: A4 - Digital Building - Extensive Computer Support for the Design, Construction, and Management of Buildings. In: U. Flemming and S. van Wyk (eds.): Proceedings of the Fifth International Conference on Computer-Aided Architectural Design Futures CAAD Futures '93. NorthHolland, Amsterdam, 1993, pp. 405-422.

Neches, R.; Fikes, R.; Finin, T.; Gruber, 1.; Patil, R.; & Senator, 1.; Swartout, W.: Enabling Technology for Knowledge Sharing. AI Magazin 1991.

Oertel, W.: FAENSY: Fabel Development System. FABEL Report Nr. 27, GMD, Sankt Augustin, 1994.

Ousterhout, I. K.: An Introduction to Tcl and Tk. Addison-Wesley Publishing, 1993.

Pu, P.: Issues in Case-Based Design Systems. In: Al EDAM, Vol. 7, Nr.2, Academic Press 1993.

Silverman, B.G.: Research Workshop on Expert Judgment, Human Error, and Intelligent Systems. Al Magazine, FALL 1993.

Walther, J.; Graether, W.; Oertel, W.; Schmidt-Belz, B.; & Voss, A.: An Open Architecture for Multiple Case Retrieval Methods. In: Keane, M.; Haton, J.P.; Manago, M. (eds): Preprints of the Second European Workshop on Case-Based Reasoning. AcknoSoft Press, Paris, France 1994, pp. 373-380.

8 Acknowledgement

We are indebted to Brigitte Bartsch-Spoerl. Her valuable contributions permitted us to improve the DOM conceptualization and functional specification considerably. It is thanks to Ulrike Eltz that a set of basic functions of the DOM development system have been implemented efficiently.