PUC

Series: Monografias em Ciência da Computação
No. 3/90

INTELLIGENT HYPERTEXT FOR NORMATIVE KNOWLEDGE IN ENGINEERING

Daniel Schwabe
Bruno Feijó
Werther G. Krause

Departamento de Informática

PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO DE JANEIRO
RUA MARQUÊS DE SÃO VICENTE, 225 – CEP-22453
RIO DE JANEIRO – BRASIL

PUC Rio - Departamento de Informática

Series: Monografias em Ciência da Computação, Nº 3/90
Editor: Paulo A. S. Veloso January, 1990

INTELLIGENT HYPERTEXT FOR NORMATIVE KNOWLEDGE IN ENGINEERING

Daniel Schwabe
Bruno Feijó
Werther G. Krause

This work has been partially sponsored by FINEP

In charge of publications:

Rosane Teles Lins Castilho
Assessoria de Biblioteca, Documentação e Informação
PUC RIO, Departamento de Informática
Rua Marquês de São Vicente, 225 - Gávea
22453 - Rio de Janeiro, RJ
BRASIL

Tel.:(021)529-9386 TELEX:31078 FAX:(021)274-4546
BITNET: userrtlc@lncc.bitnet

Intelligent Hypertext for Normative Knowledge in Engineering

Daniel Schwabe
Bruno Feijó*
Werther G. Krause**

Departamento de Informática
Pontifícia Universidade Católica - RJ
R. Marquês de S. Vicente, 225
Rio de Janeiro, RJ 22453, BRASIL
Ph.: +55 21 2744449
Fax: +55 21 529 9485
E-Mail: USERSCHW@LNCC.Bitnet
USERBRUN@LNCC.Bitnet

ABSTRACT:

We present a system that combines hypertext with a semantic representation of engineering norms. Since the representation is done via a Prolog encoding of an And/Or Graph, it is possible to discuss the relation between the execution of the (representation of the) norm and navigation in the hypertext. The system incorporates an interpretation of the norm by experts, and it is shown how this interpretation can be regarded also as an hyperview onto the hypertext.

1. Introduction

In many areas of human activity, norms set a number of requirements that control several of its properties; this is particularly true in the area of engineering. In previous works [Feijo 88], it has been argued that engineering norms, in order to be useful, must have a very logical structure, allowing no room for imprecision and ambiguity. On the other hand, these norms are typically consulted by engineers with a varied number of purposes, ranging from mere curiosity to very specific design questions.

In this paper, we present a system whose goal is to provide access to engineering norms to a wide range of users, both humans and machines. In order to achieve such a goal, a multiple representation approach is used, combining a logic programming encoding of an And/Or graph representing the norm and an interpretaton of the norm by experts[1], and a hypertext representation useful for access by humans. This approach provides a very rich interface, in good part due to the interaction allowed by the multiple representations.

In what follows, section 2. will discuss the representation of normative knowledge, in particular for engineering; section 3. will discuss an hypertext representation of

---

* Also Expert Systems Lab., Imperial College
** Also Promon Engenharia Ltda.

[1] Incidentally, the word "Intelligent" was included in the title of this paper in the sense of including this representation. It would also be appropriate to replace it by "Knowledge Based".

the norm; section 4. will discuss the integration of these representations; and section 5. draws some conclusions.

2. Normative Knowledge

The study of norms (and regulations) as a type of knowledge can be found in the areas of Office Automation [Paolini 88] and Legal Reasoning [Sergot 86, Schwabe 89]. In the case of engineering regulations, this type of knowledge presents the following characteristics:

- it is supposed to have an explicit and precise interpretation (in contrast with the "open nature" of law in the field of legal reasoning);
- it is available in written form;
- it is supposed to be complete and correct;
- it presents no uncertain facts;
- it requires no vague reasoning;
- it presents a simple structure of discourse (in contrast with the complex structures found in the area of Text Generation [Mann 88]).

The representation of this type of knowledge in an artificially intelligent system involves two main tasks: its structuring and the translation of the written provisions into a form suitable for symbolic manipulation. However, these tasks are not easily accomplished, because they should be carried out with the entire design process in mind. Moreover, revisions of the text are required when the knowledge is incomplete and/or incorrect [1]. In this case, a knowledge engineer might be required to explicate the results intended by the norm drafters.

In this context, formalization of normative knowledge allows the development of a conceptual model to render regulations more complete and with added rigour. This paper presents such a model based on a graph, Horn clause and hypertext representation.

2.1 The Graph Representation

The structure of a design code can be represented by an And/Or graph in which provisions call subprovisions (Fig.3). By definition, And/Or graphs assume the inclusive definition of OR; therefore, at least one (and possibly more) subprovisions hold.

At the leaf nodes it is possible to include the representation of additional knowledge, such as typed data items. For example, "input data item" (axial load), or "external data item" (some calculation procedure) [Feijo 89].

[1] Incorrect in the sense that the results are not those intended by the norm drafters.

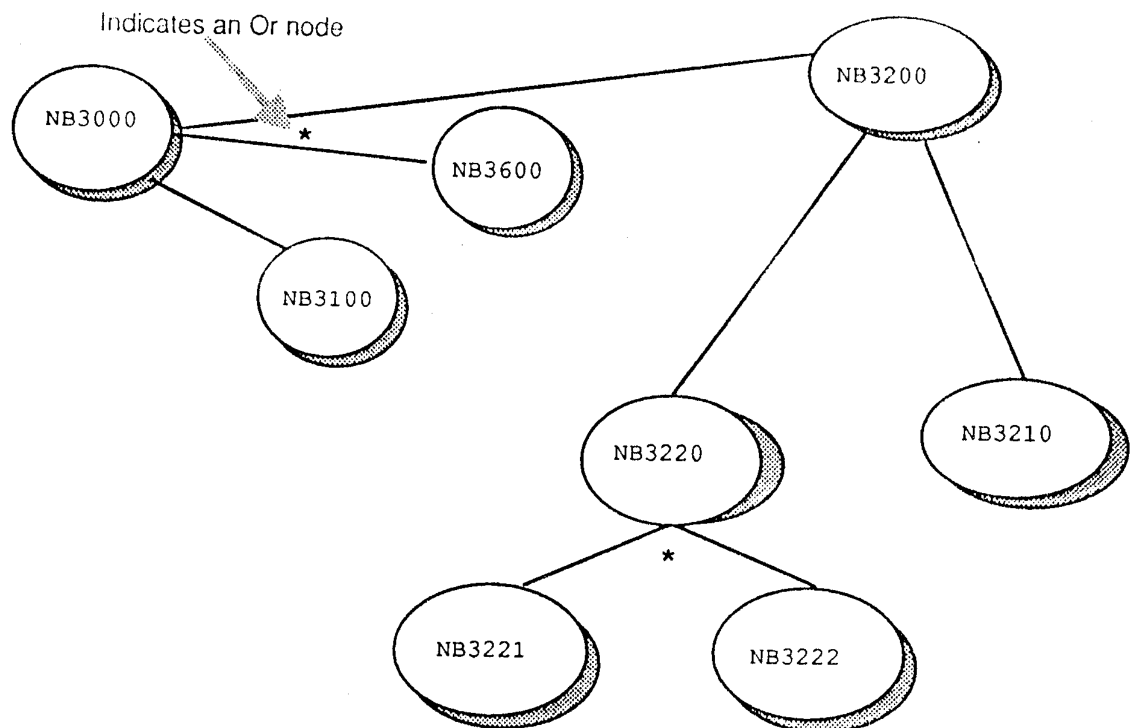

Figure 0. And/Or Graph corresponding to a norm

This kind of representation presents the following properties:

- Conformance checking should be represented by a unique AND subgraph, otherwise the norm presents ambiguities or multiple interpretations;
- Cumbersome forms of cross referencing, lack of connections and loops can be easily detected ;
- The graphs (and especially the subgraphs) show the consequences of changing a particular data item. This property can be used to support "intelligent decisions" made by Design Automation Systems. Also, it allows an effective redesign;
- These graphs share some properties with the information networks proposed by Fenves and Wright [Fenves 77]. In particular, the graph can easily be converted into a depth first spanning tree. This tree can be used by writers of design codes to identify cross references and to achieve better textual expression.

2.2 The Horn Clause Representation

An And/Or graph can be converted into a set of Horn clauses [Kowalski 79, Lloyd 84] in a very strightforward way. The logical structure of a norm can then be cast into a logic program whose execution is equivalent to the execution of the graph.

In the general case, each node of the graph may correspond to several Horn clauses. These Horn clauses must be mutually exclusive rules which should define the provisions of the norm completely. Moreover, these clauses may contain conditions which are not described explicitly in the And/Or graph, such as equations. The mutually exclusive rules assure the property of uniqueness, i.e. that the norm yields one and only one result with respect to a given decision.

For example, the logic program below corresponds to the And/Or graph described in Fig. 3; `Entity` is any piece of the system being designed that one wishes to verify conformance to the norm.[1]

```
nb3000(Entity) :-
     nb3100(Entity),
     nb3(Entity).

nb3(Entity) :- nb3200(Entity).

nb3(Entity):-  nb3600(Entity).

nb3200(Entity):-
     nb3210(Entity),
     nb3220(Entity).

nb3220(Entity):-
     nb3221(Entity),
     nb3222(Entity).

nb3221(Entity):-
     ...
```

3. Hypertext Representation of Norms

One of the earliest applications of Hypertext was for manuals [Acksyn 88, Hayes 89], many of which included normative knowledge of some kind. In representing engineering norms, the first type of link provided was simply a reproduction of the syntactic structure of the norm, which is a tree.

3.1 Views

Among the many uses of the norm, a few can be singled out as deserving special treatment, namely, those related to design problems. In these situations, designers are accessing the norm in order to either analyse a particular aspect of a design, e.g., conformance testing, or to derive (hard) constraints on the design options. In either case, a very specialized type of knowledge must be employed in order to use the norm properly; this knowledge is almost never present in the norm itself. Rather, one must use an "abstract model" which was probably present in the minds of the drafters of the norms, but is nowhere else present.

In order to support this type of access to the norm, we have included what we have called a *specialist's view*[2] of the norm. This view contains specialized knowledge pertaining to the most common uses of the norm for design problems. For instance, the knowledge about "Membrane Protection" in the norm ASME - Class 1 [ASME 84] concerns the limits that

---

[1] As argued in [Tese Bruno], a norm should be drafted in such a way that it refers to only one entity at a time. This is reflected in the example by including only *one* parameter in the predicates.

[2] We define a *view* as a (sub)set of the nodes and links of the hypertext.

stress intensities should meet; it is *implicit* that all the primary stresses are only evaluated on a membrane. This general notion is not clear for the reader of many parts of the norm where this concept of "membrane protection" is not explicitly stated. By accessing the specialist's view, this can be clarified.

From a logical programming standpoint, this view can be regarded as providing both additional predicates and control knowledge to the pure logic program of the norm itself. From an hypertext standpoint, the nodes corresponding to this view can be regarded as "commentaries", providing additional insight into the meaning of the norm. Thus, at appropriate points of the norm, a link is provided to allow access to this information.

For example, when the user consult this norm, he or she may not understand the criteria of primary membrane stress intensities. The system provides a way for the user to access the interpretation of this provision simply by clicking at a button named *Assistance* .

Figure 1 contains a schematic representation of the relation between the norm and a specialist's view. This view may sometimes include *virtual links* between nodes of the norm itself, i.e., links that are not present in the norm, but are included by the specialist. The norm itself is also called the *standard view.*

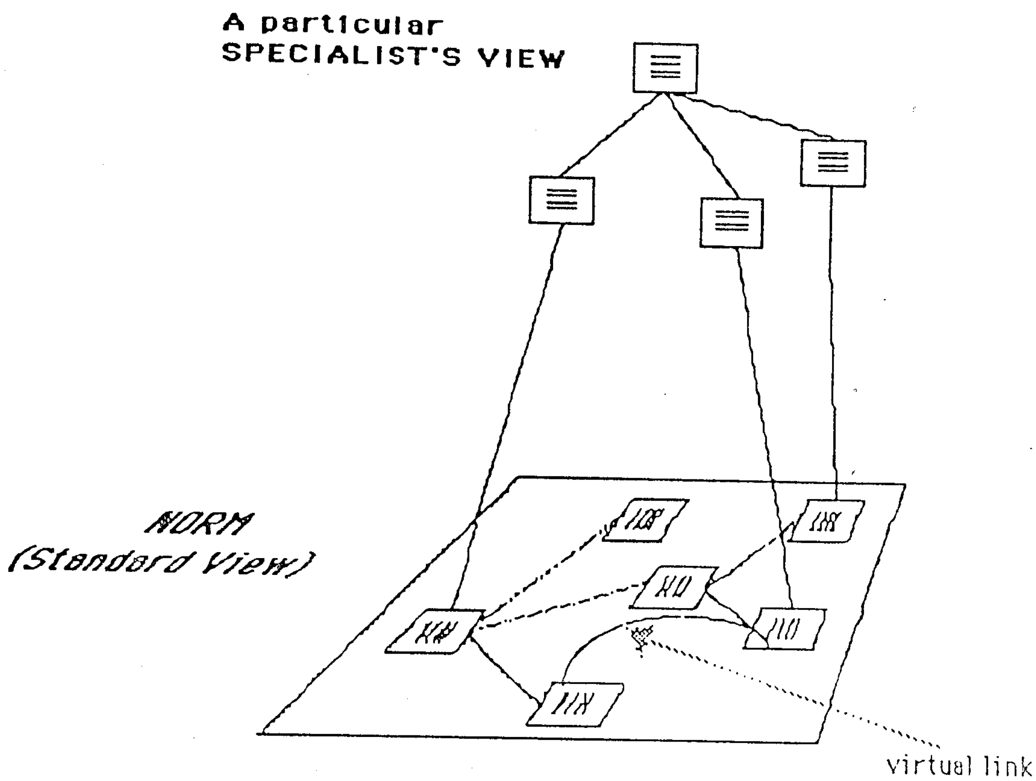

Figure 1 - Relationship between the Norm and the Specialist's View

3.2 Glossaries

As usual in this kind of document, a glossary access is also provided - by clicking on terms highlited in the text of the norm, a glossary definition of the term is presented, as well a list of other places in the norm where the term appears. The user is then free to examine other points of the norm by clicking at the appropriate entry in the index associated with any glossary entry. Figure 2 shows an example.

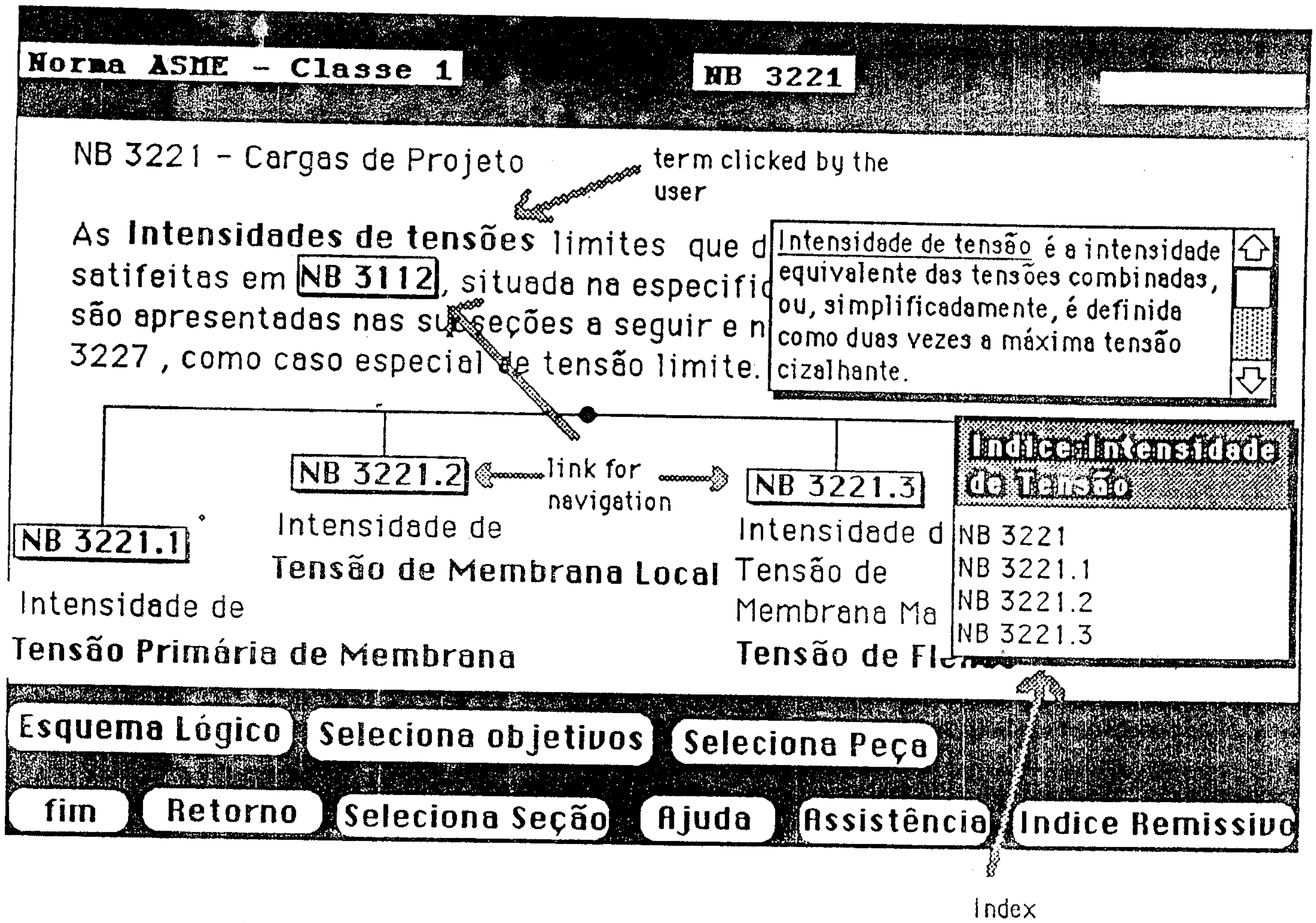

Figure 2 - Example of the use of the Glossary and the associated Index

It should be noted that glossary access like this is not a conventional type of hypertext link, in the sense that it is established *by content* and not by *position*. In other words, if a new node is included in the hypertext which contains one or more terms present in the glossary, the link to the glossary entry is automatically built by a special utility. This is only possible in systems such as Hypercad [Apple 88], which are programmable.

From a conceptual point of view, the glossary could be regarded as being on a separate "plane", in such a way that it is accessible from *any* node of the hypertext (and specialist's view). Although this way of organizing this type of information is not *necessarily* so, we feel it makes the design of the hypertext much cleaner.

3.3 User Model

Most of the above information is presented to the user according to a *user model*. When first entering the system, the user is asked about his/her level of expertise and familiarity with the norm, and according to the answers, the user is classified as beginner or advanced[1]. Therefore, in principle, most of the specialist's view is not accessible by beginners, since it covers concepts not normally known by them. Another important difference is in the help messages, which are more detailed for beginners (when relating to the norm itself, not to the system).

Since the philosophy followed in designing this system is to allow the user as much freedom as possible, we have not *prohibited* a beginner from accessing parts of the hypertext which were not included in his/her "view". Rather, when the user enters a part of the hypertext which is not included in the user's model, the system *remembers* this "exit" point, and provides an easy mean for the user to return to it later on, if so desired. Thus, this exit point functions as a safe "anchor" while the user navigates into "unknow" portions of the hypertext. The user can return to if s/he gets lost in the navigation process. Presumably, navigation is easier when the users restrict themselves to the portions of the hypertext that are part of the corresponding user model.

In Fig. 4, part of a norm is represented. Supposing the user has browsed through nodes `NB3000`, `NB3200`, `NB3220`, and `NB3221` of the norm ASME-Class 1. `NB3000` is the article about design provisions. The others contain rules about design by analysis, and about establishing load and design criteria. At this point, s/he decides to examine the specialist's view about the concept of "Membrane", mentioned in node `NB3221`. After browsing about, the user wishes to return to the norm itself, at the point s/he left it. This will be achieved by pressing the button *Return to Context*.

---

[1] We are aware that a lot of research has been and is being done on user models. However, since this is *not* the main focus of our research at this point, we have deliberately chosen a very simple one.

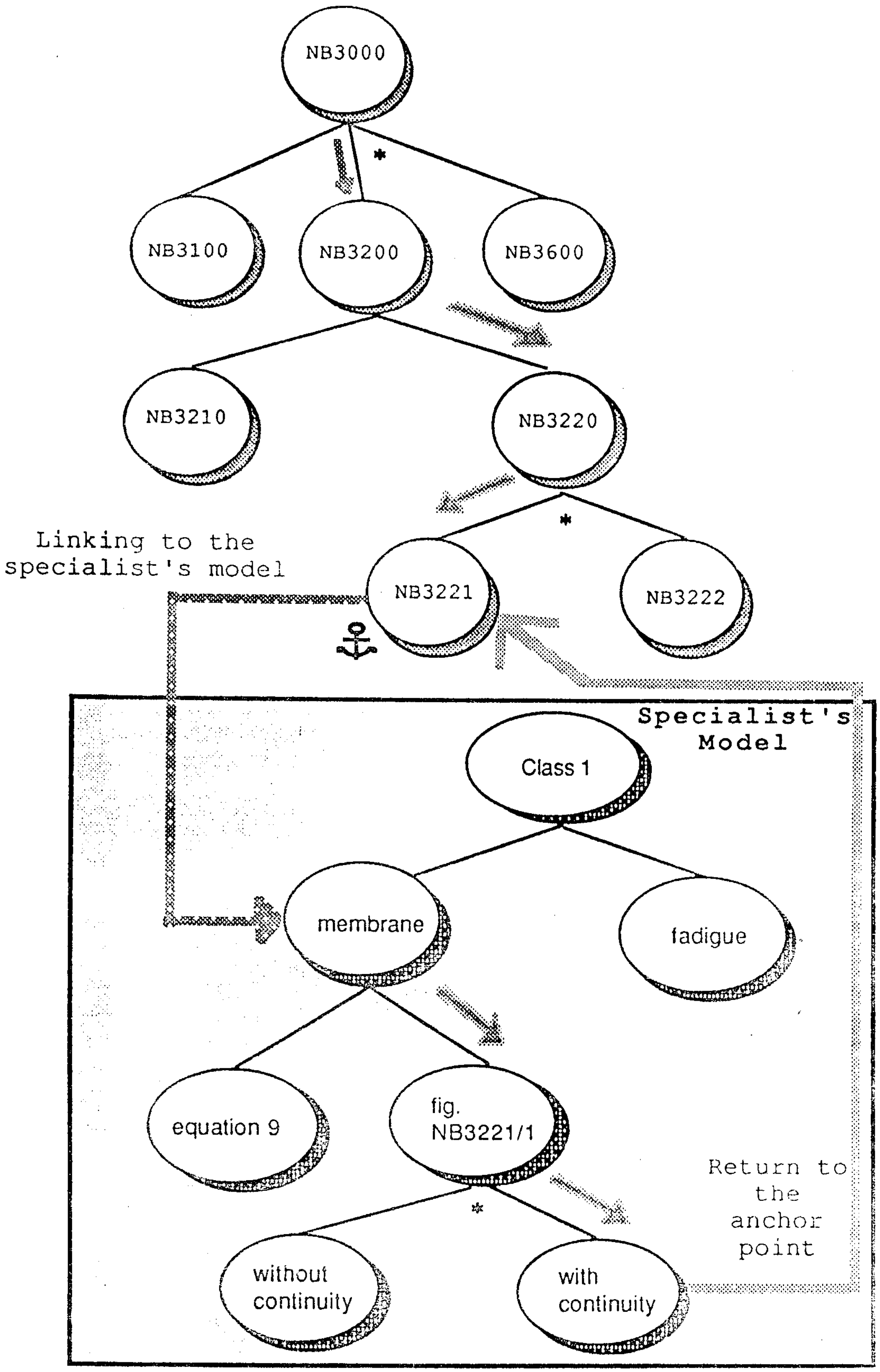

Figure 4 - Example of Navigation and Anchor Points

4. Integrating Semantics with the Hypertext

4.1 Logic Programs

As discussed previously, it is possible to derive a logic program from an And/Or graph representation of the norm. Since there is a portion of the hypertext representation of the norm that is isomorphic to the And/Or graph, it is also very simple to have a corresponding logic program associated with it. For each node, a button called *Logical Interpretation* is available which, when clicked, presents the fragment of the logic program corresponding to that part of the norm. This feature is normally available only to advanced users.

It should be noted that, in general, the logic program may contain more detail than that included in the And/Or graph, such as equations. Therefore, one should regard the graph as giving the overall logical structure of the norm, whereas the logic program gives a more detailed interpretation of it.

4.2 Executing the Norm

Once a logic program is available, it is possible to execute it. This is equivalent to posing questions about a particular object being considered, and trying to answer it from the specification in the norm. A typical example is verifying if a pipe satisfies certain restrictions imposed by the norm on the stresses it must withstand.

Normally, simple questions can be answered directly from the norm, i.e., by excuting the logic program corresponding to it. However, more complicated questions, whose answers are not evident from the norm, must be answered by looking at the specialist's model, which also has a corresponding logic program. In fact, the specialist's model can be regarded as an expert system on the subject of the norm.

From the hypertext's standpoint, a button named *Select Goal* is available, which, when clicked, shows the various user goals know to the system. Among these, there is the option to evaluate the norm, which, when chosen, initiates a dialog to obtain the specifications of the entity being evaluated, and applies the correspondig logic program (i.e., the logic program corresponding to the sub-graph rooted in the node in question) to it.

4.3 Explanations

A simple form of explanation can be obtained once an evaluation has occurred using the logic program associated with a node. A successful execution of the logic program corresponds to an And sub-graph which satisfies the And/Or graph in question.

One can regard this sub-graph as a possible sequence of nodes of the hypertext to be presented to the user; it will contain all the sections of the norm relevant to the decision just made. Thus, the user can browse through these nodes by successively clicking the button named *Explanation*. This button is available only when a successful evaluation has been made. In the case of failure, the failure node is presented to the user.

It is clear that this is a very crude form of explanation. However, it incorporates a very interesting notion, namely, that of deriving meaningful *navigation paths* in the hypertext from the semantic description. This navigation path could also be used by higher level planning processes in an design automation system.

5. Conclusions

This paper has briefly discussed the integration of a semantic description of normative knowledge with hypertext. This system incorporates a few interesting features, which illustrate the advantages of having multiple representations:

- Derivation of the structure of (part of) the hypertext from the logical structure of the norm;
- Use of multiple views;
- Use of the "trace" of the execution of a logic program as a way to define a navigation path in the hypertext;
- Incorporation of some of the user's goals in the user model, via the use of a specialist's view.

Besides these facilities, more conventional ones were also included, such as coordinated glossary and index, accessible anywhere in the hypertext. The proposed glossary is particularly interesting because it can be automatically constructed, not requiring manual insertion of links in each node of the hypertext.

A number of open questions and problems remain. First, a more sophisticated user model should be developed, incorporating more of the user's goals, as well as background. Associcated with this is the incorporation of more complex types of explanations, such as negative answers, coordination with user's intentions, inclusion of the specialist's views, etc...

Finally, it is our intention to integrate this system into a more compreehensive CAD environment, where one can, for instance, obtain all the data regarding a design from a database. Possible outputs of the system, besides evaluation of conformance, would be (hard) constraints on incompletely specified designs.

A prototype of the system described in this paper is being developed using Hypercard; more details can be found in [Krause 90].

Acknowledgements - Many of the ideas presented here are the result of discussions with P. Paolini, F. Garzotto and S. Danielluzi from the Dipartimento di Elettronica, Politecnico di Milano; and D.L. Smith from the Expert Systems Lab., Imperial College, London.

6. References

[Acksyn 88] AKSCYN,R.M., McCRACKEN,D.L. and YODER,E.A., KMS: a distributed hypermedia system for managing knowledge in organizations, Commun. ACM, 31, (7), July 1988, p. 820-835.

[APPLE 88] APPLE Computer, Inc., Hypercard User's Guide, 1988.

[ASME 84] AMERICAN SOCIETY OF MECHANICAL ENGINEERS, ASME Division 1, Subsection NB: Class 1 Components, Subsection NB 3000 (Design), USA, 1984.

[Feijo 88] FEIJO,B., Fundamental Steps towards an Intelligent CAD System in Structural Steel, Ph.D. Thesis, Expert Systems Laboratory, Imperial College, March 1988.

[Feijo 89] FEIJO,B.; DOWLING, P.J.; SMITH, D.L. , Incorporation of Steel Design Codes into Design Automation Systems, Proceedings of IABSE Colloquium, Expert Systems in Civil Engineering, Bergamo, Italy, 1989, p. 393-402.

[Fenves 77] FENVES,S.J. and WRIGHT,R.N., The representation and use of design specifications, in Hall,W.J. (ed.), *Structural and Geotechnical Mechanics*, Prentice Hall, 1977, p. 278-304.

[Hayes 89] Hayes, P.; Pepper,J., Towards an Integrated Maitenance Advisor, Proceedings of Hypertext'89, Pittsburgh, 1989

[Kowalski 79] KOWALSKI,R., *Logic for Problem Solving*, Artificial Intelligence Series, Vol. 7, Elsevier-North Holland, 1979.

[Lloyd 84] LLOYD, J.W., *Foundations of Logic Programming*, Springer-Verlag, 1984.

[Mann 88] MANN,W.C., Text generation: the problem of text structure, in McDonald,D. and Bolc,L. (eds.), *Natural Language Generation Systems*, Springer Verlag, 1988,p. 47-68.

[Paolini 88] PAOLINI,P. et al., Knowledge based document generation. In Lamersdorf,W.(ed.), *Office Knowledge: Representation, Management, and Utilization*, Elsevier Science Publ., North Holland, IFIP, 1988, p.179-195.

[Schwabe 89] SCHWABE,D.; PINHEIRO,C.S. Expert systems and social welfare benefits regulations: the Brazilian Case, *2nd International Simposium on Artificial Intelligence in Economics and Management*, Singapore, North Holland, 1990

[Sergot 86] SERGOT,M., KOWALSKI,R.A., KRIWACZEK,P. HAMMOND,P. and CORY,H.T., The British Nationality Act as a Logic Program, Commun. ACM, 29, (5), May 1986.