The Knowledge Engineering Review, Vol 9:2, 1994, 147–162

Approaches to representing and reasoning with technical regulatory information

PAUL WAI HING CHUNG

Department of Chemical Engineering, Loughborough University of Technology, Loughborough LE11 3TU, UK

DAVID STONE

The Building Directorate, Scottish Office, New St. Andrew's House, Edinburgh EH1 3SZ, Scotland

Abstract

Regulatory information has a direct and significant effect on the safety, economics and quality of many industries' operations and end-products. However, standards documents are often voluminous and can be complex. Users of such standards need to invest considerable time and effort to become familiar with the structure and content of the documents. Therefore, it has long been recognized that computer handling of this information is a potentially useful application area. This paper provides an overview of the work that has been done to provide computer systems that will help both users and authors of regulations. The emphasis is on using advanced symbolic computing techniques that would help in accessing, interpreting and applying the information.

1 Introduction

Many aspects of some industries, like the construction and process industries, are governed by regulatory information. This information has a direct and significant effect on the safety, economics and quality of the industries' operations and end-products. The information is contained in a range of document types extending from the mandatory to the advisory and includes national and local regulations, codes of practice and product standards. For the purpose of this review, the terms *regulations*, *codes of practice* and *standards* are used interchangeably.

Standards are often voluminous and can be complex, consisting of contributions from a number of authors. The information is costly to produce in terms of both time and effort. It requires the involvement of leading experts in the subject domain and, usually, protracted consultation and review procedures which lead to a cycle of revision and re-drafting.

The volume of information involved compels individual users to invest considerable time and effort in locating relevant parts. Once located there are often further difficulties associated with understanding and correctly applying the requirements. Idiosyncrasies of style and inadvertent inconsistencies or errors in the texts can, in the case of mandatory requirements, lead to conflicting interpretations by users and enforcing authorities.

Legislation and similar regulatory information has long been recognised as a potentially rich area for computer applications. Sergot (1991) provides a comprehensive survey on research carried out in the area of conventional legislation. However, conventional legislation has been found to contain many structural and conceptual problems which defy easy analysis and representation (Gardner, 1987; Susskind, 1987). Standards in technical domains do not share these problems to the same extent because they focus on the physical rather than the abstract. This paper reviews work carried out related to technical standards, in particular building standards, because it is in this domain that most of the research has been done to date. The paper attempts to compare and contrast the various approaches that have been tried.

2 Standards processing

Computer systems for standards processing can be grouped into three general categories–information retrieval, conformance checking and authoring. Information retrieval systems provide on-line access of standards documents to users. The main issue concerning the development of such a system is how to structure the documents so that a user can locate the appropriate part(s) easily. Section 3 considers two different approaches that are used.

The main body of research in standards processing is in automating design conformance checks. A *standard* is a collection of *provisions* that are intended to ensure or promote the general objective of the standard, and a *provision* is defined as a measure that will enable some entity to attain a satisfactory performance with respect to a particular limit state (Cronembold & Law, 1988).

There are different ways of implementing conformance checkers. One way is to write programs using conventional procedural languages. For example, EVNSTD (Crawley & Boulin, 1991), which stands for ENVelope STanDard, is a C program for checking energy efficiency of new buildings as specified in ASHRAE/IES Standard 90.1-1989 (ASHRAE, 1989). This hard-coding approach has been criticised for two reasons (Cronembold & Law, 1988; Garrett & Hakim, 1992). First, the program developer (a computing person) is often not the most familiar with the standards and may lead to misinterpretation of the provisions. Second, major efforts are required to rewrite or update the source code when revisions are made to the design standards.

Therefore, research effort has been concentrated on developing generic standards processors. The aim is to find a formal and yet convenient way of representing provisions so that the domain information can be easily understood by users and can be manipulated by the computer. The different approaches that have been tried are described in section 4. Issues related to integrating design checkers with conventional CAD or CAE systems are outside the scope of this paper and will not be considered.

Yang et al. (1993) extend previous work by incorporating case histories into their system. They argue that work on standards processing has taken too narrow a view by concentrating on provisions. Standards, as part of a legal system, has another source of information, case histories, which must be available in the consultation process. Therefore, they have developed a unified knowledge representation scheme for both provisions and case histories. This work will be described in section 5.

As the aim of standards processing is to have the standards available on-line on a computer and to have conformance checking carried out automatically, it is logical to provide support to authors to produce the standards on a computer in the first place. Research on authoring support system will be described in section 6.

3 Retrieval systems

Because of the volume and complexity of standards, a user, who may be a designer or a local authority official whose job is to administer the standards, has to invest considerable effort in order to understand the inter-relationship of the different parts. To facilitate user access, regulatory information is now being put onto computer databases (Vanier, 1991; Bourdeau, 1991). Irrespective of what hardware is used to store the information, a hypertext like interface for browsing is preferred. A hypertext system allows different pieces of information to be linked together and enables the user to move from one piece of information to another easily (Conklin, 1989). This flexibility is desirable because it is common that one part of a document references other clauses, tables, figures, graphs and equations, as well as other documents.

The main access problem lies in how to index an entire document or a set of documents so that the user can search for the relevant information without being overwhelmed by the volume of material available. There are basically two approaches. One is to use *keyword descriptors* and the other is a structured approach based on a *classification* of the concepts in the domain.

	Works	Functional requirements	Administrative constraints	Purposes
Descriptor 1	roofing	mechanical resistance and stability	builder obligations	all buildings
Descriptor 2	roofing	mechanical resistance and stability	owner obligations	all buildings
Descriptor 3	roofing	water tightness	builder obligations	all buildings
Descriptor 4	roofing	water tightness	owner obligations	all buildings
Descriptor 5	roof window	water tightness	-	all buildings

Figure 1

Products	Materials
metallic sheets	stainless steel

Figure 2

3.1 Keyword descriptors

CD-REEF is one of the largest scale projects in providing on-line access to regulatory information. REEF is a 15000 page encyclopedia consisting of a collection of about 1000 documents commonly used by French building professionals such as architects, engineers or contractors. To facilitate information retrieval REEF has been put on a CD-ROM to make the information accessible from a standard microcomputer (Bourdeau, 1991). Texts make up 75% of the database and the rest consist of 3500 tables, 9000 drawings and a large number of formulae. A huge amount of effort was spent digitizing all the documents.

Information contained in CD-REEF is indexed at two different levels: the document level and the information unit level. At the former level, the content of each document is characterized by two types of descriptors: general and technical. A general descriptor has four attributes: works, functional requirements, administrative constraints and purposes. A technical descriptor has two attributes: products and materials. Figures 1 and 2 are examples taken from Bourdeau (1991). The descriptors are used to index a document about roofing using large sheets of stainless steel.

At the second level, each document is divided into information units. Each unit is indexed by a set of keywords that describe its content. The keywords are first of all extracted automatically from the text. They are then checked by human experts and inappropriate words are deleted. The dictionary of keywords has about 5000 entries. Future work includes building a thesaurus so that a search can be extended to include synonyms and associated terms.

Assessing information from CD-REEF involves two steps, each corresponding to a different level of indexing.

1. Select document. This can be done by selecting directly from the catalogue of documents or by specifying general and technical descriptors to initiate a search. After the appropriate documents have been selected, the user can copy them to a file specific to the case in progress.
2. Select information unit. Again there are two ways of doing this. The first is by looking up the table of contents of the selected documents and identifying the sections that are of interest. The second is by specifying keywords. The user can choose the keywords from a list, which is generated by the system by merging the indexes of all the information units in the selected documents.

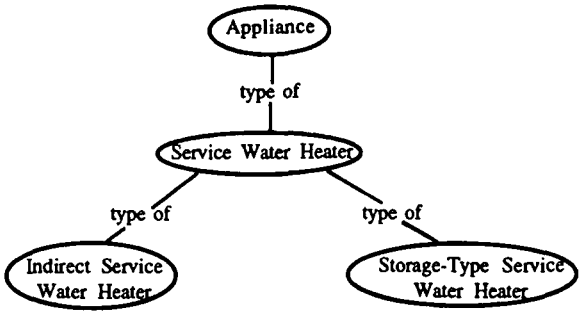

Figure 3

It is not clear how easy it is to access standards documents using CD-REEF. It could be a problem for a user to provide the appropriate general and technical descriptors, particularly if there is no guidance given by the system. A problem related to the use of keywords in step (2) is that the system has no idea how different keywords are related to one another. For example, if the user is interested in looking at regulations related to all types of heating system then he or she may have to provide an exhaustive list of keywords. There is also the risk of missing some of the keywords out.

3.2 Classification scheme

Some researchers (for example, see Harris & Wright, 1980; Hakim & Garrett, 1992) see that the way to understand the inter-relationships of the concepts contained in standards is by creating a formal structure of those concepts. An example of a recent project which gives priority to the classification problem in a standards retrieval system is the Classification System for the National Building Code for Canada (NBC) (Vanier, 1991).

The NBC Classification System has three components: a classification scheme, an article database and a retrieval interface. The classification scheme provides a way of structuring the concepts found in the NBC. It is essentially a network of nodes and links. A node represents a concept, which can be either abstract or physical. Associated with the concept is a set of properties with default values. A link represents the relationship between the two nodes that it joins together. Figure 3 shows a partial network related to the concept "service water heater". The *type-of* relation used in the example is similar to the class hierarchy as found in object-oriented systems (Graham, 1991). In a type-of hierarchy the properties of a node are automatically inherited by the child nodes. Another useful relation that is supported by the classification scheme is the *part-of* link. A *part-of* hierarchy shows the constituent parts that make up a concept. For example, a house may have the following parts: window, door, bedroom, kitchen, toilet, etc.

The classification scheme is being developed by experts using HyperCard. HyperCard was selected because it provides very good prototyping features. Each node can also be annotated by historical and contextual information. Based on counting the number of discrete words, the NBC has roughly 2000 to 3000 concepts. It is estimated that it will take approximately one year to construct the complete classification scheme for the NBC.

In the Article Database, each information unit is carefully indexed by the relevant concepts. Access to the database is via the Retrieval Interface. The user specifies the concepts which describe the project at hand and the interface program extracts the relevant information units from the article database. Because the system has an explicit model of the relationships among the concepts it is easy to locate related information by searching the classification hierarchy.

4 Representing provisions

As mentioned in section 2, the major area of work in standards processing is in the development of generic standards conformance checkers. The aim is to find a logical and expressive way of

		Rule 1	Rule 2	Rule 3	Rule 4	Rule 5	Rule 6
Conditions	The occupancy sub-group of the building is A1	True	True	True	True	True	False
	The element of structure is a separating wall	True	–	–	False	False	–
	The height of the building is greater than 28 metres	–	True	–	False	False	–
	The element of structure is a compartment wall	–	–	True	–	False	–
	The height of the building is greater than 15 metres	–	–	True	False	–	–
Actions	The minimum fire resistance of the element of structure is 60 minutes	Yes	Yes	Yes			
	The minimum fire resistance of the element of structure is 30 minutes				Yes	Yes	Yes

Figure 4

representing provisions so that they can be easily understood and updated by human users and can be manipulated and reasoned with by computers. The earliest work on developing a systematic organisation model to represent provisions was by Fenves (1966). It was based on the ideas of decision tables. Since then a considerable amount of work has been done by him and colleagues to extend, refine and demonstrate these ideas (Fenves et al., 1969; Goel & Fenves, 1969; Fenves, 1973; Fenves & Wright, 1977; Stahl et al., 1983; Fenves & Garrett, 1986; Fenves et al., 1987). The decision table representation has also been used by Cronembold & Law (1988). Other researchers have tried to represent provisions using decision trees (Lopez & Elam, 1984), production rules (Sriram et al., 1984; Rosenman & Gero, 1985; Dym et al., 1988; MacKellor & Ozel, 1991; Delis & Delis, 1992), logic (Stone & Wilcox, 1986b; Jain et al., 1989; Rasdorf & Lakmazaheri, 1990) and facts (Topping & Kumar, 1989; Rasdorf & Wang, 1988). These different approaches will be considered in turn.

4.1 Decision tables

A decision table has three parts: a set of conditions, a set of actions and a set of rules. Each rule shows the actions to be taken for a particular combination of conditions. Figure 4 shows the decision table representation of the following building regulation provision taken from Stone and Wilcox (1988):

If the occupancy sub-group of the building is A1 and either

1) the element of structure is a separating wall or;
2) the height of the building is greater than 28 metres or;
3) the element of structure is a compartment wall and the height of the building is greater than 15 metres

then the minimum fire resistance of the element of structure is 60 minutes;
in any other case the minimum fire resistance of the element of structure is 30 minutes.

Each rule in the table suggests a particular conclusion for a particular combination of conditions. For example, rule 1 is interpreted as: if the occupancy sub-group of the building is A1 and the element of structure is a separating wall then the minimum fire resistance of the element of structure is 60 minutes.

Using this approach, a document is represented by a set of inter-related tables. The tables are inter-related because the data items contained in the condition part of one table may be derived from the conclusions of other tables. The SASE (Standards, Analysis, Synthesis, and Expression) methodology developed by Fenves et al. (1987) to support the decision table representation distinguishes three kinds of data:

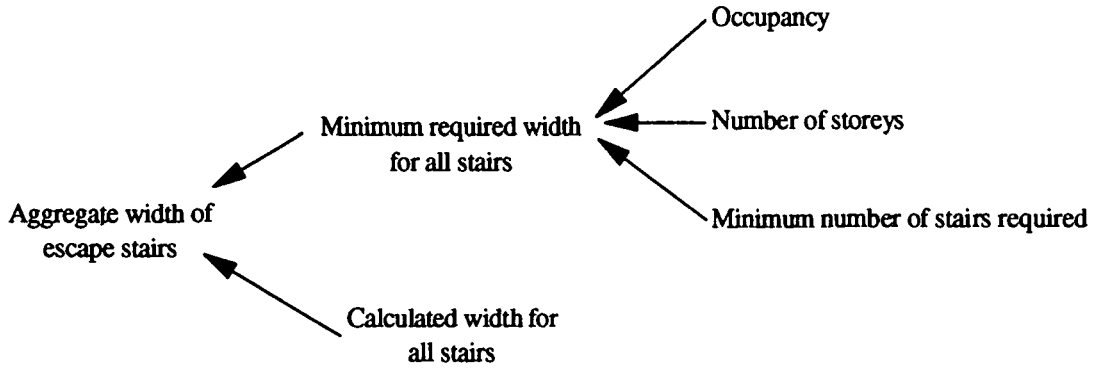

Figure 5

- basic—data items that are not derived but they contribute towards the derivation of other data items;
- derived—data items that are derived and they also contribute towards the derivation of other data items;
- terminal—data items that are derived but they do not contribute towards the derivation of other data items.

Data items found in the decision tables are organised into two kinds of information networks to show data dependency. These information networks are called ingredient network and dependence network. An ingredient network shows, for a data item D in the conclusion of a provision, all the items of information and their precedence which contribute towards inferring the value of D. For example, Figure 5 shows that the determination of the aggregate width required for escape stairs requires the determination of the calculated width required for all stairs and the minimum width required for all stairs. The latter requires the calculation of the appropriate capacity of the storeys served and the determination of the minimum number of stairways required. Ingredient networks also allow cyclic dependency to be automatically identified.

Dependence networks are the complement of ingredient networks and show, for any given condition, all the conclusions to which it contributes. For example, the height of a storey above ground level is a common ingredient that affects many conclusions. A dependence network can show all the provisions which may be critically affected by a design variable.

The decision table approach provides a compact and concise way of structuring provisions. However, to implement a standards processor based on decision tables there needs to be a decision table language, its processor or translator (Welland, 1981; Metzner & Barnes, 1977). Specifying a suitable language and implementing the processor require substantial amount of work.

4.2 Decision trees

Another representation formalism of provisions is the decision tree. For example, Figure 6 shows a partial decision tree of the provision given in the previous section. A decision tree can be constructed from the original statement or translated from a corresponding decision table. It is worth noting that for a given decision table there can be many valid decision trees. Depending upon the order in which the conditions are tested, some decision trees are smaller than others.

Similar to the practical problem with decision table, to implement a standards processor based on decision trees there need to be a suitable decision tree language, its processor or translator.

4.3 Production rules

With the advent of expert systems, rule based programming (Brownston et al., 1985) has provided a new way of implementing computer systems. A rule-based program consists of a set of rules of the general form:

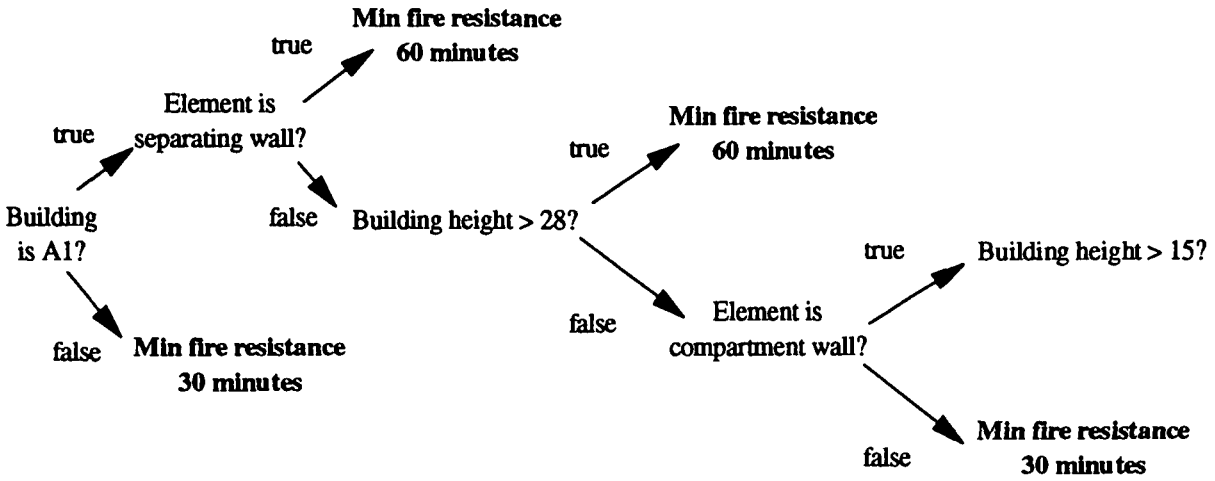

Figure 6

If Condition 1 and . . . and Condition N then Conclusion

Data are represented as facts in the system. The inference engine works in a match-select-execute cycle. In the match state, the engine finds all the rules that have their conditions satisfied by some data in the system. They are collectively called the conflict set. Each rule in the conflict set is a potential candidate for execution. In the select state, a rule is selected from the conflict set according to some criteria. The chosen rule is then executed and the engine repeats the cycle. The computation in a production-system is therefore said to be data-driven.

There is a clear correspondence between a rule and a column in a decision table. Each provision can be written as a set of production rules. Therefore, it seems that production systems provide a convenient and high level way of implementing standards conformance checkers (Sriram et al., 1984; Rosenmann & Gero, 1985; MacKellar & Ozel, 1991).

In practice, a conformance checker cannot be developed by simply rewriting provisions into formal rules. To provide a useful system the control of the execution of the rules is very important and this cannot be left to the default strategies used in the inference engine. Consider an example where the user wants the system to check whether the surface area of a particular window meets the heat conservation criterion. This query cannot be handled easily in a data-driven manner. It requires selecting the appropriate rule and then checking whether the conditions are met.

To deal with different kinds of queries control rules need to be added and control information needs to be included in the condition part of the domain rules. Therefore, translating provisions to rules become less straightforward and the meaning of the rules become obscured by additional information. Updating and modifying the system to reflect changes in the standards is much more difficult. Delis and Delis (1992) have chosen to use a rule interpreter that allows both data-directed and goal-directed reasoning. However, the standards checker performs exhaustive search in a data-driven manner.

4.4 Logic programming

The objective of logic programming is to provide a means of defining computer applications in terms of a machine intelligible form of symbolic logic. The most practical and efficient implementation of the notion of logic programming at the moment is the programming language Prolog (Clocksin & Mellish, 1984). A Prolog program consists of a set of Horn clauses, which are rules of the form

Conclusion if Condition 1 and Condition 2 and . . . Condition N.

```
min-fire-resistance(E,60) if                % The minimum fire resistance of E is 60 minutes
    element-of-structure(E,B), building(B),  % E is an element of structure of B and B is a building
    occupancy-sub-group(B,a1),               % The occupancy sub-group of B is a1
    separating-wall(E).                      % E is a separating wall

min-fire-resistance(E,60) if
    element-of-structure(E,B), building(B),
    occupancy-sub-group(B,a1),
    height(B,H), H > 28.                     % The height of B is H and H is greater than 28 meters

min-fire-resistance(E,60) if
    element-of-structure(E,B), building(B),
    occupancy-sub-group(B,a1),
    compartment-wall(E),                     % E is a compartment wall
    height(B,H), H > 15.                     % The height of B is H and H is greater than 15 meters

min-fire-resistance(E,30) if                % The minimum fire resistance of E is 30 minutes
    element-of-structure(E,B), building(B),
    occupancy-sub-group(B,a1),
    not separating-wall(E),                  % E is a not a separating wall
    height(B,H), H =< 15.                    % The height of B is H and H is less than or equal to 15 meters

min-fire-resistance(E,30) if
    element-of-structure(E,B), building(B),
    occupancy-sub-group(B,a1),
    not separating-wall(E),                  % E is not a separating-wall
    not compartment-wall(E),                 % E is not a compartment-wall
    height(B,H), not H > 28,                 % The height of B is H and H is not greater than 28 meters

min-fire-resistance(E,30) if
    element-of-structure(E,B), building(B),
    occupancy-sub-group(B,G),
    not G = a1.                              % occupancy sub-group of building is not a1
```

Figure 7

Unlike production systems, Prolog works in a goal-driven manner. A clause in Prolog can be read as "Conclusion is true if it can be shown that Condition 1 and Condition 2 and . . . Condition N are true". A clause without conditions, i.e. N = 0, is interpreted as "The Conclusion is true".

The regulation statement considered previously can be written as a complete running Prolog program as shown in Figure 7. Each clause in the Prolog program corresponds to a column in the corresponding decision table. Each symbol that begins with a capital letter is a variable. When Prolog is presented with the goal of the form *min-fire-resistance(element,X)*, it will deduce the *min-fire-resistance* rating for *element* and bind the value to variable X.

Although Prolog provides a very convenient representation and a powerful computation mechanism, it is not without problems in practice. Notice that some of the conditions are duplicated a number of times in the rules. This may present two problems. One is consistent updating. If one condition in the standards is amended it will need to be changed in several places in the program. The other problem is slow execution because the same condition has to be evaluated several times. One could rewrite the program in a more compact way by taking advantage of the control facilities provided in Prolog. However, this would obscure the meaning and maintainability of the program.

Another aspect of the sample program that requires special attention is the use of the not predicate in the fifth clause. The goal *not(Q)* in Prolog is interpreted as all ways of showing Q fail.

This interpretation is called negation as failure (Clark, 1978). It is justified whenever we can make a closed world assumption: anything which is not known is assumed to be false. However, not making a clear distinction between *not known* and *false* will sometimes lead to wrong inferences. For example, if the type of wall is not known the sample program will falsely infer that the min-fire-resistance of the element is 30 minutes. This problem may be serious because regulations are often drafted by general rules followed separately by a list of exceptions. Sergot et al. (1986) discusses the treatment of negated conditions in more detail using examples from the British Nationality Act.

Prolog's in-built control strategy entails backward reasoning. The normal query form is to ask the system to prove or find a value for a rule conclusion. In some situations, however, it is useful to be able to specify a preferred conclusion and ask the system to determine what are the necessary condition values. Stone and Wilcox (1986a, b) found that developing a standards processor in Prolog requires a great deal of programming effort. Despite the criticisms, Prolog, compared with other programming languages, is a good programming language.

4.5 Automated formalization of texts

Stone & Wilcox (1987, 1988) have also attempted to implement a system that will map document's texts directly into a machine readable and executable form. It is suggested that there are two advantages to these methods. Firstly, they obviate the need for intermediary specialists or knowledge engineers to translate the information into machine format. Secondly, they provide the possibility for authors of codes and standards to communicate directly with system functions in familiar text form without the need to understand system notions such as predicates, arguments, variables and so forth.

Automated text formalisation depends essentially upon some form of parsing. The system by Stone and Wilcox recognizes a limited set of logical connectives and arithmetic functions. For example, the provision shown in section 4.1 is mapped to the rule

```
IF "the occupancy sub-group of the building" = "A1"
   AND OR
        "the element of structure" = "separating wall"
        OR
        "the height of the building" > 28
        OR ("the element of structure" = "compartment wall"
            AND
            "the height of the building" > 15
THEN
   "the minimum fire resistance of the element of structure" = 60
ELSE
   "the minimum fire resistance of the element of structure" = 30
```

The system does not try to "understand" the original sentence during parsing. It recognizes certain keywords, arithmetic operators, mathematical functions and numeric values. Text strings between logical connectives are simply treated as variable identifiers and constant values. It is not surprising that this simplistic approach is found to be very limited. For example, it would not be able to identify that "the height of the building" and "the building's height" refers to the same thing. The investigators have abandoned pursuing this approach further.

4.6 Facts

Topping & Kumar (1989), and also Rasdorf & Wang (1988), have suggested representing Standards requirements as a collection of facts. The facts are then interpreted by a generic standards processor as and when they are needed. Topping and Kumar suggest dividing a provision

into two types of criteria: applicability criteria and performance criteria. They are analogous to the conditions and actions of a rule respectively. The general forms for the applicability and performance criteria are:

applicability(Clause Number, Expression)
performance(Clause Number, Data Item, Expression, Operator)

The outcome of converting the example building regulation statement into facts would be something like:

applicability(3.2, occupancy-sub-group(B,a1))
applicability(3.2, height(B) > 15)
applicability(3.2, height(B) > 28)
applicability(3.2, separating-wall(E))
applicability(3.2, compartment-wall(E))
performance(3.2, min-fire-resistance,30,=)
performance(3.2, min-fire-resistance, 60,=)

An analysis of the facts reveals two fundamental problems with this approach. The first problem is the treatment of disjunction. The use of a single clause number is insufficient to distinguish how the different applicability criteria relate to the different performance criteria. This may be overcome by creating artificial clause numbers. The second, and more serious, problem is the treatment of variables. Because facts are represented individually there is no information linking that the element E is part of the building B. It seems that there is no advantage of representing the applicability and performance criteria as separate facts.

4.7 Discussion

Some of the work described in this section is encouraging. It demonstrates that expert systems have a role in standards processing. However, there is no generic processor that has made it beyond the research laboratory. There is still much research to be done before such a tool will be made available. Before a usable system can be built, a better understanding of how the application software is to be used is required. This is an area that requires some research effort. The different types of queries expected from the users need to be identified and the different inference strategies that are required to support these queries need to be incorporated into the system. The rule-based approaches, both production- and logic-based, seem intuitive and natural. However, a standards processor cannot be developed simply by translating provisions into formal rules. The observation by Brown & Chandrasekaran (1989) is appropriate here: "the much talked about separation of knowledge from inference is not really true in practice for complex problems."

The issue of data representation also needs to be addressed in such a system. For example, the command "check the width of all stairs" will require the system to *locate* all the widths of stairs from the data set, but not the widths of doors or walls, etc, and then *check* them against the relevant provisions. To handle this kind of query, data needs to be organized into a data model as discussed in section 3.2 (Garrett & Hakim, 1992; MacKellar & Ozel, 1991).

5 Incorporating case rules

The work described so far has focused on the provisions specified in regulatory information and the problems of representing their content in various ways in computer systems. Yang et al. (1993) argue that this view of regulations is too narrow. Technical regulations have characteristics of legal information as well as statements of what the artifact must satisfy. When assessing whether an artifact meets a particular standard, deviation from the detailed requirements is permitted in

unexpected or exceptional circumstances provided that the intention or goal can still be achieved (McCormick & Bankowski, 1989). Therefore, case histories which allow such deviations must be available and used for the consideration of new cases. As standards are updated, or amended, new provisions derived from case histories will be incorporated.

The Knowledge-Intensive Case-Based reasoning System (KICS) by Yang et al. (1993) distinguishes two kinds of rules: rules as stated in statutory regulations, and rules derived from case histories. These rules are represented in the same format, but the former kind is marked as "strong" and the latter is marked as "weak". For example, for energy conservation purpose, a strong rule may specify the maximum area of a window in relation to the floor area in the room as given in the standards. A weak rule derived from case history may say that the area of the window may exceed the stated ratio by a certain amount if the window is double glazed and the glass is more than a particular thickness.

When KICS is asked to check a particular design parameter, it is first checked against the strong rules. If the check failed it is then checked against the related weak rules. If none of the rules are satisfied then an expert has to assess the design against the stated *intention* of the standards. If an exception is made then a new weak rule based on the case is created and incorporated into the system. KICS has not been fully implemented but it provides an interesting architecture for integrating statute and case rules.

6 Author support systems

Instead of focusing on the needs of users of regulations, some researchers (Garzotto & Paoloni, 1989; Stone & Tweed, 1991; Hakim & Garrett, 1992) have considered the kind of system functionality necessary to support authors and administrators of standards.

They argue that the published texts are only the most tangible manifestation of a much more extensive information domain and that the conventional view of standards as comprising solely the published texts is the source of many of the problems of the production of the information and the basis of many of the difficulties users experience in interpreting and applying the requirements in practice. The drafting of regulatory information requires considerable expertise and necessitates an understanding of the enabling principles of legislation, a knowledge of the underlying intent of a particular regulation or standard, a detailed knowledge of individual requirements and a familiarity with exemplars and case histories. This knowledge may be well documented but dispersed and held at minutes and records of meetings, notes of discussions, formal records of judicial determinations, written answers to queries and correspondence, research reports and so forth or it may simply be anecdotal. Partly because of the limitations of conventional documentation and partly because of the statutory status of regulatory information there is a dislocation between this experiential, operational knowledge, necessary for the proper understanding and development of standards, and the formal knowledge contained in the texts. The objective of the research work of authoring systems cited above is to find ways of binding these two domains of knowledge in order to retain and make explicit expertise in the standards domain and to ensure the proper maintenance of the domain's information.

It will be evident that the amount of information involved is potentially very large. It is for this reason that the research work cited is less concerned with the representation of the content of documents and more with the problems of information organization and information retrieval.

A major difficulty of organization is the problem of recording and maintaining links between semantically related material. The conventional way of retaining this kind of control of the domain information relies on the use of physical filing systems and individual's memories to maintain the appropriate connections and relationships. However, physical file systems become unwieldy, individual's memories are unreliable or there may be changes in personnel. The collective and individual memory that links information and gives it meaning is, then, only too easily diminished or lost. The problems of information retrieval parallel and are linked to the problems of organization. For any given task, a user typically needs to abstract only a subset of the material

available but needs to be assured that all the relevant information has been located. This problem is particularly critical for authors when drafting or revising requirements. Before proposing changes they need to explore the consequences of change relative to existing material and ensure that their proposals reflect and properly respond to current thinking and do not conflict with established principles.

6.1 Hypertext and expertext

This perceived need to improve information organization and retrieval for authors has led researchers to adopt hypertext as a primary representational paradigm. However, hypertext is not without its drawbacks. Although it does provide for more convenient and structured access to texts than linear documents, it provides no direct functionality with regard to the content of the text nodes and a proliferation of links has been shown to lead to difficulties of navigation. Research is focusing, therefore, on ways of enhancing and extending the basic hypertext model. Of particular interest are the proposals of Rada & Barlow (1989), Diaper & Rada (1989) and Diaper & Beer (1990). Their proposals seek to combine the properties of both expert systems and hypertext and they have coined the term *expertext* to describe such a system. They argue that both expert systems and hypertext share the same underlying graph-theoretic model. However, they suggest that there are substantial differences in the two approaches in their treatment of nodes and links. Whereas the nodes in hypertext are semantically rich, in that they contain natural language texts, the links have little or no semantic content. Conversely, whereas the nodes in an expert system are semantically impoverished, because they contain a formalization of the source information, the links are well specified, typically as predicate names in a rule-based system. Expertext represents an attempt to combine the best properties of expert systems and hypertext and provide systems which have the semantically rich nodes of hypertext and the well specified, computable links of expert systems.

It is interesting to look at this proposal from the perspective of how intelligence is distributed between the user and the system. In an expert system the intelligence lies primarily with the system; the user is usually relegated to responding to system generated queries. In hypertext, by contrast, the intelligence lies largely with the user, who interprets the node content but must also specify the order of link traversal. In expertext there is potentially a more balanced distribution of intelligence. Understanding and interpreting the content of nodes is the user's responsibility and is the task most appropriate to human intelligence. On the other hand, because in the expertext model there exist defined and computable links between nodes, the system's intelligence can be focused on the task of navigation, that is on selecting and presenting the most appropriate set of nodes for the user's consideration. It is argued that this distribution of intelligence should reduce, if not eliminate, the navigation problems conventionally associated with hypertext.

The expertext model potentially fits well with the problems identified in authoring and maintaining regulatory information. Firstly, the problems of the organization of material and the maintenance of links across domain information can be resolved by expertext's underlying node-and-link structure. Secondly, the problems of information retrieval can be resolved by expertext's intelligent navigation functionality. Thirdly, expertext is not inherently limited by the problems of scale or change in the domain's information. And finally, it seems possible that the functionality and intelligence of the system can be incrementally enhanced as experience in its use is gained.

A particular architecture for expertext was proposed by Rada & Barlow (1989). This has subsequently been developed and termed Headed Record Expertext (HRExpertext) by Diaper & Rada (1989) and Diaper & Beer (1990). The underlying model of HRExpertext remains the node-and-link semantic net. Nodes, however, consist of headed records which have two parts. These are, a record containing the user readable, natural language texts (or potentially any form of user understandable material) and a header which contains an abstraction of the semantic content of the record. Header material is a formalization of the record's content and is, therefore, an impoverished representation of the record but it has the property that it can be used computationally by the system to select and make available the records consistent with a user's objective.

6.2 Argumentation

We have noted that a major objective of current work is to bind the experiential and operational information generated by the administration and application of standards to the document domain. The non-document domain of information we may term argumentation. A continuing research problem is to find an appropriate rhetoric model for organizing and structuring this kind of information and linking it to the documentation domain. What is required is a method of organizing texts which allows a user to expose key relationships within the argumentation material and to make explicit the inherent processes of dialogue and negotiation. Two candidate models which have been proposed are Issue Based Information Systems (IBIS) (Kunz & Rittel, 1970; Conklin & Bergeman, 1988) and Rhetorical Structure Theory (RST) (Mann & Thomson, 1987; Mann et al., 1989).

The IBIS model suggests that argumentation material can be characterized as being the end-product of a process of dialogue or debate, involving a number of participants, about one or more issues. An issue in this sense is simply a question or unresolved problem. IBIS provides a framework for structuring and recording the elements of information generated by this process of issue resolution in a node-and-link representation. IBIS offers a relatively restricted set of node and link types. For example, nodes may be issues, positions or arguments where positions *respond_to* issues and arguments *support* or *object_to* a position.

RST, by contrast, is not primarily concerned with dialogue but with providing a way of analysing and structuring texts as written monologue. It works by splitting the text into fragments of any length and then linking these fragments to form a hierarchy. The links between fragments are commonly of a nucleus-satellite form, in which one node is ancillary to the other. An important difference between the RST and IBIS approach is that nodes in RST can be groups of sub-nodes. This allows links to be placed between large sections of the text, each of which may themselves contain a network of links and sub-nodes. RST provides a more sophisticated representation than IBIS, both in terms of the number of different relations and the definition of restrictions on how the relations should be applied. Typical relation types in an RST graph might be *background*, *concession*, *contrast*, *elaboration*, *motivation* or *means*. IBIS and RST are to some extent complimentary and some hybrid model might be desirable. For example, it may be possible to merge the different types of relations to provide a single set of relations with a rich variety of types; or to use IBIS to map out the dialogue structure of the argumentation material and then use RST relations to map out the fine detail of the issues, arguments and so forth. However, it seems likely that there is no abstract way of resolving the issue modelling the information in the argumentation domain and that it can only be resolved by pragmatic experimentation in an application context.

6.3 Hybrid system

The development and implementation of a system based on the concept of expertext and providing support for authors and argumentation material is described in Casson & Stone (1992) and Casson (1993). The system, called PLINTH (PLatform for INtelligent Hypertext), is an expertext shell which combines a variety of established representation schemes into an integrated architecture. It supports the creation and manipulation of standards documents in hypertext form but also manages the argumentation underlying the documents and provides facilities for intelligent, guided browsing. PLINTH augments hypertext networks with semantic properties. Types can be assigned to nodes and links to mark the function of the nodes in the network and the relations between them. This enables users to create, manipulate and browse argumentation material based on the IBIS model.

PLINTH then supports this augmented hypertext with rule-based navigation. The author can write sets of navigation rules to perform different text retrieval or consultation functions corresponding to different views. For example, for free browsing, the navigator follows basic structural links like "next" or "part-of" in response to commands from the reader such as "next",

"previous", "up", "back" and so on. For the consultation facilities more akin to an expert system, PLINTH uses navigation based on logical typing of nodes and links. In this case, the navigator follows the links in logical order assigning truth values to nodes by asking the reader whether or not the conditions expressed in the node text are met. These truth values are combined according to the link types to decide which provisions apply and advise the reader on how to satisfy them. It is suggested that PLINTH's underlying architecture alleviates the problems of expressiveness, parsimony, maintenance and cognitive overhead associated with solely hypertext or expert systems.

7 Conclusion

This paper has reviewed the wide variety of approaches to using information technology which are being experimented with to provide support for users and authors of regulatory information. Practical success to date has been mainly limited to systems which focus on information retrieval functions. The more advanced technologies, such as expert systems, are appealing but seem largely restricted to small scale applications or prototypes. This type of technology, which attempts to represent formally the content of regulatory texts, typically entails one or more transformations or mappings of the source texts. At the moment, these transformation processes are not automated and involve difficult technical, conceptual and practical issues, not least of which are problems of scale and change in the domain information and problems of ambiguity and the preservation of meaning. Recent research in standards processing has started to address the wider issues relating to data model, case histories and author support.

It is recognized (Reed, 1991) that, whilst work on providing computer based tools for accessing and manipulating regulatory information is technically challenging and of research interest, practical progress will depend upon establishing a common and more rigorous development framework and ultimately on its acceptance by standards' agencies and the industries involved.

Acknowledgements

Paul Chung is grateful to British Gas and The Royal Academy of Engineering for financial support through a Senior Research Fellowship.

References

ASHRAE, 1989. *ASHRAE/IES Standard 90.1-1989 Energy Efficient Design of New Buildings Except Low-Rise Residential Buildings*, American Society of Heating, Refrigerating and Air-Conditioning.

Bourdeau, M, 1991. "The CD-REEF: The French Building Technical Rule on CD-ROM". In: K Kahkonen and B Bjork (eds.), *Computers and Building Regulations*. VTT Symposium 125, Technical Research Centre of Finland.

Brown, DC and Chandrasekaran, B, 1989. *Design Problem Solving, Knowledge Structures and Control Strategies*. Pitman.

Brownston, L, Farell, R, Kant, E and Martin, N, 1985. *Programming Expert Systems in OPS5*. Addison Wesley.

Casson, A, 1993. *PLINTH: Integrating Hypertext, Semantic Nets and Rule-Based Systems in an Expertext Shell for Authors and Readers of Regulatory Documents*. AIAI-TR-142, Artificial Intelligent Applications Institute, University of Edinburgh, September.

Casson, A and Stone, D, 1992. "An expertext system for building standards". In: *Computers and Building Standards Workshop*. Montreal, Canada, May 12.

Clark, K, 1978. "Negation as failure". In: H Gallaire and J Minkes (eds.), *Logic and Data Bases*. Plenum.

Clocksin, W and Mellish, C, 1984. *Programming in Prolog (2nd Edition)*. Springer-Verlag.

Conklin, J and Begeman, M, 1988. "gIBIS: A hypertext tool for exploratory policy discussion". *ACM Transactions on Office Information Systems* **6**(4) 303–331.

Conklin, J, 1989. "Hypertext: An introduction and survey". *IEEE Computer* **20**(9) 17–41.

Crawley, DB and Boulin, JJ, 1991. "ENVSTD: A computer program for complying with building envelope requirements". In: K Kahkonen and B Bjork (eds.), *Computers and Building Regulations. VTT Symposium 125*, Technical Research Centre of Finland.

Cronembold, JR and Law, KH, 1988. "Automated processing of design standards". *Journal of Computing in Civil Engineering* **2**(3) 255–273.

Delis, EA and Delis, A, 1992. "An expert system for fire-code checking: Design, implementation and geometric algorithms". *International Journal of Artificial Intelligence* **1**(4) 597–627.

Diaper, D and Beer, M, 1990. "Headed record expertext and document applications". In: *Proceedings of Hypertext Update*, pp. 62–69. Unicom Seminars Ltd.

Diaper, D and Rada, R, 1989. "Expertext: Hyperising expert systems and expertising hypertext". In: *Proceedings of Hypermedia/Hypertext and Object Orientated Databases*, pp. 124–152. Unicom Seminars Ltd.

Dym, CL, Henchey, RP, Delis, EA and Gornick, S, 1988. "A knowledge-based system for automated architectural code checking". *Computer Aided Design* **20**(3) 137–145.

Fenves, SJ, 1966. "Tabular decision logic for structural design". *Journal of Structural Engineering* **92**(6) 473–490.

Fenves, SJ and Garrett, JH Jr, 1986. "Knowledge based standards processing". *Artificial Intelligence in Engineering* **1**(1) 3–14.

Fenves, SJ, Gaylord, EH and Goel, SK, 1969. *Decision Table Formulation of the 1969 AISC Specification*. Civil Engineering Studies SRS 347, University of Illinois, August.

Fenves, SJ, 1973. "Representation of the computer-aided design process by a network of decision tables". *Comput. Struct.* **3**(3) 1099–1107.

Fenves, SJ and Wright, RN, 1977. *The Representation and Use of Design Specifications*. Technical Note 940, National Bureau of Standards, Washington, DC.

Fenves, SJ, Wright, R, Stahl, F and Reed, K, 1987. *Introduction to SASE: Standards Analysis, Synthesis, and Expression*. NBSIR 87-3513, National Bureau of Standards, Washington, DC.

Gardner, A, 1987. *An Artificial Intelligence Approach to Legal Reasoning*. MIT Press.

Garrett, JH Jr and Hakim, MM, 1992. "Object-oriented model of engineering design standards". *Journal of Civil Engineering* **4**(3) 323–347.

Garzotto, F and Paolini, P, 1989. "Expert dictionaries: Knowledge-based tools for explanation and maintenance of complex application environments". In: *Proceedings Second International Conference on Industrial and Engineering Applications of AI and Expert Systems IEA/AIE-89*, pp. 126–134. ACM Press.

Goel, SK and Fenves, SJ, 1969. *Computer-Aided Processing of Structural Design Specifications*. Civil Engineering Studies SRS 348, University of Illinois, November.

Graham, I, 1991. *Object Oriented Methods*. Addison Wesley.

Hakim, MM and Garret, JH Jr, 1992. "Issues in modeling and processing design standards". In: *Computers and Building Standards Workshop*. Montreal, Canada, May 12.

Harris, JR and Wright, RN, 1980. *Organisation of Building Standards: Systematic techniques for Scope and Arrangement*. Building Science Series NBS BSS 136, National Bureau of Standards, Washington, DC.

Jain, D, Law, KH and Krawinkler, H, 1989. "On processing standards with predicate calculus". K Will (ed.), *Proceedings of 6th Conference on Computing in Civil Engineering*. ASCE, New York.

Kunz, W and Rittel, HWJ, 1970. *issue as Elements of Information Systems*. Working Paper 131, Centre for Planning and Development Research, University of California.

Lopez, LA and Elam, SL, 1984. *SICAD: A Prototype Knowledge Based System for Conformance Checking and Design*. Technical Report, University of Illinois.

MacKellar, BK and Ozel, K, 1991. *Artificial Intelligence in Design '91*, J Gero (ed.). Butterworth-Heinemann.

McCormick, N and Bankowski, Z, 1989. "Some principles of statutory interpretation". In: J van Dunne (ed.), *Legal Reasoning and Statutory Interpretation*, pp 41–53. Gouda Quint BV Arnhem.

Mann, WC, Mathiesson, CMIM and Thomson, SA, 1989. *Rhetorical Structure Theory and Text Analysis*. Isi/rr-89-242, Information Sciences Institute, University of Southern California.

Mann, WC and Thompson, SA, 1987. *Rhetorical Structure Theory: A Framework for the analysis of Texts*. Isi/rs-87-185, Information Sciences Institute, University of Southern California.

Metzner, JR and Barnes, BH, 1977. *Decision Table Languages and Systems*. Academic Press.

Rada, R and Barlow, J, 1989. "Expertext: Expert Systems and Hypertext. In: *Proceedings of EXSYS'89 conference*, IITT-International, Paris.

Rasdorf, WJ and Lakmazaheri, S, 1990. "Logic-based approach for modeling organization of design standards". *Journal of Computing in Civil Engineering* **4**(2) 102–123.

Rasdorf, WJ and Wang, TE, 1988. "Generic design standards processing in an expert system environment". *ASCE Journal of Computing in Civil Engineering* **2**(1) 68–87.

Reed, KA, 1991, "Building regulations and standards in the computer age. In: K Kahkonen and B Bjork (eds.), *Computers and Building Regulations. VTT Symposium 125*, Technical Research Centre of Finland.

Rosenman, MA and Gero, JS, 1985. "Design codes as expert systems". *Computer-Aided Design* **17**(9).

Sergot, M, 1991. "The representation of law in computer programs". In: T Bench Capon (ed.), *Knowledge-Based Systems and Legal Applications*. Academic Press.

Sergot, MJ, Sadri, F, Kowalski, RA, Kriwaczek, F, Hammond, P and Cory, HT, 1986. "The British nationality act as a logic program", *Communications of the ACM* **29**(5) May.

Sriram, D, Mahar, ML and Fenves, SJ, 1984. "Knowledge-based expert systems in structural design. In: *Advances and Trends in Structures and Dynamics*. NASA, October.

Stahl, FI, Wright, RN, Fenves, SJ and Harris, JR, 1983. "Expressing standards for computer-aided building design". *Journal of Computer Aided Design* **15**(6) 329–334.

Stone, D and Wilcox, DA, 1986a. "Knowledge structures and domain specific shells for building regulations". In: Proceedings of ESCAD'86 Workshop. Reading, UK, July.

Stone, D and Wilcox, DA, 1986b. "Formulation of building regulations using interactive logic programs". In: *CIB'86, Proceedings of the 10th Triennial Congress of the International Council for Building Research, Studies and Documentation*, Washington, DC, September, Vol 2.

Stone, D and Wilcox, DA, 1987. "Intelligent systems for the formulation of building regulations". In: *proceedings of 4th International Symposium on Robotics and Artificial Intelligence in Building Construction*, Haifa, Israel, June.

Stone, D and Wilcox, DA, 1988. "Intelligent information systems for building standards". In: *Proceedings of EuropIA 88, Applications of Artificial Intelligence to Building, Architecture and Civil Engineering*, Paris, France.

Stone, D and Tweed, C, 1991. "Representation issues in the design of an intelligent information system for building standards". In: K Kahkonen and B Bjork (eds.), *Building Design in Computers and Building Regulations. VTT Symposium 125*, Technical Research Centre of Finland.

Susskind, R, 1987. *Expert Systems in Law, Part 2–3*. Oxford University Press.

Topping, B and Kumar, K, 1989. "Knowledge representation and processing for structural engineering design codes". *Engineering Applications of AI* **2**, September.

Tuominen, P, 1991. "Fire Expert—An expert system for fire safety regulations". In: K Kahkonen and B Bjork (eds.), *Building Design in Computers and Building Regulations. VTT Symposium 125*, Technical Research Centre of Finland.

Vanier, D, 1991. "A parsimonious classification system to extract project-specific building codes". In: K Kahkonen and B Bjork (eds.), *Computers and Building Regulations. VTT Symposium 125*, Technical Research Centre of Finland.

Welland, R, 1981. *Decision Tables and Computer Programming*. Heydon & Son Ltd.

Yang, S, Robertson, D and Lee, J, 1993. "KICS: A Knowledge-Intensive Case-based reasoning System for statutory building regulations and case histories". In: *Proceedings of the Fourth International Conference on Artificial Intelligence and Law*, Amsterdam, June.