# Automated Generation of Rules from the National Building Code Text

Bernard Moulin[1]

## Abstract

The Canadian National Building Code (NBC) outlines a set of model technical requirements with respect of public safety in public buildings. Like most regulatory texts, NBC evolves rapidly as a consequence of the accelerated pace of building science and building material technologies. Fewer NBC users (architects, builders, engineers) will have the expertise and time to correctly identify and interpret building safety regulations, and will rely even further on a limited number of knowledgeable experts.

The reported research investigates the possibility of generating knowledge bases from the analysis of NBC text. These knowledge bases could be used by consultation systems (expert systems) which could help users in checking the characteristics of buildings against NBC prescriptions. We present SACD a knowledge acquisition system which analyses regulatory texts and generates a rule base whose logical content is equivalent to the original text content. We illustrate SACD utilization on the firewall section of the French version of NBC.

## Introduction

The National Building Code of Canada (NBC) outlines a set of model technical requirements with respect to public safety in buildings. Its present published format has evolved as a model for legislative adoption [NBC 85]. The readership, including architects, engineers, builders and building code officials (hereinafter referred to as the "users") presently numbers 60 000 for the NBC and 100 000 for the Ontario Building Code.

Few users (approx. 4%) use computer based commercial information retrieval building code products. Their lack of functionality and inherently poor design has resulted in less than favorable market acceptance of these products [Vanier 89, 90b]. Most users manually search through NBC and rely on experience and whatever indexing cues (i.e. table of contents, margin subject notes and an index). Depending on the nature and subject of the search, this is a somewhat cumbersome and time-consuming process subject to error.

The accelerated pace of building design science and building material technologies suggests a significant impact on the amount and complexity of building safety regulations which will be adopted during the next decade. In addition, the very nature of a country-wide standard implies the adoption of an even greater amount of exceptions (or constraints) depending on what province the legislation is referring to. Thus the NBC promises to be a more voluminous and complex document whereby (1) fewer users will have the expertise and time to correctly identify and interpret building safety regulations, (2) users will rely even further on a limited number of knowledgeable and expensive resources in the field (expert consultants) and (3) some provisions of the NBC text could become inconsistent or difficult to apply because of the complexity of the regulation text (for example because of

[1]Université Laval, Département d'informatique Ste Foy Québec, G1K 7P4 Canada

the multiplicity of cross-references in the text).

Several software products are available for assisting the automation of code search and code checking. They generally fall into two categories : (1) information retrieval which includes full-text (indexed) search, sequential search and hypertext access [Vanier 89, 90c and 90d], and (2) expert systems for interactive code checking.Available expert systems incorporate code checking knowledge, usually in the form of rules, to interactively prompt the user for information and then to analyse the building situation for code conformance. The user saves time as the expert systems is automatically pruning the search space and asking questions in logical order. Since these systems are not currently integrated with source text, the user may have little confidence in the justifications they provide. Because the knowledge bases are solely elaborated on the basis of human expertise in manipulating the codes, systems are not proved to be complete and coherent with respect to the source text.

There are multiple sources of juridical knowledge: statute and regulation texts, jurisprudential texts ("case law"), knowledge of legal experts, etc. Research activities have investigated different areas : formulating legislation with the aid of logical models [Sergot 86] [Bench-Capon 87], legal reasoning [Von der Lieth Gardner 87], case-based reasoning [Rissland 87] [MacCrimmon 89], developing expert systems applied to the juridical [McCarty 80] [Deschamps 87] [Nitta 88] [Schild 88] [Tidrick 88] or administrative domains [Corriton 88] [Lesaffre 90]. In the domain of law, most expert systems use "surface knowledge" which are heuristic rules obtained from human experts ("shallow models"). McCarty [1984] emphasized the need for developing expert systems using "deep knowledge" and Bench-Capon [1989] indicated that this deep knowledge can be partly obtained in formalizing legislation.

Our research aims at bridging the gap between regulatory texts and expert systems by providing a method and a tool to extract from these texts the logical knowledge which can be used to create knowledge bases. In project A.C.A.T. (Acquisition des connaissances et analyse de textes), we explore the possibility of creating knowledge bases by exploiting information contained in French versions of regulatory texts from the governments of Quebec and Canada [Moulin 89, 90-1, 90-2].

## The logical content of regulatory texts

A regulatory text may be considered as a body of knowledge that describes the characteristics of a practical domain on which the law is applied. From a cognitive point of view, we can consider that a regulatory text describes the properties of a "standard world". For example the National Building Code contains the recommended characteristics of the buildings to be constructed in Canada. This "standard world" is composed of abstract or concrete objects which have been selected by legal experts such as buildings, firewalls, floor areas etc. The regulation articles describe the properties of these objects which are mandatory, allowed or forbidden. When a user (an analyst in a governmental agency or an architect for example) wants to check a situation or a case (a building blue print for instance), s/he must find and evaluate the relevant characteristics of the given case. These observed characteristics are compared with the corresponding properties of the objects described in the standard world in order to appreciate the conformity of the observation with respect to the regulation. In doing so, the user plays a role which is similar to that of a diagnosis expert system: the regulation is equivalent to a natural language knowledge base which contains the rules to be applied for checking the case; the user who searches in the regulatory text and applies provisions, behaves as an inference engine which chains the rules, selects them and fires the appropriate ones.

However, regulatory texts cannot be processed directly by expert systems, because they are natural language texts written in a way which is less structured and concise than expert system knowledge bases. Users are often facing problems when interpreting regulatory texts, due to the lack of structure and to the proliferation of cross references. Regulatory texts are composed of three types of propositions [Gardies 80]: definitions, "normative propositions", "metatextual statements".

*Definitions* are sentences which introduce relevant domain objects in the text. They can provide interpretation problems but have no influence on the text logical content which is expressed by normative propositions and metatextual statements.

*Normative propositions* are sentences which describe prescriptions and contain expressions indicating modalities (permission, obligation, interdiction).

According to Kalinowski's suggestions [1972], we work with texts composed of "normative propositions" whose syntactic structure characterizes norms : they are built with verbs or expressions indicating explicitly a modality such as an obligation ("devoir"; must or shall) a permission ("pouvoir"; to be allowed or may) or an interdiction ("il est interdit de"; it is forbidden to).

*Metatextual statements* express properties which are not related to the application domain, but to the text structure: they correspond to cross-references from one part of the text to another one. Metatextual statements can greatly influence the logical content of prescriptive texts. The use of cross-references makes difficult the interpretation of legal texts [Tremblay 87] [Lagacé 87] [Bench-Capon 87], especially when there are chains of references introduced by expressions such as "in spite of article x", "subject to provisions of article y" etc. Cross-references are used by legal writers to ease their task of formulating concise sentences, but a proliferation of references puts a heavy burden on the reader when s/he tries to interpret a prescriptive text : the reader must understand and make explicit the chains of references in order to infer what are the precise conditions which influence the application of text prescriptions.

In project ACAT, one of our goals was to show that it is possible to generate a structured knowledge base with a logical content equivalent to that of the original regulatory text, by analysing the text at a syntactic level, using text grammars. In order to construct the knowledge base, we have to identify in the regulatory text the relevant objects of the "standard world", as well as the rules which apply on their properties.

Our approach is based on the assumption that syntactic markers (modal verbs, conjunctions, punctuation etc.) can be used to recognize the relevant objects and rules in regulatory texts. Such an approach can be applied on those regulatory texts that we call "prescriptive texts". Prescriptive texts are composed of "normative propositions" [Kalinovski 72] in which the modalities are stated explicitly using modal verbs such as "doit" ("shall"), "peut" ("may") etc. Most regulatory texts from the Government of Quebec as well as codes like the National Building Code are prescriptive texts according to our definition.

In this project we do not aim at creating a natural language understanding system. SACD identifies the logical structure of prescriptive texts, from which it generates a so called "deontic" knowledge base composed of objects associated with so called "deontic" rules equivalent to the initial text provisions. We use the terms deontic knowledge base and deontic rules in order to distinguish the output of SACD from the rules which are part of the system itself. Deontic rules differ from traditional expert system rules because they are associated with a modality. In order to reason with these rules, we have to use modal logic, and more specifically deontic logic when it comes to prescriptive texts [Kalinowski 72] [Von Wright 68] [Hilpinen 81].

In the section on The analysis of prescritive texts we introduce the model that we have developed for analysing the logical content of prescriptive texts. In the section on The Knowledge Acquistion we discuss briefly the sub-system which is used to analyse prescriptive texts. In the last section we describe the approach that is used to generate the rule base which reflects the text logical content. Examples of deontic rule generation are given.

## The analysis of prescriptive texts

We distinguish in a regulatory text such as NBC, three types of components [Moulin 88]: the text macrostructure which corresponds to the information dealing with text presentation (headings, titles, organization of the text in chapters, sections, articles etc.); the text microstructure which organizes the logical content of the text (keywords or expressions introducing conditions, exceptions, modalities, references etc.); the domanial component of the text corresponding to the information which is specific of the application domain and belongs neither to the macrostructure nor to the microstructure. SACD recognizes the macro- and micro- structures elements of a prescriptive text thanks to two text grammars. These text grammars encode in rules the syntactic associations permitted between the elements of the macro- and micro- structures in French prescriptive texts.

Analysing the text microstructure enables us to identify the text component parts which characterize the text logical content : relevant domain objects and their related prescriptions, definitions, normative propositions and their application contexts, references. Here, we present briefly the main elements of a text microstructure (for more details see [Moulin and Rousseau 90-1, 90-2]).

We call a **modal operator** any expression which points out the modality in the sentence. The modal operator characterizes normative propositions [Kalinowski 72]. the three main modalities are obligation, interdiction and permission. **An operator scope** is the part of the sentence on which the modal operator is applied. A scope that precedes a modal operator is called front-scope, whereas, the back-scope corresponds to a scope which follows the operator.

In figure 1 we present the parse proposed by SACD after analysing the microstructure elements of the first portion of provision 3.1.8(3) of the NBC French version. We give here a translation in English: Every [firewall] which separates a building or buildings with floor areas containing a Group E or Group F,

Division 1 or 2 major occupancy ......(front scope of the modal operator "shall")
**<shall>** ....... modal operator
be constructed as a fire separation of non combustible
construction having a fire-resistance rating of at least 4 h ........ (back scope)

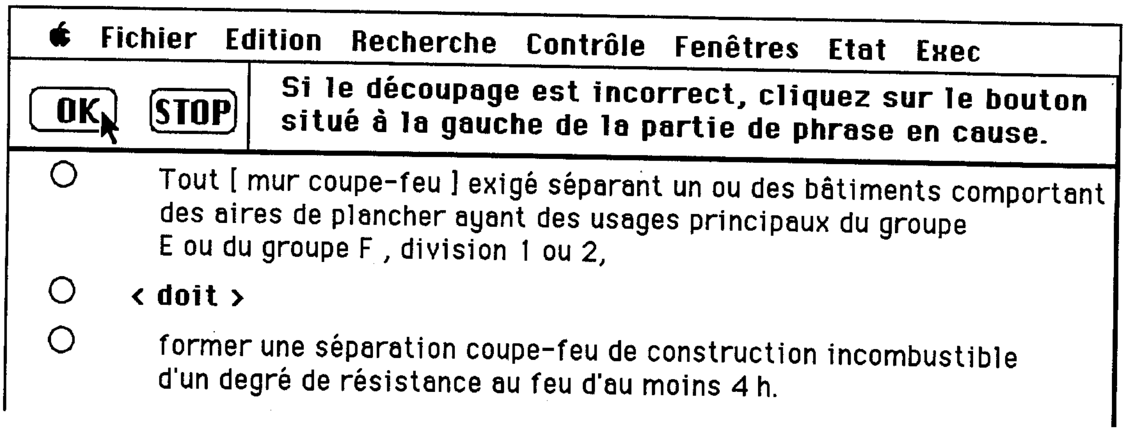


Figure 1 Parse of the first portion of provision 3.1.8 (3) of NBC French version

The modal operator plays an important role within a normative sentence: it helps identifying the object (usually situated in the front scope and emphasized with square brackets by SACD) on which holds the prescription (usually situated in the back scope): that's the case for most regulation texts from the Gouvernment of Quebec [Tremblay 84]. That is also the case of both NBC French and English versions. In our example of figure 1, the prescription is attached to the object "mur coupe feu" ("firewall").

Classical **logical operators** are also identified in the text : "et" ("and"), "ou" ("or"), "ne ... pas" ("not"), "ne ... plus" ("no ... more"), etc.

A **connector** is a conjunction which introduces a **condition** ("lorsque", "si" ("when", "if")) or an **exception** ("sauf si", "à moins que" ("except", "unless")). A connector may be situated either in a front scope or in a back scope of a modal operator. A connector is followed by a scope which is different from the modal operator scopes. In figure 2 we have the second portion of provision 3.1.8(3) of NBC French version. We give here a translation in English:

| | |
|---|---|
| However, | .... interstatement connector |
| when | ...... condition connector |
| the upper portion of a firewall separates floor areas containing other than Group E or Group F, Division 1 or 2 major occupancy, | .... scope of the connector |
| the [fire-resistance rating] of the upper portion | .... front scope of modal operator |
| **<may>** | .... modal operator |
| be reduced to 2 h. | .... back scope of modal operator |

The connector "lorsque" ("when") has been detected as well as its scope.

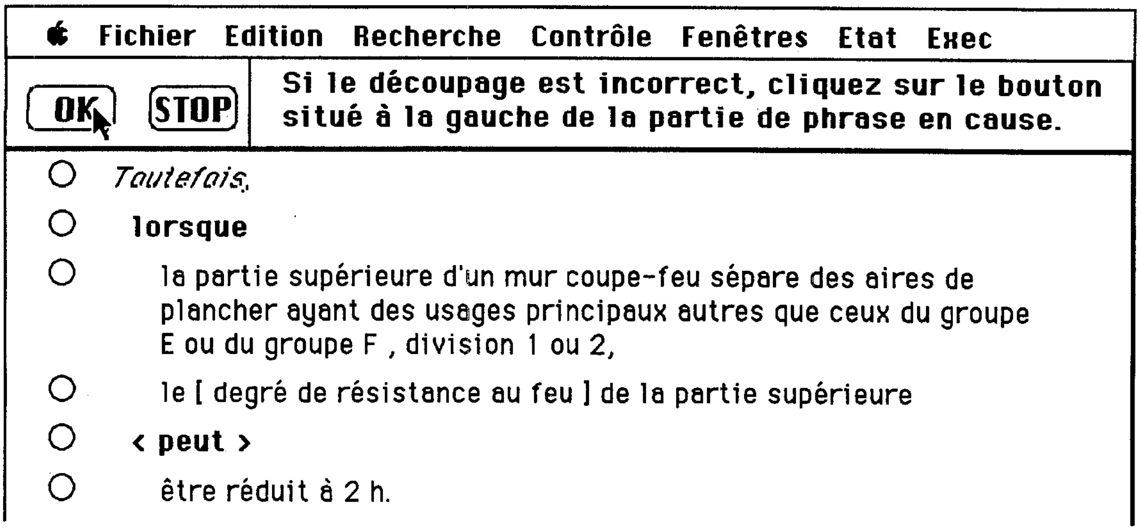


Figure 2 Parse of the second portion of provision 3.1.8 (3) of NBC French version

Internal references in prescriptive texts correspond to other microstructure elements which are detected by SACD. There are different categories of internal references [Moulin 91]: referential connectors (such as "malgré" "in spite of" etc.), referential expressions (such as "requis par" "required by" etc.), metatextual statements (such as "soumis aux exigences de" "subject to the provisions of" etc.), interstatement connectors (such as "cependant" "however" etc.). We have in figure 2 an example of the detection of interstatement connector "cependant" ("however").

As our objective is to identify the components that support the logical content of a prescriptive text, it is not necessary to analyse the text at a semantic level. For efficiency sake, our approach consists in analysing syntactically prescriptive texts, using two text grammars in order to analyse successively the text macrostructure and microstructure [Moulin 90-1, 90-2]. SACD presents the results of the parse to the domain specialist for validation. If the parse is correct, the specialist clicks on the OK button. If not, s/he clicks on the button facing the line where there is an error (see figure 1). SACD presents a menu which contains the various types of errors that can be detected according to the category of the component that has been indicated by the specialist. S/he chooses the error type and SACD presents a specific interface for correcting this error in the parse [for more details see Moulin 90-1].

Errors in the parse may be the result of wrong formulations in the original text; and the text correction fixes the error. Errors may be caused also because SACD text grammars are not rich enough to cope with some new microstructure element in this particular text. In this case, the knowledge engineer analyses the problem and modifies SACD text grammars accordingly. Hence, the grammars are improved with each new text analysed by SACD. Currently the grammars are fairly stable and require few modifications.

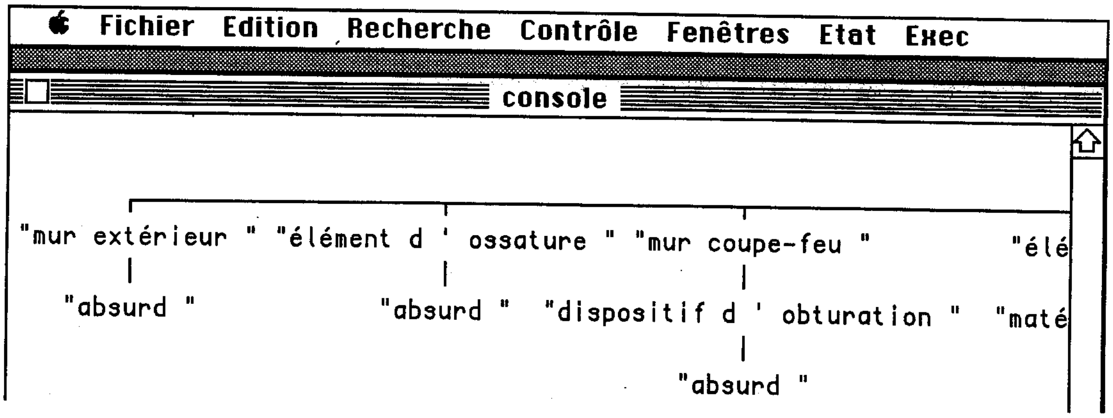


Figure 3 : A sample of the object type hierarchy created with SACD

After having completed the microstructure analysis, SACD helps the specialist to organize in a type hierarchy [Sowa 84], the objects that have been detected in the text (figure 3). For instance "firewall" is categorized as a sub-type of "wall", "fire-resistance rating" as a sub-type of "characteristics" etc. The type hierarchy is the back-bone of the deontic knowledge base to be constructed by SACD. In the last phase of the text analysis, SACD identifies anaphoric references such as personal pronouns, possessive or demonstrative adjectives and some adverbs which are used in the text to refer to an object within the same sentence or in another one [for more details see Moulin 90-1].

## The knowledge acquisition system SACD

SACD supports a knowledge acquisition approach in two main phases: the prescriptive text compilation according to the macrostructure and microstructure grammars; the integration of the compiled structures within the deontic knowledge base. Two types of individuals may use the system: the domain expert (E1) who knows the text characteristics and the intricacies of the legislation; the knowledge engineer (E2) who knows SACD and can modify the grammars if needed. SACD architecture is presented in figure 4.

The first function (P1) enables the domain expert (E1) to enter and edit the regulation text (A1). Function P2 is controlled by the domain expert (E1) and parses the text (A1) in two steps using the metalangage grammars (macro- et micro- structures text grammars) (A2) and creates the structures compiled from the text (A3). The resulting parses are checked by the domain expert. Then, function P2 interacts with the domain expert (E1) in order to classify new objects in the object type hierarchy (A4). Finally, function P2 presents the detected anaphoric references to the domain expert (E1) in order to solve them and to indicate the referenced objects: an equivalence table for anaphors is updated (A5). If problems are encountered by the system, they are stored in (A6) and will be solved later by the knowledge engineer (E2). For more details see (Moulin and Rousseau 1990-1).

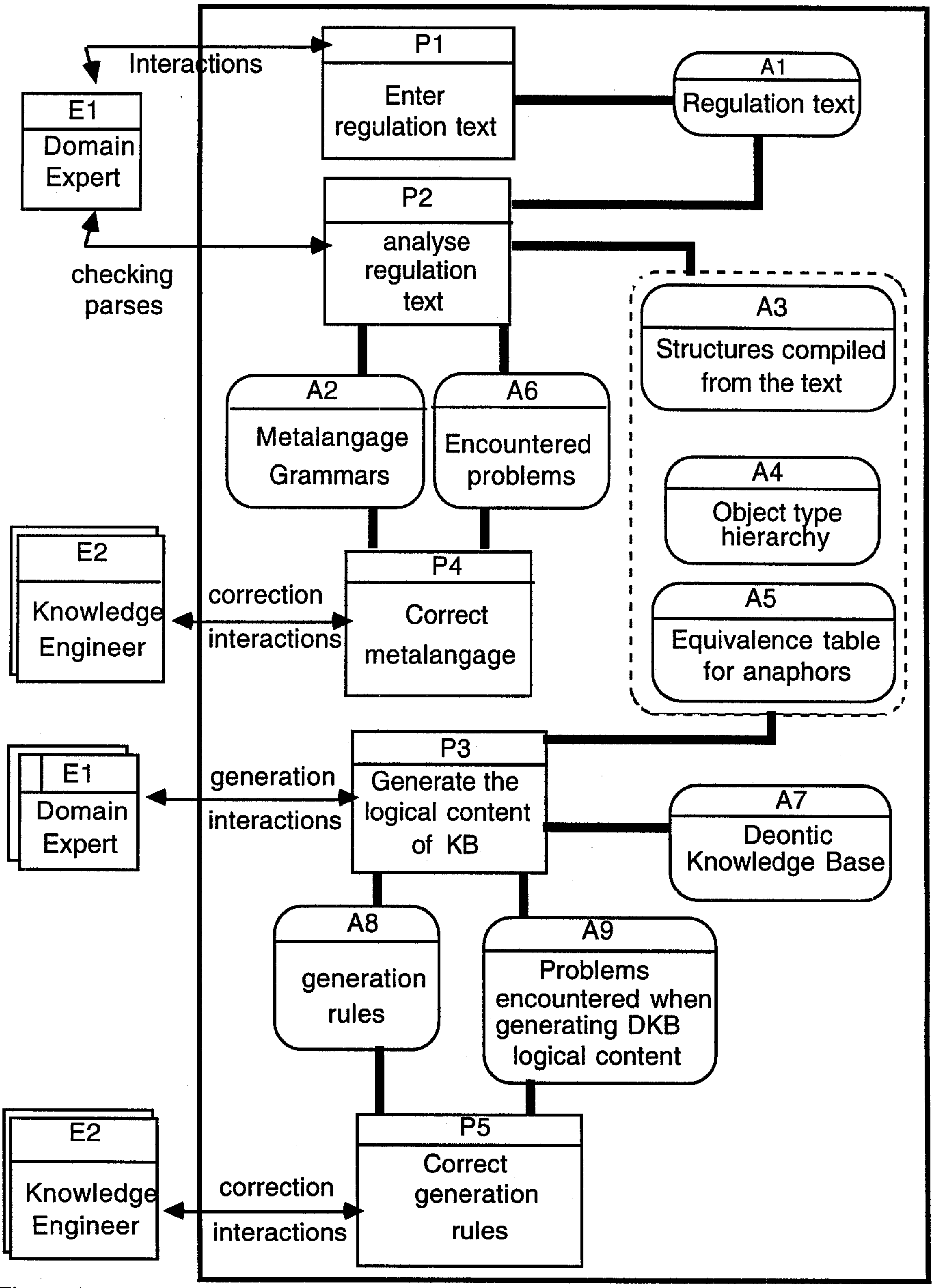


Figure 4 The knowledge acquisition system SACD

Function P3 proceeds in two steps for generating the logical content of the deontic knowledge base (DKB) (A7) in the form of extended production rules called "deontic rules". These rules are associated with a modality and attached to the corresponding objects. In a first step, P3 generates the initial content of the DKB while processing the text sentences separately: the deontic rules are obtained along with an initial application context (the rule premisses deriving from the sentence processing). Internal references may modify the application context of text articles. In a second step, P3 modifies the DKB logical content, taking into account the internal references which have been detected by function P2: the application context of some deontic rules may be modified and new rules may be added. The deontic rule generation process is detailed in the last section.

Function P4 enables the knowledge engineer (E2) to correct the metalangage grammars (A2) after studying the problems encountered during the parsing (A6). Hence SACD's text grammars are enriched as the result of the processing of new texts. In a similar way function P5 enables the knowledge engineer to correct the rules (A8) which are used to generate the logical content of the knowledge base (A7) according to the problems encountered (A9) during the generation process.

## Deontic rule generation

Starting with the structures obtained after the text compilation (A3 in figure 4), SACD generates the deontic knowledge base (DKB). The DKB differs from traditional rule bases in several ways: modalities are attached to rules; rules are associated with objects; the rules corresponding to the prescriptive text provisions interact with each other as a result of the internal references interpretation.

For example let us consider the case of a negative interstatement connector such as "however" and the two following provisions:

Pr1': If context1, X mod.op1 P1.
Pr2': However, X mod.op2 P2 if context2.

where context1 and context2 correspond respectively to the application conditions of Pr1 and Pr2; mod.op1 and mod.op2 correspond respectively to the modal operators of Pr1 and Pr2; P1 and P2 correspond respectively to the conclusions of Pr1 and Pr2; X is the object to which both provisions are associated.

Let us now examine what is the impact of the interstatement connector on the logical interpretation of propositions Pr1 and Pr2. "However" introduces Pr2 as an exception with respect to the preceding proposition Pr1. Implicitly, application context1 holds for proposition Pr2, and, application context2 does not hold for proposition Pr2.

So if we want to get two independent rules we have to transform both propositions in the following way:

Pr1': If context1 and if not (context2), X mod.op1 P1.
Pr2': If context1 and if context2, X mod.op2 P2 .

Obviously, the transformation of the proposition logical content depends on the category of the internal reference contained in the proposition. In SACD these transformations are encoded in rules for modifying contexts (contained in A8 in figure 4).

In order to obtain a DKB composed of independent deontic rules, we have selected the following principles:

- the DKB is specified with rules (called "deontic rules") in which we can indicate a

modality and characterize the object to which each rule is related;
- deontic rules should be independent from each other;
- we can associate one or several rules to each sentence of the initial prescriptive text, and each rule refers to the sentence from which it derives;
- conditions and / or exceptions expressed in a sentence are used to enunciate the basic context of the corresponding deontic rule (the basic context is equivalent to the proposition application conditions, before any transformation related to internal references processing);
- the basic context of a deontic rule can be enriched if other sentences refer (through internal references) to the sentence from which the rule derives.

The DKB is composed of independent deontic rules. In order to apply the transformations required by the logical interpretation of internal references, the DKB is generated in two phases:

- Phase 1: for each phrase (v-phrase) found in the prescriptive text, SACD generates a deontic rule (v-rule) with its basic context, without considering internal references.
- Phase 2: for each metatextual statement, SACD modifies the contexts of the rules associated with the phrases that are in the scope of the corresponding internal reference.

For example the rules that are generated by SACD after step 2 for the provisions of figure 1 and 2 are the following (the English translation is given in figure 5):

---

REGLE: règle1-n3-a3181 PHRASE: phr1-n3-a3181

PREMISSE:

----------------

SI NON(la partie supérieure d ' un mur coupe-feu sépare des
aires de plancher ayant des usages principaux autres que
ceux du groupe E ou du groupe F , division 1 ou 2 )

CONCLUSION:

----------------

Tout [mur coupe-feu ] exigé séparant un ou des bâtiments
comportant des aires de plancher ayant des usages
principaux du groupe E ou du groupe F , division 1 ou 2
/ OBLS:doit / former une séparation coupe-feu de
construction incombustible d ' un degré de résistance au feu d ' au moins 4 h

---

REGLE: règle2-n3-a3181 PHRASE: phr2-n3-a3181

PREMISSE:

----------------

SI la partie supérieure d' un mur coupe-feu sépare des aires de plancher ayant des usages principaux autres que ceux du groupe E ou du groupe F , division 1 ou 2

CONCLUSION:

-----------------

le [degré de résistance au feu ] de la partie supérieure
/ PERM:peut / être réduit à 2 h

------------------------------------------------------------------------------------------------

RULE: règle1-n3-a3181 PHRASE: phr1-n3-a3181

PREMISE:

-------------

IF NOT ( the upper portion of a firewall separates floor areas containing other than Group E or Group F, Division 1 or 2 major occupancy)

CONCLUSION:

------------------

Every [firewall] which separates a building or buildings with floor areas containing a Group E or Group F, Division 1 or 2 major occupancy
/ OBLS:shall / be constructed as a fire separation of non combustible construction having a fire-resistance rating of at least 4 h

------------------------------------------------

RULE: règle2-n3-a3181 PHRASE: phr2-n3-a3181

PREMISE:

-------------

IF the upper portion of a firewall separates floor areas containing other than Group E or Group F, Division 1 or 2 major occupancy,

CONCLUSION:

------------------

the [fire-resistance rating] of the upper portion
/ PERM:may / be reduced to 2 h

**Figure 5 : English translation of rules obtained after processing sentences phr1-n3-a3181 and phr2-n3-a3**

Notice that rule "règle2-n3-a3181" contains a premise which is the negation of the condition of provisions of figure 2. That's the logical effect of the interstatement connector "cependant" ("however").

## Concluding remarks

SACD is implemented in Prolog and runs on Macintosh platforms. It has been used for analysing relatively small regulatory texts (written in French) from the Government of Quebec composed of approximately 100 provisions, as well as portions of the National Building Code. The macrostructure analysis of a 100 articles regulation text takes around 5 minutes. The microstructure analysis of a sentence within an article depends on the complexity of the sentence structure, but it is usually completed in less than 5 seconds. Again, the efficiency of deontic rules generation depends on the number of cross-references found in the original text: deontic rule generation takes few minutes for a 100 articles regulation text.

As an extension of the current research, we will work on bigger prescriptive texts (such as NBC which is composed of more than 1000 articles) and study several issues such as: parsing efficiency, use of predefined object hierarchies, version management in order to recompile only the updated text portions and to regenerate only the corresponding parts in the knowledge base.

SACD analyses prescriptive texts at a syntactic level, using the microstructure text grammar. This kind of analysis provides meaningful results because prescriptive texts are composed of normative propositions in which modalities are stated explicitly using modal verbs like "shall", "may" etc. SACD cannot be used for analysing regulatory texts which are not prescriptive texts. However, this limitation is not really critical since most regulatory texts in Quebec have the properties of prescriptive texts according to governmental recommendations Tremblay 84].

SACD generates a deontic knowledge base (DKB) composed of deontic rules attached to objects, these objects being organized in a type hierarchy. As we mentioned earlier, the specialist can easily examine the deontic rules displayed by SACD in a natural language format in order to verify the logical content of the prescriptive text and to check its consistency. In a given application domain, object type hierarchies generated from existing texts can also be used as preliminary frameworks for the preparation of new regulatory texts in that domain. Furthermore, associated with browsing facilities, the DKB could be used to retrieve in a regulatory text the provisions related to a given subject. Instead of searching through the original text, a user could browse through the DKB and get relevant information along with its logical interpretation. These browsing facilities are currently under study.

One of our goals is to use the DKB in order to build expert systems exploiting the knowledge extracted from prescriptive texts. In a preliminary study we have showed that in its current format, the DKB can be used to develop simple expert systems for checking situations or cases against the provisions of the regulatory text. The expert system uses the premises of the deontic rules to ask questions to a user in order to describe the characteristics of the examined situation. These questions can be generated by using some interrogative expressions such as "Can you tell me if ..." appended to the proposition found in the premise or in the conclusion of the deontic rule. Those propositions can require some syntactic modification such as replacing anaphoras by the objects they refer to, changing some verb tenses etc. For example the premise of the first deontic rule of figure 5 can be used to generate the question:"Can you tell me if the upper portion of a firewall separates floor areas containing other that Group E or Group F, Division 1 or 2 major occupancy ?". By comparing the user's answers with the expected responses according to the corresponding deontic rules, the expert system can conclude that the described situation conforms to the regulation or violates it, and it can mention the regulation provisions that are violated. Such "check list" diagnosis systems can prove useful for helping people to check themselves if they conform to regulations in various domains such as "automobiles", "buildings", "municipal regulations" etc.

SACD could be enhanced in order to use the deontic knowledge bases for the creation of more sophisticated expert systems, such as an expert system for checking if building specifications conform to the Canadian National Building Code prescriptions. For this purpose, the propositions contained in the premises and conclusions of the deontic rules may require a further refinement in order to characterise which are the object properties involved in the propositions and what are their expected values. Such a refinement will require that a semantic analysis be carried on these propositions. We can anticipate that at the proposition level the semantic analysis will be much simpler than if it had been done on the whole text at once. Furthermore, the availability of the object type hierarchy, and eventually of preexisting descriptions of the domain object properties can facilitate this analysis.

## Acknowledgments

This research is supported by CEFRIO, le Centre francophone de recherche en informatisation des organisations and by the National Science and Engineering Research Council of Canada.

## Note

Note 1: In expert systems reasoning is usually based on the modus ponens inference rule which states that if A => B holds and if A is true then we can infer that B is true. In contrast with traditional expert systems which manipulate simple facts, prescriptive texts contain provisions which contain a modality (obligation, permission or interdiction). Deontic logic provides the formal apparatus to manipulate rules in which appear such modalities. For instance, suppose that the deontic knowledge base contains one rule R1: If situation X, then object A shall do P. Suppose also that in the fact base which describes the case to be checked, situation X is true. We cannot use the modus ponens inference rule and infer directly that object A has done P. We can only check if object A has done P, and if object A has not done P, we can conclude that rule R1 has been violated.

## Bibliography

[Bench-Capon 87] Bench-Capon T. J. M. ,G. O. Robinson, T. W. Routen, M. J. Sergot, "Logic programming for large scale applications in law", in [ACM 1987].

[Bench-Capon 89] Bench-Capon T. J. M., "Deep models, normative reasoning and legal expert systems", in Proceedings of the *Second International Conference on Artificial Intelligence and Law*, Vancouver BC, June 1989, ACM Press. [Boose 1990] Boose J. H., B. R. Gaines editors, proceedings of the *Fifth Banff Knowledge Acquisition for Knowledge-Based Systems Workshop*, November 1990.

[Corriton 88] Corriton J. C., F. M. Lesaffre, SYMA : "système expert en milieu administratif", in [*Rault* 1988].

[Deschamps 87] Deschamps J., M. Quenillet, "Systèmes experts juridiques: une réalité", in proceedings of the *seventh Workshop Expert Systems and their*

*Applications*, Avignon May 1987.

[Hilpinen 81] Hilpinen R. editor, *New Studies in Deontic Logic*, D. Reidel, Dordrecht.

[Gardies 80] Gardies J-L, "La structure logique de la loi", *Archives de philosophie du droit*, tome 25, pp 109-121.

[Jacobs 88] Jacobs P., ZERNIK U., "Acquiring lexical knowledge from text: a case study", Proceedings of the *AAAI88*, St. Paul, Min.

[Kalinowski 72] Kalinowski G., *La logique des normes*, Presses Universitaires de France.

[Lagace 87] Lagace J., "Expressions utilisées dans les renvois", dans *Légistique*, bulletin de rédaction législative et réglementaire, Société québécoise d'information juridique, n1 janvier 1987, pp 21-30.

[Lesaffre 90] Lesaffre F-M , L. Brunessaux, C. Oggioni, J. Zuck, "Un système expert réglementaire: ALPIN, Aide à la liquidation de pension d'invalidité", proceedings of the Ninth Workshop on *Expert Systems and their Applications*, Avignon France May 1990.

[MacCrimmon 89] MacCrimmon M. T., "Expert systems in Case-based law: the Hearsey rule advisor", in Proceedings of the *Second International Conference on Artificial Intelligence and Law*, Vancouver BC, June 1989, ACM Press.

[McCarthy 80] McCarthy J., N. S. Sridharan, "The representation of an evolving system to legal concepts: Logical Templates", in proceedings of the *Third Biennal Conference of the Canadian Society for Computational Studies of Intelligence*, Victoria, B. C. 1980.

[McCarthy 84] McCarthy J., *Intelligent legal information systems: problems and prospects, in Data Processing and Law*, C. Campbell editor, Sweet and Maxwell pub.

[Moulin 88] Moulin B., *Réflexions sur la conception et l'utilisation de bases de connaissances déontiques*, Rapport de recherche CEFRIO.

[Moulin 89] Moulin B., D. Rousseau, "Conception de bases de connaissances déontiques pour la consultation de textes de réglement", proceedings of the *Eighth Workshop on Expert Systems and their Applications*, Avignon France May 1989.

[Moulin 90-1] Moulin B., D. Rousseau, "Structuration d'une base de connaissances à partir du contenu de textes prescriptifs", Conférence sur le traitement du langage naturel, in proceedings of the *Tenth Workshop on Expert Systems and their Applications*, Avignon France May 1990.

[Moulin 90-2] Moulin B., D. Rousseau, "A knowledge acquisition system for analysing prescriptive texts", in (Boose1990).

[Moulin 91] Moulin B., D. Rousseau, "Extracting logical knowledge from prescriptive texts in order to build deontic knowledge bases", in proceedings of the *Sixth Banff Knowledge Acquisition for Knowledge-Based Systems Workshop*,October 1991. [NBC 1985] Canadian National Building Code, Associate committee of the National Buiding Code, National Research Council of Canada, Ottawa, 1985.

[Nitta 88] Nitta K., J. Nagao, T. Mizutori, "A Knowledge Representation and Inference System for Procedural Law", *New Generation Computing*, vol. 5 n. 4.

[Québec 87] Gouvernement du Qubec , *Règlement sur le transport par taxi*, révisé 1987.

[Rault 88] Rault J. C. editor, proceedings of the *Eighth Workshop on Expert Systems and their Applications*, Avignon France May 1988.

[Rissland 87] Rissland E. L., K. D. Ashley, "A case-based system for trade secrets law", in [ACM 87].

[Schild 88] Schild U. J., "JURIX : a legal expert system", in [Rault 88].

[Sergot 86] Sergot M. J., F. Sadri, R. A. Kowalski, F. Kriwaczek, P. Hammond, H.T. Cory, "The British Nationality Act as a logic program", *Communications of the ACM 29.*

[Sowa 84] Sowa J.F., *Conceptual Structures: Information Processing in Mind and Machine*, Addison Wesley.

[Tidrick 88] Tidrick T.H., "Expert consultant for law enforcement : the case of alleged revenue officer embezzlement" , in [Rault 88].

[Tremblay 87] Tremblay R., "Le renvoi", dans *Légistique*, bulletin de rédaction législative et réglementaire, Société québécoise d'information juridique, n1 janvier 1987, pp 3-20.

[Tremblay 84] Tremblay R., R. Journeault, J. Lagacé, *Guide de rédaction législative, publication du Gouvernement du Québec*, Société québécoise d'information juridique.

[Von der Lieth Gardner 87]
Von der Lieth Gardner A., *An artificial Intelligence Approach to Legal Reasoning*, The MIT Press, Cambridge, Mass.

[Von Wright 68] Von Wright G. H., "An essay in deontic logic and the general theory of action", in *Acta Philosophica Fennica*, vol 21, North Holland, Amsterdam 1968.

[Vanier 89] Vanier D. J., *Computerization of Building Regulations*, National Council of Canada, Institute for Research in Construction.

[Vanier 90a] Vanier D. J., *Hypertext - a computer Tool to Assist Building Design*, National Council of Canada, Institute for Research in Construction.

[Vanier 90b] Vanier D. J., *Expanded Index to the NBC 1990*, working paper, National Council of Canada, Institute for Research in Construction.

[Vanier 90c] Vanier D. J., Liaw S., *Electronic Browsing of the National Building Code*, National Council of Canada, Institute for Research in Construction.

[Vanier 90d] Vanier D. J., *Electronic Code Evaluation*, National Council of Canada, Institute for Research in Construction.

[Vanier 91] Vanier D. J., "A parsimonious classification system to extract project-specific building code", in proceedings of *UTT symposium series*, Helsinki, Finland May 1991.