Leda: a semi-intelligent legislative drafting support system

Voermans, W.J.M.

LEDA: A SEMI-INTELLIGENT LEGISLATIVE DRAFTING-SUPPORT SYSTEM

Wim Voermans and Egon Verharen

Summary,

In this contribution recent Dutch theoretical and practical developments are discussed in computerized - semi-intelligent - assistance of legislators. Special attention is paid to the so-called drafting-support systems. In the Netherlands two drafting-support systems are being — or have recently been — developed (LEDA, developed for the Dutch Ministry of Justice and OBW developed by the Dutch Ministry of Education and Science). This paper will deal with one of these systems (the LEDA-system) in particular. In discussing the development, the structure and particular functionalities of the LEDA-system some general characteristics and possibilities of legislative drafting-support systems will be illustrated.

1. Introduction

Over the past few years the developments in computerized - semi-intelligent -assistance for legislators have been significant. These developments are both of a theoretical and a practical nature. This paper will sketch an outline of these developments, and discuss one semi-intelligent legislative drafting tool (LEDA) more in particular.

Legislative and legal reasoning

Theoretically important is the notion that from an AI-point of view the legal decision-making process and the legislative decision-making process cannot be treated indiscriminately. The legislative decision-making process is only partly dependent on legal problemsolving, legal knowledge and legal reasoning. In comparison with other forms of legal problemsolving (like application of the law), legislative problemsolving, i.e. the decision-making process aimed at the enactment of legislation, is much more dependent on world knowledge (common sense), and it equally involves, throughout the different stages, substantial political, economic and social-scientific reasoning.[Snellen, 1987/Habermas, 1992] Furthermore, the legislative process does not primarily result in legally (in)valid conclusions, but rather in 'relatively appropriate' solutions, or in convincing arguments.[Hotz, 1984] Whether a bill is an appropriate answer to a legislative problem does only partly depend on its legal quality, and, vice versa, the correct application of legal requirements does not automatically procure good or appropriate bills.[Voermans et al., 1992] These differences between the legislative process (and its components) and the process of legal problemsolving[1] amount to the conclusion that comprehensive automation of legislative reasoning, using AI-methods and AI-techniques, is (still) not possible, due to the complexity of reasoning and the structure of the knowledge involved. This conclusion does not rule out the relevance of legal computer science and AI-techniques for certain legislative activities however, even when they depend upon (legal) knowledge. It does mean, though, that in efforts to build (intelligent) tools and systems to support legislative activities, the standard approaches of legal KBS-development will not always apply. Legislative support systems will have to be developed in accordance with the specific characteristics of legislative activities.

Practical developments

These theoretical insights in the nature of the legislative process and legislative activities have already resulted in the actual development of custom-made practical applications. Different (experimental) systems were built in order to support different legislative activities, especially during the departmental drafting stage, to which domain

J.S. Svensson, J.G.J. Wassink and B. van Buggenhout (eds.) (1993). Legal Knowledge Based Systems: Jurix '93: Intelligent Tools for Drafting Legislation, Computer-Supported Comparison of Law.

this paper will be restricted. According to their functionality, these systems can be divided in two major categories:

a. legislative analysis and review systems
b. semi-intelligent drafting-support systems

2. Legislative analysis and review systems

Legislative analysis and review systems assist legislators in determining the consistency or the consequences of already *existing* (draft) regulations. To be able to perform this functionality, natural language (draft) regulations have to be translated or modelled in terms of knowledge representation formalisms in order to allow the system to reason with it. Although the need for a formal translation can pose a serious drawback in the time-pressed legislative drafting process, these systems have obvious advantages for complex legislative drafting projects, especially when draft-regulations have considerable quantitative (e.g. financial) aspects, or when numerous behavioral possibilities and situations have to be normalized in a consistent manner (e.g. traffic regulation) [Allen et al., 1988/Den Haan et al. 1991]. An additional benefit of draft-analysis and -review systems is that these systems force legislators to think more fundamentally about the deontological structure of their drafts. This confrontation may invite them to come up with logical equivalent alternatives for certain solutions. The necessary formalization and representation of drafts can also result in blueprints for knowledge based administrative (handling)systems. This latter pungent, but in most cases still latent, feature is hardly ever discussed in legal computer science literature however.[2]
In the Netherlands, two legislative analysis and review systems have been developed by the Ministry of Social Services and Employment (ExpertiSZe[3]) and by the University of Amsterdam (TRACS[4]).

3. Semi-intelligent drafting-support systems

Where legislative analysis and review systems come in when (draft) regulations have already been made, drafting-support systems function in situations where there does *not yet exist a draft*, but - for instance - only a relatively vague notion that legislation can procure the answer to a certain (social or policy) problem.[5]

The drafting process
Drafting regulations is not just a matter of putting down policy choices into words, but involves a decision-making process in which many substantive choices regarding content, structure, structure elements and - eventually - phrasing and wording of a draft have to be made. Quite frequently legislative drafting even means that policy decisions have to be made or reviewed.[6] While making these choices a lot of requirements have to be met. These requirements are not only of a homogeneous nature, comprising legal standards (e.g. constitutional standards) and aspects of legislative policy and technique, but also of a heterogeneous nature resulting from various factual conditions related to particular subject matter, or from existing policies regarding the field of the projected draft. The drafting process is a complex decision-making process which requires great skill and knowledge. In the Netherlands most of the legislative drafting is therefore carried out by specialists within the ministerial departments. To ensure the quality of their drafts, these legislative specialists - in most cases - approach the drafting process methodically. Although these approaches vary between the different departments, some general characteristics of these approaches to legislative design can be discerned. Generally speaking, these approaches consist of the following (iterative and interdependent) steps:

1. problem definition (including the determination of the policy and of the legislative goals of the draft solution);
2. problem analysis (including the determination of the relevant legal and factual contexts);
3. generating of alternative solutions;
4. analysis of the different solutions (in the light of the goals, context and effects);
5. selection of a solution (in the light of the goals, contexts and effects);
6. implementation of a solution in a legislative text;
7. evaluation.

This model of the legislative design process (the design-step-model) does not always concur with the actual designing procedure. According to the nature of a specific project, sometimes only a few steps within this process are deemed necessary. Sometimes steps in this iterative cycle are repeated. Analytically speaking, however, this process model is empirically[7] and prescriptively[8] substantiated.
This analysis model also constitutes the pretext and the knowledge-backbone of two design support systems that have recently been developed for the Dutch Ministry for Education and Science (OBW[9]) and The Ministry of Justice (LEDA[10]).

Semi-intelligent drafting support systems
Although the open nature of legislative problemsolving and the substantive reliance on word knowledge resist comprehensive automation of legislative reasoning (see § 1), AI-techniques can be used for the developement of drafting-support systems. For instance, the two Dutch systems, LEDA and OBW, use these techniques to represent methodological knowledge according to the above mentioned design-step-model (using the frames representation formalism). In both systems the various design-steps derived from the design-step-model constitute instances within a hierachically ordened (hypertext) network. These instances, which are visually represented in the interface as different screens (OBW) or levels within a screen (LEDA), (can) possess various attributes and methods. Sometimes a level or a screen in the network will comprise (access to) textual information about the desired level- or screen-activity, and sometimes it will contain a procedural rule (or a hierarchical hypertext-link) regarding the hierarchical place and status of the level/screen and the permitted procedures between the various levels/screens.
Both systems support users by pre-structuring the drafting process and offering knowledge-based access tot relevant information. They do this by using knowledge-based drafting-templates (LEDA) combined with hypertext-based information access and document-assembly (LEDA and OBW).

In the next paragraphs we will try to illustrate in more detail the way in which AI-techniques can be used for the development of legislative drafting-support systems by discussing the development, structure and functionalities of the LEDA-system. To be able to do this it is necessary, though, to consider the background of and motivation for the development of the LEDA-system.

4. Motivation for the development of LEDA

Over the past ten years the Dutch government has — due to serious problems regarding the quality and effectiveness of legislation — become increasingly concerned with the quality of legislation. To improve the overall legislative quality, different policies were pursued and enacted [Legislation in perspective, 1991]. One of the main results of these governmental efforts and policies was the adoption of a general legislative policy, which consists of a set of measures aimed at the lasting improvement of legislative quality by setting quality criteria.[11] A substantial part of these measures concerns the fundamental drafting stage.

The Recommendations for regulations
To guarantee attention for the legislative quality more effectively during the drafting stage, new *Recommendations for regulations* were drawn up in 1993 [Recommendations, 1993]. These Recommendations consist of 346 directives and guidelines regarding important drafting issues and activities. Aside from legislative technique issues, like terminology and model clauses, they also deal with policy aspects, methodological issues, procedures, structural design etc. Although they closely resemble ordinary legal rules, they are of a different nature, however. They are not always generally binding, like legal rules, but directives that can, in certain cases, be ignored if there is a good reason to do so.[12] They constitute a mix of legal (constitutional) rules and a guidelines concerning `best practices and solutions', derived from legislative experience. Besides legal rules, best practices, and legislative quality criteria, a large amount of quality safeguards are incorporated throughout the 346 Recommendations. The Recommendations therefore can be considered a voluminous `Draftman's handbook' dealing with every important activity within the drafting process (see the design-step-model in § 2). Related to the activities in the drafting process, the Recommendations can be categorized into the following groups:

a. Recommendations concerning *preparational methodological and substantive* issues (preparatory activities);[13]
b. Recommendations concerning the *structural design* of a draft (arrangement of the elements in the draft);
c. Recommendations *concerning phrasing and terminology* (including the use of model clauses, model presentation-letters etc.);
d. Recommendations concerning *procedures.*

The text of the Recommendations, however, is not organized along the chronological and methodological lines of the drafting process, but rather thematically in the order of diminishing abstraction. This circumstance makes it quite difficult for legislators (even experienced ones) to find their way through the new Recommendations during the drafting stage. An information system, it was felt, could be the way out of these problems. This meant the start of the LEDA-project.

The goals of the LEDA-project
The main goal in the LEDA-project[14] was to make the information of the Recommendations themselves accessible in concordance with the information-need during the different stages of the drafting process. A secondary goal was to make the information, referred to in the Recommendations (secondary information), available to the users. Many Recommendations, as it happens, do not prescribe what the solution has to be in a certain factual situation - as is often the case with ordinary legal rules - but rather prescribe which activity should be undertaken at a certain moment, and what kind of information is needed to be able to perform the prescribed activity. The third goal of the LEDA-project was to offer knowledge-based drafting-support on the basis of the legislative knowledge within the Recommendations, pursuant to the knowledge-based access of the information from the Recommendations.
In 1993 the project resulted in the prototype LEDA-system, which is currently being tested and validated within the Ministry of Justice.

5. LEDA

LEDA is a prototype Legislative Design and Advisory system designed to offer access to the Recommendations (and secondary information) in a methodical way, concurrent with the stages of the drafting process, and through this offer knowledge-based support for the drafting activities of legislators regulated in the Recommendations. To this end LEDA contains four major (integrated) functionalities, namely:

a. methodological support;
b. document drafting and assembly support;
c. knowledge-based information retrieval;
d. legislative advice.

These functionalities are integrated throughout the system and can best be discussed by a description of the functionalities of the system's modular components. LEDA consists of two major modules:
1. the Preparatory (policy) Module;
2. the Basic Design Screen.

5.1 The Preparatory Module
The preparatory module in LEDA was set up to offer knowledge-based access to the Recommendations concerning substantive, methodological and structural design issues, in a way consistent with the chronology of events in the drafting stage (see for this chronology the design-step-model in § 2).

Representation
In the Preparatory Module of LEDA the different preparatory methodological activities, regulated in the Recommendations are represented in a methodological way. We have pointed out already that the Recommendations are not arranged methodologically, but thematically. In order to be able to offer methodological guidance and assistance in LEDA, we first had to distil the methodologically important issues and activities from the Recommendations, and assess their interdependencies. To discover the methodologically important elements, we used an analysis-frame, based on a quite traditional model of the different components or elements of a norm [Ruiter, 1987]. Each separate Recommendation was analyzed with the following terms derived from the norm-element model:

Recommendation (norm) *object* (or activity):
Recommendation (norm) *condition*:
Recommendation (norm) *operator*:
Recommendations (norm) *subject*:

The next step was to analyze the relations between the activities we discovered. For this we supplemented the original analysis-frame with extra slots in order to be able to conclusively asses the relations between the normalized activities. The second analysis-frame looked as follows:

Rec. object:
- activity type:
- activity trigger:
- required information input:
- information output:

Rec. condition:
Rec. operator:
(Rec. subject:)

On the basis of this two-step analysis we were not only able to distil the preparatory legislative activities and their interdependencies from the Recommendations, but we were also able to construct a hierarchical frames-representation of the different drafting activities themselves, and their mutual relations. The latter operation resulted in a model which closely resembled the design-step-model discussed in § 2, consisting of seven major consecutive design steps. Within the design steps of the model, several interrelated substantive (subordinate) activities, issues and questions, regarding the preparation of a draft and the draft structure, are represented in their turn. In this way the analysis resulted in a methodological transposition of the Recommendations into a design-step model.

An obvious advantage of the frames representation in the model was, that we were able to assess the information-basis of the different activities formulated in the model. This resulted in the conclusion that, although many activities were information-based, they relied on formally representable (e.g. legal) knowledge only to a very small extent (see also § 1). That part of the knowledge which could be formalized (e.g. the knowledge about structural design), was, together with the methodological drafting model, formalized and represented using the frames-representation formalism. Most knowledge was represented in simple frameslot procedures regarding hierarchical and referential relations and serving to address relevant blocks of information, or support limited inferencing.

Knowledge-based modelling of a hypertextnetwork
The analysis and the methodological frames representation proved that drafting activities rely strongly on information. This indicated that hypertexttechnology was a suitable candidate for the technical implementation. From a functional point of view the hypertexttechnology aims to enable users to make their way through a body of complex information in a manner that facilitates its ready appreciation or visualization. [Mital et al., 1992] From a more conceptual point of view, hypertexttechnology provides the means for non-linear text organization in computers by associating windows on the screen with objects in the database and providing links between those objects both graphically (as labelled tokens) and in the database (pointers).[Conklin, 1987] To make this possible hypertextnetworks possess nodes and links, governing the relationships between the various nodes. Links and nodes can have a variety of properties. Nodes can, for instance, consist of (or better: correspond with database objects which `contain') chunks of textual information, but they can also contain (a piece) of a knowledge-based template, which contains hypertextlinks in its turn.[15] Links can connect nodes in different ways. To establish this connection they can consist of simple or quite elaborate (knowledge-based) procedures. There are two methods for explicitly linking two points in a hypertextnetwork: by reference and by organization. The referential method supports non-hierarchical (for instance: associative) linking of nodes. The organizational method on the other hand explicitly creates hierarchical connections, by connecting a parent node with its children, thus establishing a strict tree subgraph within a hypertext network graph.
Without a further discussion of all the different potent possibilities of hypertext-technology, it will be evident that it was not hard to transpose the methodological frames-representation (within our design-step-model) into a hypertextnetwork. We used the frames-representation specifically to model the hypertextnetwork to our needs. For instance: in order to model the hierarchical links in the hypertextnetwork, we used the methodological knowledge about drafting activities represented in the frame-network. In the same way we modelled the network's referential links and inference procedures. This enabled us to create a hypertextnetwork which does not only provide very flexible information linking, but which also dynamically produces knowledge-based templates, and substantively as well as methodologically supports legislative drafting. [Verharen et al., 1992]
For the experimental realization of the system, we initially used a development tool called Toolbook (an MS-Dos version of Hypercard). The prototype of LEDA is however developed in Borland C++, using the object-oriented programming paradigm.

Functionalities of the Preparatory Module
The Preparatory Module (PM) combines the functionalities of a hypertextsystem with a knowledge-based (KB-) template system. The hypertext-based PM of LEDA permits the user not only to draft a preparatory document (e.g. a policy memorandum), but also supports the creation of a skeletal form for a KB-template, to be used for the actual structural design and formulation of a draft (Basic Design Screen). To this end the Preparatoy Module guides the user through a hypertextnetwork of semantic hierarchical and referential links. To offer guidance, the hypertextnetwork of the PM is divided into different levels, corresponding with the different methodological steps of the design-step-model derived from the Recommendations. The levels in their turn serve as a

checklist, expressing important points of attention regarding methodological aspects and the structural design of a draft.
A look at the systems' interface architecture (which closely resembles the functional architecture) may illustrate these features:

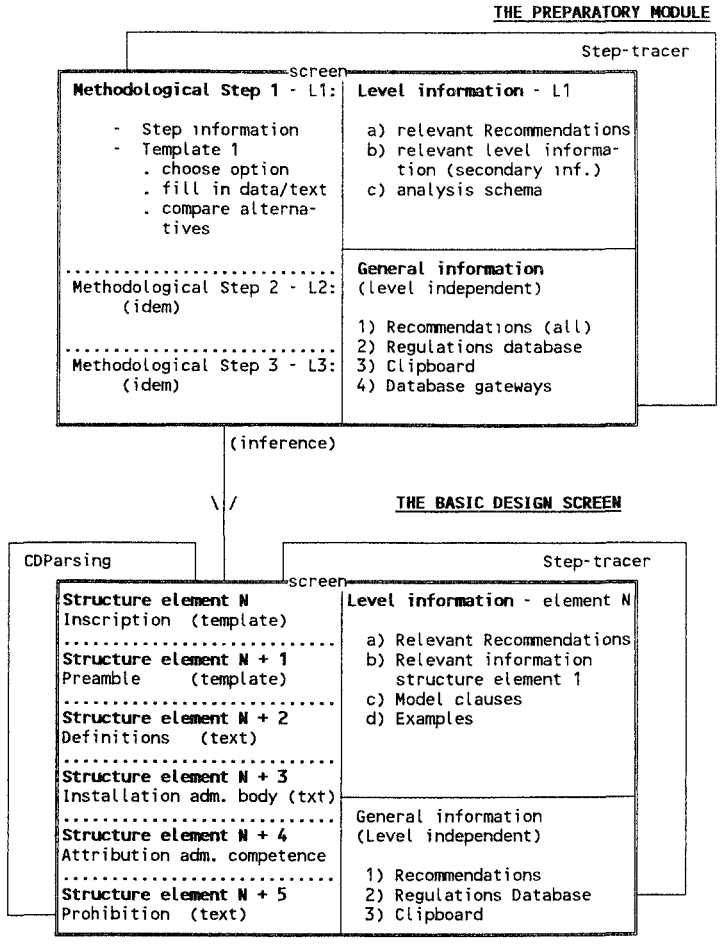

figure 1 LEDA: Interface-architecture (functional modules and components)

As figure 1 shows, the Preparatory Model consists of various methodological and consecutive levels (dotted lines on the left hand side). These methodological levels are referentially linked with level information (box at the upper right hand side). The level information component consists of (access to) the relevant Recommendations, access to relevant secondary information (as referred to by the relevant Recommendations), and a graphic template-scheme for user-analysis of certain options. Level information changes according to the level which is active (i.e. the level in which the user is working).

The methodological levels themselves consist of fields containing information (about what is to be done within a certain level) and knowledge-based templates. The level-template-documents which mainly serve to insert (or draft) text, also support the identification of important sub-items, and the choice between options. Both on the basis of the choice of the user and automatic analysis of text-input in the template, the system makes inferences regarding the arrangement of levels further down the network's path (e.g. the arrangement of the levels in the Basic Design Screen). From the point of view of the user, the levels form an interactive word-processor which provides methodological guidance and provides relevant (semantically interlinked) information. The user may progress randomly through the level-structured hypertextnetwork. This fundamental openness of the system is necessary as the user-legislator is always free — when drafting a legislative text whithout the use of the system — to deviate from the Recommendations themselves whenever there is a good reason.[16] To accommodate reluctant users, there is even a possibility of to shut down the levels altogether. What remains is a word-processor linked to information in a single default-information level explaining the methodological approach of the Recommendations, and providing (links to) the relevant Recommendations and secondary information.

To prevent getting lost in the hypertextnetwork, user-guidance is provided by the levels themselves, together with easy backtracking procedures and a step tracer, which consist of a major and minor active compass which visibly records the path hitherto followed in the network. On top of this the PM is provided with a General Information-component to offer non-hypertextual access to various internal or external databases. Users can retrieve text from these databases while working in the different levels. The text in the internal databases, however, is hypertextually linked.

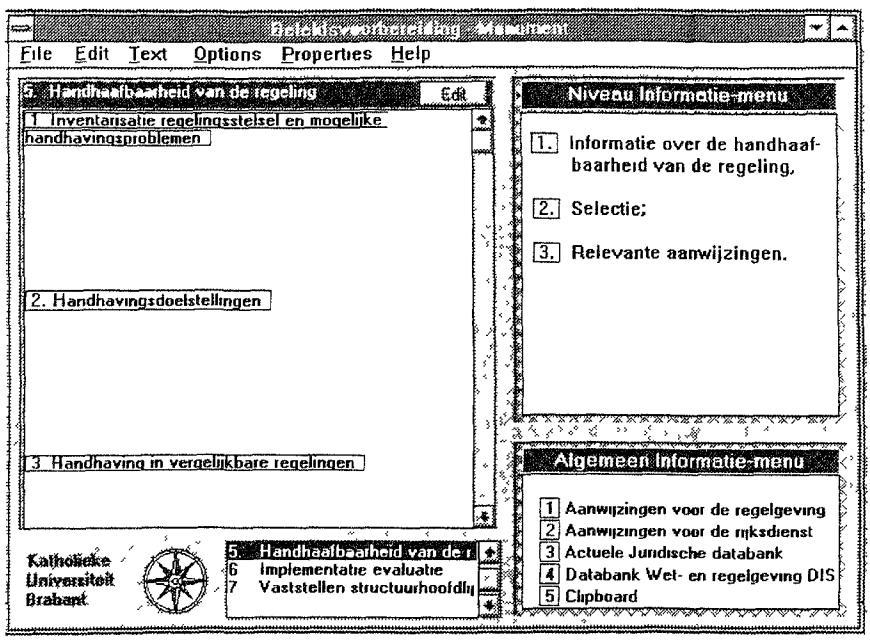

figure 2: *the PM-interface*

5.2 The Basic Design Screen

The Basic Design Screen Module (BDS) is developed and structured in a way very similar to the Preparatory Module. Like the PM it consists of a level structure, linked with level information. The levels (see the dotted line in the BDS-module of figure 1) contain templates mainly consisting of free-text fields, which allow system supported insertions (e.g. of model clauses or examples). The templates within the levels of the BDS however do not express points of attention with regard to the preparation and structural design, but important phrasing, terminology and terminology-related (substantive) issues regarding the structural elements of a draft. The arrangement of the levels in the BDS is both based on knowledge (gained from the Recommendations) and knowledge-based inferences made by the PM module. The BDS itself can be regarded as one large knowledge-based template which is shaped and directed by the PM. The BDS represents the preferred structure of a draft, modelled to the needs of the user.
Like the PM the BDS has a very open structure: the user may progress randomly, do away with the levels altogether and receive default-information, and delete or add certain levels. The user-guidance function is similar to the one in the PM. The BDS has, however, one distinctly different feature compared to the PM. It possess a conceptual dependency parser [Schank et al., 1981].

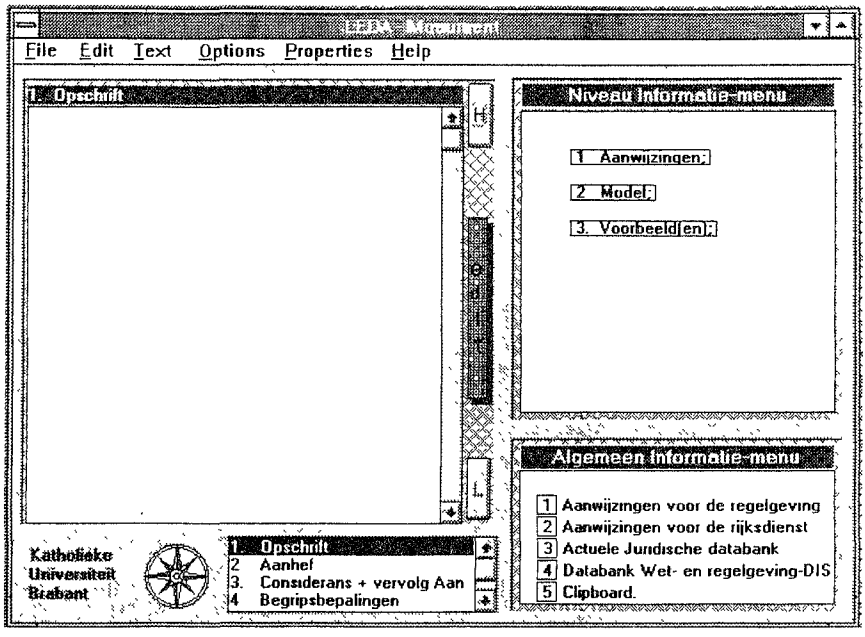

figure 3: *the BDS-interface*

The CD parser

When a user has finished the drafting of a text (within a certain level of the BDS), he may be interested to know whether he has overlooked a relevant Recommendation. To accommodate this interest LEDA possesses a conceptual dependency parser (CDP). This CDP automatically analyzes (parses) the user-inserted text in a BDS level and dynamically creates hypertextlinks to a particular relevant concept in the database or the text of a Recommendation if the text-analysis indicates the relevance. To be able to do this the CDP not only detects key-words and key-word-combinations and matches them with patterns in the database (stringmatching), but also analyzes concepts in text sentences (by using the linguistic conceptual dependency method [Schank et al., 1981])

and matches them with concepts in the database (automated conceptual information retrieval). Concepts in a user-inserted text within a level are analyzed using the norm-element model (see § 5.1) [Ruiter, 1987]. This norm-element-model distinguishes between four major concepts within a sentence expressing a norm,[17] looks as follows:

1. a (deontic) *normoperator* (expressing in a natural language terms an Obligation, a Permission, a Prohibition or a Command. Grammatically speaking this operator will always be a conjugation of a verb);
2. a *normobject* (grammatically: a set of substantive and/or adjective nouns, conjugated verbs and conjunctions constituting the direct object-clause of the sentence);
3. a *normsubject* (grammatically: a pronoun or a substantive noun combined with a definite or indefinite article constituting the subject of a sentence);
4. a *normconditon* (grammatically: verbs, (pro)nouns, conjunctions and articles constituting the adverbial clause of a sentence).

Natural language-analysis on the basis of this norm-element-model is possible because the concepts in the database, which are modelled as frames on the basis of knowledge derived from the Recommendations, have slots corresponding to this norm-element-model. The CDparser will check the patterns and concepts in the database (or knowledge base) to see which concept is applicable. An example may illustrate. Suppose a user inserts the following text in his draft:

"Our minister can set rules regarding the administration of licences."

The CD-parser in LEDA will (in this hypothetical situtation) match this piece of natural language with the concepts in the database. Two relevant concepts (frames) will prove to be applicable. First of all the concept (or pattern-concept):

Frame-Legislative Terminology[18]

type:	indication-minister
general indicator:	minister*, our minister*
Recommendations:	30,69,73,74,75
related frames:	delegation-minister, subscription-minister, attribution of administrative authority-minister
operation:	**if** (indicator **then** show_link **proc** show_link /*on demand show corresponding leaflet*/)
leaflet indication-minister:	"**Terminology-Indication of ministers** The following Recommendations concerning the way in which minister are to be indicated in legislative texts are most likely to be relevant: *73* (indication of a minister) *74* (indication of more than one minister) *75* (etc.) See also: *Delegation of regulatory authority to ministers (30,69)* *Subscription & ministers* *Attribution of administrative authority & Ministers*"

NB. The italicized texts indicate/refer to hypertextlinks which are connected to text of a Recommendation or to a leaflet (with text and links) in another concept in the database.

The second concept that will be found reads as follows:

Frame-Delegation[18]

type:	regulatory authority
operator-indicator:	*can, may, set, sets, regulate, regulates* (etc.)
object-indicator:	rule, rules, set..rules (etc.)
subject-indicator:	minister, ministers, government
condition-indicator:	{language}
operation 1:	**if** (operator_indicator, **if** object-indicator, **if** subject-indicator, **if** in norm_sentence **then** show link **proc** show link /*on demand show corresponding leaflet A {delegation of regulatory authority to ministers}*/
operation 2:	**if** (operator_indicator, **if** object-indicator, **if** in norm_sentence **then** show link **proc** show link /*on demand show corresponding leaflet B {delegation of regulatory authority}*/
leaflet A-delegation-minister:	"**Delegation of regulatory authority to ministers** The following Recommendations are most likely to be relevant: *30* (Delegation of regulatory authority to ministers) *69* (Terminology ministerial delegation) See also: *Indication of ministers*"
leaflet B-delegation:	"**Delegation of regulatory authority** The text seems to indicate delegation of regulatory authority. To whom is this authority to be delegated? *government (see: delegation of regulatory authority to government)* *a minister (see: delegation of regulatory authority to ministers)* (etc....p.m.) a mouse-click on the italicized text will indicate your choice"

NB. The italicized texts indicate/refer to hypertextlinks which are connected to text of a Recommendation or to a leaflet (with text and links) in another concept in the database.

This - due to the inherent limits of this paper - briefly illustrated form of conceptual dependency parsing, combined with automated conceptual information retrieval [De Mulder et al., 1993] is very powerful because both the concepts in the level-related text and the concepts in the database can be quite accurately defined. For the user it supplies a powerful semi-intelligent Recommendations check of his natural language draft.

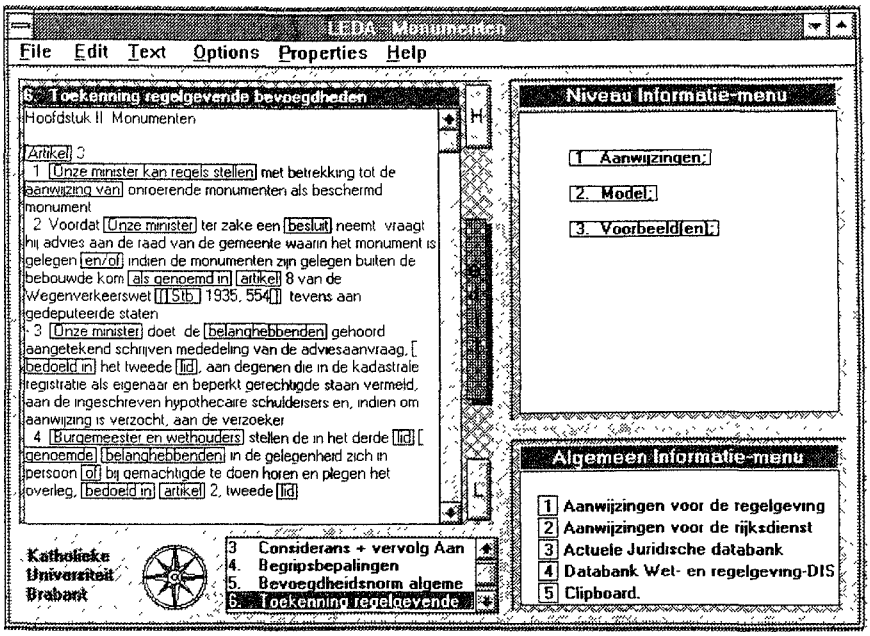

figure 4: *The CD Parser at work*

6. Conclusion

By pre-structuring the draft-process and offering knowledge-based access to relevant information LEDA (as well as the OBW-system of the Dutch Ministry of Education and Science) can be considered a first modest step on the way to a really intelligent drafting system. In a number of ways they semi-intelligently support the complex task of drafting a bill. However, in our view really intelligent legislative drafting systems can only be realized when features of legislative design-support systems are combined with the characteristics of legislative review and analysis systems. This combination of draft support and analysis/review systems is, however, for the moment, blocked by the necessity of user unfriendly and complex knowledge representation and formalization of natural (draft) language to accommodate the operation of analysis and review systems. There may be a way out of these problems, however: one day conceptual dependency parsing of natural language may well provide the solution, by allowing for automatic knowledge representation and formalization of knowledge-concepts, contained in the natural language of a draft.

Notes

1. These differences, however, are more like differences in scale than intrinsic differences. See [Wahlgren 1992, 147]. The span of this paper however does not allow for an elaboration of these theoretical questions, interesting as they may be.
2. Overhoff and Molenaar [Overhoff et al., 1991] mention this interesting possibility in discussing the relevance of the decision-table technique.
3. See for a detailed discussion of ExpertiSZe, [Wassink, 1992]
4. See [Breuker 1991/Den Haan et al., 1991].
5. However, sometimes this notion will not be vague at all. When for instance a higher ranking statute stipulates delegated regulation of a certain kind, within a certain period, legislators will not be free at all to determine whether legislation is

necessary or not. In these circumstances drafting discretion can be strictly limited (see for instance Art. 7, 22 and 40 `Wet persoonregistraties').
6. See the directives 6-18 of the *Recommendations for regulations*.
7. See the results of empirical research conducted within the Dutch Ministry of Education and Science [Wassink, 1992].
8. See directives 6-18 of the Recommendations for Regulations.
9. OntwerpBank Wet- en regelgeving (Regulations Design Bench) developed by the Dutch Departement of Education and Science and Bolesian.
10. Prototype of a LEgislative Design and Advisory system developed at Tilburg University
11. See for a more detailed discussion of these measures the contribution of dr. Ph. Eijlander in these proceedings.
12. See directive 5 of the Recommendations.
13. This group of Recommendations addresses questions like: what is the problem? What are the goals for a solution? Is legislation necessary to resolve the problem? If legislation is inevitable, what kind and sort of regulation will have to be drafted? Which substantive elements does it have to contain? What are the alternatives? Can it be enforced properly? etc., etc.
14. Subsidized by the Dutch Ministry of Justice.
15. See for a detailed discussion on the use of knowledge-based templates combined with hypertext techniques to enable userfriendly document-drafting and document assembly: [Mital et al., 1992], p. 123-166 (Chapters 7,8 and 9).
16. See directive 5 of the Recommendations for regulations.
17. The beginnning and the end of a norm sentence do not necessarily concurr with the beginning and the end of natural language sentences.
18. In the actual LEDA-system the concepts have a totally different form and substance. The concept mentioned here serves as a natural language illustration of the structure of a LEDA-frame-concept.

References

[Allen et al., 1988] Allen, Layman and Saxon, Charles, Exploring Computer-Aided Generation of Questions for Normalizing Legal Rules, in Charles Walter (ed.), *Computer Power and Legal Language*, New York, Westport, Connecticut, London, 1988, p. 243-317

[Breuker, 1991] Breuker, Joost, Towards a workbench for the legal practioner, in C. van Noortwijk, A.H.J. Schmidt, R.G.F. Winkels, *Legal knowledge based systems*; Aims for research and development, Lelystad 1991, p. 25-35

[Conklin, 1987] Conklin, J., Hypertext: an Introduction and Survey, in *IEEE Computer*, september 1987, p. 17-41

[Den Haan et al., 1991] Den Haan, Nienke and Breuker, Joost, A tractable juridical KBS for applying and teaching traffic regulations, in J.A. Breuker, R.V. de Mulder, J.C. Hage (eds.), *Legal knowledge based systems, model-based legal reasoning*, Lelystad 1991, p. 5-16

[Habermas, 1992] Habermas, J., *Faktizitt und Geltung*, Frankfurt am Main, 1992, especially p. 236, 360 and 430-435

[Hotz, 1984] Hotz, R., Strukturierung des Vorverfahrens der Gesetzgebung -Erste Schritte zu einem allfälligen Einsatz von Computern bei der Schweizerischen Gesetzgebung, in: Theo Öhlinger (Hrsg.), *Gesetzgebung und Computer*, München 1984.

[Legislation in Perspective, 1991] Netherlands Ministry of Justice, *Legislation in perspective* (a policy plan for the further development of the general legislative policy, aimed at improving the constitutional and administrative quality of government policy), The Hague 1991

[Mauldin, 1991] Mauldin, Michael L., Conceptual Information Retrieval (a Case Study in Adaptive Partial Parsing), Boston, Dordrecht, London, 1991

[Mital et al., 1992] Mital, V., Johnson, L., *Advanced Information Systems for Lawyers*, London 1992, Chapters 7-10, p. 125-166

[De Mulder et al., 1993] Mulder, R.V., Wildemast, C., Van den Hoven, J., Conceptueel geautomatiseerde juridische documentatie-systemen, in *Computerrecht*, 2, 1993, p. 69-77

[Overhoff et al.,1991] Overhoff, R.W., Molenaar, L.J., *In de regel beslist*, Den Haag 1991

[Ruiter, 1987] Ruiter, D.W.P., *Bestuursrechtelijke wetgevingsleer*, Assen 1987

[Schank et al., 1981 Schank, R.L., and Riebeck, C.K., *Inside Computer Understanding*, Hillsdale, NJ, 1981

[Snellen, 1987] Snellen, I.Th. M., *Boeiend en geboeid*, oratie KUB, Alphen aan den Rijn 1987

[Sprowl, 1979] Sprowl, J.A., Automating the legal reasoning process: a computer that uses regulations and statutes to draft legal documents, *American Bar Foundation Research Journal* 1979, p. 1-73

[Verharen et al., 1992] Verharen, E., Fridael, M., Voermans, W., Experimenteel model LEDA: kennisgebaseerd gebruik van de hypertexttechniek in een wetgevingsontwerp- en -adviessysteem, in B.R. van der Spek, R. de Hoog (eds.), *Proceeding Kennistechnologie conferentie oktober 1992*, Den Haag 1992, p. 329-340

[Voermans, 1991] Voermans, W., Computer-aided legislative design: worth while the effort?, in C. van Noortwijk, A.H.J. Schmidt, R.G.F. Winkels, *Legal knowledge based systems*; Aims for research and development, Lelystad 1991, p. 87-96

[Voermans et al., 1992] Voermans, W., Maas, F.H.D.M., Nouwt, J. en De Wild, A.H., Are Legislators beyond Help from Legal Computer-Science?, in *AIgg Kennisgeving*, jg. 5, nr. 3, september 1992, p. 2-11

[Wahlgren, 1992] Wahlgren, P., *Automation of Legal Reasoning*, Deventer/Boston 1992

[Wassink, 1992] Wassink, J.G.J., *Kennistechnologie en het ontwerpen van regelgeving*, (brochure), Den Haag 1992.