# A Use Case Analysis of Legal Knowledge-Based Systems

Thomas F. Gordon
*Fraunhofer FOKUS*
*Kaiserin-Augusta-Allee 31, 10589 Berlin, Germany*
*Thomas.Gordon@fokus.fraunhofer.de*

**Abstract.** A number of important use cases for public administration ("e-governance") applications of legal knowledge-based systems are identified. For each use case, a data flow model is developed showing processes and document types sufficient to implement the use case. Our main goal is to identify document types which, if standardized, would enable a modular set of interoperable components for building legal knowledge-based systems.

## 1 Introduction

This paper provides a high-level process analysis of the major **use cases** of legal knowledge-based systems, with a focus on public administration applications. Our analysis is based in part on prior work by the ePower project [4, 5] and the survey of AI methods and tools for supporting legal argumentation by Bench-Capon, et. al., in [1]. Use cases are a software engineering modeling and diagramming technique, standardized as part of the Unified Modeling Language. Use cases are very high level models showing how different kinds of actors interact with a system to carry out tasks. They are useful for helping to get an overview of the desired functionality of the complete system and are an "essential tool" for identifying requirements [6].

Figure 1 is a UML diagram showing all of the use cases we have identified. There are five main kinds of actors participating in these use cases: *legal analysts* (lawyers) and *knowledge engineers* (computers scientists) work together to build formal, computer models of legislation; *clients* (citizens or businesses) consult *advisors* (such as lawyers or tax consultants) take part in a structured dialog, for example to apply for social benefits or a building permit, with a *public agency*. Both the advisor (or the client) and the public agency use these models and other resources to analyze the issues of a case and to generate arguments.

The following section present process models showing the document types and processes required to implement each use case. In doing this, our main goal is to identify document types which, if standardized, would enable a modular set of interoperable components for building legal knowledge-based systems. Ideally, appropriate industry standards for the document types in these processes would enable components to be used together and allow consumers to freely choose products for each of the components. Presumably competition would help to improve quality and reduce prices while the availability of several competing products for each component would provide public administration and other users with a measure of security protecting their investments.

Thomas F. Gordon, 'A Use Case Analysis of Legal Knowledge-Based Systems' in D. Bourcier (ed.), *Legal Knowledge and Information Systems. Jurix 2003: The Sixteenth Annual Conference.* Amsterdam: IOS Press, 2003, pp. 81-90.

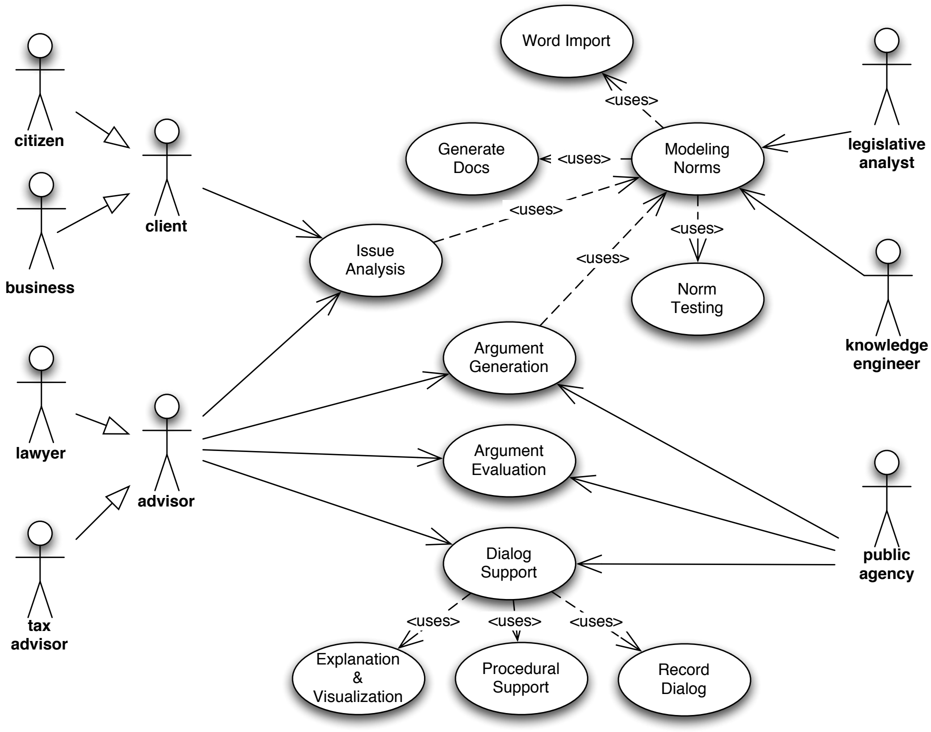

**Figure** 1: Legal Reasoning Uses Cases

## 2 Modeling Norms

Let us begin with the Modeling Norms use case. Here the task is to interpret and analyze legal sources, such as a piece of legislation, a regulation or some case law, to produce a high-level formal model of the concepts and norms in the sources. For our purposes here, it does not matter what knowledge representation methods are applied for this purpose. Figure 2 show the data flow diagram[1] for norm modeling.

There are two starting points in this data flow chart. You can start at the bottom, with an empty norm model, and simply begin to create a model. If, for example, the models are being created using UML/OCL, as in the ePower methodology, this could be done using one of the many CASE tools for UML. Presuming you have a structured, marked-up version of the legislation to be modeled, using for example the MetaLex XML document schema for legislation [2], you can start at the top by generating partial models from this structured document and then integrating these models. For example, the ePower workbench provides a tool for generating models from MetaLex files, using a natural language parser to identify concepts and relations.

Please note that the processes in this and the following data flow diagrams, such as Generate Model and Integrate Model in Figure 2, are not assumed to be performed completely automatically. Rather, the way these processes are performed is left completely open during this analysis. At one extreme, the process could be performed completely manually, in some conventional way. More ambitiously, ICT tools can be imagined which could assist a human user in performing these tasks. Most advanced would be full automation, but this is clearly beyond the state of the art for many if not most of the processes discussed in this paper.

[1] To be precise, we use UML Activity Diagrams as data flow diagrams.

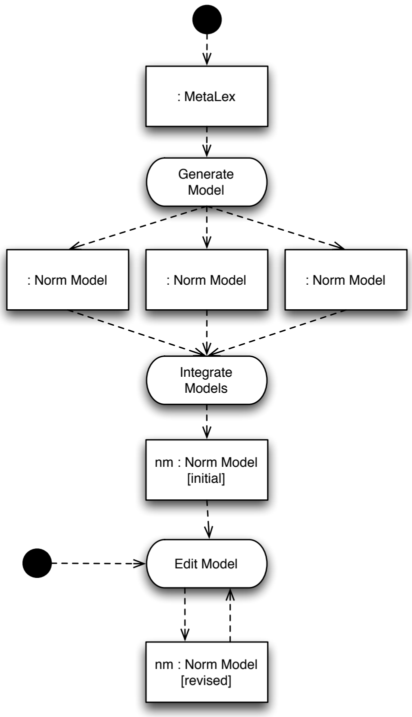

**Figure** 2: Modeling Norms

## 3 Word Import

The ePower workbench also provides a tool implementing the Word Import use case from Figure 1. The data flow diagram for the Word Import use case is shown in Figure 3.

The structure parser converts a Microsoft Word version of a piece of legislation into MetaLex, which is an XML Schema for legislation and other legal texts. The structure parser depends on the structure of the Word document having been previously marked up by applying paragraph and character styles from a particular Word template. This conversion process is made necessary by the wide spread use of Microsoft Word to draft legislation. Perhaps some day this will be done using XML editors, making this conversion process unnecessary. The Check References process of the Word Import use case makes sure that all cross references in the legislation are correct, or at least do not refer to non-existing sections or paragraphs.

In the discussion so far we have identified three document types which are candidates for a standardization effort: 1) the Word Template for marking up legislation using Microsoft Word; 2) An XML document schema such as MetaLex for marking up the structure of legislation using XML; and 3) some document type for models of norms. More will follow below.

## 4 Model Testing

Figure 4 is the data flow diagram for testing norm models. This is the first process to use argumentation, which will be explained shortly. The input to the process is some representation of a case and the norm model. The diagram illustrates the testing process for a single case, but this process usually would be iterated for some set of test cases. A case here should consist at least of the legal facts of the cases, that is the facts in terms of the legal terminology (or "ontology") used by the norm model, the legal issue or issues to be decided and the outcome or decision. For decided court cases, the outcome would be the actual decision reached by the court. But for testing models of new legislation the test cases may be "hypotheticals" with

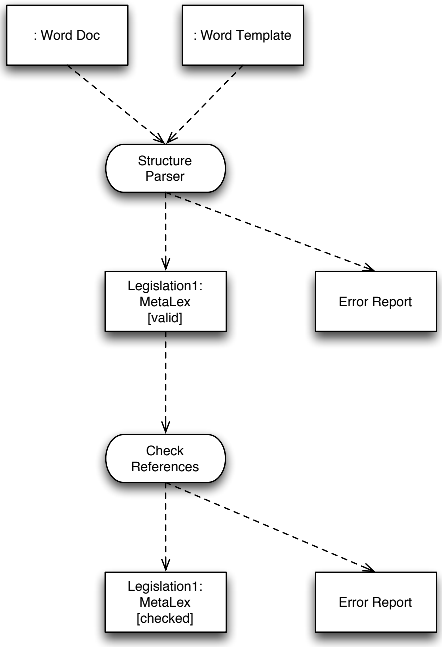

**Figure** 3: Word Import

the desired outcomes given the policy behind the legislation.

Now we explain the use of argumentation in this process. We do not restrict the norm models to logical representations, let alone to representations in monotonic logics such as first-order predicate logic (FOL). Rather, we want to also support models which use, for example, case-based reasoning or some kind of defeasible (nonmonotonic) logic. The problem here is to find a way to make the results of reasoning with such heterogenous models somehow comparable and interoperable. Our proposed solution is to have each kind of model explain its reasoning as a set of arguments in a standard format. Each argument instantiates an argumentation schema, where each kind of model can define its own schemas.

The Generate Arguments process applies the model to the facts of the case to try to resolve the issues of the case. Essentially, the issues are interpreted as queries to be posed to the model. The model may generate conflicting arguments, pro and contra some resolution of an issue. The task of the Apply Arguments process is to link up these arguments into an argumentation graph. The Evaluate Arguments process then labels the arguments, or the propositions in the premises and conclusions of the arguments. These labels record whether the argument or proposition is *acceptable* (i.e. to be considered proven or “true” given all the arguments made). The Check Results process then compares the decision of the case with the labels of the issues in the argument graph to determine whether they are consistent or compatible.

The precise form and semantics of legal arguments, argument schemas, argument graphs is an active area of research [7, 10, 15]. But all approaches fit the abstract process model presented here. Again, our goal here is not to solve these research problems but to identify document types and processes which (eventually) should be standardized. Here we see the need for standards to represent cases, norm models, arguments and argument graphs.

One important advantage of this process model is that it is easily extensible to cover reasoning with **multiple norm models**. Many legal problems can only be resolved by considering a multiple sources of norms. For example, a cross-border commercial transaction between parties in Germany and The Netherlands might be regulated by The Unitied Nations

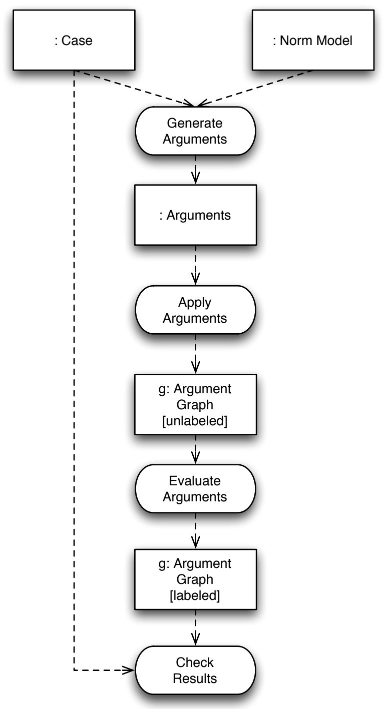

**Figure** 4: Model Testing

Convention on the International Sale of Goods[2] (CISG), laws of the European Union and the national laws of both countries, not to mention case law about all of these legal texts.

This completes our presentation of the use cases for modeling norms, except for the Generate Docs use case. This is about generating documentation of norm models from templates and style sheets in various output formats, such as HTML, PDF, or even Word. Since document generation is well understood and well supported by industry standards, we have omitted a more detailed discussion to save space.

## 5 Issue Analysis

Let us turn our attention to the more interesting use cases in which clients with the help of their advisors interact with public agencies, beginning with Issue Analysis. To illustrate the issue analysis use case, let us use the following hypothetical from [3]. A client comes into an attorney's office and tells the following story:

At the place where I work, our bookkeeper didn't give me my paycheck. Instead she signed my name on it, cashed it an left town. I asked my boss for a new check but he said he already paid me. Is he right that he doesn't owe me my wages?

Although from a common sense perspective the client's story and problem are quite clear, the problem is to identify the legal issues, if any. To do this, the story needs to be interpreted in light of knowledge of one or more areas of substantive law. Is this a commercial law problem, a labor law problem, or perhaps some kind of tort? (Or perhaps more than one of these.) That is, the lawyer needs to use his knowledge of the law to first try to narrow in on one or more areas of law which might be relevant. Then, for each of the selected areas of law, the common sense terms of the client's story need to be mapped into the technical legal terms. Branting shows the result for the US Uniform Commerical Code:

[2] http://www.cisg-online.ch/cisg/conv/convuk.htm

Under Article 3 of the Uniform Commercial Code, is a payor's obligation to a payee discharged by a negotiable instrument if the negotiable instrument is paid to a third party over a forged signature?

Only after the problem has been reformulated in this way can we begin to use some model of the selected legal domain to try to begin to analyse the legal issues and generate arguments for the client.

Although Branting's example is about a private legal conflict, the problem is just as relevant in the context of e-governance and the electronic delivery of public services. One of the goals of e-governance is to provide one-stop shops for citizens on the Web. Supposing that the vision is to provide a wide variety of government services backed by legal knowledge-based systems, how shall the citizen find out which of the many services is relevant for his or her problem or "life event"? The person at the front-desk of the one-stop shop has the role of the advisor in this scenario and faces the same issue identification problem of an attorney or other legal advisor.

Figure 5 is the data flow diagram for the issue identification process. (Incidentally, this process is called "theory construction" in [1], but since this term is also used for the process of creating a theory explaining a set of precedent cases I prefer "issue identification" here.) There are two main tasks in this process, finding models and interpreting the story of the client. We imagine there to be a library of models, including metadata indexing the models. For each model selected, the story interpretation task translates the story into legal facts and issues.

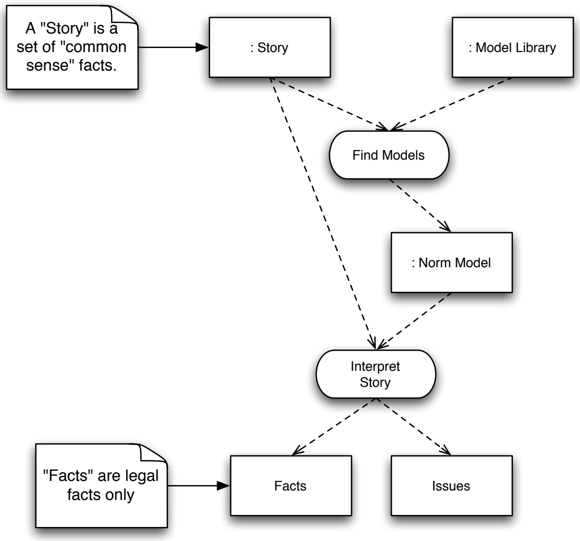
A "Story" is a set of "common sense" facts.
: Story
: Model Library
Find Models
: Norm Model
Interpret Story
"Facts" are legal facts only
Facts
Issues

**Figure** 5: Issue Identification

Document types of the issue identification process which might be suitable for standardization include the norm models, and representation of the legal facts and issues. Legal facts and issues were also a part of the case document type discussed previously, so these should be represented in an integrated fashion, as part of the same document type. The story document, on the other hand, does not seem suitable for standardization, since it is simply the story as told by the client in natural language.

It is not clear how far we can go in providing ICT support for finding models and interpreting stories. Surely some support can be provided, but these are very difficult tasks requiring a great deal of both common sense knowledge and legal expertise.

## 6 Argument Generation

The next use cases to discuss are for generating and evaluation arguments. Figure 6 is a data flow diagram for argument generation. It is similar in some ways to the model testing process discussed previously. Instead of a test case, we have the legal facts and issues which were identified by the previous process. Instead of a decision or desired outcome, some representation of the **interests** of the client are input into the process. These are used to evaluate and select arguments supporting alternative decisions or outcomes, rather than restricting attention to a single desired outcome prematurely. The argument selection process also take into account the prior state of the dialog, recorded in an argument graph, and features of the audience, to make use of rhetorical skills. The result of the argument construction process is a set of selected arguments and a updated argumentation graph including these arguments.

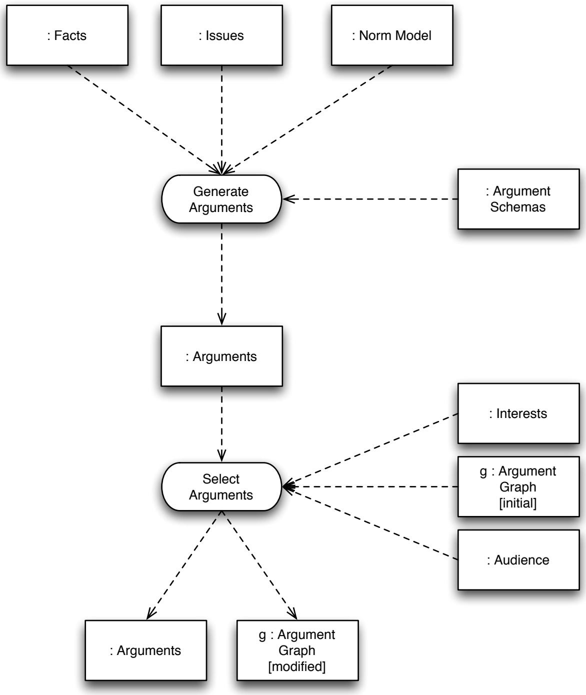

**Figure** 6: Argument Generation

There are no new document types in this process, assuming that legal facts and issues are part of the document type for cases. An exception is possibly the document type for argument schemas, which appears here for the first time. Perhaps argument schemas should be a part of document type for domain models. There seems no pressing need to standardize the representation of interests and the audience, so long as there are no tools or methods for using these representations to select arguments. We are not aware of research results in this area.

## 7 Argument Evaluation

Figure 7 is a data flow diagram for the argument evaluation process. It is deceptively simple. It takes a set of arguments and the prior state of an argument graph, extends this graph with the new arguments and updates the labels of the graph. This process may make use of the argument application and evaluation processes of the model testing process discussed previously. Although not shown in the diagram, a model of **metanorms** for resolving conflicts between norms will be used by this process. Famous examples of such norms include *lex specialis* (prefer the most specific norm) and *lex superior* (prefer the most authoritative norm). But there is a lot of hand waving going on here, since as mentioned previously this is an active area of research. There is as yet no consensus on the form (syntax) or semantics of argumentation graphs. No new document types appear in this process.

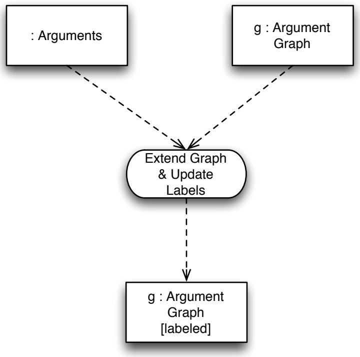

**Figure** 7: Argument Evaluation

## 8 Discourse Support

This nearly completes our discussion of the legal reasoning use cases. It remains only to discuss discourse support. We do not have data flow diagrams for the discourse use cases ready for this paper. Although my colleagues and I have done some work in this area [8, 9, 13] some effort is required to bring this prior work up to date and to integrate it into the general framework presented here. However some general requirements already seem quite clear. The discourse support system must support the use of formal, structured protocols for various kinds of dialog types [17], such as critical discussions, deliberation and negotiation. These protocols will make use of rules of procedure [14], which themselves ideally would be modeled in the same way as substantive areas of law. In the e-governance context, there may be specific administrative procedures to take into consideration. The discourse support system will need to manage the state of the dialog, keeping a record of the moves (speech acts) made and their effects. This includes managing the so-called **commitment stores** of the participants, so that we know who made which assertions of facts. These commitment stores are not only for helping to prevent the participants from contradicting themselves, but also for such mundane purposes as allowing information provided on forms to modified. Finally, the discourse support system should provide tools for presenting and visualizing arguments and argument graphs [12, 16], to make them more understandable without requiring users to become familiar with formal languages.

## 9 Conclusion and Future Work

The main goal of the work presented here has been to identify document types which, if standardized, would enable a modular set of interoperable components for building legal knowledge-based systems. Rather than taking a technology-driven approach, by starting with existing legal knowledge-based systems and looking for common document types, we have taken a requirements-driven approach here, starting with e-governance use cases. In conclusion, the following document types have been identified:

- A Microsoft Word template for marking up legislation and other sources of norms using Word;
- A markup language and exchange format for legislation and other sources of norms;
- A knowledge representation language and exchange format for formal models of norms;
- An exchange format for legal cases, including facts, issues and decisions;
- An exchange format for arguments and argument graphs; and
- Possibly a representation language and exchange format for argument schemas, if this is not part of the language for norm models.

We have conducted a survey of proprietary formats, research prototypes and proposed or emerging standards for these document types. Unfortunately, there is no room to present the results of this survey here; we will try to publish this research in another paper soon. For now, we can only say that our conclusion is that the legal knowledge-based systems industry is still in a very early stage of development.

## References

[1] Bench-Capon, Trevor, James Freeman, Hanns Hohmann and Henry Prakken ”Computational Models, Argumentation Theories and Legal Practice.” Argumentation Machines; New Frontiers in Argument and Computation. Eds. Chris Reed and Timothy J. Norman: Kluwer Academic Publishers, 2003. 85-120.

[2] Boer, Alexander, Rinke Hoekstra, and Radboud Winkels. ”Metalex: Legislation in XML.” Legal Knowledge and Information Systems (Jurix 2002). Eds. T. Bench-Capon, A. Daskalopulu and R. Winkels: IOS Press, 2002. 1–10.

[3] Branting, L. Karl. ”An Agenda for Empirical Research in AI and Law.” ICAIL Workshop on the Evaluation of Legal Reasoning and Problem-Solving Systems. Eds. Karl Branting and Steffi Brüninghaus. Edinburgh: International Association of Artificial Intelligence and Law, 2003. 28–35.

[4] Engers, Tom M. van, Rik Gerrits, Margherita Boekenoogen, Erwin Glassée and Partries Kordelaar ”Power: Using UML/OCL for Modeling Legislation – an Application Report.” International Conference on Artificial Intelligence and Law (ICAIL 2001). Ed. Henry Prakken. St. Louis, 2001. 157–67.

[5] Engers, Tom M. van, and Margherita R. Boekenoogen. ”Improving Legal Quality – an Application Report.” International Conference on Artificial Intelligence and Law (ICAIL 2003). Ed. Giovanni Sartor. Edinburgh: ACM, 2003. 284–92.

[6] Fowler, Martin, and Kendall Scott. UML Distilled – a Brief Guide to the Standard Object Modeling Language. 2nd ed: Addison Wesley Longman, Inc., 2000.

[7] Gordon, Thomas F., and Nikos Karacapilidis. ”The Zeno Argumentation Framework.” Proceedings of the Sixth International Conference on Artificial Intelligence and Law. Melbourne, Australia, 1997. 10-18.

[8] Gordon, Thomas F. The Pleadings Game; an Artificial Intelligence Model of Procedural Justice. Dordrecht: Kluwer, 1995.

[9] Gordon, Thomas F., and Oliver Märker. ”Mediation Systems.” Online-Mediation. Neue Medien in Der Konfliktvermittlung - Mit Beispielen Aus Politik Und Wirtschaft. Eds. Oliver Märker and Matthias Trénel. Berlin: Edition Sigma, 2002. 61-84.

[10] Greenwood, Katie, Trevor Bench-Capon, and Peter McBurney. ”Towards a Computational Account of Persuasion in Law.” International Conference on Artificial Intelligence and Law. Ed. Giovanni Sartor. Edinburgh: ACM, 2003. 22–31.

[11] Johnson, Peter. Legal Knowledge-Based Systems in Administrative Practice and Electronic Service Delivery (E-Government), 2000.

[12] Kirschner, Paul A., Simon J. Buckingham Shum, and Chadd Carr, eds. Visualizing Argumentation: Software Tools for Collaborative Education and Sense-Making. London: Springer-Verlag, 2002.

[13] Mochol, Malgorzata. ”Discourse Support Design Patterns.” Online Dispute Resolution – Legal and Technical Aspects. Eds. Arno R. Lodder, et al. Edinburgh, 2003. 61-74.

[14] Prakken, Henry, and Thomas F. Gordon. ”Rules of Order for Electronic Group Decision Making – a Formal Methodology.” Collaboration between Human and Artificial Societies. Ed. Julian A. Padget. Springer Lecture Notes in Ai. Berlin, 1999. 246-63.

[15] Prakken, Henry, Chris Reed, and Douglas Walton. ”Argumentation Schemes and Generalisations About Evidence.” International Conference on Artificial Intelligence and Law. Edinburgh, 2003. 32–41

[16] Rowe, G. W. A., C. A. Reed, and J. Katzav. ”Araucaria: Marking up Argument.” European Conference on Computing and Philosophy. Glasgow, 2003.

[17] Walton, Douglas N. The New Dialectic : Conversational Contexts of Argument. Toronto; Buffalo: University of Toronto Press, 1998.