School of Informatics, University of Edinburgh

Centre for Intelligent Systems and their Applications

Prototyping a Legal Decision Support System: A Case Study

by

John Kingston, Lillian Edwards, Jean Hall

Informatics Research Report EDI-INF-RR-0165

School of Informatics **November 2002**
http://www.informatics.ed.ac.uk/

Prototyping a Legal Decision Support System: A Case Study

John Kingston, Lillian Edwards, Jean Hall

Informatics Research Report EDI-INF-RR-0165

SCHOOL *of* INFORMATICS
Centre for Intelligent Systems and their Applications
November 2002

Proceedings of Lawtech 2002, the 3rd International Conference on Law and Technology, Boston, Mass., 6-7 November 2002. IASTED.

Abstract :

This paper describes a project to develop a prototype of an intelligent computer system that contains legal domain knowledge which is capable of being accessed via the World Wide Web, and which can support legal decision-making and advice in the area of Scottish divorce law. The results of both the initial feasibility analysis and the issues encountered in the course of the project are used to advise future researchers on how to carry out a prototyping project in a legal domain.

Keywords : Law, Case law, Internet, Webshell

Prototyping a Legal Decision Support System: A Case Study

John Kingston, Lillian Edwards and Jean Hall
University of Edinburgh
Edinburgh, Scotland
Corresponding author: J.Kingston@ed.ac.uk

Abstract

This paper describes a project to develop a prototype of an intelligent computer system that contains legal domain knowledge which is capable of being accessed via the World Wide Web, and which can support legal decision-making and advice in the area of Scottish divorce law. The results of both the initial feasibility analysis and the issues encountered in the course of the project are used to advise future researchers on how to carry out a prototyping project in a legal domain.

Keywords: Internet, divorce law, prototyping, knowledge based systems

1. Introduction

This paper describes work carried out on the project "Web-Based Decision Support for Divorce Lawyers"[1]. The purpose of this one year project was to develop an intelligent computer system that contains legal domain knowledge that is capable of being accessed via the World Wide Web, and which can support legal decision-making and advice. The aim was to investigate the application of technology to representing and reasoning about law by producing a prototype system, which could then be used as a basis for further development if desired. The project was carried out by the Department of Law at the University of Edinburgh, with input from the University's Artificial Intelligence Applications Institute (AIAI).

The domain chosen was Scottish divorce law – specifically, the distribution of matrimonial assets between former spouses after divorce. The overall goal was to improve access to justice by incorporating the information necessary to solve the various legal decisions that need to be made at each stage. It was planned that the system could be used either by expert users to browse through legal cases relevant to each stage in the divorce process, or by less expert users to "run" the system, allowing the system to ask for relevant information and then to make and justify its own decisions, which the users could accept or reject at each stage.

One of the main functions of prototyping within software engineering is to acquire understanding of the domain to clarify requirements for a subsequent system. This is even more true in knowledge engineering, since it's difficult to scope a knowledge based system until some of that knowledge has been acquired. The purpose of this paper is therefore to present this project as a case study of a technology prototyping project in the legal domain, discussing both the successes and the weaknesses of the project and the resulting prototype, with the aim of helping researchers with little software engineering experience to carry out successful prototyping projects in future. The framework for this discussion will be drawn from a set of feasibility-related factors for knowledge-based projects developed at AIAI (see [1] for a preliminary version of these factors). These factors are a mixture of issues to be considered (e.g. "have you considered the involvement and commitment of the domain expert; the developers; the funding body?") and heuristic rules (e.g. "Does the knowledge being collected pass the 'telephone test' – i.e. in an emergency, could it be transmitted via a telephone? If not, the knowledge probably contains spatial or perceptual elements, and will be hard to capture in a knowledge based system"). These factors have been grouped under three main headings – Organisation/Business, Technical/Knowledge and Project/People.

2. Organisational/Business Factors

The first step in developing any knowledge based system is to determine whether there is a

[1] sponsored by the UK Economics & Social Sciences Research Council (ESRC) under grant number XXXX.

"business case" for this system, as well as determining whether the system will fit into existing business practices. This case is often harder to make for knowledge based systems than for more conventional computer systems, since the easiest measure of business benefit is increased productivity, but knowledge based systems often deliver bigger gains from more accurate decision making; from freeing up key senior people from routine tasks by enabling more junior staff to take on these tasks; or from standardizing practice. There are also issues concerning the need for the system: will the task it supports continue to be performed? Would other "knowledge management" approaches (e.g. the creation of a bulletin board for discussion of issues) be more effective? Will the system effectively train its users (which is good) or become a "black box" on which the users rely unthinkingly (which is bad)?

Applying these principles to this project, we observe that the task of splitting up assets after divorce is (unfortunately) one that will continue to be performed; there's no evidence that a bulletin board would provide any significant benefits; and we believe the system will train its users, by offering them access to the relevant information at the time they need to know it As for the 'black box' problem, this was avoided by designing the system so that it provided access to transcripts of relevant legal case histories via the World Wide Web; the users were thus encouraged to access the relevant information and to make their own decisions rather than to rely completely on the system's decisions. The strongest rationale for this project, though, is provided by the nature of the divorce proceedings in Scotland. Despite the adversarial image of divorce proceedings displayed in the media, Scottish solicitors have a preference for seeking consensual negotiated settlements, and a significant number of divorce cases are settled using joint minutes of agreement and/or rubber stamping of informal arrangements that have been negotiated beforehand. It is therefore highly desirable for trainee divorce lawyers and/or mediators to have access to a decision support system to help them in preparing these negotiated settlements. The primary "business benefit" of this system therefore comes from enabling more junior staff to take on the task of dividing matrimonial assets; and since the financial benefit is primarily to society (in particular, divorcees) rather than a single commercial organization, it is appropriate that that this work should be supported by a Government-funded research council. We therefore conclude that the system has a strong business case for providing decision support to trainee lawyers and mediators. For senior lawyers, there is a lesser business case – it helps them keep up to date with the latest case histories -- although making the system available and beneficial to senior lawyers should have an important side-effect: if the senior lawyers use the system, they are more likely to trust it, and therefore to support rather than resist its introduction into the organization.

3. Technical/Knowledge factors

This set of factors is concerned with the technical issues of implementation of knowledge in the prototype system, and of the capabilities of the prototype. These factors cover those aspects of knowledge that affect the difficulty of knowledge capture, knowledge modeling, KBS design and implementation: these include the type of task that is being supported; the form of the knowledge (taxonomic, heuristic, procedural, etc.; see also the "telephone test" described above); the quantity and quality of knowledge; the type of user interface required; and so on.

3.1 Technical

A novel feature of this project was that the system was to be delivered via the World Wide Web. This approach had been tried out before on a student project [2] and had been considered sufficiently feasible to use as a basis for this project. According to the technical feasibility factors, the use of novel interfaces often causes a significant rise in the effort required on the project.

The other major technical factors relate to the completeness of the system. Is there a clear definition of when the system is finished? If not it's common to get "scope creep" – lots of requests to add an extra feature here and an extra feature there, until the prototype has almost as much functionality as the system that would result from a subsequent full-scale development project. And can the knowledge be validated? For if there is no canonical way of checking that the knowledge is correct, it's difficult to declare that the prototype is finished.

We chose to manage the risk of excessive effort on novel interfaces by using a software tool that was already Web-enabled, thus reducing the extra time load required for the researcher to adapt to this tool. Issues relating to completeness of the prototype are discussed in section 5 of this paper.

3.2 Knowledge

The knowledge relating to division of matrimonial property under Scots law that was acquired from the domain expert (and from a book that she co-authored [3]) during the project turned out to be something of a mixed bag:

- A collection of principles based on section 9 of the Family Law (Scotland) Act 1985 provide for property to be "shared fairly between the parties to the marriage", unless one party would be disadvantaged by this relationship (e.g. through taking on the financial burden of child care) or because of certain special circumstances. Each of these provisos breaks down into further sub-questions, often characterised as "X is true UNLESS Y is also true".
- The Act provides some definitions that are important e.g. the definition of what constitutes marital property. These definitions often have multiple conditions.
- However, judges have considerable discretion to vary awards not only according to disadvantages and special circumstances but also according to the total resources of the parties.

Scots law can therefore be seen as a hybrid of rule based laws similar to those found in civilian matrimonial property domains and highly discretionary divorce domains such as are typically found in England, the US common law states, and Australia [4].

In the terminology of knowledge based systems (KBS), these three types of knowledge consist of a *decision tree*; a collection of *rules*; and one or more *assessments*. The "section 9 principles" constitute a decision tree, because they require decisions to made that are based on a number of sub-factors, each of which may in turn be based on a number of factors. The definitions provided by the Act can be represented as rules. And the discretionary factors should be characterised as assessments – or strictly speaking, as decisions that result from an assessment task, for assessment is a knowledge based task in its own right. The components of an assessment task are (in simple terms) *key factors* to consider, the *ideal* and *actual value* of each factor, and a *weighting* to be applied to each factor when making the final assessment.

Applying the technical feasibility factors to this knowledge, we see that the form of the knowledge is symbolic; it is both taxonomic and procedural; and the task type is classification (within a decision tree), but a number of assessment tasks may also have to be performed. We also see that the knowledge is of high quality (i.e. definitive and reliable), and that the quantity of knowledge involves about 100-200 possible decisions, which makes this medium-sized in KBS terms. Furthermore, there is almost[2] none of the types of knowledge that knowledge based systems must work hard to cope with: temporal knowledge, spatial knowledge, reasoning from first principles (e.g. cause-effect reasoning based on the laws of physics) or real-time processing. We therefore see from the prototype that this knowledge is suitable for putting into a KBS although there is a trap for the unwary if it is believed that the task is solely a classification task, without noticing the additional assessment tasks inherent in discretionary decisions.

4. Project/People Factors

These factors ensure that all stakeholders are considered in the development of a system. Stakeholders include the management (who monitor the project, and either provide the funding or are responsible to the funding agency); the users; and the developers. For a knowledge based project, the domain expert is also a key stakeholder. Key factors to consider for each stakeholder are:

Management: Will they support the system throughout its development? Will they prevent 'scope creep' whilst allowing sensible specification changes? Will they support the organisational changes required for the system to be used?

Users: Are they able to use the system – is a brief training session required? Are they

[2] There is a small amount of temporal knowledge in the system, concerning whether some events happened before or after the "relevant date" (which will be either the date of cessation of cohabitation or the date of the summons in the action for divorce).

unwilling to use the system? Or conversely, will they place too much trust in the system?
Developers: Are they trained in knowledge acquisition, knowledge engineering, and in programming with the chosen software tool(s)?
Domain experts: Are they available? Are they willing to share their knowledge? Are they genuinely expert? Can they express their knowledge?

Since this project was funded as a small-scale research project, the 'management' was drawn from university academics and the 'developer' consisted of a single hired researcher. In order to address some of the issues listed above, the project was designed to have two investigators: one from the Department of Law, who also acted as the domain expert; and one from AIAI who could provide technical expertise to the developer where needed. As for the users, it was planned that the final system would be tested by presenting it to a number of users of different levels of skill to obtain their comments on the system itself and on the chosen interface(s).

5. Putting Plans into Practice

At the start of the project, the situation was:

- A good "business case" had been made for the development of the system and funding had been awarded.
- A researcher had been recruited to develop the system. This researcher, S, had reasonably good skills in computing and a background in humanities, but no specific experience in law and only a little experience with AI programming.
- A knowledge based system tool with the capability to be delivered over the World Wide Web had been selected. This was JESS, the Java Expert System Shell [5].
- Two university staff were to act as investigators, one providing domain expertise and the other providing AI technical expertise.
- The project plan was to capture the necessary knowledge, represent it in models, and then implement it.

However, not everything went as planned. While S was a competent programmer who learned JESS, wrote a system with a form-based interface that calculated the relevant date, and provided web links to the relevant case transcripts, S was unfamiliar with knowledge acquisition, knowledge modeling, or law. As S progressed deeper into the domain, he found it difficult to gather and represent legal knowledge. While the prototyping exercise therefore fulfilled its purpose in clarifying requirements and knowledge structure, this required extra effort from both domain expert and the developer, and so the project fell behind its timescales.

There was also an issue of S's commitment to the project; not long before the end of the project, he resigned and took up a PhD in a humanities subject instead. The project was therefore left partially complete with 3 months' worth of funding left. A second researcher, J, was recruited; she had significant experience in building models for software engineering and in general programming but had little experience as a programmer of AI software. She was familiar with an academically developed Web-enabled software tool called Webshell [6], however, which provides a decision tree representation based on an exception table, and is able to use a factor-based AI algorithm when no exceptions can be specified. Because of the short remaining time on the project, and because JESS' license had become more restrictive since the start of the project, it was decided that the remainder of the project would be implemented using Webshell.

J worked hard to bring the project to a satisfactory conclusion. She generated models of knowledge using a directed graph format for the principles and rules and a custom format [7] for discretionary knowledge, until she had modeled all the relevant sections of the Family Law Act; see Figure 1 for an example. She discussed and repeatedly revised these models with the domain expert; and she implemented the resulting models (except for the parts that S had already implemented) in Webshell. However, her industry uncovered other issues that needed to be resolved, but with little time remaining to resolve them. These issues included:

- Where the system requirements differ for expert lawyers and trainee lawyers, which type of user should be targeted? This became important when deciding what parts of the knowledge could safely be left out of the prototype. The domain expert was uncomfortable about cutting any corners in accurate representation of matrimonial property law while the developer felt there was insufficient time to model all this knowledge accurately and implement it, and discussions about the best places to cut

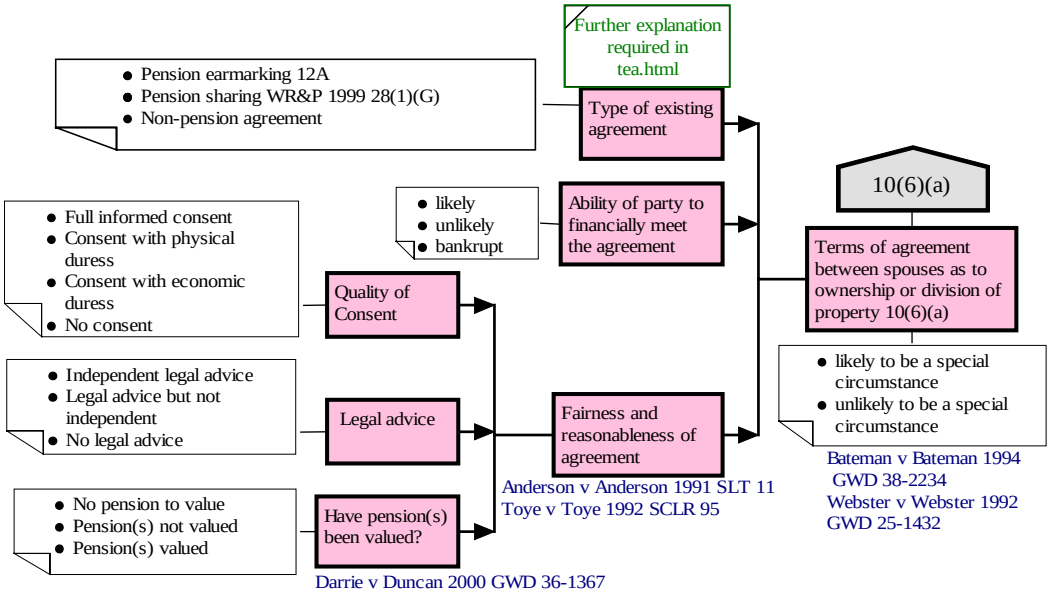

Figure 1: An example model

corners often centred on who the target user would be.

- Technical limitations of Webshell: its database-based design allows users to provide answers to multiple choice questions but precludes users from entering numerical values or other dynamically defined data.
- While JESS runs as a Java applet, Webshell requires a Web server with two or three different packages running on it. Setting up all these packages took some time. The issue of allowing the two systems to communicate also raised some difficulties.
- Finally, the attempt to gather and then to verify a lot of knowledge in a short period of time was constrained by the limited availability of the domain expert. This led to the implementation of some models before they had been thoroughly checked, and then to subsequent revisions of the implementation.

Looking at these issues from the viewpoint of the feasibility factors, we see that the issue of the target users arose because of a case of "scope creep" – or rather, because of difficulties in "scope reduction". Because the initial specification targeted the system at both senior lawyers and trainees, the domain expert wanted the system to reflect all the relevant knowledge for all the relevant people; in other words, to implement most of the functionality required from a full scale system. The need to "cut corners" was certainly not part of the original project plan, and can be largely attributed to difficulties with project staffing; but the real weakness here was that the investigators did not define in advance a set of criteria for determining when the prototype would be considered complete. Even if these criteria had to be cut back at a later stage, their existence would have made discussions on related issues much more focused.

The inability of Webshell to accept numerical values required some workarounds to be devised, one of which came into conflict with the two types of knowledge identified (classification knowledge for principles and assessment knowledge for discretionary factors). The workaround in question was to transfer the responsibility for decision making to the user in "bite-sized chunks" (i.e. highly specific decisions) and then to apply logic to combine these decisions upwards in the decision tree. The issue arose when determining if either party had received economic disadvantage from the marriage; the user would be asked up to ten questions regarding the husband, followed by the same questions regarding the wife, and the system would then conclude that one, both or neither had suffered economic disadvantage. However, while it is logically correct (in a classification task) to state that both husband and wife have suffered economic disadvantage if both of them have positive answers to any of the ten questions, no matter how small the financial value of the disadvantage, the domain expert understandably insisted that the system should be able to determine if one partner had suffered a

much more significant economic disadvantage than the other. Since Webshell cannot accept numerical values, it was unable to calculate the financial value of each disadvantage (equivalent to the weighting in an assessment task). In the end, this had to be resolved by asking the user "On balance, did one partner suffer a greater economic disadvantage?" In short, the prototyping exercise again succeeded in highlighting a requirement for the full scale system – this time, a technical requirement on the software tool.

The issue of Webshell's server, and the problem caused by S's departure, serve to highlight the importance of considering and attending to the needs of all the stakeholders in the project -- including computer support personnel.

6. Discussion

We have developed a prototype of a web-based system to provide decision support to divorce lawyers, and a set of models representing the knowledge in that system. The prototyping exercise has achieved many of its goals of making the requirements for a more detailed system clearer, despite time constraints.

We have succeeded in producing models capturing both the relevant law and domain expertise. These models have been thoroughly checked by a domain expert and have been tested by being implemented in a prototype; this implementation has served as an aid to knowledge acquisition, and has encouraged the domain expert to consider certain areas of the domain more deeply, which may lead to future research. In fact, the models constitute a more significant deliverable from the prototyping project than the software; for while the software implementation is a necessary prelude to any future software project in this area, as well as having some benefits in knowledge acquisition and requirements clarification, it is the models that are most likely to be of direct use to other researchers.

Our advice to future researchers would be:

- Initial feasibility analysis is important; without such an analysis, the project may not even be funded.
- Prototyping is a good way of clarifying requirements and feasibility-related issues for a full scale project.
- For a prototyping project in particular, it is important to be clear about the goals of the project and the criteria for determining that a prototype implementation is finished.
- The need for developers to have expertise in two very different areas implies that two developers may be better than one for such projects.
- Trying out multiple software tools on a project (for whatever reason) gives extra information about the best tools for final implementation, but has hidden costs in tool learning and computer support.
- Be sure to develop models of the knowledge as well as an implemented system.

References

[1] R. Inder, R. Aylett, D. Bental, T. Lydiard & R. Rae. *Study on the Evaluation of Expert Systems Tools for Ground Segment Infrastructure* (Artificial Intelligence Applications Institute, AIAI-TR-84, 1991).

[2] M. Dale, *Building Knowledge Based Decision Support for Divorce Lawyers on the Internet* (University of Edinburgh, MSc project report, 1999).

[3] L. Edwards and A. Griffiths. *Family Law* (W.Green: Edinburgh, 1997).

[4] S.Duguid, L. Edwards & J. Kingston. A Web-based Decision Support System for Divorce Lawyers. *International Journal of Law, Computers and Technology*, *15*(3), 2001, 265-279.

[5] E. Friedman-Hill, *JESS: the Rule Engine for the Java platform* (Sandia National Laboratories, http://herzberg.ca.sandia.gov/jess/, 2002).

[6] A. Stranieri, A. & J. Zeleznikow, WebShell: A knowledge based shell for the world wide web. *Proc. ISDSS2001- Sixth International Conference on Decision Support Systems*. Brunel University, London, 2001.

[7] A. Stranieri, J. Zeleznikow & J. Yearwood. Argumentation structures that integrate dialectical and monoletical reasoning. *Knowledge Engineering Review*, forthcoming, 2002.