Information & Communication Technology Law,
Vol. 11, No. 1, 2002

Using Web-based Legal Decision Support Systems to Improve Access to Justice

JOHN ZELEZNIKOW

Joseph Bell Centre for Forensic Statistics and Legal Reasoning, Faculty of Law, University of Edinburgh, Old College, South Bridge, Edinburgh EH8 9YL, UK
Telephone: +44 131 650 9704; E-mail: john.zeleznikow@ed.ac.uk

ABSTRACT *There are an increasing number of litigants who are forced to represent themselves in court. This causes havoc in the judicial system and raises issues of access to justice. We believe that important support for unrepresented litigants can be provided by the construction of web-based legal decision support systems. We discuss tools we have constructed for building web-based legal decision support systems and give examples from the domains of Family Law and eligibility for legal aid. We also illustrate how such decision support tools can help litigants negotiate their disputes.*

1. Introduction

The opening session of the Third International Symposium on Judicial Support Systems had as its theme 'What can judicial decision support systems do to improve access to justice?' This article was presented in a much-reduced version as a paper at the symposium with the title 'Legal Aid and Unrepresented Litigants: Building Legal Decision Support Systems for Victoria Legal Aid'.

In this article we discuss the demands that the rise of *pro se* litigation poses for the judicial system and how community legal services can help meet these challenges through the development of web-based decision support systems. In particular, we discuss three systems we have built for Victoria Legal Aid which help improve both the efficiency of Victoria Legal Aid and the quality of its advice.

1.1 The provision of services by community legal centres

Branting (2001) notes that increasing numbers of litigants represent themselves in court. This swelling tide of *pro se*[1] litigants constitutes a growing burden not only on the judiciary but also on the entire legal process. Typically, unrepresented litigants:

(1) Extend the time taken for litigation—due to their lack of understanding of the process.
(2) Place themselves at a disadvantage compared to their opponent(s).
(3) Place the judicial decision-maker in the difficult position of deciding how much support and forbearance the decision-maker should offer to the *pro se* litigant.

ISSN 1360-0834 print/ISSN 1469-8404 online/02/010015–19 © 2002 Taylor & Francis Ltd
DOI: 10.1080/13600830220133530

A recent study conducted for the American Bar Association in the Supreme Court of Maricopa County, Arizona, USA indicated that at least one of the parties were self-represented in over 88% of domestic relations cases and both parties were self-represented in 52% of the cases. Meachem (1999) reports that 24,416 of the 54,693 cases opened in the US Court of Appeals in 1999 were filed by *pro se* appellants. This figure, whilst on first glance may not appear alarming, does need to be considered in light of the fact that many *pro se* appellants have neither the financial resources nor the legal skills to conduct their own appeals.

Quatrevaux (1996) notes that there is a shortfall in legal systems for poor persons in the United States. Branting (2001) claims that domestic abuse victims are particularly likely to have few resources and little opportunity to obtain the services of a lawyer. He states that the growth of the consumer movement has increased the trend for *pro se* litigation. The growing availability of books, document kits and computerised forms—together with the increasing availability of legal materials on the World Wide Web—has increased the opportunities for *pro se* litigants to organise their own litigation.

Black (1990) states that legal aid in the United States is a countrywide system, administered locally, through which legal services are rendered to those in financial need and to those who cannot afford private counsel. The Sixth and Fourteenth Amendments to the US Constitution guarantee the constitutional right of a criminal defendant to a court appointed attorney if he/she is financially unable to retain private counsel. The Legal Services Corporation was established by the Legal Services Corporation Act of 1974 to provide financial support for legal assistance in non-criminal proceedings to persons financially unable to afford legal services. The Corporation makes grants to and contracts with individuals, firms, corporations, organisations, and state and local governments for the express purpose of providing legal services to indigent clients.

The Donald Berman Laboratory for Information Technology at La Trobe University has focused upon building legal decision support systems with a variety of partners including Victoria Legal Aid (VLA) (Zeleznikow & Stranieri, 2001). VLA—based in Victoria, Australia—is a government-funded provider of legal services for disadvantaged clients (www.legalaid.vic.gov.au). Its goals include providing legal aid in the most effective, economic and efficient manner, and pursuing innovative means of providing legal services in the community.[2]

VLA focuses much of its attention upon Criminal and Family Law. VLA handles 80% of all Victorian defended criminal cases. This is a great burden on its resources, since Victorian courts require that defendants, who face the possibility of incarceration and do not have the financial resources to hire legal counsel, receive legal representation. Indeed, the Victorian Supreme Court has indefinitely postponed trials in which a defendant has not been represented.[3]

VLA also handles many Family Law disputes. As is the case for criminal matters, applicants for legal aid must meet certain financial guidelines. In addition, they need to be pleading a case that has *a reasonable chance of success*.[4] It is not uncommon for the wife, husband and children in a Family Law dispute to be supported by VLA. One party will receive legal support from VLA; the others will have private lawyers who are paid by VLA. Many prospective VLA Family Law clients exceed the financial guidelines required for VLA assistance. Since they cannot afford private counsel, they often represent themselves—a very undesirable situation.

1.2 The benefits of decision support systems for the legal community

When considering decision making as a knowledge-manufacturing process, the purpose of a decision support system is to help the user manage knowledge. A decision support system fulfils this purpose by enhancing the user's competence in representing and processing knowledge. It supplements human knowledge management skills with computer-based means for managing knowledge. A decision support system accepts, stores, uses, receives and presents knowledge pertinent to the decisions being made. Its capabilities are defined by the types of knowledge with which it can work, the ways in which it can represent these various types of knowledge, and its capabilities for processing these representations.

Over the past ten years, the Donald Berman Laboratory for Information Technology and Law has built a variety of legal decision support systems covering the domains of:

(a) Worker's compensation (Zeleznikow, 1991),
(b) Credit Law—IKBALS (Zeleznikow *et al.*, 1994),
(c) Family Law Property Division—Split-Up (Stranieri *et al.*, 1999),
(d) Refugee Law—EMBRACE (Yearwood & Stranieri, 1999),
(e) Family Law Negotiation—(Bellucci & Zeleznikow, 2001),
(f) Copyright Law—RightCopy (Stranieri & Zeleznikow, 2001a),
(g) eligibility for legal aid—GetAid (Stranieri & Zeleznikow, 2001b), and
(h) the sentencing of criminals (ongoing).

The development of our legal decision support systems has led to:

(i) Consistency—by replicating the manner in which decisions are made, decision support systems are encouraging the spread of consistency in legal decision making.
(ii) Transparency—by demonstrating how legal decisions are made, legal decision support systems are leading to better community understanding of legal domains. This has the desired benefit of decreasing the level of public criticism of judicial decision making.[5]
(iii) Efficiency—One of the major benefits of decision support systems is to make firms more efficient. For example, since VLA handles a large number of cases (and rejects even more) and, unlike most private legal firms, it does not bill either by the hour or even by each individual case, it is most concerned about being efficient. About 60% of VLA's labour costs are devoted to assessing whether applicants are eligible for legal aid. Clearly, it would be preferable if VLA lawyers spent their time providing clients with legal aid, rather than assessing the eligibility of clients for grants.
(iv) Enhanced support for dispute resolution—Users of legal decision support systems are aware of the likely outcome of litigation and thus are encouraged to avoid the costs and emotional stress of legal proceedings.

While we do not claim that the construction of legal decision support systems will have a drastic effect on improving access to justice, we make the argument that the construction of such systems for community legal centres will improve their efficiency and increase the volume of advice they can offer. Until recently, most legal decision supports systems were rule-based and developed to run on personal computers. While personal-computer-based tools are fine for lawyers, they may not be easily accessible to *pro se* litigants. Reasons for this difficulty include their lack of an awareness of such systems, and the high cost of purchasing relevant software. Currently, very few legal decision support systems are available on the World Wide Web.

The Australasian Legal Information Institute (AustLII—www.austlii.edu.au) provides free Internet access to Australian legal materials. AustLII's broad public policy agenda is to improve access to justice through better access to information. To that end, AustLII has become one of the largest sources of legal materials on the net, with over 7 gigabytes of raw text materials and over 1.5 million searchable documents. AustLII publishes public legal information—that is, primary legal materials (legislation, treaties and decisions of courts and tribunals) and secondary legal materials created by public bodies for purposes of public access (law reform and Royal Commission reports for example). It does not have any decision support systems on its Internet site. The British and Irish Legal Information Institute (BAILII—www.bailii.org) provides access to the most comprehensive set of British and Irish primary legal materials that are available for free and in one place on the Internet. CanLII (www.canlii.org), now a permanent resource in Canadian Law, was initially built as a prototype site in the field of public and free distribution of Canadian primary legal material.

In this article we discuss how the provision of web-based systems enhance access to justice. In particular, we discuss how two systems we have built (Split-Up and GetAid) improve access to justice. The construction of these systems has led us to build two support tools for building legal decision support systems on the World Wide Web. WebShell is a knowledge-based system shell that enables knowledge-based systems to be developed and executed on the World Wide Web. WebShell models knowledge using two distinct techniques: decision trees for procedural type tasks and argument trees for tasks that are more complex, ambiguous or uncertain. We have used Toulmin's (1958) theory of argumentation to structure knowledge in a specific manner and to build ArgumentDeveloper, a knowledge management tool for structuring legal knowledge in the form of Toulmin arguments

2. The provision of decision support services for Victoria Legal Aid

2.1 Rule-based legal decision support systems and the World Wide Web

Most legal decision support systems built for commercial use model fields of law that are complex but not discretionary. For example, SoftLaw (Johnson & Mead, 1991) has developed systems with tens of thousands of rules. Most of these systems have advised upon the determination of pension benefits. The Social Security Act is difficult to master because it is large and complex, yet few decisions based on the Act involve the exercise of discretion.

There seems little doubt that the trend toward rule-based systems to encode large and complex legislation will continue to a substantial extent as claimed by SoftLaw (2000). This is assured by the increasing public demand for more transparency and consistency in government decision making alongside the continuing enactment of increasingly complex legislation. In addition to complex statutes, there is a growing body of regulations that often encode procedural and interpretive rulings. For instance, the Australian Migration Act (1964) is a relatively small Act with a straightforward structure, yet the regulations that govern the interpretation of key terms occupies many volumes.

The majority of knowledge-based systems in commercial use have not been designed to execute on the World Wide Web. There are a number of reasons for this. First of all, few expert systems shells have been developed for web environments. Those that have been developed are typically very expensive and beyond the reach of most user groups. Furthermore, traditional rule-based system architectures are not particularly well suited for web-based shells. For example, the traditional separation of domain knowledge from control knowledge (Shortliffe, 1976) requires that the inference engine scans large segments of the knowledge base in order to find candidate rules to fire. If both inference engine and knowledge bases reside and execute on the server, then the time required for this in a web-based knowledge-based system (in addition to transmission delays from the client to the server and the time required for the resolution of rule conflicts) is prohibitive. Furthermore, the opportunity for potentially any number of simultaneous users to access a web-based knowledge-based system places real constraints on concurrency control mechanisms.

Huntington (2000) states that difficulties with the introduction of web-based expert systems diminish if shells are designed to execute largely on the client's machine as opposed to the server. Java applets are promoted for this. However, the appeal of this approach is diminished because client side shells are difficult to realise in practice. The knowledge base and inference engine components of a knowledge-based system are typically large programs that require substantial resources and time to download. Furthermore, execution on the client side is likely to be limited to users with powerful computers which restricts the universality of the approach.

JESS, the Java Expert System Shell, encodes CLIPS rules (see http://herzberg.ca.sandia.gov/jess/). Grove & Hulse (1999) have described a server side architecture using JESS. This architecture supports client side knowledge-based development. The disadvantages of client side systems include the additional overhead in the transmission of knowledge bases and inference engine programs, security concerns and the restrictions on universality due to the resource demands placed on the client.

Although server side applications such as Jnana (see www.jnana.com) have been developed for web environments they are typically too expensive to enable small-to-medium sized enterprises to use them. WebShell (Stranieri & Zeleznikow, 2001c) is a server side web-based shell that is both small and simple. The separation of domain knowledge from control knowledge that was the hallmark of traditional expert systems has been relaxed. This allows the shell to be compact, fast and inexpensive to build. Knowledge is stored in a relational database and modeled using two representations: a variant of a standard decision tree and an argument tree. This facilitates ongoing maintenance.

The decision tree representation is particularly well suited to parts of domains that are procedural. This captures knowledge where decision-makers have little discretion or room for interpretation. The argument tree representation involves knowledge where an expert has more discretion. A knowledge-based system for a rule-based, procedural domain can be built in WebShell using only decision trees. On the other hand, a knowledge-based system for a discretionary domain can be built using only argument trees. However, the two representations

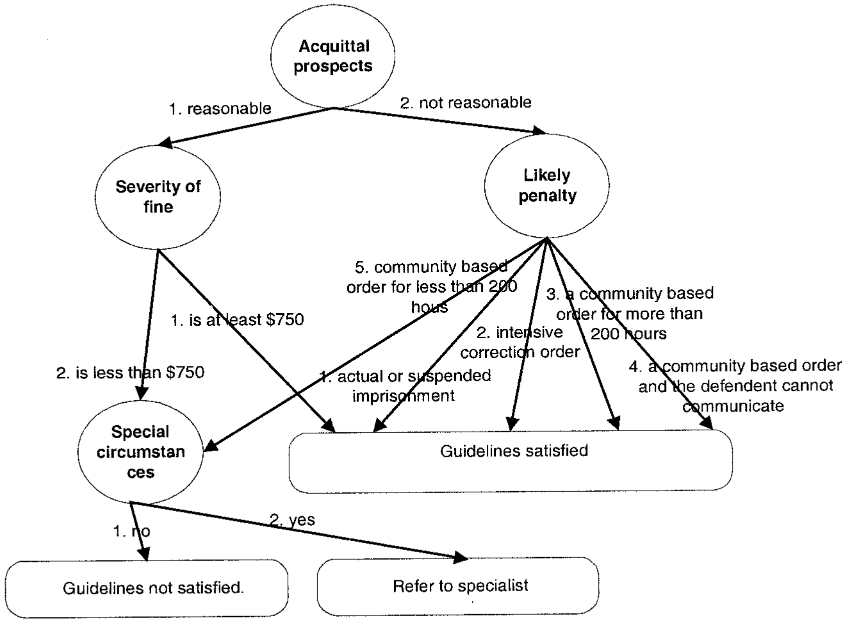

Figure 1. Decision tree for eligibility for legal aid.

are tightly integrated for domains that involve a mix of procedural and non-procedural knowledge.

2.2 GetAid—A web-based decision support system for determining eligibility for legal aid

When an applicant approaches VLA, his/her application is assessed to determine whether he/she should receive legal aid. This task chews up 60% of VLA's operating budget, yet provides no services to its clients. After passing a financial test, applicants for legal aid must pass a merits test. The merits test involves a prediction about the likely outcome of the case if it were to be decided by a Court. VLA grant officers, who have extensive experience in the practices of Victorian courts, assess the merits test. This assessment involves the integration of procedural knowledge found in regulatory guidelines with expert lawyer knowledge that involves a considerable degree of discretion.

A decision tree is a directed graph in which the nodes represent domain concepts and possible values for each concept are captured in arcs emerging from each node. Leaf nodes represent conclusions. Figure 1 depicts a decision tree that represents reasoning used by VLA lawyers to determine whether an applicant for legal aid, who is scheduled to appear in a minor (Magistrates) court, has met statutory guidelines.

Stranieri & Zeleznikow (2001c) introduced a variation on the conventional decision tree. They labeled nodes and arcs in a pre-specified manner which allowed for their conversion into sets called 'sequenced transition networks' (STN). The sets are stored as tuples in a relational database. The decision trees used for representing procedural knowledge in WebShell differ cosmetically from the conventional decision tree depicted in Figure 1. The variation, illus-

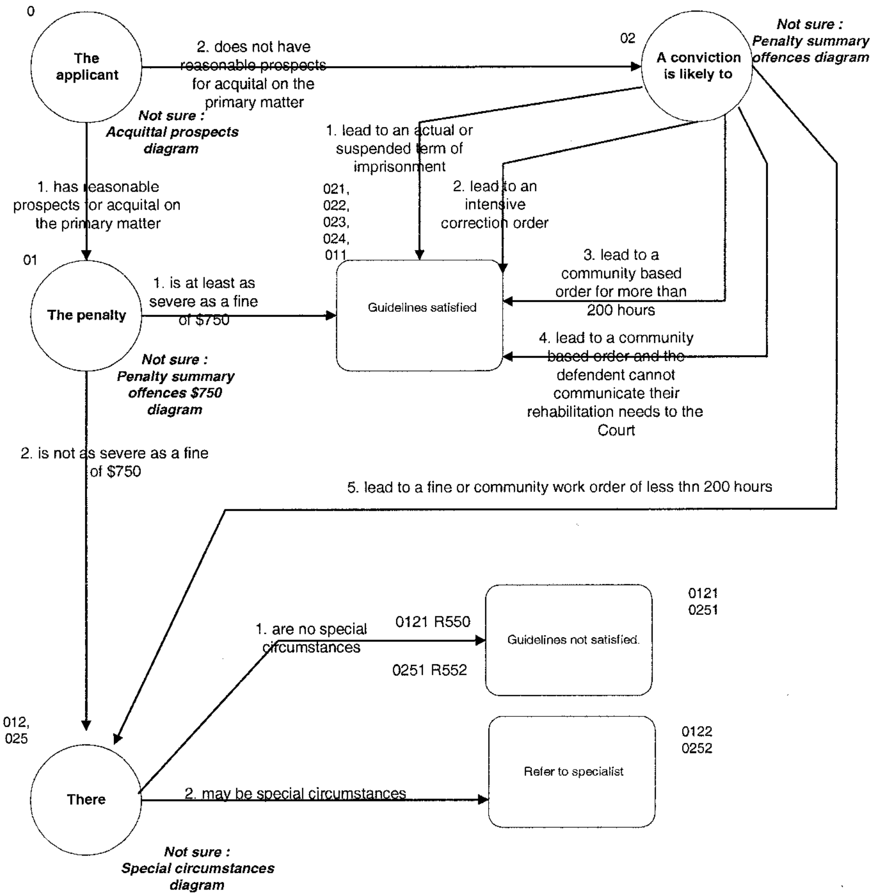

Figure 2. STN for eligibility for legal aid.

trated in Figure 2, enables user prompts and explanations to be more easily generated directly from information in the tree and facilitates the maintenance of the system.

When using a conventional expert system, the decision tree is converted into rules. During a consultation, the expert system shell's inference engine generates user prompts from the rules and, based on user responses, fires rules until a conclusion is reached (see Figure 3). The approach of Stranieri & Zeleznikow (2001c) differs in that the decision tree is not converted into rules. Instead, the information in the tree is entered into a relational database in a way that facilitates the retrieval of a sequence of user prompts leading to a conclusion.

The STN-based approach can facilitate the development of knowledge-based systems for knowledge that can readily be captured as a decision tree. However, not all knowledge is procedural. For example, the node in Figure 2 relates to the prospects for acquittal of an applicant for legal aid. Determining the possibility of a successful defence to a charge requires legal expertise. Such knowledge cannot readily be modeled using decision trees.

Toulmin (1958) concluded that all arguments, regardless of the domain, have a structure that consists of six basic invariants: claim, data, modality, rebuttal, warrant and backing. Every argument makes an assertion based on some data.

Figure 3. Screen of the first user prompt generated by the inference engine.

The assertion of an argument stands as the claim of the argument. Knowing the data and the claim does not necessarily convince us that the claim follows from the data. A mechanism is required to act as a justification for the claim. This justification is known as the 'warrant'. The backing supports the warrant and is typically a reference to a statute or a precedent case in a legal argument.

The Toulmin Argument Structure (TAS) has often been adapted to structure knowledge, yet most studies that apply the Toulmin structure do not use the original structure but vary one or more components. The variation employed in WebShell draws a distinction between a generic argument and an actual argument and is described fully in Stranieri *et al.* (2002).

Important variations pertinent to the use of the argument structure in WebShell are:

- *Variable-value representation for data and claim values.* We adopt a variable/value representation for claim and data items in order to structure linguistic variables more tightly than suggested by Toulmin and enable the user prompts to be generated more easily.
- *Inference mechanism.* We replace the Toulmin warrant component with inference mechanisms and a reason for relevance.

Since experts could not readily represent knowledge about an applicant's prospects for acquittal as a decision tree, we decided to model the process as a tree of Toulmin arguments. The first of these is illustrated in Figure 4, where only claim variables/values and data variable/values are included. During knowledge acquisition, the expert is prompted to articulate factors (data items) that may be relevant in determining a prospect for an acquittal claim, without any consideration as to how the factors may combine to actually infer a claim value. For every factor presented, a reason for the item's relevance must be given. The next step in the knowledge acquisition exercise using the generic argument is to expand each data item. For example, the expert is asked to describe relevant factors for determining the strength of the Crown case.

Once the tree is developed as far back as the expert regards appropriate for the task at hand, attention is then focused on identifying one or more inference mechanisms that may be used to infer a claim value from data item values. It

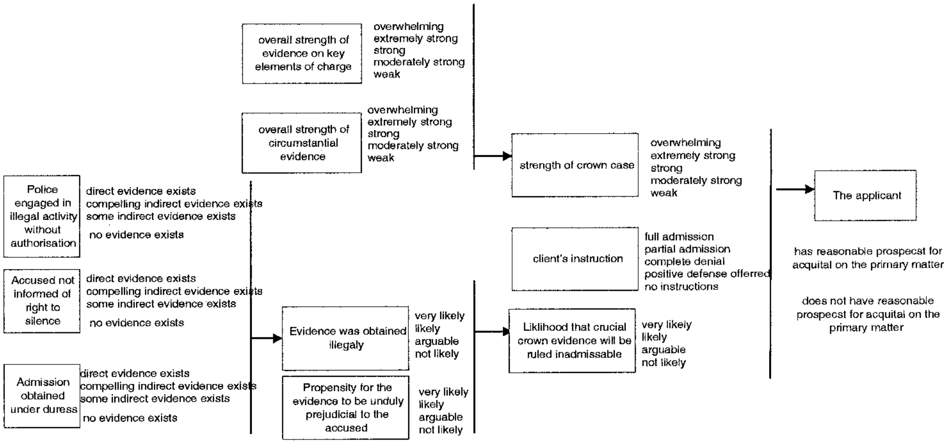

Figure 4. Argument tree for acquittal prospects.

was difficult for the principal domain expert to articulate the ultimate argument (the argument shown on the extreme right of Figure 4). She could not express her heuristics as rules because the way in which the factors combine is rarely made explicit. Her expertise was primarily a result of the experience she had gained in the domain. Although it is feasible to attempt to derive heuristics, the approach we used was to present a panel of experts with an exhaustive list of all combinations of data items as hypothetical cases and prompt the panel for a decision on acquittal prospects. Six experts and the knowledge engineer were able to record their decision in all of the exhaustive hypothetical cases (for that argument) in approximately 40 minutes. The decisions from each rater were merged to form a data set of 600 records that were used to train neural networks.

The inference mechanism in WebShell consists of two components: a lookup table for exceptions and a weighted sum formula. Once the user has supplied values for data items, the WebShell inference engine attempts to look up a claim value in the lookup table of exceptions. This table stores values that are exceptions to the weighted sum formula that are detected during the evaluation phase of knowledge-based system development. If no entry is found in the lookup table, the inference engine applies a weighted sum formula according to weights associated with each data item. Using a lookup table to store the mapping between data values and claim values also enables the use of inference methods other than neural networks.

Neural network inferences can be implemented by storing all possible data item inputs and corresponding claim value outputs in the lookup table. A real time, web-based implementation cannot rebuild a neural network for each inference without causing consultation delays, so storing all inputs/outputs as a lookup table enables fast inferences even when the source was a neural network.

A user consults the GetAid system via the web pages that are generated from the decision tree described above. Suppose a user follows the 'Not-Sure' link on the web page depicted in Figure 4. She is taken to a page that presents three user prompts that derive directly from the argument depicted in Figure 4: 'strength

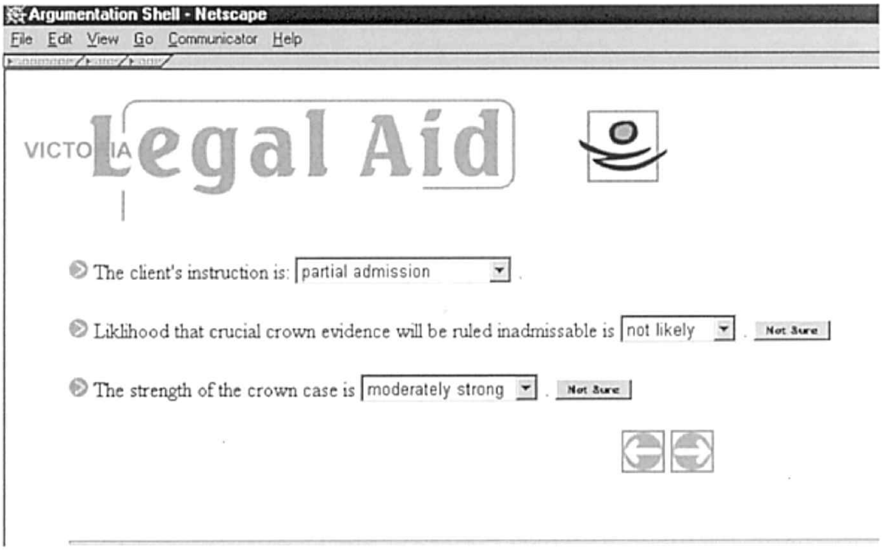

Figure 5. WebShell based on argument tree.

of the Crown case', 'client's instruction' and 'likelihood that Crown evidence is ruled inadmissible'. This page is illustrated in Figure 5. The user is presented with a consistent user interface throughout and is generally unaware that some pages are generated from the argument tree and others from the decision tree (see Figures 5 and 6).

The PHP program that implements the argument-based inferences is somewhat more complex than the STN but it is still a small and relatively simple program that executes on the server side very quickly and is not memory intensive. The GetAid system is currently being tested by VLA experts and is being developed in conjunction with web-based lodgement of applications for legal aid (Hall *et al.*, 2002).

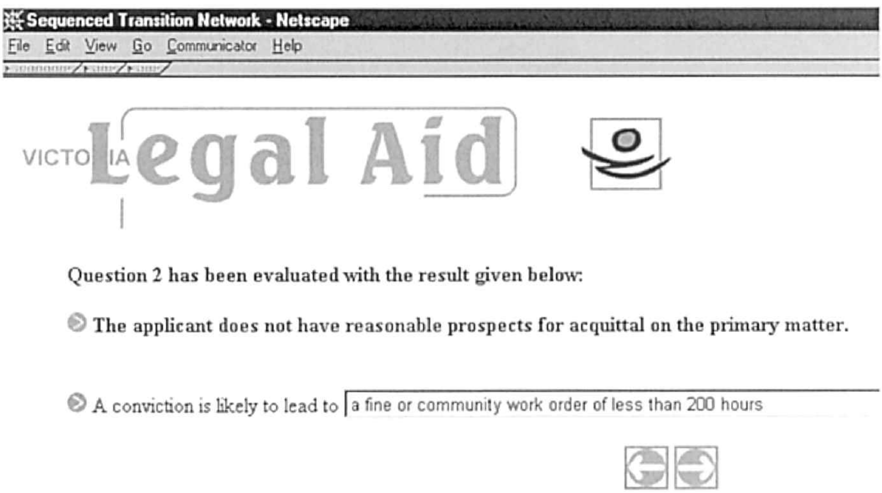

Figure 6. WebShell based on argument tree.

We are also using WebShell to represent legal knowledge about:

(a) Family Law Property Division in Australia,
(b) Family Law Property Division in Scotland,
(c) computer copyright, and
(d) sentencing of criminals in Victoria, Australia.

The impact of the availability of GetAid on clients, staff and VLA as a whole is drastic and is currently being evaluated (see Hall *et al.*, 2002). For example, we need to consider the possibility of job transfers and losses at VLA and whether VLA clients will accept using web-based decision support systems for decision making. Zeleznikow (2000) has noted that divorcees with little knowledge of Australian Family Law have often been surprised at predictions provided by the Split-Up system. Ultimately, it is not wise for systems such as Split-Up to be utilised by users with little Family Law knowledge since such users cannot identify unusual (or hard) cases. A further major problem is that naive users of decision support systems are not able to make accurate decisions about the facts of a case. Determining facts will always require a human decision. For this reason, we expect lawyers rather than litigants to be the main users of systems such as GetAid.

2.3 Placing Split-Up on the World Wide Web

Split-Up provides advice on property distribution following divorce (Zeleznikow & Stranieri, 1998). In developing Split-Up, we used Family Law experts to identify factors pertinent to a property distribution following divorce. We then assembled a data set of past cases that were fed to machine-learning programs. In this manner, Split-Up learned the way in which judges weighed factors in past cases, without resorting to developing rules.

In the Split-Up system, the relevant variables were structured as data and claim items in 35 separate arguments. The claim items of some arguments were the data items of others, resulting in a tree that culminated in the ultimate claim that indicated the percentage split of assets a judge would likely to award the husband. The tree of variables is illustrated in Figure 7. In 15 of the 35 arguments, claim values were inferred from data items with the use of heuristics, whereas neural networks were used to infer claim values in the remaining 20 arguments. The neural networks were trained with data from only 103 cases. This was feasible because each argument involved a small number of data items due to the decomposition described in Figure 7.

In consultation with domain experts, 94 variables were identified as relevant for the determination of a percentage split of the common pool. The way the factors combine was not elicited from experts as rules or complex formulas. Rather, values on the 94 variables were extracted from cases previously decided, so that a neural network could learn to mimic the way in which judges had combined variables. The relevant variables were structured as separate arguments following the argument structure advanced by Toulmin (1958).

Figure 8 illustrates one argument from the Split-Up. We see from the Figure that there are three data items (‘contributions’, ‘level of wealth’, ‘needs’) which lead to the determination of what percentage of marital property the husband receives. In the argument in Figure 8, the inference mechanism is a neural network. The network, once trained with appropriate past cases, outputs a claim

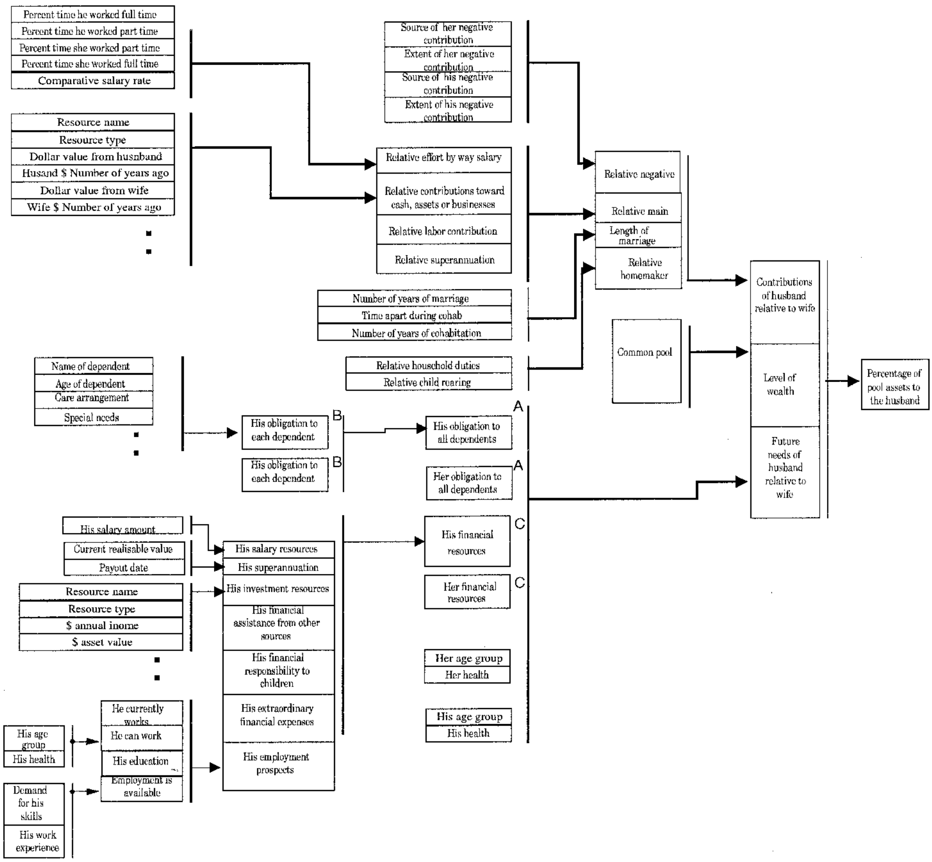

Figure 7. Tree of actual arguments for split up.

value ('percentage split of assets to husband') given values of the three data items.

The Split-Up system produces an inference by the invocation of inference mechanisms stored in each argument. However, an explanation for an inference is generated after the event, in legal realist traditions, by first invoking the data

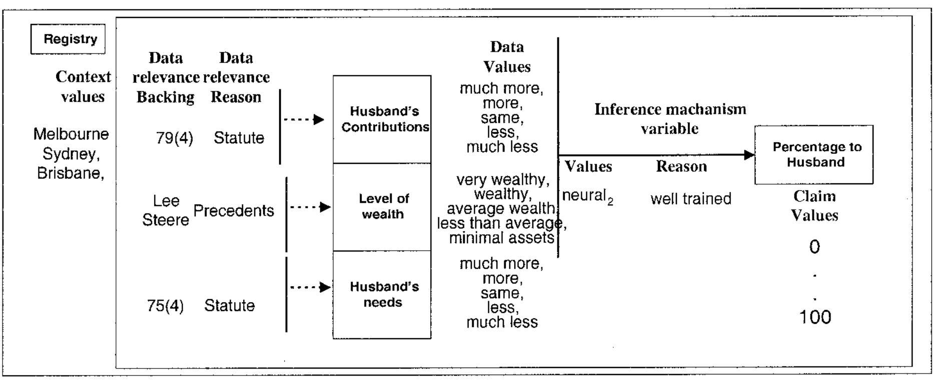

Figure 8. Generic argument for percentage split of assets to the husband.

items that led to the claim. Additional explanatory text is supplied through the provision of reasons for relevance and a backing. If the user questions either data item value, he/she is taken to the argument that generated that value as its claim.

The Split-Up system performed favourably on evaluation, despite the small number of samples (Stranieri & Zeleznikow, 1999). Currently, the tree of arguments is being modified in conjunction with domain experts from Victoria Legal Aid to accommodate recent changes in legislation and practice—in particular:

(a) The recent tendency by Family Court judges to view domestic violence as a negative financial contribution to a marriage.
(b) The re-introduction of spousal maintenance as a benefit to one of the partners. Under the 'clean-break philosophy', Family Court judges were reluctant to award spousal maintenance, since it would mean one partner would continue to be financially dependant on his/her ex-partner. However, the increasing number of short, asset-poor, income-rich marriages has led to a re-consideration of the issue of spousal maintenance.
(c) The need to consider superannuation and pensions separately from other marital property.

With the provision of domain expertise and financial support from VLA, we are currently developing a web-based version of Split-Up using the web-based shell WebShell and the knowledge management tool ArgumentDeveloper. As a web-based system, Split-Up will inform divorcees of their rights and support them to commence negotiations pertaining to their divorce.

3. Decision support systems that provide advice upon alternative dispute resolution

3.1 Alternative dispute resolution and online dispute resolution

Katsh & Rifkin (2001) state that, compared to litigation, alternative dispute resolution has the following advantages:

(a) lower cost,
(b) greater speed,
(c) shows more flexibility in outcomes,
(d) is less adversarial,
(e) is more informal,
(f) is solution- rather than blame-oriented, and
(g) is private.

Many of the above factors offer major benefits to *pro se* litigants. Bellucci & Zeleznikow (2001) have been developing computer software to support alternative dispute resolution. While we have developed our systems in the domain of Australian Family Law, we argue that the techniques developed are generic and can be utilised in many negotiation domains.

Chung & Pak (1997) stress that, although dispute resolution is a human problem, computers are already at the bargaining table to transform the negotiation process. The Harvard Negotiation Project of Fisher & Ury (1981) claimed that the few generally accepted principles of successful negotiation require world knowledge beyond the scope of current artificial intelligence methods. Principled negotiation advocates separating the problem from the people. It promotes a focus on the interests of the party rather than allowing negotiation to deteriorate into a contest of 'who will back down first' as occurs in positional negotiation.

In developing principled negotiation, Fisher and Ury introduced the notion of 'know your best alternative to a negotiated agreement' (BATNA)—that is, the reason you negotiate with someone is to produce better results than would otherwise occur. If you are unaware of what results you could obtain if the negotiations are unsuccessful, you run the risk of entering into an agreement that you would be better off rejecting *or* rejecting an agreement you would be better off accepting.

In most legal conflicts, once a settlement is reached the parties to the agreement are not required to have an ongoing relationship. This is not the case in Australian Family Law. Family Law (Ingleby 1993) varies from other legal domains in that in general:

1. There are no winners or losers—in most common law domains one party to a legal dispute wins a case whilst the other loses. In civil matters, under the cost indemnity rule, the loser of a litigated case pays the costs of the winner. Save for exceptional circumstances, both parents following a divorce receive a portion of the property and have defined access to any children.
2. Parties to a Family Law case often need to communicate after the litigation has concluded. Hence, the Family Court encourages negotiation rather than litigation.

Split-Up can be used to determine one's BATNA. It first shows both litigants what they would be expected to be awarded by a court if their relative claims were accepted. It gives them relevant advice as to what would happen if some or all of their claims were rejected. Users are able to have dialogues with the Split-Up system about hypothetical situations and learn about the strengths and weakness of their claims. Suppose the disputants' goals are entered into the system to determine the asset distributions for both W and H in a hypothetical example. For the example taken from Bellucci & Zeleznikow (2001), the Split-Up system provided the following answers as to the percentages of the marital assets received by each party:

	W's (%)	H's (%)
Given one accepts W's beliefs	65	35
Given one accepts H's beliefs	42	58
Given one accepts H's beliefs but gives W custody of the children	60	40

Clearly custody of the children is very significant in determining the husband's property distribution. If he were unlikely to win custody of the children, the husband would be well advised to accept 40% of the common pool (otherwise he would also risk paying large legal fees and having ongoing conflict).

Online dispute resolution is alternative dispute resolution conducted over the

Internet (Katsh & Rifkin, 2001). The attractions of online dispute resolution include:

(1) When one is online, one can engage in activities at a distance that previously required physical presence.
(2) When one is online, one can do things quickly.
(3) When one is online, one can perform information-processing tasks beyond the capability of humans.

In the next section, we discuss our research on building decision support systems to support alternative dispute resolution. We hope to eventually place such systems on the World Wide Web.

3.2 Negotiation support systems in Australian Family Law

Mediators often encourage disputants to resolve their conflict through the use of compromises and trade-offs. Once the trade-offs have been identified, other decision-making mechanisms must be employed to resolve the dispute (Wellman, 1990).

Game theoretic techniques and decision theory were the basis for AdjustedWinner (Bellucci & Zeleznikow, 1998), which implemented the procedure of Brams & Taylor (1996). AdjustedWinner is a point-allocation procedure that distributes items or issues to people on the premise of whoever values the item or issue more. The two players are required to explicitly indicate how much they value each of the different issues by distributing 100 points across the range of issues in dispute. The AdjustedWinner paradigm is a fair and equitable procedure. At the end of allocation of assets, each party accrues the same number of points.

An influence diagram (Howard & Matheson, 1981) gives us a simple visual representation of a decision problem. It provides a technique for identifying and displaying the essential elements of a decision—namely the values of issues and how they influence each other. Influence diagrams use nodes to represent uncertain quantities or random variables. Lines joining the nodes represent conditional dependence/independence between the nodes. In Family_Winner (Bellucci & Zeleznikow, 2001), influence diagrams (trade-off maps) have been used to model the thought processes of the parties. In a negotiation, two influence diagrams are drawn up to represent the trade-offs disputants bring into the dispute. They show the effect of issues on each other (their values are entered by parties at the beginning of the negotiation) and track changes caused by the allocation or loss of issues. These changes cause the application of trade-offs to clearly represent a party's current standing in the negotiation.

Holsapple & Whinston (1996) claim that there are six common techniques for decision making. One of these is the Analytic Hierarchy Process (AHP). AHP decomposes the overall decision objective into a hierarchic structure of criteria, sub-criteria and alternatives. Using the AHP, a decision-maker then compares each pair of criteria by answering the question, with respect to meeting the overall decision objective, as to which of the two criteria is most important. The result is a matrix of pairwise decisions.

The decision-maker proceeds to the first level of sub-criteria beneath the level of main criteria and repeats the process. He/she compares each pair of sub-criteria by deciding, with respect to satisfying a certain criterion in the prior level, which of these two sub-criteria is more important and by how much. Then the decision-maker uses the numeric scale discussed to represent these judgements. The result is a matrix of pairwise sub-criteria comparisons for each criterion in the prior level. The decision-maker repeats this process for succeeding levels of sub-criteria until they reach the lowest level of the hierarchy, which consists of specific comparison of the alternative with respect to sub-criteria in the prior level. For each of these sub-criteria a matrix of pairwise comparisons results from answering the question: 'With respect to a certain sub-criterion, which alternative is preferred and by how much?' For each matrix of pairwise comparisons, the AHP mathematically derives a ratio scale of relative magnitudes, which are expressed in terms of priority units.

While AHP is used to formulate alternatives, Family_Winner uses its concept of hierarchical decomposition to determine the order by which allocation is to occur, and to enable the formation of sub-issues or items. The disputants must indicate the issues in dispute, decompose the issues into sub-issues until their positions are reflected in the sub-issues. Each issue is broken down so that allocation values are in binary form: each issue is allocated to either H or W. It uses a theory of pairwise comparisons to determine who is allocated the item or issue. Upon reaching the lowest level in the hierarchy (as specified by the disputants), the system mathematically calculates the value of each sub-issue or item with respect to the relative super-issue or items. It does so for each party. Once complete, the system calculates which party is allocated particular sub-issues or items through pairwise comparisons over the derived values from both parties.

The algorithm supports the process of negotiation by introducing importance values to indicate the degree to which each party desires to be awarded the issue being considered. The system uses this information to form trade-off rules. The trade-off rules are used to allocate issues according to the logrolling strategy. The system makes this analysis by transforming user input into trade-off values, used directly on trade-off maps, which show the effect of an issue's allocation on all unallocated issues.

Users of the Family_Winner system enter information (such as the issues disputed, indications of their importance to the respective parties and how the issues relate to each other) with respect to whether issues should be resolved independently or within the constraints of mutual exclusiveness. An analysis of the aforementioned information is compiled, which is then translated into graphical trade-off maps. The maps illustrate the relevant issues, their importance to each party and trade-off capabilities of each issue. The system takes into account the dynamics of negotiation by representing the relations that exist between issues. Maps are developed by the system to show a negotiator's preferences and relation strengths between issues. It is from these maps that trade-offs and compromises can be enacted, resulting in changes to the initial values placed on issues.

The user is asked if the issues can be resolved in its current form. If so, the system then proceeds to allocate the issue as desired by the parties. Otherwise, the user is asked to decompose an issue chosen by the system as the least contentious. Essentially the issue on which there is the least disagreement (one party requires it greatly whilst the other party expresses little interest in the issue) is chosen to be the issue first considered. Users are asked to enter sub-issues. As issues are decomposed, they are stored in a decomposition hierarchy, with all links intact. This structure has been put in place to recognise there may be sub-issues within issues on which agreement can be attained. For example, a 50/50 split of all property may not be agreed upon. However, after following a process that splits property so that the H keeps the house and the W keeps the apartment, it may then be possible to reach agreement. It is important to note that the greater the number of issues in dispute, the easier it may be to allocate issues, as the possibility of trade-offs increases. This may seem counter intuitive, but if only one issue needs to be resolved, then suggesting trade-offs is not possible.

This process of decomposition continues through the one issue, until the users decide the current level is the lowest decomposition possible. At this point, the system calculates which issue to allocate to which party, then removes this issue from the parties respective trade-off maps, and makes appropriate numerical adjustments to remaining issues linked to the issue just allocated. The resulting trade-off maps are displayed to the users, so they can see what trade-offs are made in the allocation of issues. When all issues are allocated at the one level, then decomposition of issues continues, re-commencing from the top level in a sequential manner.

The algorithms implemented in the system support the process of negotiation by introducing importance values to indicate the degree to which each party desires to be awarded each issue. We assume the importance value of an issue is directly related to how much the disputant wants the issue to be awarded to him/her. The system uses this information to form trade-off rules

While we have developed this algorithm in the domain of Australian Family Law, the techniques used do not inhibit its use in other negotiation domains. The concept of a decomposition hierarchy to store issues and sub-issues and the use of trade-off maps, for example, can certainly be applied to a number of disputes for successful resolution.

The system is being subjected to rigorous evaluation by Family Law mediators. Using Family_Winner, mediators can learn possible trade-offs. It will provide them with an understanding of the negotiation process and likely outcomes. The availability of such negotiation support systems, which provide unbiased support for prospective litigants, will hopefully lead to a diminution in conflict. The savings in both financial and emotional expenditure will be significant.

Litigation can be damaging to both parties in a dispute. It is a zero-sum game, in that what one party wins the other loses.[6] Mediation can strive to reduce hostility between the parties, to fashion an agreement about tasks each party is willing to assume and to reach agreement on methods for ensuring certain tasks have been carried out. It can lead to a win-win result.[7]

4. Conclusion

The growing rise in the number of litigants who represent themselves in court has undesirable consequences for the administration of justice. Our approach to help alleviate this problem is to develop web-based legal decision support systems. We have illustrated techniques for building web-based legal decision support systems, especially with regard to the granting of legal aid and the distribution of marital property following divorce.

Our work on developing web-based legal decision support systems has led to the construction of two tools for building such systems: GetAid—an expert system shell for building web-based knowledge-based decision support systems and ArgumentDeveloper—a tool for managing knowledge in the form of Toulmin arguments. We have also illustrated techniques for supporting computer-based alternative dispute resolution. We are currently extending such techniques to online dispute resolution.

While our research has been primarily devoted to developing legal decision support systems, we are of the view that the systems developed can help deliver improved access to justice. But how do we demonstrate our contention? Current research (Hall *et al.*, 2002) is focusing upon evaluating legal decision support systems. Such research is necessary to demonstrate the benefits of building a diverse range of systems in Family Law, eligibility for legal aid and alternative dispute resolution. These techniques are also being used to evaluate systems in Copyright Law, Refugee Law and sentencing—systems we have built but not described in this article. In future articles we will also conduct an in-depth comparison of our legal decision support systems—especially in the domain of Family Law.

Notes

1. A *pro se* litigant is one who does not retain a lawyer and appears for himself in court.
2. As set out in the Legal Aid Act 1978.
3. In *Graham vs Victoria Legal Aid* (2001) VSC 90 (3 April 2001) Supreme Court of Victoria, Criminal Division No. 1496 of 2000, Teague J made an order that VLA provide legal assistance to Mr Graham who was held in custody on a charge of murder. *DPP (Cth) vs His Honour Judge Wodak & Ors* (1998) VSC 15 (13 August 1998) considered the case of Mr Philip Chee Ming Ng. Mr Ng was arrested in Melbourne and charged with a number of drug offences pursuant to both Commonwealth and State Regulations relating to the importation of heroin into Australia. They decided Mr Ng should receive legal aid.
4. Something which requires legal expertise to ascertain.
5. Judges of the Family Court of Australia are worried about criticism of the court, which has led to the death of judges and physical attacks on courtrooms. They believe enhanced community understanding of the decision-making process in Australian Family Law will lead to reduced conflict.
6. It is actually worse than a zero-sum game and indeed can often lead to a lose-lose result. This is because of the large legal fees arising from litigation.
7. For example, if both parties value the list of items in dispute, it is not uncommon (as long as they do not value the items in an identical manner) for each party to receive 70% of their requested points.

References

Bellucci, E. & Zeleznikow, J. (1998) A comparative study of negotiation decision support systems. *Proceedings of the 31st Hawaii International Conference on System Sciences* (Los Alamitos, CA, IEEE Computer Society), pp. 254–262.

Bellucci, E. & Zeleznikow, J. (2001) Representations for decision making support in negotiation. *Journal of Decision Support* 10(3–4).

Black, H.C. (1990) *Black's Law Dictionary* (St Paul, MN, West Publishing Company).

Brams, S.J. & Taylor, A.D. (1996) *Fair Division: From Cake Cutting to Dispute Resolution* (Cambridge, Cambridge University Press).

Branting, L.K. (2001) Advisory systems for *pro se* litigants. *Proceedings of the 8th International Conference on Artificial Intelligence and Law* (St Louis, MO, ACM Press), pp. 139–146.

Chung, W.W.C. & Pak, J.J.F. (1997) Implementing negotiation support systems: Theory and practice. *Proceedings of the 30th Annual Hawaii International Conference on Systems Science—Mini Track on Negotiation Support Systems* (IEEE Press).

Fisher, R. & Ury, W. (1981) *Getting to YES: Negotiating Agreement Without Giving In* (Boston, MA, Houghton Mifflin).
Grove, R.F. & Hulse, A.C. (1999) An Internet-based expert system for reptile identification. *Proceedings of the 1st International Conference on the Practical Application of Java (PAJava99)* (Practical Application Co Ltd), pp. 165–173.
Hall, M.J.J., Stranieri, A. & Zeleznikow, J. (2002) A strategy for evaluating web-based decision support systems. Submitted to *ECIS2002— the 10th European Conference on Information Systems*, Gdansk, Poland, June 6–8.
Holsapple, C.W. & Whinston, A.B. (1996) *Decision Support Systems—A Knowledge-based Approach* (St Paul, MN, West Publishing Company).
Howard, R.A. & Matheson, J.E. (1981) Influence diagrams. In R.A. Howard & J.E. Matheson (Eds) *Readings on the Principles and Applications of Decision Analysis*, Vol 2 (Strategic Decision Group).
Huntington, D. (2000) Web-based expert systems are on the way: Java-based web delivery. *PCAI Intelligent Solutions for Desktop Computers* 14(6): pp. 34–36.
Ingleby, R. (1993) *Family Law and Society* (Sydney, Butterworths).
Johnson, P. & Mead, D. (1991) Legislative knowledge-based systems for public administration—Some practical issues. *Proceedings of the 3rd International Conference on Artificial Intelligence and Law* (Oxford, ACM Press), pp. 108–117.
Katsh, E. & Rifkin, J. (2001) *Online Dispute Resolution: Resolving Conflicts in Cyberspace* (San Francisco, CA, Jossey-Bass).
Meachem, L. (1999) *Judicial Business of the United States Courts* (technical report) (Washington, Administrative Office of the United States Courts).
Quatrevaux, E. (1996) *Increasing Legal Services Delivery Capacity Through Information Technology* (Technical Report No. LSC/OIG-95–035) (Legal Services Corporation). (Available online at http://:oig.lsc.gov/tech/techdown.htm)
Shortliffe, E.H. (1976) *Computer-based Medical Consultations: MYCIN* (New York, Elsevier). Softlaw (2000) www.softlaw.com.au (accessed 1 June 2001).
Stranieri, A. & Zeleznikow, J. (1999) Evaluating legal expert systems. *Proceedings of the 7th International Conference on Artificial Intelligence and Law* (Oslo, ACM Press), pp. 18–24.
Stranieri, A. & Zeleznikow, J. (2001a) Copyright regulation with argumentation agents. *Information and Communications Technology Law* 10(1): pp. 109–123.
Stranieri, A. & Zeleznikow, J. (2001b) WebShell: A knowledge-based shell for the World Wide Web. *Proceedings of ISDSS2001—The 6th International Conference on Decision Support Systems* (London, Brunel University), pp. 218–226.
Stranieri, A. & Zeleznikow, J. (2001c) WebShell: The development of web-based expert systems. *Research and Development in Expert Systems XVIII. Proceedings of ES2001—The 21st SGES International Conference on Knowledge-based Systems and Applied Artificial Intelligence* (London, Springer Verlag), pp. 245–258.
Stranieri, A., Zeleznikow, J., Gawler, M. & Lewis, B. (1999) A hybrid-neural approach to the automation of legal reasoning in the discretionary domain of Family Law in Australia. *Artificial Intelligence and Law* 7(2–3): pp. 153–183.
Stranieri, A., Zeleznikow, J. & Yearwood, J. (2002) Argumentation structures that integrate dialectical and monoletical reasoning. *Knowledge Engineering Review* (forthcoming).
Toulmin, S. (1958) *The Uses of Argument* (Cambridge, Cambridge University Press).
Wellman, M.P. (1990) *Formulation of Tradeoffs in Planning under Uncertainty* (London, Pitman Publishing).
Yearwood, J. & Stranieri, A. (1999) The integration of retrieval, reasoning and drafting for Refugee Law: A third generation legal knowledge-based system. *Proceedings of the 7th International Conference on Artificial Intelligence and Law* (Melbourne, Australia, ACM Press), pp. 117–137.
Zeleznikow, J. (1991) Building intelligent legal tools—The IKBALS project. *Journal of Law and Information Science* 2(2): pp. 165–184.
Zeleznikow, J. (2000) Building judicial support systems in discretionary legal domains. *International Review of Law, Computers and Technology* 14(3): pp. 341–35.
Zeleznikow, J. & Stranieri, A. (1998) Split-Up: An intelligent decision support system which provides advice upon property division following divorce. *Journal of Law and Information Technology* 6(2): pp. 190–213.
Zeleznikow, J. & Stranieri, A. (2001) The use of legal decision support systems at Victoria Legal Aid. *Proceedings of ISDSS2001—The 6th International Conference on Decision Support Systems* (London, Brunel University), pp. 186–192.
Zeleznikow, J., Vossos, G. & Hunter, D. (1994) The IKBALS project: Multimodal reasoning in legal knowledge-based systems. *Artificial Intelligence and Law* 2(3): pp. 169–203.