A Multi-Agent Architecture for Online Dispute Resolution Services

Brooke ABRAHAMS,[1] John ZELEZNIKOW
School of Information Systems, Victoria University

Abstract: Argumentation theory is often used in multi agent-systems to facilitate autonomous agent reasoning and multi-agent interaction. The technology can also be used to develop online negotiation and mediation services by providing argument structures that assist parties involved in a dispute to resolve outstanding issues or avoid future disputes. While Alternative Dispute Resolution (ADR) represents a move from a fixed and formal process to a more flexible one, Online Dispute Resolution (ODR) moves ADR from a physical to a virtual place. The research aims to capitalise on the recent trend towards ODR by creating a JADE based multi-agent ODR environment. The utility functions and argument structures of two existing ODR applications are being re-deployed as Web based intelligent agents capable of intuitively coordinating during a negotiation. One agent uses expert knowledge of the Australian Family Law domain to recommend a percentage property split, while another uses heuristics and game theory and combines this split with a significance rating of items provided by each party, to allocate issues and advise upon possible trade-offs. The ultimate aim is to provide disputants with an integrated ODR environment offering a range of services to assist them in achieving fairer outcomes.

Keywords: Alternative dispute resolution, Bayesian reasoning, Argumentation theory, JADE, Multi-agent systems, Online Dispute Resolution.

1. Introduction

Recently, argumentation theory has become an increasingly popular method of specifying autonomous agent reasoning and facilitating multi-agent interaction. The theory can be used by agents, for example, for belief revision and decision-making under uncertainty and non-standard preference policies, and provides tools for designing, implementing and analysing sophisticated forms of interaction among rational agents as described by [1]. The technology can also facilitate online negotiation and mediation services by providing argument structures that assist parties involved in a dispute to resolve outstanding issues or avoid future disputes.

The Laboratory of Decision Support and Dispute Management at Victoria University in Melbourne Australia, has successfully developed decision support

[1] Corresponding author: Brooke Abrahams, School of Information Systems, Victoria University. E-mail: brooke.abrahams@vu.edu.au.

systems in Australian Family Law. The project team has further used domain expertise to construct a variety of Family Law negotiation support systems.

The Split-Up project [2] used Toulmin's theory of argumentation [3] to model how Australian Family Court judges exercise discretion in distributing marital property following divorce. The prototype used machine learning to model how judges perform a percentage distribution of assets. Whilst the Split-Up system was not originally designed to support legal negotiation, it is capable of doing so. Split-Up can be directly used to proffer advice in determining a 'Best Alternative to a Negotiated Agreement' (BATNA). This point is illustrated by [4].

Family Winner [4] is an application that uses a variety of artificial intelligence and game theoretic techniques to advise upon structuring the mediation process and advising disputants upon possible trade-offs. Heuristic utility functions were developed from cases supplied by the Australian Institute of Family Studies. Family Winner operates best when it is possible to allocate points to issues, and creative decision-making is not required.

Having successfully overseen the development of these applications, the research laboratory is now focussing on the development of a new multi-agent online dispute resolution (ODR) environment. The aim is to re-deploy the utility functions and argument structures of Split-Up and Family Winner as Web based intelligent agents that can intuitively coordinate during a negotiation to assist parties involved in disputes to achieve fairer outcomes. A BATNA agent uses expert knowledge of the Australian Family law domain, combined with Toulmin's argumentation theory and Bayesian reasoning[2], to recommend a percentage property split. An Asset Divider agent uses heuristics and game theory and combine this percentage split with a significance rating of items provided by each party, to allocate issues and advise upon possible trade-offs.

The paper commences with some background information about ODR and briefly describes its place in the field of Alternative Dispute Resolution (ADR). A proposed framework for a multi-agent ODR environment is then presented and multi-agent interaction is described in detail, as well as the utility functions and behaviour of individual agents. Finally, the paper outlines the project team's ultimate vision, which is to deploy the architecture as an integrated ODR environment, offering disputants a range of negotiation and mediation services.

2. Online Dispute Resolution

Alternative dispute resolution (ADR) is generally defined as processes that are 'alternative' to traditional court proceedings (litigation). The ADR movement has progressively played an increasingly important role in the move away from authoritarian and top down social and institutional structures to more open, accountable and inclusive arrangements [5]. Online dispute resolution extends this trend even further. While ADR represents a move from a fixed and formal process to a more flexible one, ODR (by designating cyberspace as a location for dispute resolution) moves ADR from a physical to a virtual place.

Although ODR sites have primarily been used for Internet-related disputes, ODR can also facilitate resolution of disputes that have not originated online. For instance, many blind-bidding sites that exist can be used to solve financial disputes, such as insurance claims, that are not necessarily related to e-commerce. In addition,

[2] http://journals.cambridge.org/action/displayAbstract?fromPage=online&aid=76835

considering the ease with which the younger generation uses online tools, it seems reasonable to suggest that within the next decade, ODR will become a central method of dispute resolution.

SmartSettle [3] assists parties in overcoming the challenges of conventional negotiation through a range of analytical tools. It is designed to clarify interests, identify trade-offs, recognise party satisfaction, and generate optimal solutions. The aim is to better prepare parties for negotiation and support them during the negotiation process. Applications such as Smartsettle are becoming popular alternatives to litigation. This is possibly because many people are starting to believe that for most conflicts, ODR is a better dispute resolution mechanism due to its convenience, low cost and speed. The benefits of ODR are described in detail by [6].

Other interesting and related research includes work presently being undertaken by [7] who use a multi-agent approach to simulate negotiation and decision making in the Rungis wholesale fruit and vegetable market[4] in France, and the work of [8] who are developing an integrated software framework for the rapid construction of a Web-based negotiation support systems.

3. A Multi-agent Online Dispute Resolution Architecture

The project team believes that there are a number of advantages in using a multi-agent approach to develop ODR systems. Firstly, the loosely coupled nature of multi-agent systems can reduce the complexity of adding additional services. Services can be added somewhat independently by creating new domain agents, thus eliminating the need for major modification of existing programming code. Another advantage is that by using a dedicated agent development environment such as JADE[5], communication protocols are readily available. External agents can also access services offered via the interface agent using the JADEGateway class. Communication protocols in JADE are defined by the ACL language specified by FIPA[6]

The JADE main container also provides two special agents; 1) an Agent Management System (AMS) that ensures that each agent has a unique name, and allows agents on external containers to be terminated; and 2) a Directory Facilitator (DF) that lists services offered by agents so that other agents can find them. These two special agents are very useful for managing independent services. JADE can also run in any J2EE compliant container and with most of the popular database management systems[7]. The system is configured to run on a Tomcat server using MySQL and JDBC for database connectivity. The system architecture is presented in Figure 1.

[3] http://www.smartsettle.com

[4] http://www.rungisinternational.com/pages/gb/presentation/mar_hist.asp

[5] JADE version 3.6 available for download at: http://jade.tilab.com

[6] http://www.fipa.org

[7] http://www.theserverside.com/news/thread.tss?thread_id=14185

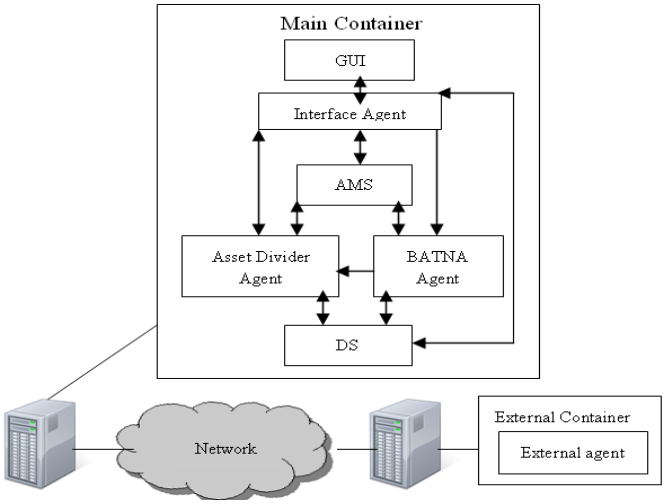


Figure 1. Multi-Agent Architecture

3.1. Interface Agent

The interface was designed as the system's gateway to external resources. Supported by a JSP graphical user interface (GUI) to accept user input, it also provides access to services for agents on external containers through the JADEgateway class. In the case of a marital property dispute, the user is presented with a series of screens similar to Split-Up prompting them to enter facts about a marriage and the party's financial contributions to it. This data is received by the Interface agent from the GUI in the form of XML. It is then transformed into the ACL format to be passed to the BATNA agent.

Another series of screens accepts the same user input as Family Winner about items in dispute, including an associated importance value that indicates the significance of each item to the disputants. Once again, the data is received by the Interface agent in the form of XML, transformed into the ACL format and passed to the Asset Divider agent.

3.2. BATNA Agent

Toulmin argument structures provide a mechanism for decomposing a task into sub-tasks. In Split-Up, ninety four arguments were identified during expert/engineer interactions for the determination of an appropriate percentage split of assets of a marriage. That is, the task of determining a percentage split was decomposed into ninety four sub-tasks. Many of these arguments produced claims which were in turn used as data for other arguments. All arguments ultimately contributed to three culminating arguments which were then fed into a final top level argument named the Percentage Split argument, the claim of which presented a solution to the problem. The claims for arguments in Split-Up were mainly inferred from data values with the use of

a neural network. The inputs into the network were the data items for the argument. The network's output represented the claim of the argument.

The BATNA agent uses the same Toulmin argument structures that were implemented in Split-Up. Bayesian reasoning, however, is used instead of a neural network to infer argument claims. With Split-Up it was later found during a controlled experiment that 16 variables actually produced a more accurate prediction of judgements than when 94 variables were used. A possible explanation offered by [4] was that judges rarely used many of the other 78 variables when distributing property. It was therefore decided that the BATNA agent would only use 16 variables to formulate argument claims.

3.2.1. BATNA Agent Process Flow

At runtime, the BATNA agent makes a JDBC connection to a legal database and extracts data about previous cases. The agent receives user input from the Interface Agent. The agent uses its business logic (described in 3.2.2) to formulate argument claims and determine a percentage property split. This percentage split is then sent to the Asset Divider agent.

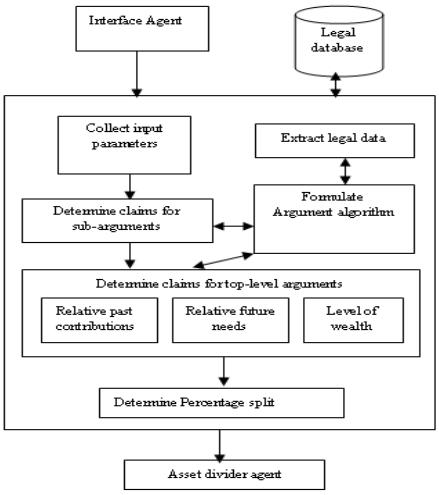


Figure 2. BATNA Agent Process Flow

3.2.2. BATNA Agent Utility Function

Bayesian reasoning is a statistical approach to uncertainty management in expert systems that propagates uncertainties based on the Bayesian rule of evidence [9]. Eq.(1) is known as the Bayesian rule. The concept considers that event A is dependent upon event B.

$$p(A|B) = \frac{p(B|A) \times p(A)}{p(B)} \quad (1)$$

$p(A|B)$ is the conditional probability that event A occurs given that event B has occurred; P(A) is the probability of event A occurring;

Bayes' theorem can be transformed to the following equation:

$$p(C_1|A_1A_2\ldots A_n) = \frac{p(A_1|C_i) \times p(A_2|C_i) \times \ldots \times p(An|C_i) \times p(C_i)}{\sum_{K=1}^{m} p(A_1|C_k) \times p(A_2|C_k) \times \ldots \times p(A_n|C_k) \times p(C_k)} \quad (2)$$

In expert systems, an expert determines the prior probabilities for possible hypothesis $p(H)$, as well as the conditional probabilities for observing evidence E if hypothesis H is true $p(E|H)$ [10]. In the architecture presented here, the BATNA agent itself fills the role of the expert by using statistical analysis of previous cases to determine prior probabilities and conditional probabilities of $p(H)$ and $p(E|H)$.

Example 1

Let us say that in a hypothetical property dispute Eq. (2) has been applied to determine the claims of all sub-arguments in the BATNA argument tree, leaving only the following three top level arguments to be processed before a final percentage split is inferred:

$A1$	The wealth of the couple can be considered average
$A2$	The wife in future will need more
$A3$	The wife in the past has contributed more

Table 1: Top Level Arguments

Three possible outcomes[8] (claims) are now compared:

$C1$	70% of property awarded to wife
$C2$	60% of property awarded to husband
$C3$	50% split

Table 2: Possible Outcomes (Claims)

The BATNA agent has calculated the following conditional probabilities of observing each argument for the three claims:

	Hypothesis		
	i = 1	***i = 2***	***i = 3***
$p(Ci)$	0.45	0.35	0.20
$p(A1\|Ci)$	0.25	0.60	0.55
$p(A2\|Ci)$	0.80	0.40	0.65
$p(A3\|Ci)$	0.70	0.00	0.80

Table 3: Conditional Probabilities for Argument Claims

[8] In reality there would be many possible outcomes. To keep the explanation simple only the likelihood of three are compared here.

Thus, by applying Eq. (2):

$$p(C_1|A_1A_2\ldots A_n) = \frac{0.25 \text{ x } 0.8 \text{ x } 0.7 \text{ x } 0.45}{0.25 \text{ x } 0.8 \text{ x } 0.7 \text{ x } 0.45 + 0.6 \text{ x } 0.4 \text{ x } 0.0 \text{ x } 0.35 + 0.55 \text{ x } 0.65 \text{ x } 0.8 \text{ x } 0.2}$$

$= 0.52$

$$p(C_2|A_1A_2\ldots A_n) = \frac{0.6 \text{ x } 0.4 \text{ x } 0.0 \text{ x } 0.35}{0.25 \text{ x } 0.8 \text{ x } 0.7 \text{ x } 0.45 + 0.6 \text{ x } 0.4 \text{ x } 0.0 \text{ x } 0.35 + 0.55 \text{ x } 0.65 \text{ x } 0.8 \text{ x } 0.2}$$

$= 0$

$$p(C_3|A_1A_2\ldots A_n) = \frac{0.55 \text{ x } 0.65 \text{ x } 0.8 \text{ x } 0.2}{0.25 \text{ x } 0.8 \text{ x } 0.7 \text{ x } 0.45 + 0.6 \text{ x } 0.4 \text{ x } 0.0 \text{ x } 0.35 + 0.55 \text{ x } 0.65 \text{ x } 0.8 \text{ x } 0.2}$$

$= 0.48$

C_1 is now considered the most likely outcome based on the Bayesian forecast. The BATNA agent has predicted that out of three possibilities, the most likely outcome is that a judge would award 70% of marital property to the wife. This percentage split is passed to the Asset Divider agent.

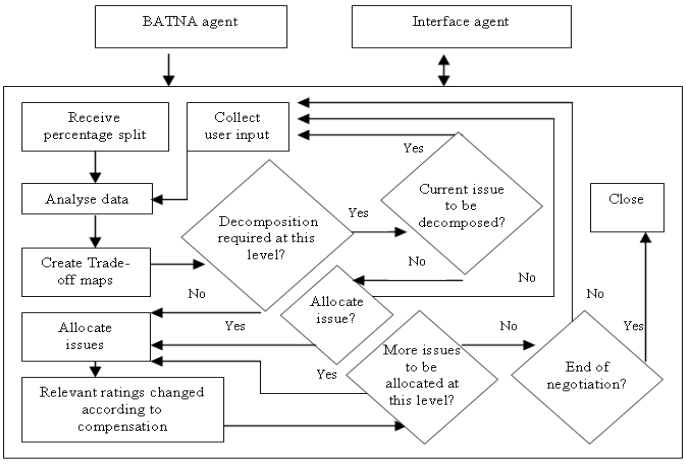
It should be noted that the Bayesian reasoning forecast method used here assumes conditional independence of evidence. To ensure validity of outcomes, results need to be thoroughly tested and compared to outcomes of the Split-Up application.

3.3. Asset Divider Agent

This section describes the process flow and utility function of the Asset Divider agent.

3.3.1. Asset Divider Agent Process flow

The Asset Divider agent receives a property percentage split from the BATNA agent, and collects user input about issues in dispute and their significance rating via the Interface agent. It applies game theory and heuristics to form trade-off rules based on this input. Issues are decomposed into sub-issues and allocated to each party in accordance with a logrolling[9] strategy. Trade-off maps are then produced and sent back to the Interface agent to assist parties in evaluating possible trade-offs between issues. Figure 3 shows the process flow of the Asset Divider agent.



pa sues to find issues that one

Figure 3. Asset Divider Agent Process Flow

3.3.2. Asset Divider Agent Utility Function

3.3.2.1. Defining the problem

The set of issues in dispute is: $D = X \cup Y$ where X = $\{X1, X2, \ldots, Xn\}$ is the set of issues that H sees as in dispute and Y = $\{Y1, Y2, \ldots, Yn\}$ is the set of issues that W sees as in dispute. H and W give a significance value (rating) to each of the issues in $D = \{D1, D2, \ldots, Dk\}$ where $m, n \leq k \leq m+n$. These significance values (or ratings) are denoted $xD = \{xD1, xD2, \ldots, xDk\}$ and $yD = \{yD1, yD2, \ldots, yDk\}$ respectively. Eq. (3) normalises each party's significance values, so that they both initially sum to one hundred and are then adjusted to incorporate the 70/30 percentage split received from the BATNA agent.

$NEW(xDi) = ((xDi \times 100) \times (200 \times 0.3))/\sum xDi$ and $NEW(yDi) = ((yDi \times 100) \times (200 \times 0.7))/\sum yDi$ where $i \in \{1, 2, \ldots, k\}$ (3)

Each issue can be decomposed into sub-issues $Di = \{Di,1, \ldots, Ddi,g(i)\}$, where $g(i)$ is the number of sub-issues for issue Di.

The rating of an issue refers to the value of an issue to a party. The rating of a parent issue is its numerical rating provided by disputants while the rating of a sub-issue is represented by a percentage of the parent issue's rating. The value of sub-issues, with respect to the rating of their parent issues is calculated next and is defined as a P-rating.

So the initial issue (such as child welfare) is now deleted from the list of issues to be considered and replaced by the sub-issues. The p-ratings take into account the ratings of both issues and sub-issues. P-ratings incorporate the influence of a parent issue to form the rating of a sub-issue. P-ratings are calculated according to the following equation:

If sub-issue Di is given ratings $\{xDi,1, \ldots, xDi,g(i)\}$ where $\sum xDi,j = 100$; and $\{yDi,1, \ldots, yDi,g(i)\}$ where $\_yDi,j = 100$; then the p-rating for Xdi,j is $xdi \times xdi,j/100$ and the p-rating for Ydi,j is $ydi \times ydi,j/100$ (4)

It should be noted that only the ratings of the initial issues and sub-issues are normalised. So after the initial normalisation, there is no reason why ratings or sub-ratings should sum to 100.

Example: Suppose, Party H gives issue 1 a rating of 60, and issue 2 a rating of 40. Suppose further that issue 1 has sub-issues 11 and 12 and that party H gives them ratings of 10 and 90 respectively. Then Issue 11 has a p-rating of 6 (10% of 60 = 6), and Issue12 has a p-rating of 54 (90% of 60 = 54).

3.3.2.2. Choosing the order of allocation

The order in which issues are considered for allocation is then calculated. Specifically, the function described in (5), choose (i) calculates the numerical difference between the ratings set by both parties towards the same issues.

Let set $D = \{d_1, d_2, \ldots, d_k\}$ be the set of differences between the ratings of the issues in dispute, where $d_i = |x_{Di} - y_{Di}|$ with $i \in \{1, 2, \ldots, k\}$. The issue with the highest d_i value will be presented first.

choose(1) = max $\{d_i : 1 <= i <= k\}$
The choose function, choose (i), for $i > 1$, operates on revised ratings. So choose(2) will be the maximum of the differences in revised ratings with: (a) The first issue allocated is removed from the list of revised ratings; (b) The revised ratings following the allocation of the first issue are used. The function is defined recursively.

(5)

The disputants can choose to either decompose the issue into sub-issues or directly allocate it. Example: Suppose Party H has issue1 with value of 60, issue 2 with value of 40 and issue 3 with a value of 0. Party W has issue1 with a value of 50, issue 2 with a value of 30 and issue 3 with a value of 20. The difference calculation for issue1 is 10, while the corresponding calculation for issue2 is 10 and the corresponding calculation for issue 3 is 20. Therefore D is the set {10,10,20}. Since issue 3 has the highest value of 20 in set D, the system will suggest to the disputants that they negotiate over issue 3 first.

3.3.2.3. Allocating Issues

Once a decision on which issue to distribute has been made, the issues need to be distributed. Issues need to be distributed by taking into account each parties significance factors. For example, if $D1$ is distributed first. H had a rating of 0 for $D1$ whilst W gave it a rating 20. Thus W is awarded $D1$. H needs to be compensated because W is awarded issue 3. Thus at any step, a function is required to keep a record of how many points each disputant has received at time t. Let us call this function GAIN(z,t). The eventual goal is to have GAIN(H,FINAL) fairly close to GAIN(W,FINAL).

In the example above, GAIN(H,1) = 0 and GAIN (W,1) =20.

(6)

3.3.2.4. The top level utility function

If an issue does not require decomposition or has been subdivided appropriately, the issue is allocated according to the issue's importance rating. The ratings of issues are hence compared. Essentially, the party whose rating is greatest is allocated the issue. If the ratings are of equal value, then the next issue to be considered for allocation is presented. Formally, this algorithm is presented as follows:

If $x_{Di} \geq y_{Di}$ then issue i is allocated to H, else issue i is allocated to W, where $i \in \{1, 2, \ldots, k\}$

(7)

3.3.2.5. Performing Trade-Offs

Once an issue (or issues) has been allocated, the remaining issues are affected to varying degrees, according to trade-offs executed as a result of the allocation. The extent to which the ratings of issues change is dependent on whether an issue is lost or gained, the ratings of issues forming trade-offs, and strength of the trade-off (represented by relationship figures). The values of these variables are combined to form a series of graphs, used to extract the amount of change affecting ratings. Once the issues and sub-issues have been allocated, trade-offs are needed to compensate the loser of the issue or sub-issue. To support the awarding of compensation, the Asset Divider agent develops Trade-off Maps. These diagrams are indicative of possible trade-offs between pairs of issues. A detailed discussion of trade-off maps can be found in [11].

4. Future Work

The research is being conducted in conjunction with industry partners the Queensland branch of Relationships Australia[10] and Victoria Body Corporate Services[11]. The first stage involved setting up a multi-agent architecture and establishing basic communication between a series of generic agents. The architecture is similar in design to one that was implemented in the AcontoWeb [12] system, which was built to facilitate the querying of travel and accommodation Web sites in a semantic Web environment.

The first agents to be deployed assist in resolving family disputes. The utility functions described in sections 3.2.2 and 3.3.2 are added to the generic agents to form two domain agents that can intuitively coordinate to assist parties during a marital property negotiation. The Split-Up project, from which the argument structure of the BATNA agent is based, is now somewhat dated. Split-Up used a neural network and machine learning to infer outcomes, whereas the BATNA agent uses a Bayesian reasoning approach. Once the BATNA agent is fully functional, a series of tests will be conducted to compare outcomes of Split-Up with the outcomes of the BATNA agent. If the outcomes are favourable, the BATNA agent will then be modified to include current case data and incorporate recent changes to Australian Family Law. If test results are non favourable in comparison to Split-Up, the BATNA agent's utility function and argument structure will need to be re-adjusted and refined. Other forecast methods such as certainly factor reasoning[12] may also be considered.

To satisfy the needs of both industry partners, agents are being developed to assist with body corporate disputes. Like the agents described in this paper, these agents will be expertly engineered, this time using domain knowledge of Victorian property law. Plans are also underway to develop a mediator agent that could guide disputants through a mediation process, using linguistic analysis to identify dispute agenda items, and automatic text summary to clarify the opening positions of parties. A range of linguistic tools such as these are now available for use in application development via the open source Java platform LingPipe[13].

[10] http://www.relationships.com.au/who-we-are/state-and-territory-organisations/qld

[11] http://www.vbcs.com.au/

[12] http://www.cse.unsw.edu.au/~billw/cs9414/notes/kr/uncertainty/uncertainty.html

[13] LingPipe version 1 available for download from: http://alias-i.com/lingpipe/index.html

5. Conclusion

The paper has presented a multi-agent framework providing decision support for disputes that parties attempt to resolve in cyberspace. The approach taken is to merge techniques developed from argumentation, artificial intelligence, and game theory to provide decision support in a multi-agent online environment. Apart from merely resolving disputes, it is anticipated that developing a negotiation support system will enable the continuation of constructive relationships following disputes. The project wishes to combine integrative bargaining, bargaining in the shadow of the law and formulation to develop decision support systems that support mediation and negotiation. The system, which is being developed in conjunction with industry partners Victoria Body Corporate Services and Relationships Australia, will respect ethical and legal principles and rely upon processes that are not only fair but are perceived by the parties to be fair. The ultimate aim is to provide disputants with an integrated ODR environment offering a range of services to assist them in achieving fairer negotiated outcomes.

6. References

[1] Rahwan, I. 2005, 'Guest Editorial: Argumentation in Multi-Agent Systems', *Autonomous Agents and Multi-Agent Systems*, vol. 11, no. 2, pp. 115-25.

[2] Zeleznikow, J. 2004, 'The Split-up Project: Induction, Context and Knowledge Discovery in Law', *Law Probability and Risk*, vol. 3, pp. 147-68.

[3] Toulmin, S. 1958, *The Uses of Argument*, Cambridge University Press, Cambridge.

[4] Zeleznikow, J., Bellucci, E., Uri, J.S. & Mackenzie, G. 2007, 'Bargaining in the shadow of the law - using utility functions to support legal negotiation', paper presented to Proceedings of the 11th international conference on Artificial intelligence and law, Stanford, California.

[5] Condliffe, P. 2008, 'The Rise of ADR', in *Conflict Management - A Practical Guide*, LexisNexis Butterworks, Brisbane Australia.

[6] Rule, C. 2002, 'Advantages of ODR', in *Online Dispute Resolution*, Jossey-Bass, San Francisco, California.

[7] Caillou, P., Baptisca, T. & Curchod, C. 2008, 'Multi-agent Based simulation for Decision-Making: an application to Rungis food market', paper presented to Group Decision and Negotiation, Coimbra Portugal.

[8] Szufel, P. & Wojewnik, P. 2008, 'Universal software platform for construction of web-based negotiation support systems', paper presented to Group Decision and Negotiation 2008, Coimbra Portugal.

[9] Negnevitsky, M. 2002, 'Uncertainty Management in Rule-Based Expert Systems', in *Artificial Intelligence - A Guide to Intelligent Systems*, Pearson Education Limited, Essex, pp. 55-84.

[10] Ng, K.-C. & Abramson, B. 1990, 'Uncertainty management in expert systems', *IEEE Expert Magazine*, vol. 12, no. 6, April, pp. 29-48.

[11] Bellucci, E. 2004, 'Developing Compensation Strategies for the Construction of Negotiation Deciison Support Systems', Latrode University.

[12] Dai, W. & Abrahams, B. 2005, 'A multiagent architecture for Semantic Web resources', paper presented to IEEE/WIC/ACM International Conference on Intelligent Agent Technology, Compiegne, France.