SITUATED DECISION SUPPORT FOR MANAGING SERVICE LEVEL AGREEMENT NEGOTIATIONS

Rustam Vahidov[1], Dirk Neumann[2]

[1] John Molson School of Business
Concordia University, Montreal, Canada
rVahidov@jmsb.concordia.ca

[2] Institute of Information Systems and Management (IISM)
Universität Karlsruhe (TH)
neumann@iism.uni-karlsruhe.de

Abstract

In this paper we propose a situated decision support system for efficient negotiating of service level agreements (SLAs) in Grids. Situated decision support systems effectively combine human judgment with autonomous decision making and action by agents. The intuition of using this approach for SLA negotiations lies in the monitoring and controlling of the fleet of local agents negotiating single services from multiple service providers by the use of a "manager" agent and human decision maker. We show through numerical experiments that the approach performs well under a set of simplifying assumptions.

1. Introduction

Quite recently, Grid computing has been increasingly gaining traction in a number of application areas. Grid computing denotes a computing model that distributes processing across an administratively and locally dispersed infrastructure. The ability to connect numerous heterogeneous computing devices, allows the creation of virtual computer architectures from otherwise idle resources.

According to market surveys is the average productivity of IT infrastructures extremely low: mainframes are up to 40 % of the time idle; UNIX servers even less than 10 % of the time used; individual PCs reach a utilization rate of at most 5 % [2]. This daunting ineffectiveness of IT infrastructures and associated costs entail that enterprises seek to use Grid technologies to outsource IT services. Rather than investing and maintaining proprietary infrastructures enterprises tap to the Grid and consume IT services on-demand. Service providers like SUN and Amazon accommodate this need by offering CPU hours and storage over web-services.

Outsourcing of IT services is, however, not a panacea as proposed by critics Nicholas Carr [2]. Any outsourced IT service incurs high risks, as the deployment is outside the control of the enterprises. Service providers address this risk by referring to formal contracts, so-called Service Level Agreements (SLAs). SLAs define the common understanding about services (e.g., priorities, responsibilities, guarantees and penalties once SLAs are violated) between service providers and consumers. Essentially, SLAs are the crucial instrument for service consumers to formulate guarantees on the service quality delivered by the service provider. Nonetheless, SLAs are also an important tool for service providers to advertise free service capacities as well as to manage their internal resources.

It is thus not astounding that SLA lifecycle management is currently a hot topic [6, 10]. One major success has been the specification of the de-facto standard *WS-Agreement* as a description language for SLAs by the Open Grid Forum (OGF). WS-Agreement allows the formulation of well defined and comprehensive contracts, which minimizes the risks of disputes resulting from unclear formulations among service provider and consumer. Clearly, a description language is merely prerequisite for well defined SLAs. What is equally important is a procedure with which SLAs are created. This becomes even more important, if SLAs are dynamically adapted dependent on resource availability and demand. If the SLAs are well-negotiated, the likelihood that balanced SLAs are signed is much higher. In this spirit, the Grid Resource Allocation Agreement Protocol Working Group (GRAAP-WG) (http://forge.gridforum.org/projects/graap-wg) of OGF set on their agenda for future work to define so-called negotiation profiles or protocols for SLA negotiation as part of the WS-Agreement specification [22].

SLA negotiations can become very complex as service consumers need to negotiate with multiple service providers to reserve different resources. The resource reservations that are

1530-1605/08 $25.00 © 2008 IEEE

needed are typically described via a workflow. Service consumers either want to have all resources reserved along the workflow or none at all. This is reasoned by the fact that failure to reserve one single constituent of the workflow results in unsuccessful termination of the service [15]. Due to this complexity the use of software agents have been proposed for managing SLA negotiations [4, 7, 19].

In this work our interest is in one-to many SLA negotiations involving one service provider and multiple potential customers. In such settings the complexity involved in tracking and management of multiple on-going negotiations can be alleviated by means of intelligent support (e.g. software agents). However, in our opinion, a balanced approach combining autonomous action with overall human control and decision-making is a more adequate approach, as it allows for timely intervention and improves the predictability of business outcomes. In this respect the recently introduced model of "situated decision support" seems to be a promising framework to apply to SLA negotiations [26, 28]. Situated decision support advocates abandoning the traditional "toolbox" type of decision aids in favor of a more connected and active mode. The major components of situated decision support system include sensors, effectors, manager, and active user interface. In this setup, decision support expands to include problem sensing and action generation and monitoring of the implementation of decisions in addition to the conventional intelligence/design/choice phases. It also allows flexibility for effectively combining autonomous action by the system with judgmental input from the human decision makers.

Since the fields of decision and negotiation support are closely related, in this work we show the potential applicability of the situated decision support model to managing multiple concurrent SLA negotiations, where a managing entity monitors and controls the local negotiation agents. The contribution of this paper is threefold:

- Firstly, we provide motivation for the adoption of situated decision support model for SLA negotiations;
- Secondly, we briefly describe a situated decision support framework for SLA negotiations.
- Thirdly, we illustrate the approach through numerical simulation experiments for the scenario involving two issues.

The remainder of this paper is structured as follows. Section 2 provides the background on SLAs, SLA negotiation and on agent strategies in automated negotiations. Section 3 motivates and describes the situated decision support framework and shows how it can be used for SLA negotiation. Section 4 provides an evaluation on the basis of numerical experiments showing that situated decision support is advantageous for SLA negotiations. Section concludes with a summary and an outlook on future research.

2. Background

This section provides brief overview of service level agreements (definition, content of typical SLAs), SLA negotiations (common protocols) and agent-based strategies. It further discusses why situated DSS model is deemed promising for use in SLA negotiations.

Service Level Agreements

A SLA is defined a contract between a service provider and consumer that specifies the rights and obligations of the provider and the penalties that will be applied if those obligations are not satisfied.

Since the late 80s SLAs have been used by telecom operators as contracts with their corporate customers. With the recent trend of outsourcing and application service providing, IT departments have adopted the idea of using service level agreements with their internal or external customers.

SLAs for Grid computing are not very different than those for other services (e.g. computation, or storage services). But while the elementary issues of an SLA can be the same, negotiating SLAs for Grid services is aggravated by the fact that Grid services are typically composed of several basic services. This implies that it is essential needs to manage concurrent SLA negotiations with multiple service providers [24].

Typical attributes of SLAs for Grid are compiled in Table 1.

Attributes	Attribute Levels
Operating System	Linux Solaris Windows
Service Availability	99,999 % 99,99 % 99,9 %
Price	< 1,000 €/month < 5,000 €/month < 10,000 €/month > 10,000 €/month

CPU type	Single CPU Dual CPU 4 CPU > 4CPU
CPU utilization	1,000 CPU hours 5,000 CPU hours 10,000 CPU hours
Storage (RAM)	1 GB 2 GB 4 GB > 4 GB
Bandwidth	1 Mb 10 Mb 100 Mb 1 Tb
Deployment	Webservice Source code others
Co-allocation	Service is provided by a *single* provider Service is provided by a *multiple* provider

Table 1: Common Attributes and attribute levels for Grid services

Service Level Agreements Negotiation

It is commonly accepted that the success of SLA negotiations relates to the specification and implementation of a protocol that creates legally-binding agreements. This protocol needs to describe a set of domain-independent messages, together with an abstract schema, such that it can be exchanged by all negotiators [15]. Several bargaining protocols that meet those requirements have been proposed by [4, 7, 9, 20, 25].

The bargaining protocols share a very similar structure but differ in some details. For example the protocol proposed by [25] involves multiple rounds among service consumers (e.g., co-allocators) and providers (e.g., schedulers) until they reach an agreement. The providers post their offers on request to the consumers, who can either select among the available offers or enter re-negotiation by relaxing some constraints.

Another noteworthy effort has been proposed by [17] who combine the negotiation protocol with a template-based contract generation process. At the end of the negotiation protocol a fully-fledged SLA is generated as WS-Agreement instance.

Agent-based Negotiation

SLA negotiations can involve a number of participants both on the consumer as well as the provider side trying to reach an agreement in multiple bi-lateral interactions. A given provider, for example, could have several on-going negotiations at the same time and has to set the objectives, constraints and strategies in those negotiation instances in such a way that maximizes the achievement of business objectives, while avoiding over-commitment, i.e. promising more than the provider can deliver. Clearly, with the growing complexity of the composite services and increasing customer base, automated means of supporting multiple negotiations could help the providers to better handle SLA negotiation processes. In this respect, employing intelligent agents seems to be an adequate approach. Below we briefly describe some agent-based approaches to automated negotiations, in particular for the case of multi-bilateral settings.

Most of the related work in the area of agent-based negotiation has been devoted to the design of strategies for agents. The fundamental work has been undertaken without considerations to SLA negotiations. Nonetheless, as those approaches are domain-independent, they are highly relevant for SLA negotiations.

Faratin et al. have proposed a “smart” strategy for autonomous negotiating agents [8]. Agents following this strategy would try to make trade-offs in a manner that the newly generated offer is similar to the opponent’s last offer, before trying a concession. Another work in this direction seeks to map business policies and contexts to negotiation goals, strategies, plans, and decision-action rules [16].

While fully automated negotiations may not always be a feasible choice, agents could also act as intelligent assistants, helping the users by providing advice, critiquing user’s own candidate offers as well as the offers by an opponent, and generating candidate offers for a user to consider [3, 14].

In multi-bilateral negotiations a negotiator may be having several concurrent negotiation processes taking place at the same time and involving multiple opponents A fuzzy set-theoretic approach to analysis of alternatives in multi-bilateral negotiations has been proposed in [29]. The authors have considered scenarios involving RFQ sent by one seller to multiple buyers (agents) with the purpose of deciding which potential buyers to negotiate with. They used fuzzy-relational approach to obtain a partial rank-order of the prospective buyers.

A setup where multiple agents negotiate autonomously and one agent is designated as a coordinator has been proposed in [18] and [21]. In [18] a buyer agent runs multiple concurrent

negotiation threads that interact with several sellers. The coordinator agent provides each thread with reservation values and negotiation strategy. The threads report back to the coordinator, which then advises them on the possible changes to reservation values or strategies. In [21] a similar approach has been used including coordinating agents and multiple "sub-buyer" agents. In [5] a protocol for handling many-to-many concurrent negotiations for internet services has been proposed.

In most of the above models of one-to-many negotiations it is assumed that the party on the one side can only have one agreement as an outcome of negotiations with multiple opponents (i.e. XOR case). In this work we interested in managing multiple SLA negotiations where each negotiation may fail or succeed regardless of others (OR case). In such settings the system could learn from the agreements made recently and take into account other relevant information (e.g. market situation) to direct concurrent negotiations.

3. Situated DSS for SLA Negotiation

When negotiation mechanism is used as one of the means to conduct daily basic business activities, much human effort may be required. Employing automated software components to conduct or assist human negotiators in conducting regular SLA negotiations may be a promising solution to achieve significant cost and time savings. However, with automated negotiations there is a potential danger that the overall process and the important business outcomes may become somewhat unpredictable. Moreover, often the course of negotiations depends on other factors, lying outside the domain of the expertise or sensory capabilities of the agents. For example, customer behavior may be heavily affected by the latest important economic, technology-related and other types of events. This calls for a type of solution allowing certain degree of automation, but allowing the control by the user of the overall process.

In light of the above considerations we propose a type of solution that would effectively combine human judgment capabilities with autonomous actions by agents. The key motivation here is to relieve human users from the necessity of being involved in each and every SLA negotiation process. Rather, the system should allow the human decision maker to effectively and efficiently manage the fleet of negotiating agents to meet and maintain higher-level business targets.

The field of decision support systems (DSS) is very much related to the area of negotiations and negotiation support systems [13]. In the recent DSS literature there has been important research streams directed towards building "active" and agent-based systems that would transform the original "toolbox" model of DSS into an active participant in the decision-making process, which could perform some decision-related tasks in an autonomous fashion [1, 11, 23, 27].

One such model introduced recently is known as *"situated decision support system"*, or *"decision station"* [26, 28]. Situated DSS looks to combine the benefits of agent technologies and those of decision support systems to facilitate active problem sensing, decision implementation and monitoring in addition to pure decision support. In essence, situated DSS expands the traditional model from purely "problem-solving/decision making" frame to situation assessment and action generation.

Situated DSS is made up from different active (agent) components in addition to the traditional "toolbox" of data, models, and knowledge. The components include: sensors (for information search and retrieval), effectors (for affecting current state of affairs), manager (for deciding how to handle a particular situation), and active user interfaces (for intelligent interaction with the user).

The composition and interactions of different components of the situated DSS model provides an appealing blueprint for facilitating one-to-many SLA negotiations with limited autonomous action and overall control of the process by a human decision maker. Various negotiation-related tasks can be mapped to specific agents in the model. For example, tasks including monitoring service levels, sensing potential problems, generating alerts, and predicting market trends for SLA could be delegated to the "sensor" agents. Controlling the fleet of negotiating agents by monitoring their progress and instructing them on the adjustments to negotiation strategies, reservation levels and preference structures can be delegated to some extent to the "manager" agent. Human decision maker could set the limits of authority of this agent and intervene if necessary. Each negotiation instance could be delegated to a particular "effector" agent that conducts a given negotiation and reports on the progress.

The model for supporting multiple SLA negotiations is shown in figure 1.The situated DSS model is essentially hierarchical involving three layers. At the bottom layer, which could be

called "operational" the agents perform negotiations. They are given the preferences, aspiration and reservation layers and strategies to follow and try to negotiate effectively with a given opponent. The basic cycle of generating an offer by these agents could be described as: retrieve preference structure (possibly updated by the "manager" agent); compare past offer and counter-offer; generate new offer according to adopted strategy. Automated negotiations have been studied extensively in the recent past and thus we do not focus on the detailed design of negotiating agents. One possibility is to employ "smart" strategy by the negotiating agents proposed in [8].

The second layer contains the manager agent, which makes use of the traditional DSS components (data, models and knowledge) to manage the negotiating agents. The manager monitors key indicators of the negotiation processes, such as number of agreements reached, proportion of failed negotiations, resource consumption and others and decides whether to intervene or not. For example, if memory resource becomes scarce, then in the next round of SLA negotiations the manager would instruct the agents to stress more the importance of memory resource in the utility calculation. One way to encode the knowledge of the manager could be through "If-Then" (possibly fuzzy) rules. The following simple example from a manager policy in SWRL [12] is the following rule which expresses that if memory units sold is somewhat larger than memory units estimated, the importance of memory units is slightly increased:

> *Function(?MemoryUnitsSold, ?MemoryUnitsEstimated, ?MemoryImportance, ?Increment) ←*
> *builtin:greaterThanOrEquals(?MemoryUnitsSold, ? MemoryUnitsEstimated),*
> *,builtin:add(?MemoryImportance,?Increment,? MemoryImportance)*

Such rule, when invoked would change the preference structure for the negotiator agents, which then would be willing to give up less memory units as a trade-off compared to other issues.

This level could be termed "planning" layer.

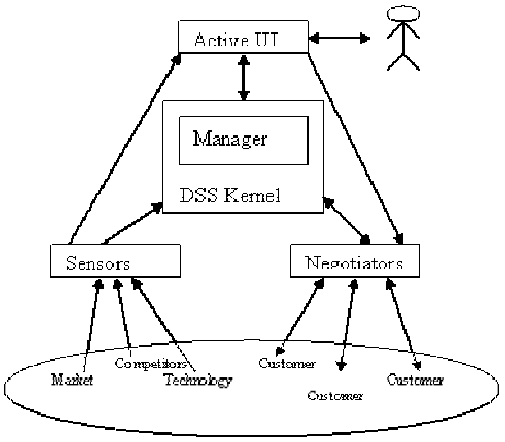

Figure 1: Situated DSS for managing multiple SLA negotiations

The manager agent would base its decisions solely on the information available to it, i.e. reports received from the sensors and negotiators. However, in reality, there are many other sources and types of information that may not be accessible to the agent. Furthermore, often judgmental input would be required to set the limits of authority for the manager agent, set the objectives and constraints (e.g. to maintain the ratio of failed vs. successful negotiations at a certain level) and attune its parameters, e.g. its risk profile, or the thresholds for sensitivity for reacting to undesirable developments. This is accomplished by the user through active user interface. Active user interface facilitates effective interaction with the user, while learning user preferences. This layer could be called the "judgmental" layer. Components of situated DSS could send various alerts to the user when human intervention may be desirable.

4. Numerical Experiments

To illustrate the situated DSS approach with the example and obtain preliminary empirical results we have conducted simulation experiments described in this section.

4.1 Data Generation

We have used the scenario where service provider negotiates with multiple prospective customers the terms of the SLA. For the simplicity, we considered only two issues: price of service and number of CPU hours. Simplifying assumptions were made, e.g. the negotiation ended the same day as it began. We did not actually simulate every negotiation process, as automated

negotiations have been studied in the past, and the focus of the work is on the overall performance of situated DSS that includes negotiating agents as components. We generated various values for reservation levels for the incoming customers and their preference structures. We furthermore assumed that if agreement between a customer and an agent was possible it would be a Nash solution. Thus, we assumed that the parties (consumers and negotiating agents) followed reasonable strategies to achieve mutually beneficial outcomes. The latter assumption has been adopted, since in [8] strategies were proposed for autonomous negotiations that lead to desirable outcomes for the parties involved.

In simulated settings we had included 20 days of SLA negotiations with twenty potential service consumers generated each day. The consumer reservation levels for price were generated from normal distribution around the "market" price, which was set to $100 per 1000 CPU hours. The requested number of CPU hours was also distributed normally. The number of available CPU hours was set to two millions and this "capacity" did not change throughout the period. Each service consumer was assigned a separate negotiator and the reservation levels were provided by the "manager" (agent). In addition, we have incorporated the preferences for the number of CPU hours vs. unit price by introducing discount options, which were also controlled by the manager. In particular, if agents committed more (less) CPU hours than targeted for a given period, the manager would decrease (increase) the discount values for additional CPU hours. This was accomplished through "If-Then" type of rules, similar to that described in the previous section.

On a single day a number of agreements were made ranging from 0 to 20. Then the outcomes were analyzed and corrections were made by the manager to price limits and discounts using simple rules, e.g.: "if the number of successful negotiations is smaller than expected by a given margin then adjust the reservation level for price downward". The manager could make such adjustments only within the limits specified by the human user. In the extreme case no such adjustments could be allowed.

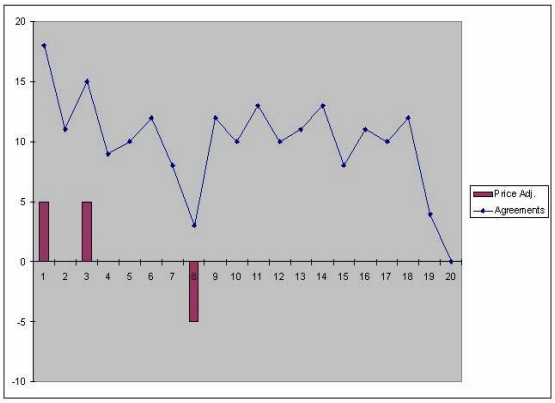

Figure 2: Effect of price level adjustments on number of agreements reached

4.2 Results

Figure 2 shows the number of service level agreements reached throughout the 20-day period as it is affected by the manager's adjustments. Vertical bars represent increases and decreases to price levels set by the manager agent and communicated to the negotiator agents. For example, in period 8 the manager drops the reservation levels for negotiating agents to increase the number of deals made. The number of agreements reduces to zero by the end of the period as the capacity limit for CPU hours is reached. Figure 3 shows the effect of adjustments to the discount levels on the number of CPU hours.

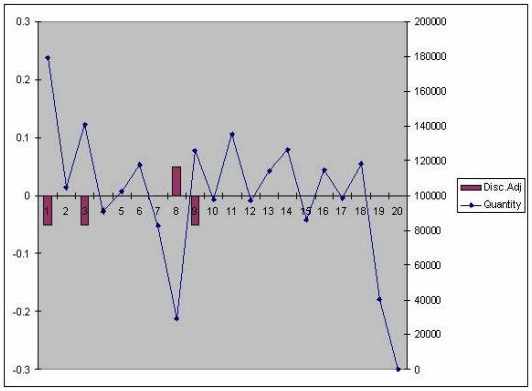

Figure 3: Effect of adjustments to discount levels on the number of CPU hours

As mentioned earlier, the freedom given to the manager in setting the levels is controlled by the human decision maker. In setting these levels the human decision maker should use his or her knowledge of the market situation and make a judgment. The settings could also translate into

the various levels of risk the decision maker is willing to take. For example if these are set high and tight there is a risk that not all CPU hours would be sold by the end of the period. If the spread between the upper and lower levels is large, then manager would start with higher levels and then gradually decrease them to be aligned with the market, thus possibly losing some early opportunities.

The above figures were based on a scenario when the market price remains stable. In case the market price is initially much lower than the price levels set by the manager and then gradually increasing, the manager would have to make several adjustments to its reservation level before it could get the needed number of agreements. This situation is shown in figure 4.

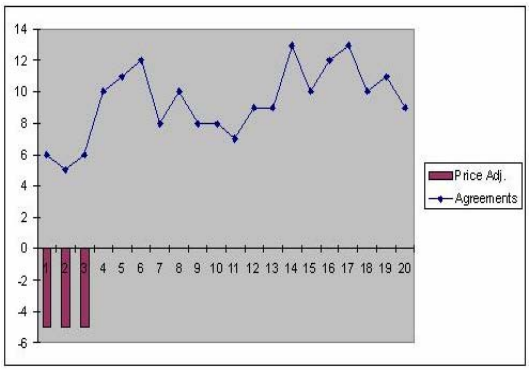

Figure 4: Adjustments when market price increases from low to high

Also, in case of a sudden change in market price the manager would be able to adjust, though it would take some time and result in the loss of some possibilities. In such cases making early interventions as a result of rightly exercised judgment by the human decision maker would be beneficial. Such a combination of human judgment with the automated capabilities could be beneficial to the successful operation of the business.

Finally, figure 5 compares profits achieved by the fixed pricing policy (price was fixed equal to the market price) vs. negotiation based policies based on the profits averaged over 20 runs.

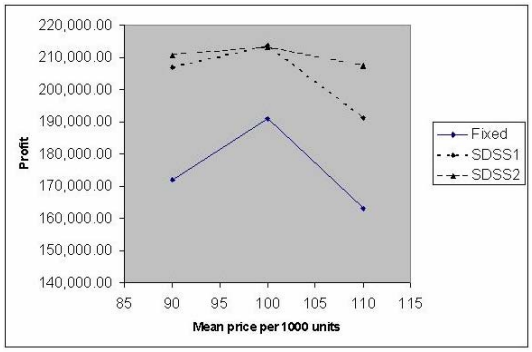

Figure 5: Comparison of average profits achieved by fixed pricing vs. situated DSS-driven negotiation-based approaches

The negotiation-based scenarios divide into two: the one where manager's actions were restricted (SDSS1), and the one where there are little restrictions (SDSS2). The latter case promises highest profits, though also least control that might jeopardize higher-level policies. As one can see the more adaptive situated DSS-based solutions perform better than the fixed price based one.

Table 2 shows the average profits made by using the fixed pricing and situated DSS approaches under various price settings. The column headings reading "price" refer to constant price set in case of the fixed pricing and to the initial reservation price in cases of situated DSS.

Method	Price = 90	Price = 100	Price = 110
Fixed	171.8	190.9	163.0
SDSS1	206.9	213.4	191.1
SDSS2	210.6	213.0	207.4

Table 2: Mean Profits earned by situated DSS vs. fixed pricing (in $1000)

5. Conclusion

This work proposed applying situated decision support approach to managing automated SLA negotiations. The framework is based on the model for situated decision support that effectively combines human judgment and autonomous decision making and action by agent components. The key idea behind the approach lies in the managing the fleet of negotiating agents by the use of a "manager" agent and human decision maker. We have illustrated the approach through simulation experiments performed under a set of simplifying assumptions.

Possible future work could be directed towards extending the situated DSS to workflow cases,

where the service consumers engages into concurrent negotiations with many service providers such that SLAs will be made with respect to all constituents of the workflow. In addition, future work will comprise the implementation of a user-friendly prototype involving multiple negotiated issues and empirical testing involving human subjects.

References

[1] Angehrn, A.A. and S. Dutta, *Case-Based Decision Support.* Communications of the ACM, 1998. **41**(5): p. 77-86.

[2] Carr, N., *The End of Corporate Computing.* MIT Sloan Management Review, 2005. **46**(3): p. 66-73.

[3] Chen, E., R. Vahidov, and G.E. Kersten, *Agent-Supported Negotiaitons in the e-Marketplace.* Int. J. Electronic Business, 2005. **3**(1): p. 28-49.

[4] Czajkowski, K., et al. *SNAP: A Protocol for Negotiating Service Level Agreements and Coordinating Resource Management in Distributed Systems*. in *8th Workshop on Job Scheduling Strategies for Parallel Processing*. 2002. Edinburgh, Scotland.

[5] Dang, J. and M.N. Huhns, *Concurrent Multiple-Issue Negotiation for Internet-Based Services.* Internet Computing, 2006. **10**(6): p. 42-49.

[6] Deora, V. and O. Rana, *SLA Enforcement in Grid using Runtime Violation Prediction and Reaction.* Working Paper, 2007.

[7] Eymann, T., et al. *On the Design of a Two-Tiered Grid Market Structure*. in *Business Applications of P2P and Grid Computing, MKWI 2006*. 2006.

[8] Faratin, P., C. Sierra, and N.R. Jennings, *Using Similarity Criteria to Make Issue Trade-Offs in Automated Negotiations.* Artificial Intelligence, 2002. **142**: p. 205-237.

[9] Gimpel, H., et al. *PANDA: Specifying Policies for Automated Negotiations of Service Contracts*. in *1st International Conference on Service Oriented Computing (ICSOC 03)*. 2003. Trento, Italy.

[10] Hasselmeyer, P., et al. *Towards Autonomous Brokered SLA Negotiation*. in *eChallenges 2006*. 2006. Barcelona.

[11] Hess, T.J., L.P. Rees, and T.R. Rakes, *Using Autonomous Software Agents to Create Next Generation of Decision Support Systems.* Decision Sciences, 2000. **31**(1): p. 1-31.

[12] Horrocks, I., et al., *Swrl: A semantic web rule language combining owl and ruleml.* Technical report, 2004.

[13] Jarke, M., M.T. Jelassi, and M.F. Shakun, *MEDIATOR: Towards a Negotiation Support System.* European Journal of Operational Research, 1987. **31**(3): p. 314-334.

[14] Kersten, G.E. and G. Lo, *Aspire: Integration of Negotiation Support System and Software Agents for E-Business Negotiation.* International Journal of Internet and Enterprise Management (IJIEM), 2003. **1**(3): p. (in print).

[15] Kuo, D., M. Parkin, and J. Brooke. *Negotiating Contracts on the Grid*. in *eChallenges*. 2006. Barcelona, Spain.

[16] Li, H., S.Y.W. Su, and H. Lam, *On Automated e-Business Negotiations: Goal, Policy, Strategy, and Plans of Decision and Action.* Journal of Organizational Computing and Electronic Commerce, 2006. **13**: p. 1-29.

[17] Ludwig, H., et al. *Template-Based Automated Service Provisioning - Supporting the Agreement-Driven Service Life-Cycle*. in *International Conference on Service Oriented Computing*. 2005.

[18] Nguyen, T.D. and N.R. Jennings. *Coordinating multiple concurrent negotiations*. in *3rd Int. Conf. on Autonomous Agents and Multi-Agent Systems*. 2004. New York, NY.

[19] Ouelhadj, D., et al., *A Multi-agent infrastructure and a Service Level agreement Negotiation Protocol for Robust Scheduling in Grid Computing*, in *Advances in Grid Computing - EGC 2005*, T.P.A.R. Peter M.A. Sloot Alfons G. Hoekstra, Marian Bubak, Editor. 2005, Springer: Heidelberg. p. 651-660.

[20] Parkin, M., D. Kuo, and J. Brooke. *A Framework & Negotiation Protocol for Service Contracts*. in *Proceedings of the IEEE International Conference on Services Computing*. 2006.

[21] Rahwan, I., R. Kowalczyk, and H.H. Pham, *Intelligent Agents for Automated One-to-Many e-Commerce Negotiation.* Australian Computer Science Communications, 2002. **24**(197-204).

[22] Rana, O., *OGF-19 Report.* Technical Report, 2006.

[23] Rao, H.R., R. Sridhar, and S. Narain, *An Active Intelligent Decision Support System.* Decision Support Systems, 1994. **12**(1): p. 79-91.

[24] Sahai, A., et al. *Specifying and Monitoring Guarantees in Commercial Grids through SLA*. in *Conference of Cluster Computing and Grids (CCGrid)*. 2003.

[25] Siddiqui, M., A. Villazon, and T. Fahringer. *Grid Capacity Planning with Negotiation-based Advance Reservation for Optimized QoS*.

in *Proceedings of the ACM/IEEE Conference on Supercomputing*. 2006. Tampa, USA.

[26] Vahidov, R. *Decision station: a Notion for a Situated DSS*. in *35th Hawaii International Conference on System Sciences*. 2002.

[27] Vahidov, R. and B. Fazlollahi, *Pluralistic Multi-Agent Decision Support System: A Framework and an Empirical Test.* Information & Management, 2004. **41**(7): p. 883-398.

[28] Vahidov, R. and G.E. Kersten, *Decision Station: Situating Decision Support Systems.* Decision Support Systems, 2004. **38**(2): p. 283-303.

[29] Van de Walle, B., S. Heitsch, and P. Faratin. *Coping with One-to-Many Multi-criteria Negotiations in Electronic Markets*. in *12th International Workshop on Database and Expert Systems Applications*. 2001.