A Framework for Automated Negotiation of Service Level Agreements in Services Grids

André Ludwig[1], Peter Braun[2], Ryszard Kowalczyk[2], and Bogdan Franczyk[1]

[1] University of Leipzig, Faculty of Economics and Management,
Information Systems Institute, 04109 Leipzig, Germany
{Ludwig, Franczyk}@wifa.uni-leipzig.de
[2] Swinburne University of Technology, Faculty of Information and Communication Technologies, Hawthorn, Victoria 3122, Australia
{PBraun, RKowalczyk}@it.swin.edu.au

Abstract. An important aspect of managing service-oriented grid environments is negotiation of service level agreements. In this paper we propose a framework in which we adopt the three-layer architecture of agent-based negotiation to the problem of service level agreement negotiation in services grids. We report on the first experience with an implementation of the framework in the context of the WS-Agreement specification provided by the Global Grid Forum and present lessons learnt when using this framework in a simple practical scenario.

1 Introduction

Grid computing has emerged as a new paradigm for next-generation distributed computing. It supports a notion of virtual organizations that can share resources for solving large problems in science, engineering, and business.

Service-orientation in grid computing focuses on virtualization of grid resources such as computational resources, storage resources, networks, programs, databases and so forth, and representing them by means of an extensible set of services that may be accessed, shared and composed in various ways [12]. The Open Grid Services Architecture (OGSA) [13] has taken up this approach and introduced the concept of grid services. At the same time integration and management of distributed applications by means of services is the objective of Web Services [36]. In an attempt to take advantage of progress in these two areas, the Globus Alliance [16] in conjunction with industry support has further developed the existing Web Service standards and the OGSA specification, and proposed the WS-Resource Framework (WSRF) [3]. WSRF supports creation, addressing, inspection, and lifetime management of resources as stateful services. It defines the semantics of WS-Resources and summarizes how interoperability between components from different sources can be enhanced using a service-oriented resource view [5]. Rather than shared usage of computer resources in computational grid infrastructures, services grids use grid paradigms in the context of services providing service-oriented applications on demand.

C. Bussler et al. (Eds.): BPM 2005 Workshops, LNCS 3812, pp. 89 – 101, 2006.

One of the most important aspects of service-oriented computing environments is that their administration and management is driven by individual organizational goals and application requirements. In order to support cross-enterprise dynamic composition and enactment of services, a number of fundamental issues regarding management of service quality and regulation of service behavior must be addressed. Some of these issues are: (a) How can the behavior of services be adjusted dynamically and who does that? (b) If services are created dynamically based on requirements of the consumer, how do participants find a mutually acceptable configuration? (c) How can these agreed service configurations be stored?

The key concept in addressing these issues is service level agreement (SLA). Similarly to commercial situations where "best effort" service guarantees are not sufficient, the agreement documents that specify what the user receives from the offered resources and its relevant performance guarantees are required in the form of SLA. SLAs capture the mutual responsibilities of the provider of a service and its client with respect to functional and non-functional parameters. For example, an agreement may define bounds on service response time and availability, or other service level objectives that describe the required quality of a service. Hence the main motivation for creating SLAs between providers and consumers is to get a reasonable certainty of the provided service behavior.

In a distributed cross-enterprise services grid numerous services interact with each other simultaneously, taking the roles of a provider and a consumer at the same time. The conditions of each of these relationships need to be represented in a SLA document. Keeping track of creating such SLAs, monitoring and evaluating service performance against them, and triggering appropriate actions in cases of SLA violation and exceptions are tasks of overwhelming importance. They include analysis of which part of SLA is violated and which party is responsible for it, what consequences arise from the violation for the overall system, and what the monetary and legal impacts are for the participants. Currently these tasks are performed by humans and require substantial manual effort, hindering broader adoption of services grids across enterprises as manual connection and contract negotiation are too costly on a large scale. Therefore, automation support for these tasks, especially for negotiating SLAs, is required. This automation must include automated creation of SLAs (e.g. as the result of negotiation), and other tasks during SLA lifecycle including their fulfillment and termination. In this context a flexible and precise SLA language, appropriate SLA templates, and a standardized SLA terminology are needed.

In this paper we propose a framework for automated negotiation of service level agreements in services grids, with the focus on the agreement creation phase. The framework adopts the three layer architecture of agent-based negotiation [21] to grid service agreements, involving decomposition of the negotiation into the negotiation objects, negotiation protocols and decision making models that are represented as different services. In addition to presentation of the theoretical framework, we also demonstrate its adaptability in a practical scenario and report on our first experiences in implementing it in the context of the Web Service Agreement specification (WS-Agreement) [1].

The paper is structured as follows. In the next section we briefly summarize related work concerning service level agreements in service-oriented environments. In Section 3 the concept of service-based representation of agreements is introduced.

Section 4 discusses techniques for negotiating service level agreements and the proposed framework for automated negotiation of SLAs in services grids is described in Section 5. In section 6 we provide more details on a prototypical framework implementation. Finally, the conclusions are presented in section 7.

2 Related Work

Research in grid service management has resulted in various approaches for grid resource reservation [6, 8, 38, 18] and quality of service delivery at the resource level [17]. An important part of these approaches is dedicated to the question of how to manage a grid resource in relation to an agreement document defining the resource consumption and provision. A standard concept of arranging and coordinating the services on the Grid are SLAs [22]. Accordingly, different specifications for describing and managing SLAs in XML-based representations are proposed in the Web Service Level Agreement (WSLA) [26], by SLAng [24] and in HP reports [31].

In order to realize an agreement represented by a SLA, several approaches define general frameworks for Grid resource reservation, acquisition, task submission, and binding [38, 18]. In contrast to these advanced reservation and balancing techniques, the Service Negotiation and Acquisition Protocol (SNAP) introduces a protocol for managing the process of negotiating an access to the resources and their use in a distributed system [4]. To represent these SLAs for every grid service running on behalf of a client, the corresponding SLA service can be instantiated. It contains and validates the SLA according to the WS-Agreement specification [1].

Although most of the above work recognizes SLA negotiation as a key aspect of SLA management, they usually provide little guidance of how negotiation (especially automated negotiation) can be realized. In a more general context, automated negotiation has been an important part of agent research (e.g. [21, 33]). They propose negotiation mechanisms including different interaction protocols and decision making models for negotiation in multi-agent systems. In this paper we adapt the agent-based negotiation approach for dynamic automated negotiation of SLAs in service grids, and provide a practical framework where different negotiation protocols and decision making strategies can be realized by service-based SLAs.

3 Service-Based Representation of Agreements

This section gives a concise overview of how relationships between Grid participants can be modeled and managed in a standardized way. As stated before, the relationships between service providers and clients are represented in SLAs to express agreement about the behavior of a provided grid service. In service-oriented grid environments every element is represented as a service, e.g. a WS-Resource service. Following this notion of service orientation and virtualization of resources, SLAs can also be represented by a WS-Resource service. Such an approach is proposed by the WS-Agreement specification published by the Global Grid Forum (GGF) [14]. The fundamental idea of WS-Agreement is the representation of a SLA as WS-Resource service in an agreement service. It describes an XML-based language for specifying an agreement between a service provider and a consumer, and a protocol for creation of an agreement using agreement templates. In this way each WSRF compliant agreement service represents an SLA and provides interfaces through which the provider and customer service management applications interact with each other. As described previously each WSRF service is capable of supporting lifecycle mechanisms.

The WS-Agreement specification consists of two basic layers (figure 1):

– the agreement layer: which provides a Web service-based interface that represents SLAs,
– the service layer which represents the application-specific layer of the provided business service.

The agreement layer represents a manageable interface for contacting and interacting with a service provider. It publishes information like acceptable agreement terms and enables the creation of agreement service instances in a factory service. The agreement service facilitates the representation of an agreement, captures the agreement terms, manages service lifetime and provides agreement composition capabilities.

The WS-Agreement model covers all periods of the SLA lifecycle. It contributes an abstract but substantial interface description for SLAs between providers and consumers and encourages the approach of service-orientation in grid computing environments. However, it does not specify how the service provider and consumer can come to an agreement and how the agreement process can be supported.

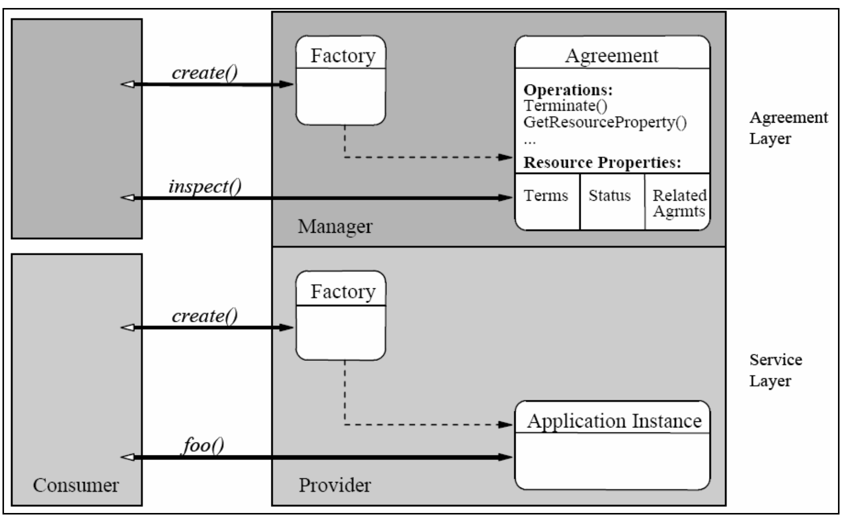

Fig. 1. WS-Agreement service model [1]

4 Negotiating Service Level Agreements

While all phases of the agreement lifecycle involve complex processes and require extensive investigation, the discussion in this work is restricted to the first period, i.e.

the creation of SLAs. The fundamental question of creating an agreement is: how do participants successfully create agreements? Humans, when faced with the need to reach an agreement on a variety of issues make use of negotiation, a process by which a joint decision is made by two or more parties. Typically, the parties first verbalize contradictory demands and then move towards an agreement by a process of concession making or search for new alternatives [29]. However, the scalable deployment and open architecture of WSRF environments enable a multiplicity of services with an unlimited number of service characteristics. Different organizational goals, service requirements and oppositional objectives require policies and technologies to manage the heterogeneity of a grid and make service negotiation a complex process. Currently operations like SLA creation and negotiation are subject to manual and human influence and call for additional support, e.g. for automated negotiation. In automated negotiations a broad range of issues have to be analyzed. That includes issues about the necessary negotiation interactions, characteristics of the negotiated services, and rules what decisions have to be made at what time [25].

A commonly recognized approach to automated negotiation is based on structuring its mechanism into: negotiation objects, negotiation protocols, and decision making models [25].

Negotiation protocols define a set of rules that prescribe the circumstances under which the interaction between agents takes place, called the rules of encounter [25]. They cover the permissible types of participants, the negotiation states, the events that cause negotiation states to change and the valid actions of the participants in particular states. While negotiation protocols are quite different for different categories of negotiation, they have one thing in common: interaction protocols expand the scope from the exchange of single messages to complete multi-step transactions (also called conversations or dialogues).

Negotiation objects are described by the range of issues over which agreement must be reached. The object can contain multiple attributes. These attributes can be classified as:

- service-specific attributes, such as quality of service (QoS), service level or other technical specifications,
- transaction attributes that are generic for the service, such as price, timings, penalties and so on.

Moreover, attributes may be:

- non-negotiable (i.e. having a fixed value),
- negotiable (i.e. having multiple possible values).

Decision making models provide a computational apparatus for making negotiation decisions according to the participants' negotiation strategies. The negotiation strategy governs the participant's general behavior and best course of actions and policies to achieve a goal. The sophistication of the model and the decisions that have to be made are influenced by the negotiation protocol in place, by the nature of the negotiation object, and by the range of operations that can be performed on it [23]. Examples of decision making models used for automated negotiation are game theory based models [27, 32], heuristic approaches [29, 30] and argumentation-based approaches [28, 28, 33].

5 SLA Negotiation Framework

The application of the WS-Agreement model enables the representation of SLA relationships by agreement services and provides standardized interfaces for agreement negotiations. However, the WS-Agreement specification gives no recommendations for how to come to these agreements and how to integrate the actual negotiation of these agreements into one context. This work proposes a negotiation framework for service level agreements (Fig. 2).

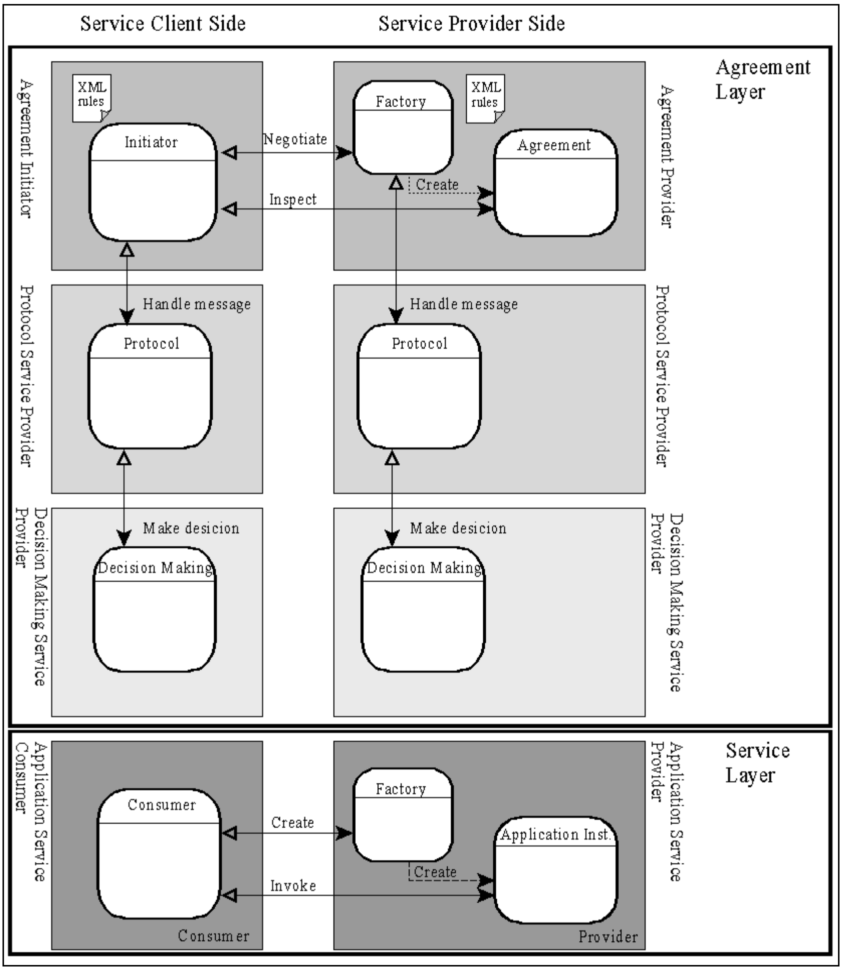

Fig. 2. SLA negotiation framework architecture

The underlying principle of the framework is decomposition of the negotiation mechanism into its basic elements: negotiation object, negotiation protocol, and decision making model in the context of service level agreements and on the basis of

WSRF grid mechanisms. By modularizing and structuring the agreement negotiation into its fundamental components it allows for dynamic adjustment of agreement policies and integrates interaction mechanisms, decision making management and dynamic control of service behavior. The framework is horizontally separated into the service client and service provider sides and vertically divided into the agreement and service layers, adopting the conceptual layered WS-Agreement service model. While the service layer is adopted from the specification without changes, the agreement layer extends the WS-Agreement service model. It consists of additional stand-alone components which fulfill well-defined, autonomous tasks during SLA negotiation. These components are:

- Agreement Provider and Initiator respectively,
- Protocol Service Provider,
- Decision Making Service Provider.

All components are encapsulated in their own services and can be offered by different parties following the service-orientation approach. The tasks of each component are described below.

Agreement Provider. The agreement provider represents a service provider in contractual matters. It provides interfaces that are necessary for interacting with a provider during service negotiation. It is responsible for describing the negotiation object (i.e. an application service) and its attributes (e.g. functional and non-functional properties). Beside that the agreement provider creates SLA documents in the form of agreement instances. The Agreement Provider incorporates the WS-Agreement model and has WSRF-compliant agreement factory and agreement port types.

The agreement factory service provides a manageability interface for negotiating with an agreement provider and is responsible for the interaction with an agreement initiator. It includes receiving and sending messages and advertising supported agreement templates. It facilitates the creation of agreement service instances and SLA lifetime management.

The agreement service represents the result of a successful negotiation in the form of a stable service level agreement between a service client and a service provider. It embodies a well understood service description and captures a mutual understanding of the expected application service behavior.

Neither the agreement factory nor the agreement service implement any negotiation logic itself – they provide only negotiation interaction interfaces and the SLA documents. Once a negotiation opponent (i.e. the agreement initiator) sends a message to the agreement factory service it forwards this message to a protocol service provider.

Protocol Service Provider. In order to make the agreement factory service independent from negotiation protocol-specific processing, it uses external protocol services. The protocol service decides who can do what and when, and how to react to events during negotiation. It enforces a coordinated behavior during a negotiation following the normative rules of the employed protocol. This includes rules about the types of participants, the negotiation states, events and actions that are taken on them. The protocol service offers interfaces that are appropriate for handling the received messages for the Agreement Provider.

Since the negotiation protocols can be different for different categories of negotiation, it is essential that the negotiating parties have a common understanding of meaning and order of the messages and their consequences. A convenient way to ensure mutually coordinated negotiation behavior is to use the negotiation protocols specified by standardization institutions such as FIPA [9]. Examples of FIPA protocols commonly used for automated negotiation between agents are FIPA's contract net protocol [10] and the iterated contract net protocol [11]. In the proposed framework a protocol service provider may offer various negotiation protocols that an agreement factory service can choose from. It also allows for multiple protocol service providers to offer numerous negotiation protocols that can be used as needed.

Nevertheless, the protocol service does not make decisions in response to the received messages, such as proposal assessments, evaluations or counter-offer generations. These operations are handled by an external decision making service.

Decision Making Service Provider. The decision making model of a negotiation is encapsulated in a decision making service. Similar to the protocol services several decision making service providers may offer various decision making models with different levels of sophistication encapsulated in numerous services.

In this context an important question is: how does the decision making service know the preferences and business rules (e.g. SLA parameter acceptance thresholds) of the actual application service provider? First of all, the service provider needs to formally define these preferences and business rules to make them available for processing by individual decision making services in a standardized way. For that purpose this framework incorporates the syntax and semantics of the Policy-driven Automated Negotiation Decision-Making Approach (PANDA framework) [15], which facilitates automated decision making during negotiation.

The PANDA framework defines so called decision strategy rules in a structured XML syntax. The basic building block of a strategy is a single rule, consisting of a condition part and an action to be performed if the condition is satisfied. The conditions are Boolean expressions and an action is a series of data sources. Each negotiation object that influences the decision is represented by a rule and only if the condition of the object's rule is fulfilled the action will be executed. The condition includes a Boolean operator, the minimal utility acceptance threshold and the relative utility weight. Additionally each rule encapsulates the parameters that describe the utility function for a certain object of negotiation.

6 Implementation Details

The proposed negotiation framework has prototypically been implemented and demonstrated with a simple business scenario in a Grid service environment. All components involved in the framework are implemented as Web services and hosted by the WSRF.NET 2.0 platform [36, 37], an implementation of the WSRF specification running on Microsoft's Internet Information Server. The presented services are developed using C# programming language in Microsoft's Visual Studio .NET 2003 environment.

The negotiation scenario presents a business model where a financial service provider offers financial services on the basis of Web services to several clients. One of these services that is implemented in our scenario simulates the evaluation of a person's credit history and anticipates the credit worthiness on a given taxpayer number. The service can be provided with different configurations described by several attributes. These are the service level, i.e. gold, silver, bronze describing the comprehensiveness of the calculated score, a quality of service index, an abstract value that includes availability of the service and the response time dependent on a requested level of throughput, the price per invocation and a minimal number of invocation requests. All of these attributes - service level, QoS value, price, and minimal invocation requests - are open for negotiation and form a multi-dimensional negotiation space.

After the service provider and its client allocate their extensible decision rules to the agreement initiator service and the agreement provider service, respectively, negotiation of the financial services can be initiated and executed. The demonstration integrates protocol services based on FIPA's contract net and iterated contract net protocols, and simple decision making services based on heuristic negotiation strategies. Figure 4 shows a screenshot of the user interface.

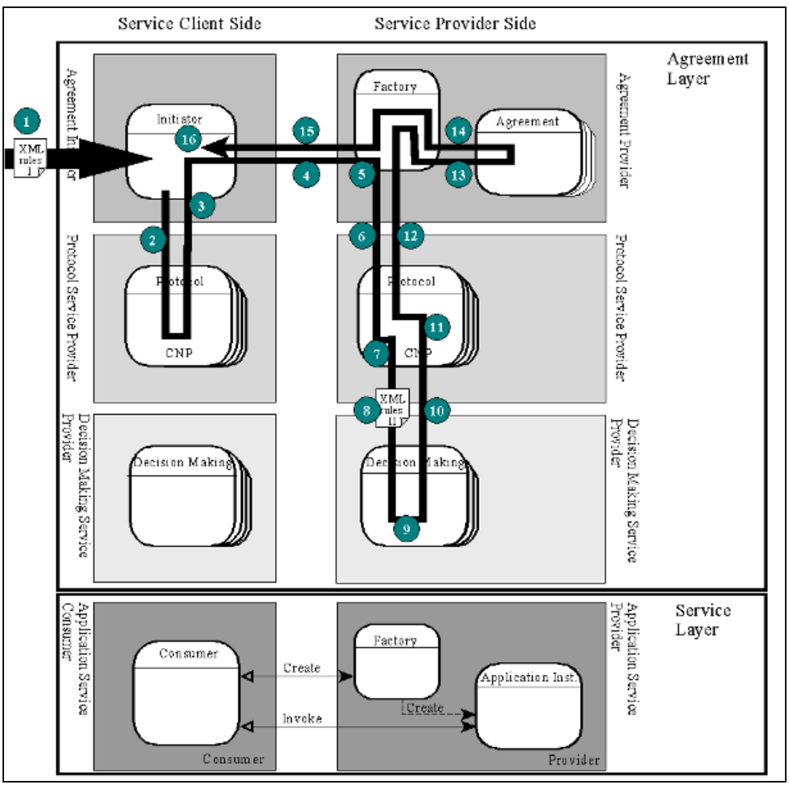

Fig. 3. Message flow during a call for proposal

The steps occurring during a call for proposal and proposal making executed one after another are illustrated by an arrow in Figure 3 and described below:

1. The whole negotiation process is initiated by an external enactment, i.e. manually by the service client. During this step the client's decision rules (XML rules I) are provided to the agreement initiator service. Also the choice of a negotiation protocol can be pre-defined here. However, the agreement initiator service can also choose a suitable negotiation protocol itself, i.e. if the agreement factory service insists on a particular protocol. As an example the FIPA contract net protocol (CNP) mentioned above is used.
2. The agreement initiator service sends a message to a suitable CNP protocol service to start negotiation. As the agreement negotiator maintains no negotiation logic, it just invokes the negotiation process. Together with this request it assigns the extensible decision rules of the client to the protocol service.
3. The CNP initiates negotiations with a 'call for proposal' message (CFP) that is sent to the service provider according to a pre-defined syntax (ACL). The protocol service creates such a CFP message and returns the ready-for-sending message to the agreement initiator.
4. The agreement initiator service contacts the agreement factory service and sends the CFP message.
5. The agreement factory service, as the manageable interface for contracting with a service provider, receives the message. As the decision rules of the contracting parties are usually contrary and kept private, the factory service stores another set of decision rules (XML rules II) for the service provider. This service provides only an interface and does not implement any operations itself. It analyses the value for the protocol suggested by the agreement initiator and assigns the message together with the decision rules to a suitable CNP protocol service.
6. The message together with the XML rules II are send to a CNP protocol service.
7. The protocol service analyses the message and decides on the consequences when receiving a CFP. In this example it decides to make a proposal. However, generation of a proposal is part of the negotiation strategy and is therefore encapsulated in an external decision making service.
8. The CNP protocol service sends a call for generating a proposal to the decision making service. It attaches the XML rules II.
9. The decision making service creates a proposal on the base of the decision rules.
10. The generated proposal is returned to the protocol service.
11. The protocol service creates a proposal message compliant with the ACL syntax and embeds the values of the created offer.
12. Afterwards it returns the proposal message to the agreement factory service.
13. The agreement factory service creates a new instance of an agreement service and writes the values of the received proposal message into the agreement instance.
14. The end-point reference of the agreement service instance is retuned to the agreement factory service.
15. The agreement factory service sends the proposal message together with the end-point reference locator of the created agreement service instance back to the agreement initiator.

16. The agreement initiator triggers further actions following the sequence flow of the CNP. In particular it is able to request the agreement service instance, i.e. for evaluating the proposal by the client's decision making service.

Figure 4 shows a screenshot of the messages recorded at the demonstration GUI during a call for proposal.

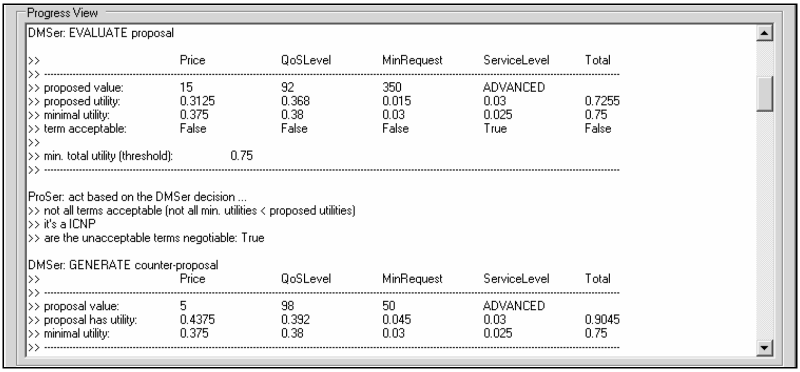

Fig. 4. Recorded messages on the SC GUI

7 Conclusions

This paper proposed a framework for dynamic creation of service level agreements based on automated negotiations between service providers and consumers. It provides several advantages over the existing approaches. The separation of the agreement and service layers as adopted from the WS-Agreement specification allows for a distinct encapsulation of the negotiation and application logics. It enables flexible relationships between the application and agreement service providers, and scenarios in which a number of application service providers can use various agreement providers to negotiate contracts with their clients. As described above, for that reason the service providers can leave their individual negotiation pre-configurations (e.g. supported contract types, acceptable agreements and negotiation constraints) encapsulated in the decision rules at the agreement provider side. Accordingly, the service clients can leave their decision rules at the agreement initiator component.

The second advantage comes from the modularization of the agreement layer. The participants – the agreement factory service in particular – have the flexibility to choose a suitable protocol depending on parameters such as the characteristics of the negotiation object or certain negotiation requirements of the client, e.g. if the client insists on negotiating on the basis of a particular protocol. In the same way it supports a flexible and dynamic choice of decision making models. It is possible to choose different levels of sophistication when making a decision and even changing the decision making model during a conversation is feasible.

The third advantage concerns the scalability and extensibility of the framework and its used components. Due to its modular architecture, additional protocol services or decision making models can easily be integrated to change the negotiation behavior of the participant. We expect that this can significantly reduce the time necessary to reach an agreement and that it can allow making a large number of transactions within a small amount of time automatically.

Acknowledgments. This work has been partly supported by the EU FP6 Integrated Project on Adaptive Services Grid (EU-IST-004617) and the Adaptive Service Agreement and Process Management (ASAPM) in Services Grid project (AU-DEST-CG060081). The ASAPM project is proudly supported by the Innovation Access Programme - International Science and Technology established under the Australian Government's innovation statement, Backing Australia's Ability.

References

[1] Andrieux, A., Czajkowski, K., Dan, A., Keahey, K., Ludwig, H., Pruyne, J. Rofrano, J., Tuecke, S., Xu, M.: Web Service Agreement Specification (WS-Agreement) 1.1 (2004)
[2] Chinnici, R., M. Gudgin, J.-J. Moreau, et al.: Web Services Description Language (WSDL) Version 1.2 (2003)
[3] Czajkowski, K., Ferguson, D.F., Foster, I., Frey, J., Graham, S., Sedukhin, I., Snelling, D., Tuecke, S.,Vambenepe, W.: The WS-Resource Framework (2004)
[4] Czajkowski, K., Foster, I., Kesselman, C., Sander, V. and Tuecke, S.: SNAP: A Protocol for Negotiation of Service Level Agreements and Coordinated Resource Management in Distributed Systems. Job Scheduling Strategies for Parallel Processing: 8th International Workshop (JSSPP 2002), Edinburgh (2002)
[5] Czajkowski, K., Ferguson, D., Foster, I., Frey, J., Graham, S., Snelling, D., Tuecke, S., From Open Grid Services Infrastructure to Web Services Resource Framework: Refactoring and Evolution (2004)
[6] Degermark, M., Kohler, T., Pink, S. and Schelen, O.: Advance reservations for predictive service in the internet. ACM/Springer Verlag, Journal on Multimedia Systems, 5(3) (1997)
[7] Emorphia: FIPA-OS; Available at: http://www.emorphia.com/research
[8] Ferrari, D., Gupta, A. and Ventre, G.: Distributed advance reservation of real-time connections. ACM/Springer Verlag, Journal on Multimedia Systems, 5(3) (1997)
[9] FIPA: Foundation for Intelligent Physical Agents (FIPA); http://www.fipa.org
[10] FIPA-CNP: FIPA Contract Net Interaction Protocol Specification; http://www.fipa.org/specs/fipa00029/SC00029H.html
[11] FIPA-ICNP: Iterated Contract Net Interaction Protocol Specification; http://www.fipa.org/specs/fipa00030/SC00030H.html
[12] Foster, I., Kesselman, C. and Tuecke, S.: The Anatomy of the Grid: Enabling Scalable Virtual Organizations. In: International Journal of High Performance Computing Applications. 15 (2001) 3, pp. 200-222.
[13] Foster, I., Kesselman, C., Nick, J. and Tuecke, S.: The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration. In: Open Grid Service Infrastructure WG, Global Grid Forum (2002), pp. 5-25.
[14] GGF: Global Grid Forum; http://www.ggf.org
[15] Gimpel, H., Ludwig, H., Dan, A. and Kearney, B. D.: PANDA: Specifying Policies for Automated Negotiations of Service Contracts. In: IBM Research Report RC22844 (2003)
[16] Globus: The Globus Alliance; http://www.globus.org

[17] Guerin, R. and Schulzrinne, H.: Network quality of service. In Foster, I. and Kesselman, C. (editors). The Grid: Blueprint for a Future Computing Infrastructure. Morgan Kaufmann Publishers (1999), pp. 479-503
[18] Hafid, A., Bochmann, G. and Dssouli, R.: A quality of service negotiation approach with future reservations (nafur): A detailed study. Computer Networks and ISDN Systems, 30(8) (1998)
[19] IBM, BEA, and Microsoft: WS-Addressing (2004)
[20] Jade: Java Agent Development Framework; http://sharon.cselt.it/projects/jade/index.html
[21] Jennings, N. R., Parsons, S., Sierra, C. and Faratin, P.: Automated Negotiation. Proc. 5th International Conference on Practical Application of Intelligent Agents and Multi-Agent Systems (PAAM-2000), Manchester, UK (2000) pp. 23-30
[22] Keller, A., Kar, G., Ludwig, H., Dan, A. and Hellerstein, J. L.: Managing Dynamic Services: A Contract Based Approach to a Conceptual Architecture. Proceedings of the 8th IEEE/IFIP Network Operations and Management Symposium (NOMS 2002) (2002) pp. 513–528.
[23] Laasri, B., Laasri, H., Lander, S. and Lesser, V.: A Generic Model for Negotiating Agents. In: International Journal on Intelligent and Cooperative Information Systems. 1 (1992) 2, pp. 291-317.
[24] Lamanna, D., Skene, J., Emmerich, W.: SLAng: A Language for Defining Service Level Agreements In Proc. of The International Workshop on Future Trends of Distributed Computing Systems (FTDCS'2003), San Juan, IEEE Computer Society Press (2003)
[25] Lomuscio, A. R., Wooldridge, M. and Jennings, N. R.: A classification scheme for negotiation in electronic commerce (eds. F. Dignum and C. Sierra). In: Lecture Notes in Computer Science. 1991 (2001), pp. 19-33.
[26] Ludwig, H., Keller, A., Dan, A., and King, R.: A Service Level Agreement Language for Dynamic Electronic Services. Proceedings of the 4th IEEE International Workshop on Advanced Issues of E-Commerce and Web-Based Information Systems (WECWIS 2002) (2002), pp. 25–32.
[27] Neumann, J. V. and Morgenstern, O.: The Theory of Games and Economic Behaviour. Princeton University Press, Princeton, US (1944)
[28] Parsons, S., Sierra, C. and Jennings, N.: Agents that reason and negotiate by arguing. In: Journal of Logic and Computation. 8 (1998) 3, pp. 261-292.
[29] Pruitt, D. G.: Negotiation Behaviour. Academic Press, New York (1981)
[30] Raiffa, H.: The Art and Science of Negotiation. Harvard University Press, Harvard, US (1982)
[31] Sahai, A., Durante, A., Machiraju V.: Towards Automated SLA Management for Web Services, HPL-2001-130 (2002)
[32] Sandholm, T. W.: Distributed Rational Decision Making. In: G. Weiss (ed.) Multiagent Systems, MIT Press, Cambridge, US (1999) pp. 201-258.
[33] Sierra, C., Jennings, N. R., Noriega, P. and Parsons, S.: A Framework for Argumentation-Based Negotiation. Proc. 4th International Workshop on Agent Theories, Architectures and Languages, Rode Island, USA, pp. 177-192
[34] Tuecke, S., Czajkowski, K., Foster, I., Frey, J., Graham, S., Kesselman, C., Snelling, D. and Vanderbilt, P.: Open Grid Services Infrastructure (OGSI) 1.0. (2003), pp. 7-12.
[35] University of Virginia, Grid Computing Group: WSRF.net (2004)
[36] W3C: Web Services (2004). http://www.w3.org/2002/ws/
[37] Wasson, G.: WSRF.NET Programmer's Reference (2005)
[38] Wolf, L. and Steinmetz, R.: Concepts for reservation in advance. Kluwer Journal on Multimedia Tools and Applications, 4(3) (1997)