A Reference Architecture for Automated Negotiations of Service Agreements in Open and Dynamic Environments*

Manuel Resinas, Pablo Fernández, and Rafael Corchuelo

ETS Ingenieria Informatica
Universidad de Sevilla, Spain
http://www.tdg-seville.info

Abstract. The provision of services is often regulated by means of agreements that must be negotiated beforehand. Automating such negotiations is appealing insofar it overcomes one of the most often cited shortcomings of human negotiation: slowness. In this article, we report on a reference architecture that helps guide the development of automated negotiation systems; we also delve into the requirements that must automated negotiation systems must address to deal with negotiations of service agreements in open environments. Finally, we analyse how well-suited current software frameworks to develop automated negotiation systems are for negotiating service agreements in open environments. This approach is novel in the sense that, to the best of our knowledge, no previous article compares extensively automated negotiation frameworks in the context of negotiating service agreements in open environments nor provides a reference architecture specifically designed for this scenario.

1 Introduction

Agreements play a major role to regulate both functional and non-functional properties, as well as guarantees regarding the provisioning of a service [1,2]. Many authors have focused on automating the negotiation of such agreements as a means to improve the efficiency of the process and benefitting from the many opportunities that electronic businesses bring [3,4].

Negotiating service agreements in an open environment poses a number of specific problems: on the one hand, creating agreements regarding the provisioning of a service involves negotiating on multiple terms, e.g., response time, security features or availability, not only the price, as is the case in typical goods negotiations; on the other hand, open environments require to deal with heterogeneous parties, to negotiate with partial information about them, and to cope

* This work has been partially supported by the European Commission (FEDER), Spanish Government under CICYT project WebFactories (TIN2006-00472), and by the Andalusian local Government under project Isabel (P07-TIC-2533).

with the dynamism inherent to service markets. Examples of open environments include both inter- and intra-organisational systems, e.g., corporate grids.

In this paper, we focus on software frameworks that help develop bargaining systems to negotiate service agreements in open environments.

Building on our analysis of the current literature, we conclude that no current framework is complete regarding the specific problems mentioned above. The goal of this article is to provide a reference architecture that gives a foundation to build such a framework. In Section 2, we detail the specific problems that raise automating the negotiation of service agreements in open environments; in Section 3, we analyse current proposals on automated negotiation frameworks; in Section 4, we outline a reference architecture for automated negotiation of service agreements in open environments to guide the research efforts in this area; in Section 5, we present the conclusions from our analysis.

2 Problems

Automated negotiation systems must cope with the following problems when facing automated negotiations of service agreements in open environments:

1. *Negotiations are multi-term.* Negotiations of service agreements usually involves many terms such as availability, response time, security or price. Therefore, it would be desirable for an automated negotiation system to:

   (1.1) Support negotiation protocols that allow the negotiation of multiple terms, such as most bargaining protocols.
   (1.2) Manage expressive agreement preferences: User preferences can be expressed in several ways (usually constraints and rules). Multi-term negotiations require preferences to capture relations between terms and, hence, enable making trade-offs during negotiations.

2. *Parties are heterogenous.* In open environments, parties may implement a great variety of negotiation protocols and present very diverse behaviours during the negotiation. To adapt to this variability, it would be desirable for an automated negotiation system to:

   (2.1) Support multiple negotiation protocols: Since there is no standard negotiation protocol, different parties may implement different negotiation protocols. Therefore, an automated negotiation system should support several negotiation protocols to avoid losing business opportunities.
   (2.2) Negotiate the negotiation protocol: Since the best negotiation protocol depends on the concrete situation [5], it is convenient a pre-negotiation phase, in which a negotiation protocol is agreed.
   (2.3) Support multiple negotiation intelligence algorithms: The effectiveness of a negotiation intelligence algorithm depends on the behaviour of the other parties [5]. Therefore, the automated negotiation system should support several negotiation intelligence algorithms to face the different behaviours of the other parties during the negotiation.

3. *Partial information about parties.* The knowledge about a party is important to strengthen our negotiation capabilities [6]. However, automated negotiation systems usually have only partial information about them [4]. Therefore, it would be desirable for an automated negotiation system to:
   - **(3.1)** Manage different types of knowledge about the other parties, namely: knowledge about the service or product the other party wish to sell or buy, about the other party itself (*e.g.* its reputation), and about the negotiation behaviour of the party (*e.g.* its temporal constraints).
   - **(3.2)** Diverse query capabilities: Automated negotiation systems may query information directly to the other party (*e.g.* as a template that should be filled [2]) or they may query third parties to obtain information related to another party (*e.g.* a reputation provider).
   - **(3.3)** Build analysis-based models of parties: Automated negotiation systems can analyse previous negotiations to build models and, later, use them to make better decisions during the negotiation process [6].
4. *Markets are dynamic.* Service markets can be extremely dynamic because services are not storable, which means that resources not used yesterday are worthless today [7], and, hence, providers may lower the cost of their services when their resources are idle. As a consequence, it would be convenient for an automated negotiation system to:
   - **(4.1)** Support several negotiations with different parties at the same time, so that the automated negotiation system can choose the party with which the most profitable agreement can be made.
   - **(4.2)** Select negotiation intelligence algorithms dynamically: Simultaneous negotiations with other parties can have an influence on the negotiation intelligence algorithms employed in a particular negotiation (*e.g.* if a profitable agreement has been found, the system can negotiate more aggressively with the others).
   - **(4.3)** Support decommitment from previously established agreements: In a dynamic market, new advantageous offers may appear at any time during the negotiations. Hence, it is very convenient to be able to revoke previous agreements, possibly after paying a compensation [8].
   - **(4.4)** Supervised creation of agreements: To avoid committing to agreements that cannot be satisfied, the automated negotiation system should be supervised by external elements such as a capacity estimator to determine whether an agreement can be accepted or not.
   - **(4.5)** Build market models: The characteristics of the market may have an influence on the negotiation process [3]. Therefore, it is convenient to build market models to obtain information such as the market reservation price of a service [6].

Last, but not least, a good software framework must be designed with a clear separation of concerns in mind. The separation of concerns indicates how independent is each part of an automated negotiation system from the others. A clear separation of concerns eases the addition of new negotiation protocols or intelligence algorithms without changing the other parts of the system.

3 Analysis of Current Frameworks

Negotiation frameworks focus on the reusability of the different parts of an automated negotiation system. (Note that we focus on software frameworks, not on conceptual frameworks like [9,4].) There are two kinds of negotiation frameworks: protocol-oriented frameworks [10,11,12,13], which deal with the negotiation protocol and interoperability problems amongst automated negotiation systems and intelligence-oriented frameworks [14,7,15,16,17], which focus on the decision-making and world-modelling of automated negotiation systems.

Protocol-oriented frameworks

Kim et al. [11] This article describes a web services-enabled marketplace architecture. It enables the automation of B2B negotiations by defining negotiation protocols in BPEL4WS and developing a central marketplace that runs a BPEL engine that executes the process. The authors also propose a semi-automatic mechanism based on a pattern-based process models to build BPEL negotiation processes.

Rinderle et al. [12] The authors propose a service-oriented architecture to manage negotiation protocols. They define a marketplace, which contains a set of statechart models specifying negotiation protocols. The negotiating parties map the statechart models onto BPEL processes and use them to carry out the negotiation.

Bartolini et al. [10] The authors present a taxonomy of rules to capture a variety of negotiation mechanisms and a simple interaction protocol based on FIPA specifications that is used together with the rules to define negotiation protocols. The authors also define a set of roles that must be implemented to carry out a negotiation process. In addition, the authors define an OWL-based language to express negotiation proposals and agreements.

SilkRoad [13] It consists of a meta-model, the so-called roadmap, intended to capture the characteristics of a negotiation process and an application framework, the so-called skeleton, that provides several modular and configurable negotiation service components.

Intelligence-oriented frameworks

Ashri et al. [15] In this article, two architectures for negotiating agents are described. However, the architecture is described from an abstract point of view and the authors do not provide any details on its components. In addition, it lacks some advanced features to deal with dynamic markets.

PANDA [7] It is a framework that mixes utility functions and rules to carry out the decision-making process. The decision-making component is composed of rules, utility functions and an object pool, which deals with the knowledge about the other parties and the market. However, the object pool is not implemented, but only vaguely specified. Furthermore, it does not support querying other parties to get information; it does not provide mechanisms to change negotiation intelligence algorithms at runtime; it does not

Table 1. Comparison of automated negotiation frameworks (I)

Proposal	**(0)**	**(1.1)**	**(1.2)**	**(2.1)**	**(2.2)**	**(2.3)**
Protocol-oriented frameworks						
Kim et al. [11]	+	+		+		
Rinderle et al. [12]	+	+		+		
Bartolini et al. [10]	+	+		+		
Silkroad [13]	+	+		+		
Intelligence-oriented frameworks						
Ashri et al. [15]	∼	+	N/A	+		N/A
Ludwig et al. [16]	∼	+	+	+		+
PANDA [7]	∼	+	+	+		+
DynamiCS [14]	+	+	N/A	+		+
Benyoucef et al. [17]	+	N/A	N/A	+		+

(0) Clear separation of concerns (2.1) Multiple protocol support
(1.1) Multi-term negotiation protocols (2.2) Negotiability of protocols
(1.2) Expressive agreement preferences (2.3) Multiple negotiation intelligence algorithms

support decommitment from previous agreements; and it does not allow a pre-negotiation phase.

Ludwig et al. [16] In this article, a framework for automated negotiation of service-level agreements in service grids is presented. This framework builds on WS-Agreement [2] and provides a protocol service provider and a decision making service provider to deal with the negotiation process. However, this proposal has important shortcomings in dynamic markets since it does not deal with partial information about third parties properly.

DynamiCS [14] It is an actor-based framework, which makes a neat distinction between negotiation protocol and decision making model and uses a plug-in mechanism to support new protocols and strategies. Nevertheless, the framework is not well suited to deal with partial information about third parties and does not cope with dynamic markets.

Benyoucef et al. [17] Their approach is based on the separation of protocols, expressed as UML statecharts, and strategies, expressed as if-then rules. Later, these UML statecharts are transformed into BPEL processes that are executed in a e-negotiation server, and the negotiation strategies are executed by software agents in automated e-negotiation interfaces. In addition, additional services can be composed to complement them. However, how this composition takes place is vaguely defined. Furthermore, the authors do not provide any details on the preferences they manage and whether they support multi-term negotiation protocols or manage different types of knowledge about parties. Another drawback is that it does not seem to be able to build analysis-based models of parties and its capabilities to deal with dynamic markets are limited.

Tables 1 and 2 depict how current negotiation frameworks deal with the problems automated negotiation systems must face in service negotiations in open environments (*cf.* Section 2): a + sign means that the proposal successfully ad-

Table 2. Comparison of automated negotiation frameworks (II)

Proposal	**(3.1)**	**(3.2)**	**(3.3)**	**(4.1)**	**(4.2)**	**(4.3)**	**(4.4)**	**(4.5)**
Protocol-oriented frameworks								
Kim et al. [11]								
Rinderle et al. [12]								
Bartolini et al. [10]								
Silkroad [13]								
Intelligence-oriented frameworks								
Ashri et al. [15]	+		+					
Ludwig et al. [16]		∼						
PANDA [7]	+		∼	+			+	
DynamiCS [14]								
Benyoucef et al. [17]	N/A	+	∼		+		+	

(3.1) Different types of knowledge (4.2) Select dynamically intelligence algorithm
(3.2) Diverse query capabilities (4.3) Decommitment support
(3.3) Analysis-based models (4.4) Capacity factors in binding decisions
(4.1) Simultaneous negotiations (4.5) Market models

dresses the feature; a ∼ sign indicates that it addresses it partially; a blank indicates that it does not support the feature; and N/A means the information is not available. The conclusions we extract from this analysis is that protocol-oriented frameworks need to be complemented with decision-making and world-modelling capabilities by means of either ad-hoc mechanisms or an intelligence-oriented framework. Regarding intelligence-oriented frameworks, although current solutions successfully deal with multi-term agreements (1.1 and 1.2) and cope with heterogeneous parties (2.1, 2.2 and 2.3) reasonably well, they lack dealing with partial information about parties (3.1, 3.2 and 3.3) and dynamic markets (4.1, 4.2, 4.3, 4.4 and 4.5).

4 A Reference Architecture for Automated Negotiation Frameworks

To overcome the problems of current negotiations frameworks described in the previous section, we have developed the NegoFAST reference architecture. Its goal is to define the data model, processes and interactions for which an automated negotiation framework should provide support. NegoFAST is divided into a protocol-independent reference architecture, the so-called NegoFAST-Core, and protocol-specific extensions. This allows to deal with a variety of different protocols while keeping the other elements of the reference architecture reusable. In this paper, we just focus on NegoFAST-Core, although a bargaining-specific extension (NegoFAST-Bargaining) has also been developed (more details can be found at `http://www.tdg-seville.info/projects/NegoFAST`).

To describe the NegoFAST-Core reference architecture (*cf.* Figure 1), we decompose it into modules, roles, interactions and environment. Modules are

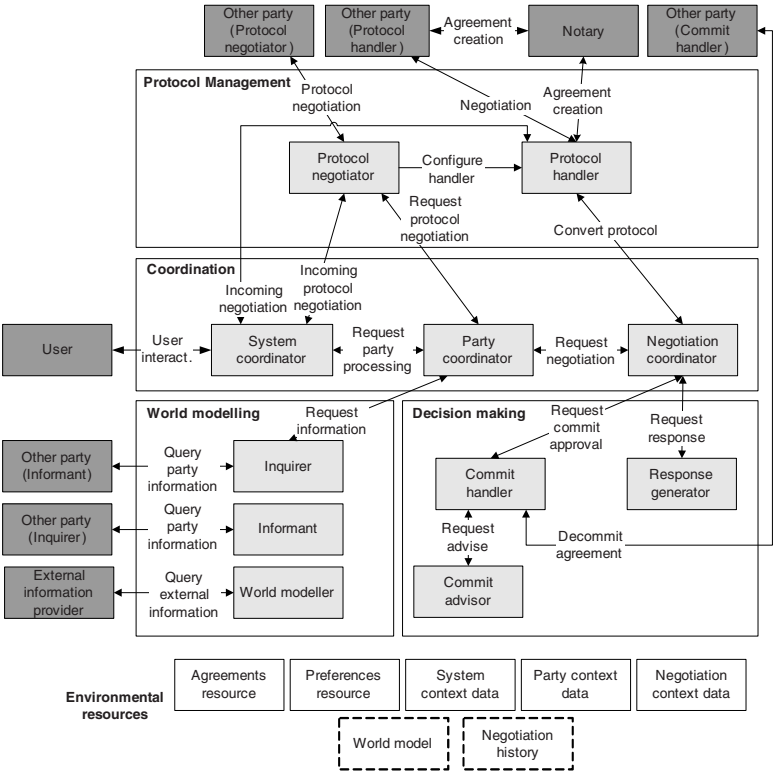

Fig. 1. The NegoFAST-Core reference architecture

depicted as big boxes and are composed of several roles (depicted as small light grey boxes) with arrows connecting them, which represent their interactions. The environment is divided into several resources, which are depicted as white boxes. Finally, elements that are external to the architecture are depicted as small dark grey boxes.

The aim of the protocol management module is to provide the elements that are necessary to deal with the selection and the execution of concrete negotiation protocols and to make the other roles of the architecture independent from them. Protocol management is composed of two roles:

ProtocolNegotiator. Its goal is to select and configure, if necessary, in cooperation with the other negotiating parties, the protocol that will be used during the negotiation process.

ProtocolHandler. It deals with the interaction with the other parties following a negotiation protocol by transforming the syntax of the negotiation protocol into negotiation messages that are understood by the other roles in NegoFAST-Core.

The goal of the decision making module is to provide mechanisms to determine the behaviour of the automated negotiation system during the negotiation. It is composed of three different roles:

ResponseGenerator. Its goal is to determine which messages are sent to the other parties during the negotiation.

CommitHandler. It is responsible for deciding whether and when the system should commit to a proposal and also deciding the decommitment from already established agreements if a more appealing agreement is found.

CommitAdvisor. It analyses the feasibility of accepting an agreement based on domain-specific knowledge (*e.g.* the provider's capacity to provision a proposal) and gives a recommendation.

The goal of the world modelling module is to obtain and manage knowledge about other parties and the market. It is composed of the following roles:

Inquirer. The *Inquirer* is the role in charge of obtaining more information about the other parties by polling their *Informants*.

Informant. It is responsible for publishing all public information that can be useful to other parties in order to evaluate the chances to make an agreement with it.

WorldModeller. Its goal is to build up a model of the other parties together with a model of the market. They are based on information supplied by *ExternalInformationProviders* and previous negotiations.

NegoFAST-Core defines three coordination levels that are coordinated by the three roles of which the coordination module is composed:

SystemCoordinator. It coordinates the interaction with the *User* and the negotiation requests from other parties. Furthermore, it stops the system when a termination condition holds such as reaching a preestablished negotiation deadline or achieving a desired number of agreements. It stores its status in environmental resource *SystemContextData*.

PartyCoordinator. It coordinates the interactions with the other parties before the actual negotiation takes place. This includes getting the information about the party that is necessary to start a negotiation with it by means of the *Inquirer* and agreeing on a negotiation protocol with the other party using the *ProtocolNegotiator*. Its status is stored in environmental resource *PartyContextData*.

NegotiationCoordinator. It coordinates the execution of the negotiation protocol, handled by the *ProtocolHandler* with the decision-making roles of the system. Furthermore, it should be capable of managing several negotiations simultaneously. Its status is stored in environmental resource *NegotiationContextData*.

Additionally, NegoFAST-Core defines the following environmental resources, which are data stores that can be read, modified or both by the roles:

AgreementsResource. It stores all agreements with other parties within the current system context to enable the comparison of agreements already reached with current negotiations and to allow the decommitment of one of them if necessary.

PreferencesResource. It allow the roles in NegoFAST to have access to the user preferences and to evaluate and compare agreements and proposals.

SystemContextData. It stores information managed by the *SystemCoordinator*, which includes: the moment when the system context started, the party references that has been received and the result of their processing.

PartyContextData. It stores information managed by the *PartyCoordinator*, which includes the information gathered by the *Inquirer*; the negotiation protocol selected, and the result of the negotiation.

NegotiationContextData. It stores information managed by the *NegotiationCoordinator*, which includes the current state of the negotiation context and the negotiation messages that have been exchanged with the other parties.

WorldModel. It stores the knowledge the automated negotiation system has about the other parties, the market and the domain the negotiation is about. For instance, knowledge about a the preferences and negotiation style of a party and the market price for a given service.

NegotiationHistory. It stores past negotiations. It is mainly intended for building models based on previous interactions. The *NegotiationHistory* can be seen as a list of the environmental resources of all system contexts that have been processed by the automated negotiation system.

5 Conclusions

Automated negotiation frameworks that help develop bargaining systems to negotiate service agreements in open and dynamic environments. However, current frameworks are not complete with regard to a variety of problems that may arise in such environments (*cf.* Sections 2 and 3). To overcome these issues, we have developed the NegoFAST-Core reference architecture. It provides a founding for the development of negotiation frameworks suited to negotiate service agreements in open environments. NegoFAST-Core can be complemented with protocol-specific extensions such as NegoFAST-Bargaining, which is a bargaining-specific extension we have already developed.

The advantages of having such reference architecture is that it defines the data model, processes and interactions for which an automated negotiation framework should provide support. Furthermore, it provides a common vocabulary to compare different automated negotiation frameworks.

To validate our approach, we have materialised the reference architecture into a software framework. Furthermore, we have used this software framework to implement three different use cases, namely: a computing job submitter, a computing job hosting service and a system to search for equilibrium strategies. The implementations of the software framework and the use cases can be downloaded from `http://www.tdg-seville.info/projects/NegoFAST`.

References

1. Molina-Jimenez, C., Pruyne, J., van Moorsel, A.: The Role of Agreements in IT Management Software. In: de Lemos, R., Gacek, C., Romanovsky, A. (eds.) Architecting Dependable Systems III. LNCS, vol. 3549, pp. 36–58. Springer, Heidelberg (2005)
2. Andrieux, A., Czajkowski, K., Dan, A., Keahey, K., Ludwig, H., Nakata, T., Pruyne, J., Rofrano, J., Tuecke, S., Xu, M.: WS-Agreement Specification (2007), `http://www.ogf.org/documents/GFD.107.pdf`
3. Sim, K.M., Wang, S.Y.: Flexible negotiation agent with relaxed decision rules. Systems, Man and Cybernetics, Part B, IEEE Trans. 34(3), 1602–1608 (2004)
4. Luo, X., Jennings, N.R., Shadbolt, N., Leung, H.F., Lee, J.H.: A fuzzy constraint based model for bilateral, multi-issue negotiations in semi-competitive environments. Artif. Intell. 148(1-2), 53–102 (2003)
5. Jennings, N.R., Faratin, P., Lomuscio, A.R., Parsons, S., Wooldridge, M., Sierra, C.: Automated Negotiation: Prospects, Methods and Challenges. Group Decision and Negotiation 10, 199–215 (2001)
6. Zeng, D., Sycara, K.: Bayesian Learning in Negotiation. Int. J. Hum.-Comput. Stud. 48(1), 125–141 (1998)
7. Gimpel, H., Ludwig, H., Dan, A., Kearney, B.: PANDA: Specifying Policies For Automated Negotiations of Service Contracts. In: Orlowska, M.E., Weerawarana, S., Papazoglou, M.P., Yang, J. (eds.) ICSOC 2003. LNCS, vol. 2910, pp. 287–302. Springer, Heidelberg (2003)
8. Sandholm, T., Lesser, V.: Leveled commitment contracts and strategic breach. Games and Economic Behavior 35(1), 212–270 (2001)
9. Faratin, P., Sierra, C., Jennings, N.R.: Negotiation Decision Functions For Autonomous Agents. Int. J. of Robotics and Autonomous Systems 24(3-4), 159–182 (1998)
10. Bartolini, C., Preist, C., Jennings, N.R.: A Software Framework For Automated Negotiation. In: Choren, R., Garcia, A., Lucena, C., Romanovsky, A. (eds.) SELMAS 2004. LNCS, vol. 3390, pp. 213–235. Springer, Heidelberg (2005)
11. Kim, J.B., Segev, A.: A web services-enabled marketplace architecture for negotiation process management. Decision Support Systems 40(1), 71–87 (2005)
12. Rinderle, S., Benyoucef, M.: Towards the automation of e-negotiation processes based on web services - a modeling approach. In: Ngu, A.H.H., Kitsuregawa, M., Neuhold, E.J., Chung, J.-Y., Sheng, Q.Z. (eds.) WISE 2005. LNCS, vol. 3806, pp. 443–453. Springer, Heidelberg (2005)
13. Strbel, M.: Design of roles and protocols for electronic negotiations. Electronic Commerce Research 1(3), 335–353 (2001)
14. Tu, M., Seebode, C., Griffel, F., Lamersdorf, W.: Dynamics: An actor-based framework for negotiating mobile agents. Electronic Commerce Research 1(1 - 2), 101–117 (2001)
15. Ashri, R., Rahwan, I., Luck, M.: Architectures for negotiating agents. In: Mařík, V., Müller, J.P., Pěchouček, M. (eds.) CEEMAS 2003. LNCS (LNAI), vol. 2691, pp. 136–146. Springer, Heidelberg (2003)
16. Ludwig, A., Braun, P., Kowalczyk, R., Franczyk, B.: A framework for automated negotiation of service level agreements in services grids. In: Bussler, C.J., Haller, A. (eds.) BPM 2005. LNCS, vol. 3812, pp. 89–101. Springer, Heidelberg (2006)
17. Benyoucef, M., Verrons, M.H.: Configurable e-negotiation systems for large scale and transparent decision making. Group Decision and Negotiation 17(3), 211–224 (2008)