Electron Commer Res (2009) 9: 317–349
DOI 10.1007/s10660-009-9040-6

A knowledge-based framework for complex, proactive and service-oriented e-negotiation systems

Giannis Koumoutsos · Kleanthis Thramboulidis

Published online: 22 April 2009

Abstract Advances in information technology and knowledge management change the way that e-negotiations, which constitute an important aspect of worldwide e-trading, can be structured and represented. In this paper, a novel approach that focuses on knowledge modeling, formalization, representation and management in the domain of e-negotiation is described. The proposed approach exploits Ontologies, Service Oriented Architectures, Semantic Web Services, software agent platforms, and Knowledge-Bases to construct a framework that favors dynamically adapted negotiation protocols, negotiation process visualization and management, modeling and preference elicitation of the negotiated object and automatic deployment of negotiation interfaces. Negotiation process, protocol and strategy are examined, and a hybrid approach that integrates rules and workflow diagrams to describe and represent them is introduced.

Keywords Negotiation support systems · Negotiations domain modeling · Negotiation protocol · Negotiation process

1 Introduction

Most of today's commerce, nearly 80% at year 2000 [1], is performed through negotiated trade. E-business, which is the use of Internet and Information Technology (IT) in all business processes, includes e-commerce and is the direction traditional businesses aim at. E-negotiations represent a very important aspect of worldwide e-trading and will play a crucial role in improving e-commerce.

G. Koumoutsos (✉) · K. Thramboulidis
Electrical & Computer Engineering, University of Patras, Patras, Greece
e-mail: koumouts@ece.upatras.gr

K. Thramboulidis
e-mail: thrambo@ece.upatras.gr

E-negotiations try to "virtualize" the real life negotiation procedures in the Internet-based world-wide market. The basic model of negotiations assumes the existence of at least two parties that share an important objective even though they have some significant difference(s). The purpose of the negotiation conference is to compromise the difference(s). Most of nowadays e-negotiations are manual, involving only humans. The main goal is to design and develop systems that will semi-automate the negotiation process. Autonomous software agents will take most of the burden carried out today in negotiations by humans.

Auctions are currently the most widely used form of e-negotiations. Auctions, due to their low cost server-based implementation, have rapidly proliferated on the Internet with multiple alternative schemes [2]. More complex forms of negotiations, such as bargaining for multiple and/or multi-attribute objects in parallel—combined negotiations—or sequentially, are also supported by e-negotiations. Complex negotiations tend to provide more alternatives to participants resulting in hard to predict sequences of exchanged messages and complicated decision making processes. Despite the growth of e-auctions, complex negotiations have not been extensively studied and mature tools for their automation are not available.

Negotiation protocol and negotiation strategy are among the key issues in e-negotiations. A negotiation protocol is a set of rules which govern the negotiation, whereas a negotiation strategy is a decision making model that participants employ, in order to achieve their goal.

Three categories of systems related to e-negotiation are defined in [3]: (a) negotiation support system (NSS), which assist users in communication and decision-making tasks; (b) negotiation software agents, which replace users in their communication and decision-making activities; and (c) e-negotiation media which are information systems that provide the infrastructure required to implement the rules of communication of a negotiation protocol. Regarding e-negotiation media, servers implement multiple protocols whereas applications usually implement a single protocol.

Negotiation systems are categorized according to [4] as passive, active and proactive:

(a) Passive systems are fast and sophisticated communication, calculation and visualization tools with no knowledge of the content in which they are used;
(b) Active systems may facilitate, support and mediate the negotiation process, incorporating the appropriate knowledge for responding any time invoked by the user; and
(c) Proactive systems use another level of knowledge that allows them to work independently of their users in monitoring of the process, making suggestions and criticizing without any request.

In this paper we describe a framework to facilitate the development of e-negotiation systems characterized by dynamic multi-protocol adaptation. According to this the negotiation protocol is not hardcoded into the negotiation media but it is selected and dynamically integrated into the negotiation system before the actual negotiation process. The specification of the selected protocol is available to the participants in a format ready for execution by a generic inference engine. The proposed proactive NSS framework, which exploits the Service Oriented Architecture (SOA)

paradigm and Knowledge Bases (KBs), facilitates the: (1) creation of dynamically adapted protocol specifications; (2) negotiation process visualization and management; (3) automated deployment of the required negotiation interfaces; and (4) strategy specification from abstract high level policies of the negotiating participants. It should be noted that strategy derivation is not examined; it is only used in our case study in order to prove the applicability of the framework.

A Basic Negotiation Protocol (BNP) is defined to capture the common concepts and functionality in e-negotiation protocols. The BNP is a collection of the fundamental vocabulary and rules that govern any type of negotiation. The BNP may further be specialized to specific protocols, such as the English auction, which are called "negotiation schemes". Using BNP participants can perform simple negotiations or even negotiate the rules of the actual negotiation scheme. The negotiation strategy depends on the selected negotiation scheme and is private for each participant.

In the development process of the framework the main focus was on e-negotiation knowledge modeling, formalization, representation and management. This knowledge concerns:

(a) Communication infrastructure between human and intelligent agent participants;
(b) Constraints and restrictions that participants should comply to, during a negotiation conference. These are given in the form of a dynamically adapted negotiation protocol; and
(c) The location and preference elicitation of the negotiated object.

The e-negotiation knowledge is modeled and formalized using semantic web technologies to be accessible by both humans and machines. XML-based and Semantic Web Services (SWSs) technologies along with formalization through declarative rules, which can be executed by a generic reasoning engine, ensure the required machine-to-machine interoperability. For human participants the role of representation and visualization of this knowledge is stressed. Solutions are proposed for negotiation process representation and preference elicitation.

Moreover, the modeling of the negotiated object, which may be real world object or concept, is crucial in the process of searching, locating, understanding and negotiating for the object. Ontologies and semantic web technologies are used to address the above issues. A fuzzy logic based approach is proposed for preference elicitation to aid end users specify their preference based on the well known applicability of objects in their life avoiding cumbersome technical details.

The remainder of this paper is organized as follows: In Sect. 2, we discuss related and background work. In Sect. 3, the proposed layered architecture is presented and in Sect. 4 an overview of the framework is given. Our approach on modeling the negotiation domain, the negotiated object and the negotiation process is presented in Sect. 5 and a negotiation case study is described in Sect. 6. Our proposal on location and preference elicitation of the negotiated object is given in Sect. 7. In Sect. 8 the importance of the Graphical User Interface (GUI) is explained and our approach is presented, while in Sect. 9 important implementation details are given. Finally, in Sect. 10, the paper is concluded and future work is given.

2 Related and background work

Many research groups work in the area of e-negotiations. Most of these deal with auctions, as for example the AuctionBot [5], which supports the pre-configuration of various auctions, and Generic Negotiation Platform [6], which separates auction specifications from the logic of the server. The meet2trade auction platform [7] is a generic market server for designing and combining various auction formats. It is a software suite for supporting and automating the complete auction design and execution process.

More recent research focuses beyond auctions to more complicated negotiation schemes as the ones of our approach. Kim and Segev [8] provide a statechart description for one e-negotiation protocol and the corresponding BPEL4WS process. Chiu et al. [9] presents an approach for developing e-negotiation plans providing meta-models for e-contract templates and e-negotiation processes. To the same direction a complete framework is described by Benyoucef and Rinderle [10] in a service oriented environment using statecharts for model-driven development as well as automatic generation of orchestrated Web Services (WSs). They report on a model-driven approach and a framework for rapid and user-friendly development of configurable service oriented e-negotiation systems. The need for a formal specification of negotiation protocols that are graphically designed and then mapped into web service orchestrations is stressed. A declarative language is used to specify negotiation strategies and an algorithm to map statechart models of negotiation protocols into web service orchestrations is implemented.

Kersten et al. [11] explored more complicated protocols proposing a configurable e-negotiation server to support bargaining. He broke down the negotiation process into well-defined phases, and defined a set of rules that govern the processing of proposals, decision-making, and communication of participants.

In [12] a set of Web services that support the process of negotiation and bargaining to facilitate the matching of supply and demand of Web services is proposed. As a market broker, these web services would help to discover and select the most appropriate available service for a specific request, bargain on the price and generate a contract.

INSS [13] is a system designed to support different negotiations on the Web. INSS is a generalization of the INSPIRE system and part of the InterNeg site. INSPIRE focuses on negotiation analysis and skills testing using a specific negotiation case, between a producer and buyer, INSS allows you to choose the negotiation protocol and set up a new negotiation case or problem.

Automating e-negotiations is of great importance and the question of the automation level and the way to achieve it is still open. According to [14] the state of the art in electronic negotiations shows fully-automated, process support and hybrid negotiation models. Towards fully automated negotiations, knowledge structures are used in order to provide the required autonomicity to negotiating software agents. On the other hand, in process support models, final decision making is undertaken by humans aided by the software agents in a passive, active or proactive way.

Towards automation of the negotiation process, declarative rules [15] and ontologies [16] were also utilized. Bartolini proposes in [15] a simple interaction protocol

which is able to support any mechanism that can be described with a taxonomy of declarative rules. A similar approach based on Bartilini's approach that uses agent technologies and tools to provide an initial implementation is presented by Badica et al. [17]. Tamma [16] uses the taxonomy proposed in [15] of rules, utilizing ontologies to make the representation of the rules of encounter explicit, machine readable and sharable; agents willing to participate to a negotiation session commit to the shared ontology, which represents the mechanisms governing the negotiation.

An approach aiming on the integration of e-negotiations within business processes and workflow management systems is presented in [18]. A generic model is developed based on Petri nets for simulation and verification purposes. A bi-lateral multi-attribute negotiation process model was defined where major activities involved were identified and implemented using Petri nets. It is an attempt to bring e-negotiations closer to their natural space of business processes and workflows, based on a mathematical formalism that will aid in consistency checking.

Negotiation support in nowadays information systems is examined in [19]. Weigand claims that negotiation processes are much harder to formalize than the business processes in the fulfillment phase and gives a formal analysis of electronic negotiations from a communication perspective, in particular, Habermas' theory of communicative action. Using this perspective, he makes a distinction between norm-oriented, goal-oriented and document-based negotiation and proposes instead of traditional modeling methods that adopt a data-oriented view, a communication-oriented view that provides more insight in the logic of negotiation processes.

The e-negotiation agency (eNAs) [20] offers a platform for autonomous agent-based negotiation. Negotiation agents can engage in automated negotiations using different negotiation mechanisms on behalf of the users. eNAs provides a variety of negotiation mechanisms involving different negotiation protocols, objects and decision mechanisms. It can support negotiations ranging from basic single-attribute bi-lateral negotiations to complex multi-attribute multi-lateral negotiations.

Kowalczyk extends this system towards fuzzy systems presenting Fuzzy e-Negotiation Agents (FeNAs) [21] system that assumes limited common knowledge and imprecise, vague preference. The FeNAs use the principles of utility theory and fuzzy constraint-based reasoning to find a consensus that maximizes the agent's utility. The algorithm uses the notion of fuzzy similarity to find negotiation solutions that are beneficial to both parties.

Wang in [22] presents a fuzzy-set based, two-phase evaluation model, allowing mobile agents to evaluate and filter online e-shops, and evaluate offers autonomously. The fuzzy evaluation criterion takes into account not only the attributes of offers, but also the reputation of corresponding e-shops. An auction-like negotiation model is presented, where the consumer agent can autonomously determine the initial offer and negotiate with multiple shops simultaneously. The final best offer is determined not only by the attributes of the offers, but also the attributes of the e-shops as well as the preference the consumer specified.

In [23] the idea that agents can increase the likelihood and quality of an agreement by exchanging arguments which influence each others' states is presented. Rahwan suggests that exchanging arguments is sometimes essential when various assumptions about agent rationality cannot be satisfied. To this end, he identifies the main research

motivations and provides a conceptual framework through which the core elements and features required by agents engaged in argumentation-based negotiation are outlined. In the same field of research Amgoud in [24] proposes a unified and general framework highlighting the role of argumentation and the study of outcomes. She explains agreements related to agents theories and defines the notion of concession in argumentation based negotiations.

The proposed highly customizable and dynamically adaptable environment that tries to imitate human negotiations favours argumentation based negotiations [23]. With the embedded reasoning and inference capabilities agents have the possibility to reason their positions, evaluate and even correct their point of view using the arguments of the negotiation participants. We apply a communication perspective on e-negotiations [19] that is widely used in non-electronic, face-to-face negotiations.

In [25] we find an extended study on bilateral, multi-issue negotiations. Three different procedures that can be used for this process are presented: the package deal procedure in which all the issues are bundled and discussed together, the simultaneous procedure in which the issues are discussed simultaneously but independently of each other, and the sequential procedure in which the issues are discussed one after another. The key problem of deciding which one to use in specific circumstances including time constraints and information uncertainty is examined. It is proven that the package deal is in fact the optimal procedure for each participant although it may be computationally more complex than the other two procedures, as it generates Pareto optimal outcomes.

An interesting approach that investigates the problem of users searching for the most preferred item in an online catalog is presented in [26]. To address this problem generally known as preference-based multi-criteria product search, a system, called example critiquing is proposed. Example critiquing aims at assisting the user to not only find the product that he is looking for, but also helps him process tradeoff information in order to achieve better decision accuracy.

Another approach for building a system for automated agent negotiation incorporating defeasible logic is presented in [27]. It uses the JADE agent framework and declarative negotiation strategies implemented with the defeasible reasoning system DR-DEVICE. An ontology is used to define elements of the world and speech-acts such as inform are used from the FIPA Agent Communication Language (ACL) to express agents intention to describe or alter the world.

There are several other technologies and approaches used extensively in our framework which are not directly connected to the e-negotiations domain. A Knowledge Base (KB) is a special kind of database for knowledge management. It stores knowledge in a machine-readable format, usually for the purpose of having automated deductive reasoning applied to the well formalized knowledge. It contains a set of data, often in the form of rules that describe the knowledge in a logically consistent manner. A KB may use an ontology to specify its structure and its classification scheme.

Service-oriented computing and Service Oriented Architectures (SOA) are being promoted as the next evolutionary approach to address the need for building larger, more complex software systems. SOA, which is not only an architecture but also a programming model, defines a new way of thinking about building software systems. A service-oriented architecture is essentially a collection of services along with an infrastructure that enables these services to communicate with each other.

A very interesting technology for describing processes and protocols is OWL-P [28]. Web Ontology Language (OWL) [29] and SWRL [30] are used to specify interactions as rule based commitment protocols. The authors separate public protocols from private policies thus allowing protocols to be easily reused and extended. This approach was our guide in describing the relations between negotiation protocol, process and strategy.

Regarding the orientation type, norm-oriented and goal-oriented negotiation processes are distinguished in [19]. Negotiators in norm-oriented negotiations act based on obligations and social commitments as these derive from negotiation steps. On the other hand, negotiators in goal-based negotiation, act based on well defined goals and objectives of the negotiation process. In the proposed framework both approaches are combined to exploit their advantages.

Information regarding e-negotiations should be represented in a way that is accessible and interpreted by both humans and machines. This vision is consistent with the Semantic Web initiative [31], which enriches the current Web through the use of machine-processable information about the semantics of information content. This will allow automatic resource integration and provides interoperability between heterogeneous systems. Ontologies have been developed in the semantic web research area to support searching and interpretation of the massive amounts of heterogeneous information in the Internet. OWL is the W3C standard for ontological modeling. OWL-S is an ontology within the OWL-based framework of the Semantic Web, for describing SWSs. It provides a standard vocabulary that can be used together with the other aspects of the OWL description language to create service descriptions and enable users and software agents to automatically discover, invoke, compose, and monitor Web resources.

A lot of works have been published in the e-negotiations domain with multiple views and approaches. Through the understanding of these works we make an effort to combine their benefits in a framework that will cover in depth several aspects of e-negotiations. We have defined a layered architecture and we use modern semantic web technologies and tools to provide solutions in a generic framework that provides the basis for multiple approaches and extensions to be applied. It is an attempt to bring the required knowledge to the center of our discussion on e-negotiations. We focus on solving knowledge management issues in order to provide a novel framework for modeling and deploying e-negotiations. Through knowledge management we provide a holistic approach that incorporates and unifies most of the existing approaches in the e-negotiations domain.

3 The architecture

In order for two or more entities to negotiate a communication interface as well as the semantics of the exchanged messages should be agreed. Negotiating participants have also to agree on the negotiation specific semantics and the negotiation protocol. They have to select or construct a specific negotiation scheme such as the one presented in Sect. 6 as well as to agree on the negotiated object specification.

To favor flexible, customizable and automated e-negotiations, the layered architecture shown in Fig. 1 was defined. The proposed architecture does not represent a

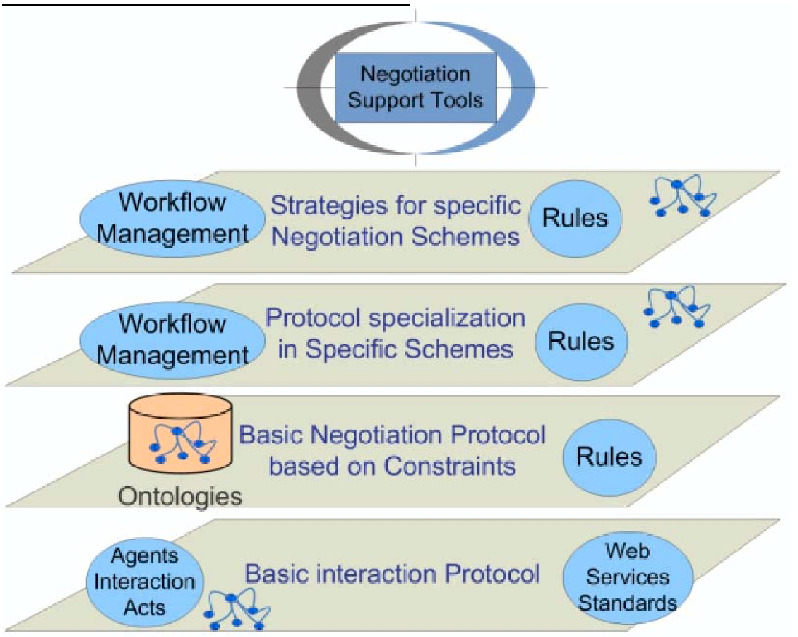

Fig. 1 The proposed e-negotiations architecture

strict layered architecture where a layer above can only be accessed by a layer below. It is closer to service-oriented stack models, developed from organizations such as OMG and FIPA and specified to supplement network-oriented models. The layers that constitute the proposed Architecture are:

(a) The basic interaction protocol layer,
(b) The Basic Negotiation Protocol layer,
(c) The specific scheme negotiation layer,
(d) The strategy layer.

The basic interaction protocol, which is the bottom layer, provides the basic communication infrastructure required to exchange messages in e-negotiations. Issues such as locating the negotiating participant, transporting the exchanged messages and defining the semantics of these messages are essential for any interaction.

The semantics of the exchanged messages in this layer are defined using Fipa agent Communication Acts [32] that can be considered as a complete set of interaction messages between autonomous agents. Communication Acts are formulated in an ontology keeping the name and semantics as described in FIPA specification. Key messages defined in this specification for the e-negotiation domain include among others: accept-proposal, agree, cancel, call for proposal (cfp), confirm, disconfirm, failure, inform, not-understood, propose, query-if, query-ref, refuse, reject proposal, request, request-when, request-whenever, subscribe, inform-if, inform-ref, proxy and propagate.

WS's technology was adopted to realize this layer. Existing WS standards are used to fulfill the interface and transportation requirements. WS technology allows the de-

finition of interfaces and the dynamic location and invocation of the corresponding services while the definition of semantics is done using other compliant technologies such as ontologies. With this approach interaction issues, as well as the required dynamic handling of concepts and semantics needed in e-negotiations are addressed. WSs are appropriate for deploying e-negotiation systems since they constitute a standardized and flexible integration technology widely adopted by the market [8]. WSs can guarantee the required interoperability and are capable of supporting dynamic relationships between participants.

The second layer, i.e., the BNP, captures the basic functionality involved in any negotiation scenario. This layer was defined to provide all these common activities found in e-negotiation scenarios and to formalize the constraints that should be satisfied by any negotiating participant. Constraints can be broadly classified as either enabling or limiting; they define the context within which participants have to move during the negotiation and serve the vital enabling role of establishing the negotiation system. Rules, which are used for the prototype implementation of the proposed BNP, are formalized using concepts from the well defined negotiation domain.

The third layer, i.e., the specific scheme negotiation layer, is used to define a specific negotiation scheme as a specialization of the BNP. A specific scheme, as for example auction or bargaining, must be selected during the first steps of any negotiation. This layer is mainly managed by human participants. The negotiation scheme selection is crucial for the strategy specification in the above layer. This is why visualized Workflow Management tools and the corresponding standards were examined for the specification, representation and monitoring of the negotiation scheme. The use of Workflow Management allows for a better understanding, design and deployment of complex negotiation schemes, including combined negotiations.

Finally, in the negotiation strategy layer, private strategies are defined for each participant depending on participant's policies and the specific negotiation scheme that was adopted. The same semantic tools and ontology-based vocabulary used in the lower layer are utilized for strategy specification to get a flexible and easy to understand negotiation architecture. Visualization provides a user friendly way of studying a specific scheme and through this the correct strategy specification. Strategy specification and automated generation are not among the main objective of this paper. A way of producing strategy from participants' high level policies is considered in the context of the case study to prove the applicability of the layered architecture.

The negotiation framework that is based on the proposed architecture contains all software entities required in e-negotiations. In our approach that utilizes a Service Oriented Environment (SOE) for the implementation of negotiation services, these entities include a dynamically created WS Client for using such an environment. The framework will also provide KB management features utilizing ontologies, rules and inference to explore the knowledge needed for negotiations.

4 The e-negotiation framework

The proposed framework, which is depicted in Fig. 2, allows a consumer and a merchant to negotiate for a specific object. Within the Knowledge layer we have included

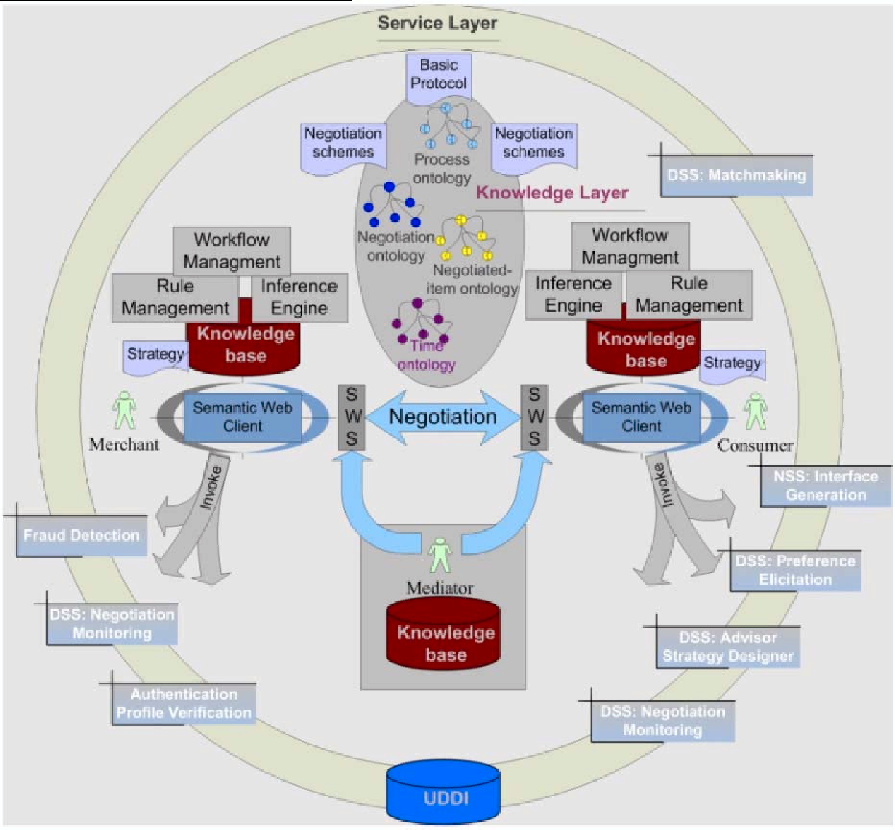

Fig. 2 The proposed e-negotiation framework

the static representation that ontologies provide, along with the dynamics of the domain given by rules and workflow diagrams. A repository was defined to contain the public knowledge captured by the Negotiation ontology, the Process ontology we have adopted, as well as the Time and Negotiated Object ontologies. The declarative BNP specification and a few well-known widely-used Negotiation schemes are also publicly available.

Each participant uses the publicly available Knowledge, an Inference Engine as well as Workflow and Rule Management tools. Knowledge in the form of ontologies and rules can be exploited better and faster if stored and managed locally. This is why a local KB containing the appropriate part of publicly available knowledge is exploited by each participant. Through Rule Management each participant can inspect and understand the BNP and edit the rules that depict his strategy. The Inference Engine is used to draw conclusions by executing the negotiation scheme and strategy rules to aid, as a first step acting as an NSS, or fully automate the negotiation process acting as an autonomous agent. It is obvious that the difference between NSSs and autonomous agents is mainly related with the completeness of the knowledge managed by machines. The Workflow Management is used to visualize the negotiation

process to aid participants in understanding the negotiation schemes and design the appropriate strategy.

According to the traditional approach every protocol specification has a proprietary implementation in order to be deployed and executed. In our specification that is based on declarative logic-programming the inference engine reasons over the specific scheme and participant's strategy for the execution of the negotiation process. Instead of altering the entire negotiation engine, only the declarative rules must be replaced in order to implement another negotiation scheme or strategy. Hence, the inference engine, which is the core component of our negotiation framework, is domain independent and it can be optimized in terms of scalability, security, and processing time.

For the negotiation to take place the participants have to generate the appropriate communication interface. In our approach negotiation interface generation is based on SWS standards and depends on the negotiation scheme specification. According to the selected negotiation scheme, which will be formalized in an XML-based language, each participant can dynamically produce using an XSLT engine the XML-based WSDL description of their negotiation interface. The negotiation can also involve a mediator that can be some kind of third party broker hosted outside the main participants. Any kind of mediation can be added in our framework as long as it exploits a knowledge base and the appropriate interfaces.

All activities that must be performed during a negotiation can be implemented using a SOE based on SWSs. These SWSs will be deployed by NSS providers that will conform to the proposed architecture and freely, or after an agreement, exploited by the negotiating participants. A small number of services, depicted in Fig. 2, were developed and used in our negotiation scenario to demonstrate the applicability of the proposed approach. A Web Client, which is dynamically produced in the framework from the services' WSDL description, exploits the provided web services environment. WS were selected as a promising technology to realize the SOA environment from multiple independent providers. This technology also provides the dynamic client creation and service invocation through the WSDL and the appropriate APIs. For example, a Decision Support Service (DSS) for Preference Elicitation can be implemented as SWS for the negotiated object. This service will be able to locate the ontology of the negotiated object, parse it and assist in preference elicitation by using multiple levels of abstraction and appropriate interfaces. An example of this approach is presented in Sect. 8, where a multilayered notebook ontology is exploited to adjust to the understanding level of each participant. Our system acts proactively by fetching from the services environment everything the user could need in each phase of the negotiation. For example, when a proposal contains concepts never used before, the agent searches and retrieves exact definitions, specifications and opinions from advisor services or other agents. A proactive agent may also access a strategy advisor to suggest the next step to be taken or even criticize the course of actions already taken.

Decision making is based to predefined preference and strategy specification and it is supported by the SOE. We did not focus on complicated decision making models, strategy and behavior specification algorithms and techniques, since our main goal is to create a generic environment for e-negotiations to be used as the infrastructure

for using any of these approaches. This is the abstraction offered by the SOE as a place for accommodating services implementing negotiation activities that can be dynamically exploited and orchestrated by our framework.

There is a need for consumers to elicit, before the negotiation, their preference over the negotiated object. In order to grasp and understand the vocabulary describing the object, negotiating participants access knowledge repositories containing the specification of the object in the form of ontologies. They will also access merchant services to acquire other important parameters for preference elicitation such as cost, shipping, etc. SWS are used for location and dynamic invocation of services along with mediator services to access the knowledge repositories and perform the semantic processing required.

The proposed SOA-based framework intends to enable negotiators to set up and customize their specific object acquisition and negotiation environment that best fits their needs. The framework is a combination of local and remote services orchestrated, as well as merchants and trusted mediators that are only used and bound together at the time of use in the particular negotiation process.

WSDL was used for interface description of SWS which are stored in a UDDI. Ontologies are written in OWL using as editing tool the protégé 3.3 and SWRL was selected as a scripting language for our rules incorporating the SWRL plug-in. Jess, the Java™ Expert System Shell and the SWRLjess plug-in were used to implement the proposed generic inference engine.

5 Modeling the negotiation domain

E-negotiation systems need a basic vocabulary which all participating should be able to easily understand and use. This vocabulary will help the common understanding of crucial concepts along with their interdependencies and will assist the conceptual design, offering the necessary abstraction for the development of generic e-negotiation engines. Several attempts were made to this direction each with a different point of view of the domain. Wurman et al. [2] focuses on auctions, whereas Lomuscio et al. [33], stresses on automation aspects.

We adopted the Montreal Taxonomy [34] which can be considered as an abstract collection of the most important concepts of e-negotiations and provides a framework that can be used for descriptive and prescriptive purposes. It covers aspects such as the roles, process (offer specification, offer submission, offer analysis, offer matching, offer allocation, offer acceptance), information revelation, and business model implementation of e-negotiations. To exploit the advantage of ontology over a simple taxonomy, that is the reasoning capability provided through a generic reasoning engine, we selected to proceed to the construction of a negotiation ontology.

Figure 3 presents part of the ontology that was produced by the ontoviz plug-in of protégé 3.3. This ontology models the negotiation domain, based on the above described architecture and Montreal Taxonomy principles. The relations between negotiation Protocol, Process and Strategy are depicted to justify the proposed layered architecture.

During the execution of a specific, complex negotiation scheme a process instantiation is produced depending on the exact course the negotiation will follow.

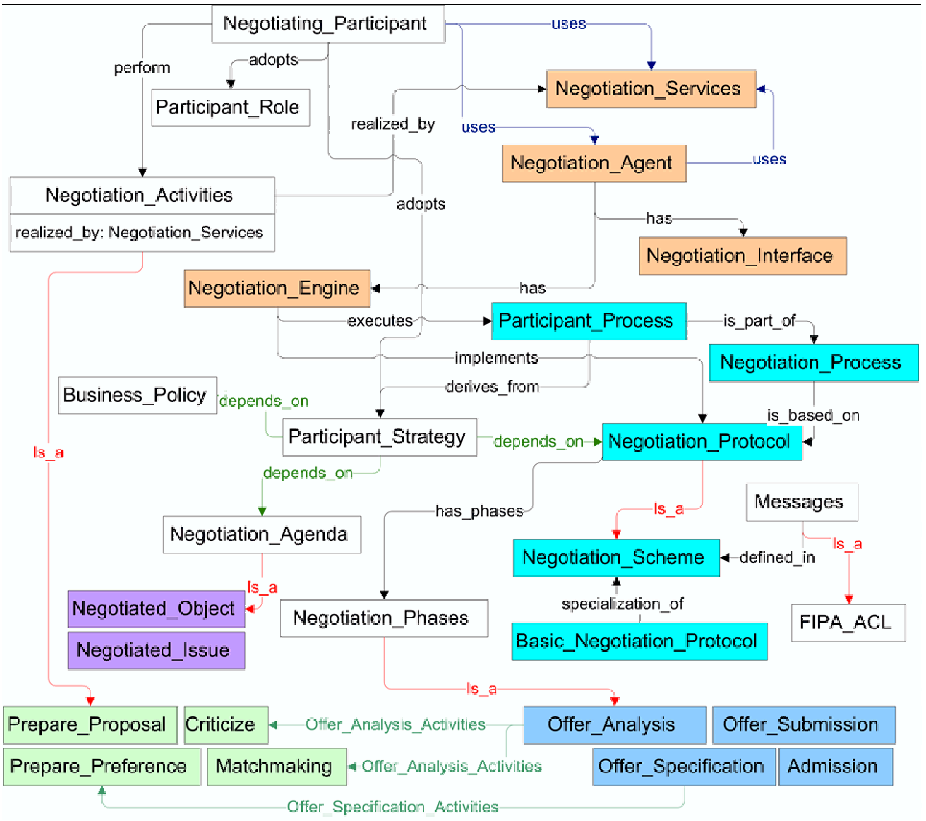

Fig. 3 Negotiation ontology produced in Protégé ontoviz

At every state of the negotiation the possible next steps to be taken for every participant are defined by the BNP. The participant chooses among the possible activities based on his strategy and he routes with his choices the negotiation scenario according to his interests. In this way he generates the Participants_Process that is executed by the Negotiation_Engine of the participant. The Participant_Process is_part_of the general Negotiation_Process which describes the whole negotiation process and derives_from Participant_Strategy which is private for each participant. The strategy adaptation depends_on the selected Negotiation_Protocol, the exact Negotiation_Agenda which can either be a Negotiated_Issue or a Negotiated_Object and the more abstract Business_Policy or objectives. Negotiating_Participant adopts a Participant_Role and performs Negotiation_Activities during a negotiation. These Negotiation_Activities are realized_by Negotiation_Services that are used by Negotiation_Agents and Negotiating_Participant. Agents have a Negotiation_Engine that implements the selected Negotiation_Protocol executing Participant_Process and a Negotiation_Interface for conducting negotiations. The term Negotiation_Interface is here used to mean the dynamically created interface, by each participant conforming to the Negotiation_Scheme, adopted for a specific negotiation.

Participants go through distinctive Negotiation_Phases and execute, specific well-defined Negotiation_Activities. For example, each participant in any kind of negotiation has to examine the received proposal and decide his next step based on the negotiation protocol and strategy. In our modeling approach we have identified the most important activities, which are tasks that each participant performs, and depicted them as concepts in our ontology in an attempt to capture their semantics. Among the concepts depicted we discriminate: Prepare Preference, Send, Receive, Prepare Proposal, Matchmaking-Critic, etc. The semantics of these activities must be clearly specified in order for these activities to be implemented by well defined Negotiation_Services, which participants can locally implement or remotely invoke in a SOE.

Another very important issue is the Negotiated-Object or Negotiated_Issue ontology. Negotiations depend a lot on the object that is under negotiation, since different approaches are needed depending on the Negotiation_Agenda. For example negotiation schemes for multi-attribute objects can become very complicated. An agreement on the negotiated object properties that should not be altered, such as minimum and maximum values of properties and flexible or don't care properties, has to be established in an early step. Participants should be allowed to express these constrains upon the concepts of the negotiated object ontology.

5.1 Modeling the negotiated object

Searching, locating and negotiating about an object presumes a good knowledge of the object's characteristics. For example when searching for a notebook the consumer will need to answer the question "What I need the notebook for?" or "What is the applicability of a notebook?" Modeling plays an important role in specifying object's multiple aspects. The object's ontology may include uncountable concepts to fully "characterize and capture the essence of being" according to Aristotle who first specified the term ontology. This is not efficient in computer science applications where only specific characteristics are selected depending on the objectives of modeling in the domain of interest. Except of capturing the basic characteristics of the object with our modeling approach we intent to address the following requirements:

(a) Application based end-user selection. This allows the consumer which is not familiar with the object's specific characteristics, to chose based on the applicability of it. For example, in the notebook case study the consumer is not expected to understand and use in his preference the CPU clock speed. Instead of it he will be able to select by defining the kind of applications that he intend to execute or even select in a fuzzy way his preference as for example Processing Power: Medium.
(b) User friendly presentation of Knowledge that is hidden inside the model of the object. Although Ontologies are formal specifications created to enable mainly machine-to-machine interoperability and communication a way to present this knowledge to users for preference elicitation and other activities must be defined.

Application based end-user selection of objects needs to be supported by a KB incorporating knowledge that answers the question "What this object does?". This

can be one important aspect of the object's ontology that will include an in depth specification of the applicability of the object. In many cases the application based selection may lead to similar products, i.e., palmtop instead of notebook, if it is used in the search for object process. This is discussed in more detail in Sect. 6.

For the notebook example the user must answer a number of questions such as: Will you play computer games? Will you watch movies or connect it to a High Definition Multimedia Interface (HDMI) device? Will you carry it a lot? Do you have any specific applications to deploy? To answer this last question the system can proactively access the specific application's ontology, retrieve the computational requirements and decide on the appropriate choice.

In order to provide user friendly representation we will need to extract the appropriate knowledge from our ontology formalization (OWL DL in our case) and present it in a friendly way as for example an HTML template. The appropriate knowledge for such a process must be incorporated in the object's ontology. For the creation of the object template each concept of the object may include except of a name and description other attributes such as unit and type_of_value. In [35] the type_of_value attribute can be numerical and non-numerical. Numerical can be Any_Number for attributes that can take any numerical value or a Range of Numbers. Non-Numerical can be a Set_of_Labels when there is an associated list of predefined textual values and there is no order among them or Ordered_Set_of_Labels defining an order for associated values. The Range_of_Numbers should also bare an attribute indicating that they are: Flat (all values are equally preferred), More Is Better (MIB indicates that higher values are more preferred) and Less Is Better (LIB for preferring lower values).

For the notebook's ontology we decided to connect it with a well known model in the domain keeping the philosophy of going from the general to more specific instances of knowledge in each domain. This structure has been proved extremely important when dealing with great mash of knowledge upon which automated reasoning and inference has to be applied. The relations between concepts in these multiple layers of abstraction provide the infrastructure for creating a more human-like-thinking approach on modeling knowledge and reason on it.

For the modeling of notebook example we adopted the FIPA device model [32] that captures the most important concepts of a generic electronic device. We created the notebook ontology in Protégé using as basis the FIPA device ontology adding other concepts in order to provide the application end user selection as well as the ability to visualize the ontology.

5.2 The need for negotiation process modeling and representation

After capturing the vocabulary of the negotiation domain and the knowledge concerning the negotiated object using ontologies there is a need to specify and represent the negotiation process. To better understand, manage and study the negotiation process we decided to incorporate in our framework the capability to visualize the workflow of negotiation schemes.

The negotiation process is produced by participants making decisions and taking courses of actions inside the constraints of the selected negotiation scheme. In order

for participants to understand and even modify an existing negotiation scheme there is a need for visualization of the different phases, constraints, messages and states that are involved. Visualization of the complete negotiation process is also very important in strategy specification. We incorporate the workflow technology for process visualization aiming at:

1. The understanding of the selected negotiation scheme before the negotiation is lunched and the definition of the strategy. The representation and visualization will depict only the main path of the negotiation scheme, avoiding messages such as cancel or not understood which may alter the resulting negotiation process. Participants are able to exchange the scheme representation and even negotiate using BNP about the negotiation scheme to be used.
2. The management and supervision of the negotiation process during the negotiation session. The negotiation process of the actual negotiation session is formulated step by step, as each participant makes decisions, based on his interests. To aid human participants in supervising the negotiation process a way to visualize this process is needed in semi-automated negotiations.
3. The study of the course of actions taken in the negotiation session after its termination. This is required in semi-automated or fully automated negotiations in order to improve adopted strategies and the behavior of the negotiation agent.

In all the above cases we are facing the same problem of generating a workflow from the specification of the selected and executed negotiation protocol. Negotiation protocols are conventionally modeled as centralized flows, specifying specific well defined steps for each participant. However, because of the exceptions and opportunities that arise in open environments and complex negotiation schemes, exact steps, relationships and activities cannot be preconfigured in detail [36]. Thus, flow-based models are inherently ill-suited in the face of evolving requirements and diversity management. Flows are not amenable to reuse via serialization, refinement or aggregation.

Another alternative for the specification of negotiation protocols is using rules. Bartolini et al. [15] has proposed a complete framework for implementing portable agent negotiations that comprises: (1) a taxonomy of declarative rules which can be used to capture a wide variety of negotiation mechanisms in a principled and well-structured way and (2) a simple interaction protocol, which is able to support any mechanism which can be captured using declarative rules. The rules, which are used for enforcing the negotiation mechanism, are organized into a taxonomy that includes: rules for participants' admission to negotiations, checking validity of proposals, protocol enforcement, updating negotiation status and informing participants, agreement formation and negotiation termination.

Rules have the advantage of easily describing complex systems, while statecharts result in the definition of large number of states and very complicated descriptions. For example, in the interaction protocols described in FIPA specification using UML notation, some common messages such as 'cancel' or 'not understood', are not included in the main path specification to avoid complicated hard to understand graphical representations.

On the other hand rules lack the graphical notation used by statecharts, which provide an easier and faster to be understood protocol specification. Graphical rep-

resentation of protocols is based on the fact that although many rules may actually be involved in negotiations, only a few of them are used at any given point in a negotiation process. We decided to combine rules and workflow diagrams because the negotiation process has to be complaint with the corresponding protocol but the protocol rules can be optimally managed and understood via a workflow.

In an approach similar to Bartolini we identify distinct phases in the negotiation process and capture rules for these phases. In the negotiation ontology we capture all important phases, messages and activities. We next use a workflow generation engine to produce a visualized workflow of the negotiation scheme before the negotiation session and the executed negotiation process during or after the negotiation. In order to implement our workflow engine we had to select a workflow representation formalism in order to use the concepts and semantics of this particular formalism.

The graphical modeling approach to be adapted for the specification and representation of negotiation processes should provide:

(a) User friendly visualization.
(b) Expressiveness required to support the majority of standard workflow patterns.
(c) The ability of serialization and publishing (e.g. XML compatible).
(d) Clear and well defined semantics.

To this direction Benyoucef [10] identifies four major alternatives for specifying protocols as processes: natural language, agent coordination languages, finite state machines and statecharts. He suggests statecharts as the best choice as they have good formal basis and can be serialized, visualized and executed. The FSM notation has been used in [37] and [38] to model the negotiation process domain. More specifically a very simple representation that tries to catch only basic states in an English auction negotiation mechanism is given in [38]. In [37] the states of the negotiation are also represented as FSM states and the transitions between states are associated with "send" and "receive" actions. The produced diagram is very complex and uses numerous states to catch the internal behaviour for each participant for bilateral bargaining negotiation scheme. In [39] an FSM ontology was presented as our first approach in modelling and representing a negotiation process.

Process Specification Language (PSL) provides another alternative for modeling processes and it was used for this purpose in [16] although the applicability is not proven. PSL gives a representation of manufacturing processes and has recently become a standard. It is based on semantics represented using KIF (Knowledge Interchange Format) and would be a convenient choice as it is already formed as an ontology with a core and multiple extensions. Nevertheless it is complicated, hard to understand and implement since it includes multiple concepts.

The need for displaying the various phases of the negotiation process and the graph-structure of the selected formalism led us to consider the XML Process Definition Language (XPDL) [40]. XPDL is a WfMC standard used for process definition with main focus on human processes and work distribution. It better fits in our case compared to BPEL that is a process execution language with main focus on web services interactions.

In XPDL a Pool acts as the container for flow objects (i.e. activities) and Sequence Flows (i.e. transitions) between them. The Sequence Flow may cross the boundaries

between Lanes of a Pool, but cannot cross the boundaries of a Pool. The interaction between Pools, as is the case of a negotiation between two participants, is shown using Message Flow. Lanes are used to subdivide a Pool. All the activities within a Lane may inherit one or more properties from the Lane. In our case, lanes have the characteristics of the negotiation phase the participant is into.

XPDL visualization results in graphs combining activity and interaction diagrams with all desired capabilities. It gives us the way with Pools and Lanes to clearly separate phases and within them give an exact description of activities and messages. Our workflow generation engine consist mainly of an XSLT document that transforms the XML (OWL, SWRL) and rule based protocol specification to the XPDL process specification. The applicability of the selected approach is further explained in the following case study.

6 A negotiation case study

To demonstrate the applicability of the proposed framework we describe in this section a negotiation case study for acquiring a notebook from a merchant. The example is not complex but it is appropriate to demonstrate the features of the proposed framework. A two-step bargaining scheme, which allows merchants to improve their proposal only once before the final decision, was adopted in the presented negotiation case study. In Fig. 4 the UML sequence diagram of the selected negotiation scheme is given.

Before the actual propose-counter_propose process, the client-Negotiation initiator sends his preference on an unusual characteristic of the notebook as for example the High Definition Multimedia Interface (HDMI). After the confirmation on this specific characteristic the client sends a detailed notebook specification asking for the providers to make their suggestions. The merchants answer with their prepositions; the client processes the prepositions and sends back to the merchants his choice. A new proposal is allowed to be sent by merchants. In our example negotiation scheme the 'not understood' and 'query' messages are allowed along with the advertise-push model, which is expressed via 'inform' messages. This means that any of these messages is allowed at any time, as is the case for the *cancel* message of the FIPA specification.

Figure 5 depicts the XPDL graphic representation of the example negotiation scheme. The internal process of a participant is shown and the interactions between participants are depicted as dotted arrows connecting the two pools. The figure describes the main course of the example scheme without representing exceptions or other allowed messages. The initial message that is send to merchants, i.e. the start message (1 in Fig. 5) contains information about the client, a reference about the negotiated object and the negotiation scheme to be used. After the confirmation to the start message (1a) which means that the merchants accept to launch negotiations with these characteristics, the client executes the admission process (2) which can include a number of activities, and enters the Prepare Preference activity (3) in the Preference Elicitation phase. The client then prepares and sends a *query_if* (4) containing the characteristic "HDMI interface" and receives a *confirm* or a *disconfirm*

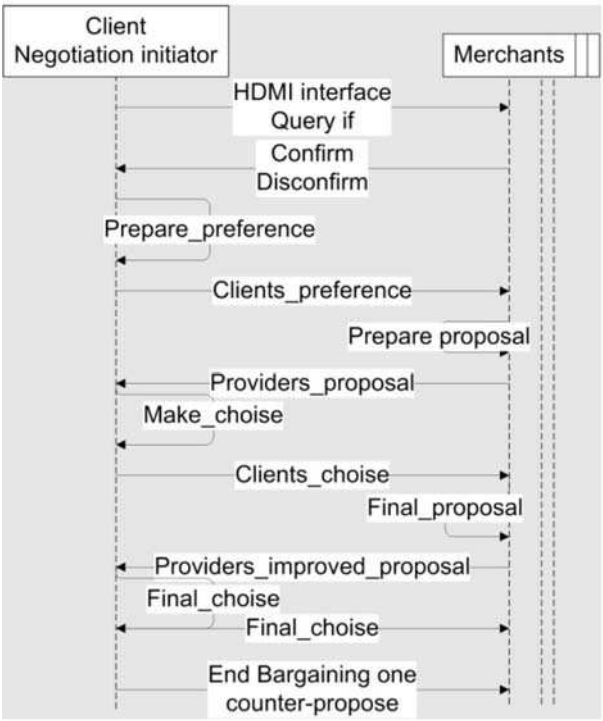

Fig. 4 The UML sequence diagram of the selected negotiation scheme

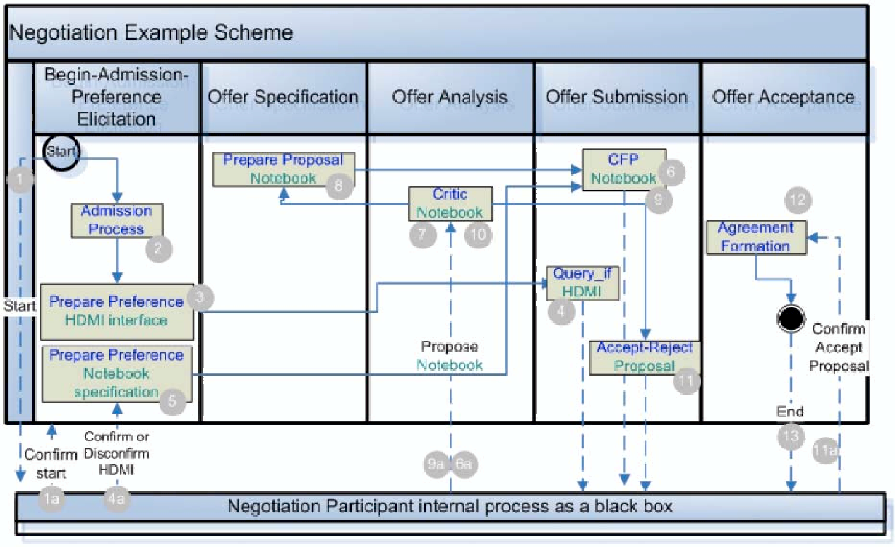

Fig. 5 Negotiation example scheme in XPDL notation

message (4a). The client decides to continue or terminate the negotiation depending on its strategy. In the former case he enters another Prepare Preference activity (5) for a detailed notebook specification and sends a *CFP* (6). The client receives a propose message (6a), criticizes the notebook specification in the Offer Analysis phase (7), Prepares a Proposal in the Offer specification phase (8) and sends a new *CFP* (9). The final *propose* message (9a) is received, the proposal is criticized (10) in order to be accepted or rejected (11). If a proposal is accepted the Agreement Formation sub-process (12) is launched before the end message (13) that terminates the negotiation. For the specific scheme specification the BNP messages that are not allowed are excluded and the restriction about the limited counter-proposals is added.

A participant when receives a *cfp* (6 and 9) or a *Propose* (6a and 9a) message, either enters the Offer Analysis negotiation state or sends an *Accept-Reject Proposal* (11) after entering the Offer Submission phase. This can be expressed using the rule that is given in informal notation below.

Rule *Receive* (*proposal or cfp*) → *Phase* (*Offer Analysis*) *or* (*Phase* (*Offer Submission*) *and Send* (*Reject or Accept*))

In the Offer Analysis phase the participant will perform a Match-making or Critic on the received proposal. These two activities describe a similar process with critic being more sophisticated and used when negotiating for multi-attribute objects. The corresponding rule is:

Rule *Phase* (*Offer Analysis*) → *Activity* (*Matchmaking*: *attribute*) *or Activity* (*Criticize*: *attributes*)

The participant will move to the offer specification (7) or offer submission (10) phase depending on the result of the above activities, which is based on the adopted strategy. In the Offer Specification phase and if matchmaking fails to find an exact match an alternative proposal will be produced or a *Reject* message will be send. This is described with the following rules:

Rule *Receive* (*CFP*) *and Matchmaking* (*true*) → *Phase* (*Offer Specification*)

Rule *Receive* (*CFP*) *and Matchmaking* (*false*) → *Phase* (*Offer Specification*) *and Activity* (*Prepare_Alternative_Proposal*)

Rule *Receive* (*CFP*) *and Matchmaking* (*false*) → *Phase* (*Offer Submission*) *and Send* (*Reject*)

At any point of the negotiation, participants can also issue a *query if*, *query ref* and *not understood* message that would obviously alter the final graphical representation. The above choices of the participant are specified in the BNP. If the *not understood* is not allowed and excluded from the scheme specification then the rules from the BNP that contain it are cancelled. In this way we move from the BNP to specific schemes.

The BNP provides the general basis for negotiations and specific schemes further specialize the negotiation protocol. These schemes specify also the environment and

the alternatives that participants may follow at each step of the process. The decision at each step will be done by the participant that may be a human or an autonomous agent in an automated negotiation process. The decision making will create the executed negotiation process out of the negotiation protocol that is not directly executable in an inference engine as it contains the above "or" rules that provide the options at each step.

Decision is based on the strategy that is preconfigured for specific scheme and derives from the more abstract and general policies. The strategy is checked for consistency according to BNP and selected scheme and drives the decision making in a rule based declarative approach. Usually the user has to manually specify the strategy according to the selected negotiation scheme. There is a need to study the scheme and design his course of actions precisely for the whole negotiation at each decision point. The result of such a process is a complicated rule-based specification that incorporates both the scheme and the strategy.

A more dynamic approach for producing the strategy from abstract policies should be implemented towards automation. Goal-oriented inference better fits automated strategy specification based on selected scheme and higher policies. Norm-oriented, forward inference is used to infer our available courses of action. For the strategy, which is a future work, a backward reasoning engine, e.g. Algernon, is more appropriate for goal-based reasoning. Two inference engines are required:

(a) One inferring the permitted next steps of the protocol, applying forward reasoning and
(b) One inferring the actual next step inside the protocol constraints, applying backward reasoning from goals and higher layer policies.

In our example scheme the merchant receives a *query_if* with the HDMI attribute. He goes to the offer_analysis Process_Phase and performs the matchmaking activity to ensure that he can provide it. In SWRL formal notation:

- Receive(query_if) ∧ Negotiated_Object(?y) ∧ multiattribute(?y, true) ∧ Negotiation _Engine(?z) → Process_Phase(offer_analysis) ∧ matchmaking(?z, ?y)

Depending on the result of the matchmaking activity the merchant will confirm or disconfirm the query_if about the specific Negotiated_Object:

- Receive(query_if) ∧ matchmaking_result(?y, true) ∧ Negotiated_Object(?y) ∧ Negotiation_Engine(?z) → Process_Phase(offer_submission) ∧ send(?y, confirm)
- Receive(query_if) ∧ matchmaking_result(?y, false) ∧ Negotiated_Object(?y) ∧ Negotiation_Engine(?z) → Process_Phase(offer_submission) ∧ send (?y, disconfirm)

This is the protocol for the *query_if* case. We assume that the initiator sends the *query_if* to decide according to the answer, if he is willing to begin the negotiation. It is just like the case of a human entering a shop and querying for a specific unusual characteristic on an object. This is part of his strategy derived from his preference policy: "notebook must include HDMI interface" combined with a conclusion or advice from the preference elicitation phase stating: "HDMI interface on notebooks is rare".

After receiving the *confirm* message the KB infers that the characteristic is included in the negotiated object and the client proceeds to prepare his preference in the preference elicitation phase.

- Send(query_if) ∧ Receive(confirm) ∧ Concept(?x) ∧ Negotiated_Object(?y) → has(?y, ?x)
- Send(query_if) ∧ Receive(confirm) ∧ Negotiated_Object(?y) ∧ multiattribute(?y, true) → Process_Phase(preference_elicitation) ∧ Prepare_Proposal(?y)

In case of receiving a disconfirm message to his query_if message the clients KB infers that the characteristic is not included in the negotiated object:

- Send(query_if) ∧ Receive(disconfirm) ∧ Concept(?x) ∧ Negotiated_Object(?y) → not_have(?y, ?x)

The abstract policy of the client states that: "a notebook in not accepted without an HDMI interface" that the client marked as a must_have in the preference elicitation phase:

- Concept(?x) ∧ Negotiated_Object(?y) ∧ must_have(?y, ?x) ∧ not_have(?y, ?x) → not_accepted(?y)

In this case the strategy specifies two alternatives based on the clients preference and weather the specific concept is a must_have or a should_have for the negotiated object:

- Concept(?x) ∧ Negotiated_Object(?y) ∧ must_have(?y, ?x) ∧ not_have(?y, ?x) → Send(reject)
- Concept(?x) ∧ Negotiated_Object(?y) ∧ should_have(?y, ?x) ∧ not_have(?y, ?x) → Prepare_Alternative_Proposal(?y)

The backward reasoning engine that executes the strategy searches among the protocol provided alternatives, the one that leads to the desired activity or phase in the negotiation process. For the Send(reject) it enters Process_Phase(offer_ submission) as dictated by the protocol in order to send the reject message. In the same way for the Prepare_Alternative_Proposal it enters the Process_Phase(preference_elicitation).

7 Getting to negotiations

In this section we will describe how the semantic environment and technologies like KBs and SOA can be utilized to aid and automate processes before reaching the actual negotiation. In order to get to the negotiation phase the consumer has to:

(a) Locate the preferred object and the e-shop services that provide it.
(b) Elicit his preference for all issues involved in acquiring the object: customization for multi-attribute objects, cost, shipping etc.
(c) Select a negotiation scheme, which should be supported by the contacted e-shops, and a strategy.

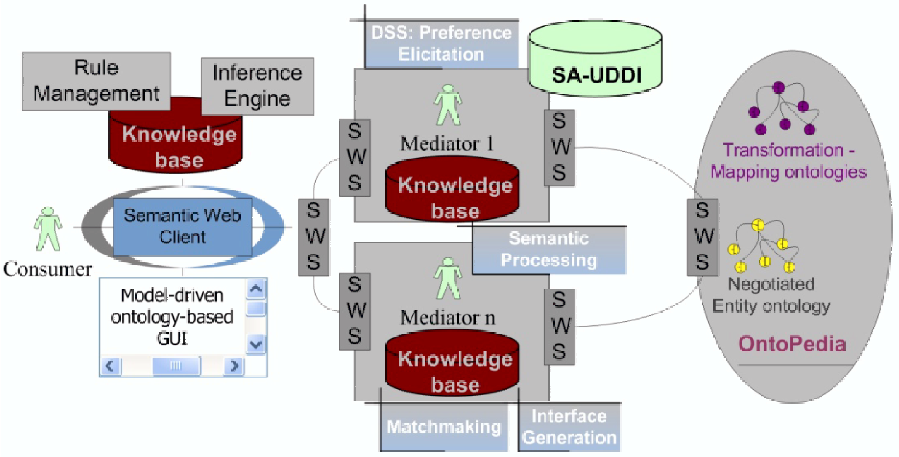

Fig. 6 Location and preference elicitation framework

Preference elicitation can be done before locating real objects and supplying services as it can make search criteria more specific. On the other hand, important parameters that will be involved in preference elicitation such as cost, can only be acquired in the required detail after contacting supplying services and actual products. This means that the preference elicitation activity will begin for the consumer to better understand the negotiated object and will continue after locating services and products for the final configuration and understanding.

For preference elicitation the consumer will need to locate the object ontology that will provide the object's essence. He will contact and query a service with access to a common, widely adopted, distributed knowledge repository. We call this repository Ontology Encyclopedia (OntoPedia) as it will contain all common, important, well defined, accessible knowledge. A query can include some keywords based on end-users' knowledge and the interfacing service will search for ontology concepts to match. Search may be restricted only to head concepts to accelerate the search process and provide accurate results. This is why "notebook" can be included as a concept in numerous ontologies. Techniques from ontology alignment domain can be used providing suggestions on concepts' similarity in the form of similarity files.

For the search process in the OntoPedia we selected the architecture depicted in Fig. 6 that is based on a Mediator SWS between the Knowledge Repository and the Consumers. Our choice aims at reliving the ontology and semantics unaware end-user which will remain to a well known keyword-based search. Mediator Services will concentrate the bulk of intelligence needed in our scenario resolving Knowledge Management issues. Semantic Web and related technologies are expected to reach a point of maturity that will make it easier to implement standardized parts of Knowledge Management locally to end user equipment. Until then Mediators will access such repositories formulating queries based on users keywords, after Semantic processing that will reduce required processing in OntoPedia repositories. The Consumer will receive again through SWS the answer containing the ontology or ontology reference specifying the object. The process of parsing the ontology for graphical

presentation and final preference elicitation follows. Again this process can be executed by the Mediators that will produce an appropriate user interface for preference elicitation or hosted locally in the end-user device.

It should be noted that different implementation scenarios exist regarding distribution of functionality in users or SWSs to better exploit the semantic Web advantages. It is a matter of choice and architecture what functionalities will run locally and this decision mainly depends on the tools that will evolve to exploit the semantic Web. It is expected that functionalities described above will soon be part of the next generation semantic web browsers. The user's search criteria will be formalized using SWRL that can describe any kind of restrictions upon ontology concepts.

In the user friendly graphical representation of the object, the user is allowed to create his customized specification. In our environment this can be accomplished in two layers of abstraction: the application-based selection or a more domain-technology depended, based on users' knowledge in the domain. This is due to the modeling of the objects presented in Sect. 5.1 and the knowledge included in the object ontology. By providing the application-based layer of abstraction to end-users selection, we reach an upper level of Goal-based policy for object selection [41]. This permits greater flexibility and frees end-users from having to know low-level details, at the cost of requiring sophisticated modeling. Our KB will drive decision inside the restrictions of the specified application level requirements.

For other issues like pricing and shipping which are very important in the final preference elicitation, the need for contacting e-shops and merchants through supplying services is obvious. Such services will be located in a Semantically Annotated UDDI (SA-UDDI). The ontology reference with a "buy" or "acquire" keyword would be enough to locate all available services exploiting the advantage of semantic based search versus simple keyword based search. Upon locating such services their dynamically invoke-able interface is retrieved and the consumer's agent is able to use these services. From this point and forth the negotiation process with all available alternatives based on the merchants interface can be launched.

8 The graphical user interface

Even though the GUI plays a very important role in e-negotiations it has not drawn the attention of researchers. In an e-negotiations environment where declarative rule based knowledge formalization is adopted along with extensive use of SOA with the use of SWS a model driven ontology-based GUI may further increase the effectiveness of the environment. Among the fundamental requirements from the GUI we discriminate the followings:

(a) Dynamic creation from parsing and visualizing the knowledge embedded in ontologies.
(b) An environment to edit ontologies and rules based on their concepts, to adjust and configure the system. Within this environment semi-automated ontology alignment will also take place.
(c) Users query answering and justification for the course of actions and automated decisions of the system.

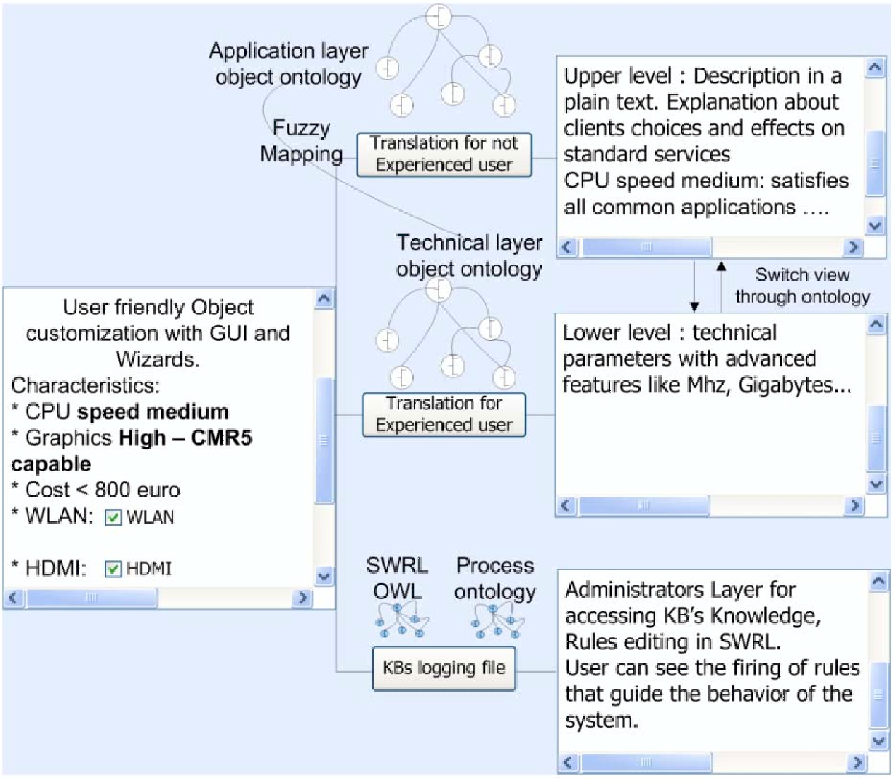

Fig. 7 The proposed GUI

(d) Workflow management and visualization of the complete preference elicitation and negotiation process.

We propose and implement an ontology-based dynamic GUI produced by ontology parsing and KB activities visualization. Furthermore we use multiple levels of understanding with dynamic mappings between them utilizing the retrieved object ontology. In particular our GUI is based on the principles of semantic oriented systems that can dynamically evolve and extended.

In Fig. 7 an abstract representation of the proposed GUI is presented. On the left hand is the interface that is used by the user to specify his preference with simple clicks and ticks. On the right, three accessible layers that can be used to aid the preference elicitation process and manage the system are provided. On the first layer the detailed description of each concept as produced by the parsing of the ontology is provided. Each concept in an ontology has a standard attribute called description that contains plain text for human understanding. If this description is not enough for the user to completely understand the meaning, SWS and SOA are used to locate further information. Further explanations for effects of user preference on usual applications will appear to drive decision. On the second layer a technical description of the pref-

erences is depicted. This is the technical notebook specification which the merchant will accept and use.

The third layer accesses another level of the system providing tools for administrating and verifying the KB. A logging file is provided that captures rules firing and results produced at each step of the preference elicitation and negotiation process. By inspecting this file a naive user can understand the operations in the KB and adjust it assuming the existence of a user friendly tool.

9 Framework implementation details

Our KB containing the required ontologies for the domain modelling is depicted in Fig. 8. The prototype KB implementation is based on the Protégé environment. Protégé's GUI was used for building our OWL DL formalized ontologies and SWRL rules through the SWRL plug-in [26]. OWL DL is well suited to represent "structured" knowledge by classes, properties and taxonomies, supporting automatic classification and class recognition of instances. SWRL is better suited to express "deductive" knowledge by rules composed of atoms.

The knowledge is fed to an inference engine which in our case is JESS through the SWRL-JESS plug-in and API. JESS is a forward chaining engine based on RETE algorithm that makes it much faster than a simple set of cascading *if.. then* statements in a loop. JESS provides a rich and flexible environment for creating rule-based systems. Since it is written in Java it provides platform portability, extensibility and easy integration with other Java code or applications. The rules of JESS allow one to build systems that reason about knowledge that is expressed as facts. Facts are a collection of knowledge nuggets and this collection is known as the working memory. Rules can only react to additions, deletions, and changes to working memory.

An example rule expressed in SWRL inferring the need to perform the criticize activity in the case of a multiattribute negotiated object follows:

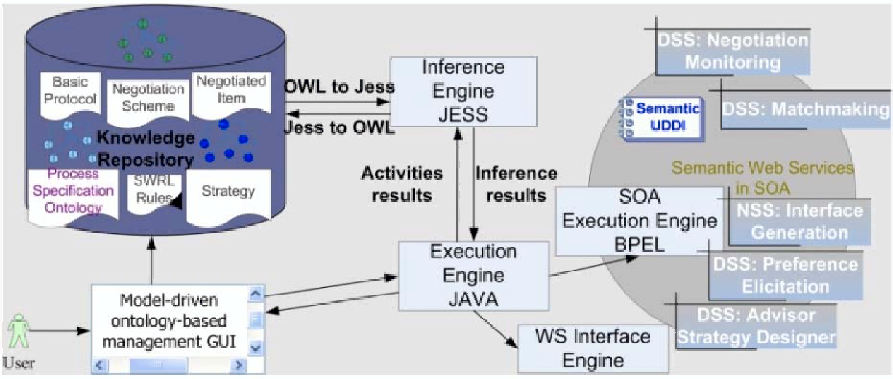

Fig. 8 Knowledge and Service based framework implementation details

$$\text{Process\_Phase(offer\_analysis)} \wedge \text{Negotiated\_Object(?y)}$$
$$\wedge \text{multiattribute(?y, true)} \wedge \text{Negotiation\_Engine(?z)}$$
$$\rightarrow \text{criticize(?z, ?y)} \wedge \text{Process\_Phase(offer\_specification)}$$

The following is the inferred result after importing the rule into JESS working memory and running the JESS engine:

(assert(Process_Phase(name offer_specification)))

[(assert(criticize Negotiation_Engine_1 Negotiated_Object_1))]

The asserted facts fire the transition of the system to its new state and trigger the execution engine instance to perform a criticize activity on the negotiated object instance.

JESS, in order to reason about things that happen outside of its working memory, uses Shadow facts that act as bridges. Shadow facts are backed by a Plain Old Java Object (POJO) that holds the same data as the fact. Static shadow facts will have their slots updated from the backed Java object during a reset. These facts can be used in rule-based programs that reason about static value objects whose properties do not change over time or change only occasionally or at well-defined times. Dynamic shadow facts on the other hand use property change listeners to dynamically update their slot values as the Java object changes. These facts are used in our dynamic fast evolving e-negotiation system as it allows our shadow facts to stay continuously up to date.

JESS inference modifies the knowledge inside the KB and outputs activities that need to be performed by our engine in the processes of locating, understanding, preference elicitation and negotiation for an object. Our execution engine is developed in JAVA using the ability of JESS to be integrated with JAVA programs using shadow facts. The execution engine contains well defined activities which are invoked with specific parameters and return their results to the inference engine.

The activities implemented can be hardcoded in procedural programs or dynamically invoked in a SOA environment. Multiple services implementing important activities can be built in a distributed fashion and be used by our system through dynamic location and invocation exploiting SWS technologies. This will be done through searching on a semantic UDDI, retrieving OWL-S like specifications and comparing to the requested one. For more complicated activities more than one service can be used in a BPEL orchestration tool. The system is accessible through our model driven ontology based GUI to the user that can be either a knowledge engineer, a domain expert or a simple consumer.

For the SOA exploitation and SWS invocation the Eclipse java based API was utilized. Default code is appended inside each activity when its skeleton is created from the ontology specification which searches the SWS environment for specific services. When our system is deployed, engineers can hardcode each activity's behavior to implement it locally. The SWS will be implemented by numerous service providers. For our demonstration a number of SWS were implemented and dynamically invoked by the WS client that was created from their WSDL descriptions.

One important issue is the translation of the OWL-based document which describes the specific negotiation scheme, to a WSDL document which specifies the WS created to handle the negotiation for each party. XSLT is used to translate and map between XML-based documents.

In an argumentation based negotiation where a participant needs to give an advice or share an observation such as: "A notebook with independent graphic card memory, not shared, usually means an expensive (<800 euro) notebook". How can this knowledge be dynamically added to the KB when an advisor service suggests it? Pragmatics about reputation retrieved from other users or trusted services must also be dynamically added to the knowledge base. With plugging in new knowledge, dynamic improvement and adjustability of the KB is enabled. It may have the form of an advice aiming the end-user or the form of rules that will alter the behavior and decision making of our KB. There can also be a rule that will trigger the execution of a specific activity performed as a procedural attachment in our system.

If the knowledge is not directly connected with an activity or a communication act, it can be appended if the appropriate semantics and concepts are included in the target KB. This means that the dynamic plug-and-play can be incorporated in the KB if the appended rules are formed only with well known concepts. This is why the advisor suggesting the rule and the end-user's KB should share the same conceptualization and formalization in their ontology-based modeling. In our case both of them have to base their KB in the notebook ontology as an extension to the FIPA Device ontology and formalize their rules in SWRL. For this case we have implemented a "copy-paste" procedure in Java for adding the new knowledge to JESS. In this process we must be very careful about conflicts that may arise and lead to an unable to function KB. Reasoning is the process that can locate such conflicts.

JESS and the execution engine play a very important role in the process of updating the activities in our KB. Updating or adding an activity means that we provide the code for the execution engine hardcoded or as an orchestration of simple, existing services. It is a patch adding in software engineering language with the difference that the knowledge and behavior added can easily be inspected and understood by KB's human administrators. The whole rule-based declarative programming approach aims and achieves this fundamental goal compared to other software engineering approaches. It brings machine-executable knowledge representation and formalization closer to humans than machines, making it accessible by non software experts.

In our attempt to capture preferences based on the applicability of negotiated objects we have adopted a fuzzy-based approach. The main reason is that these preferences in real life scenarios tend to be vague and uncertain with no crisp values. The fuzzy approach was also used in evaluating the acceptance of the incoming offers. When using conventional mathematical methodology the evaluation process is becoming more complicated as the number of negotiation issues grows.

On the implementation and formalization point of view, OWL DL used in our approach becomes less suitable in domains where concepts to be represented do not have a precise definition. Application based preference, elicitation and the matchmaking between multi-attribute object specifications for proposal validation calls for another more fuzzy approach. In this paper we consider a fuzzy extension of OWL-DL.

The main feature of fuzzy SHOIN(D) is that it allows us to represent and reason about vague concepts [42].

In application based preference elicitation the user selects the application and a fuzzy mapping algorithm translates it to specific technical requirements. In proposal validation fuzziness also provides an excellent solution in multi-attribute object matchmaking. For the proposal validation, the defuzzification can give as the winner through the matching factor of the proposal.

Since we have already defined these concepts and correctly modeled our fuzzy functions we can use fuzzy rules that the end user can easily understand and learn. For example we can add into our KB the following fuzzy rule in informal notation:

If graphic_chip_memory is High and CPU_speed is Medium then notebook_gamming_ability is High

The ability to write such rules as an advice in argumentation based e-negotiation systems or in a process to dynamically update our KB is very important.

Facts and rules in JESS standard implementation cannot capture any uncertainty or imprecision that may be present in the domain that is being modeled. This is why we have incorporated to JESS an extension which allows some forms of uncertainty to be captured and represented using fuzzy sets and fuzzy reasoning. NRC FuzzyJ Toolkit, a Java API that allows one to express fuzzy concepts using fuzzy variables, fuzzy values and fuzzy rules is exploited along with a Java API called FuzzyJess that integrates the FuzzyJ Toolkit and JESS.

For the representation of Fuzzy concepts, we use fuzzy variables, fuzzy sets and fuzzy values. Fuzzy variables consists of a name (Processing_power), its units (MHz), a range (for example, from 1 to 5), and a set of fuzzy terms that can be used to describe specific fuzzy concepts for this variable. The fuzzy terms are defined using a name such as *High*, *Medium* and *Low*, together with a fuzzy set that identifies the degree of membership of the term over the range of the fuzzy variable. The fuzzy variable terms along with the operators *and*, *or* and *not* and the left and right parentheses provide a simple grammar towards fuzzy linguistic expressions.

Define globals to hold our FuzzyVariables:

(defglobal ?*FuzzyGamingPowervar* = new FuzzyVariable “Gaming_power” 1.0 5.0 “MHz”))

Terms for the gaming power Fuzzy Variable:

(?*FuzzyGamingPowervar * addTerm “Low” (new RFuzzySet 1.0 2.5 ?rlf))
(?*FuzzyGamingPowervar * addTerm “Medium” (new TriangleFuzzySet 1.5 2.5 3.5))
(?*FuzzyGamingPowervar * addTerm “High” (new LFuzzySet 2.5 3.5 ?llf))

Rule gaming_High:

(defrule gaming-High “If graphic_chip_memory is High and CPU_speed is Medium then notebooks gamming_ability is High”
(graphic_chip_memory ?g&:(**fuzzy-match** ?g “High”))
(CPU_speed ?c&:(**fuzzy-match** ?c “Medium”))
=> (assert (gamming_ability (new FuzzyValue ?* FuzzyGamingPowervar * “High”))))

The Semantic Web Rules Language (SWRL) can be seen as the union of two decidable fragments of First Order Logic: OWL-DL and Datalog. However, SWRL’s expressive power costs in its decidability through sound and complete reasoning.

Decidability is a very important requirement and the luck to fulfill it can reduce the applicability of the SWRL approach. This is the reason for selecting parts of SWRL to use that remain decidable although they have limited expressive power.

Our proposed approach exploits structures and technologies which an end-user will have available to explore the Semantic Web. Things like web services, inference engines on easy to access rule bases and dynamic use of SOA environments will be embodied to future end-user browsers. We need to make software agents cleverer and the first step is to find an easy way to formalize our knowledge to be used by software agents as an expert system does. Protégé-like ontology and knowledge management can be our guide towards more end-user friendly environments to harvest the semantic web.

10 Conclusions and future work

In this paper a framework for efficient development as well as automation of e-negotiation systems is presented. A layered architecture is proposed to capture the negotiation domain main concepts: interaction protocol, negotiation protocol and negotiation strategy. Standards and technologies from the WSs' and agents' domains are exploited to provide the communication infrastructure required for e-negotiations. BNP captures the basic functionality involved in any negotiation scenario and it is further specialized to specific negotiation schemes. Rules are used for the specification of the BNP and the workflow management approach was used for scheme representation and visualization.

The proposed proactive NSS framework exploits Ontologies, SOA, SWS, software agent platform, and Knowledge-Bases to facilitate the: (1) creation of dynamically adapted protocol specifications; (2) negotiation process visualization and management; (3) automated deployment of the required negotiation interfaces and exploitation of the negotiation services; (4) negotiated object modelling and visualization towards application based selection; and (5) appropriate environment in order to understand and locate the object to be negotiated. The main focus and novelty for solving the above issues is the knowledge representation, formalization and management approach in all phases of the negotiation process.

A negotiation ontology captures key concepts to enable the description of complicated negotiation schemes. This ontology-based vocabulary that adopts the Montreal Taxonomy will be used for the BNP and specific scheme specification. A number of different approaches for describing the negotiation process are presented and our choice on using XPDL is explained. An example negotiation scheme is presented to demonstrate the applicability of the proposed framework.

For the modeling of the negotiated object an ontology is proposed to capture all important parameters of the object. The need for extending the negotiation object's ontology towards application based selection and appropriate presentation and visualization is stressed. The proposed extensions include concepts to provide the desired functionality and will be exploited by a multi-level view GUI that is also presented for the preference elicitation process. The GUI will also provide the appropriate interface for knowledge management to adjust systems functionality.

We are currently working to address problems that have appeared in declarative protocol specification using OWL and SWRL. Management of rules and consistency checking in the protocol specification process is hard and time consuming. Also the need for adding a reasoner to our prototype implementation is obvious in order to enforce the applicability of our approach in real world scenarios.

XSLT transformations and automatic generation of WSs is an essential issue in our framework as it provides the appropriate communication infrastructure. An effort to incorporate OWL-S in WSs description replacing WSDL is in progress. XSLT will also be used to capture the appropriate knowledge from OWL negotiation schemes specifications and create the OWL-S described negotiation services.

The strategy semi-automated specification, from higher level policies is also a feature we have to add to our system. We are currently working on making the two inference engines, the backward and the forward one to work "back to back".

Acknowledgements This paper is part of the 03ED375 research project, implemented within the framework of the "Reinforcement Programme of Human Research Manpower" (PENED) and co-financed by National and Community Funds (20% from the Greek Ministry of Development-General Secretariat of Research and Technology and 80% from E.U.-European Social Fund).

References

1. Hurwitz Report. (2000). *Negotiated trade: the next frontier for B2B e-commerce* (Technical Report). The Hurwitz Group.
2. Wurman, P., Wellman, M., & Walsh, W. (2001). A parameterization of the Auction Design Space. *Games and Economic Behaviour*, *35*, 304–338.
3. Bichler, M., Kersten, G., & Strecker, S. (2003). Towards a structured design of electronic negotiations. *Group Decision and Negotiation*, *12*, 311–335.
4. Kersten, G. E., & Lai, H. (2006). *Negotiation support and e-negotiation systems* (Technical report, InterNeg Research Report 13).
5. Wurman, P. R., Wellman, M. P., & Walsh, W. E. (1998). The Michigan Internet AuctionBot: a configurable auction server for human and software agents. In *Agents'98* (pp. 301–308). New York: ACM Press.
6. Benyoucef, M., Keller, R. K., Lamouroux, S., Robert, J., & Trussart, V. (2000). Towards a generic e-negotiation platform. In *Proceedings of the sixth international conference on re-technologies for information systems*, Zurich, Switzerland.
7. Gimpel, H., Mäkiö, J., & Weinhardt, C. (2005). Multi-attribute double auctions in financial trading. In *7th international IEEE conference on e-commerce technology*, Munich, Germany.
8. Kim, J. B., & Segev, A. (2005). A web services-enabled marketplace architecture for negotiation process management. *Journal on Decision Support Systems*, *40*(1), 71–87. Special Issue on Web Services and Process Management.
9. Chiu, D. K. W., Cheung, S. C., Hung, P. C. K., Chiu, S. Y. Y., & Chung, A. K. K. (2005). Developing e-negotiation support with a meta-modeling approach in a web services environment. *International Journal on Decision Support Systems*, *40*(1), 51–69. Special Issue on Web Services and Process Management.
10. Benyoucef, M., & Rinderle, S. (2006). Modeling e-negotiation processes for a service oriented architecture. *Group Decision and Negotiation*, *15*, 449–467.
11. Kersten, G., Law, K. P., & Strecker, S. (2004). *A software platform for multi-protocol e-negotiations* (InterNeg Research Report 04/04). April 2004. Available at http://interneg.org.
12. Tung, B., & Gachet, A. (2006). Web services for negotiation and bargaining in electronic markets: design requirements, proof-of-concepts, and potential applications to e-procurement. *Group Decision and Negotiation*, *15*, 469–490.
13. Internet Negotiation Group. http://interneg.org/interneg/tools/inss/.

14. Buttner, R. (2006). A classification structure for automated negotiations. In *Proceedings of the 2006 IEEE/WIC/ACM international conference on web intelligence and intelligent agent technology*.
15. Bartolini, C., Preist, C., & Jennings, N. (2003). Architecting for reuse: a software framework for automated negotiation. In J. Odell & G. Weiss (Eds.), *LNCS: Vol. 2585. Agent-oriented software engineering III, F*. Berlin: Springer.
16. Tamma, V., Wooldridge, M., & Dickinson, I. (2002). An ontology based approach to automated negotiation. In *Proceedings AMEC'02* (pp. 219–237). Berlin: Springer.
17. Badica, C., Ganzha, M., & Paprzycki (2006). Rule-based automated price negotiation: overview and experiment. In *ICAISC* (pp. 1050–1059).
18. Simon, C., & Rebstock, M. (2004). Integration of multi-attributed negotiations within business processes. *Business Process Management*, *2004*, 148–162.
19. Weigand, H., & Schoop, A. (2003). B2B negotiation support: the need for a communication perspective. *Group Decision and Negotiation*, *12*, 3–29.
20. Kowalczyk, R., & Bui, V. (2001). On constraint-based reasoning in e-negotiation agents. *Dignum and Cortes*, *26*, 31–46.
21. Kowalczyk, R. (2002). Fuzzy e-negotiation agents. *Journal of Soft Computing*, *6*(5), 337–347. Special Issue on Fuzzy Logic and the Internet.
22. Wang, Y., & Ren, K. L. (2005). Towards autonomous and automatic evaluation and negotiation in agent-mediated Internet marketplaces. *Electronic Commerce Research*, *5*, 343–365.
23. Rahwan, I., Ramchurn, S. D., Jennings, N. R., McBurney, P., Parson, S., & Sonenberg, L. (2004). Argumentation-based negotiation. *The Knowledge Engineering Review*, *18*(4), 343–375.
24. Amgoud, L., Dimopoulos, Y., & Moraitis, P. (2007). A general framework for argumentation-based negotiation. *ArgMAS*, *2007*, 1–17.
25. Fatima, S. S., Wooldridge, M., & Jennings, N. R. (2006). Multi-issue negotiation with deadlines. *Journal of Artificial Intelligent Research (JAIR)*, *27*, 381–417.
26. Pu, P., Chen, L., & Kumar, P. (2008). Evaluating product search and recommender systems for E-commerce environments. *Electronic Commerce Research*, *8*(1–2), 1–27.
27. Skylogiannis, T., Antoniou, G., Bassiliades, N., Governatori, G., & Bikakis, A. (2007). DR-NEGOTIATE—A system for automated agent negotiation with defeasible logic-based strategies. *Data and Knowledge Engineering*, *63*(2), 362–380.
28. Desai, N. J., Mallya, A. U., Chopra, A. K., & Singh, M. P. (2005). OWL-P: a methodology for business process modeling and enactment. In *Proceedings of the AAMAS 2005 workshop on agent oriented information systems*.
29. W3C, OWL Web Ontology Language Overview. http://www.w3.org/TR/owl-features. Accessed 25 November 2008.
30. W3C, SWRL: A Semantic Web Rule Language Combining OWL and RuleML. http://www.w3.org/Submission/2004/SUBMSWRL-20040521. Accessed 25 November 2008.
31. Berners-Lee, T., Fischetti, M. (1999). *Weaving the Web: the original design and ultimate destiny of the World Wide Web by its inventor* (1st Ed.). San Francisco: Harper Collins.
32. FIPA The Foundation for Physical Intelligent Agents. http://www.fipa.org/repository/index.html. Accessed 25 November 2008.
33. Lomuscio, A. R., Wooldridge, M., & Jennings, N. (2002). A classification scheme for negotiation in electronic commerce. In *LNCS: Vol. 1991. Agent mediated electronic commerce: the European AgentLink perspective* (pp. 19–33). Berlin: Springer.
34. Strobel, M., & Weinhardt, C. (2003). The Montreal taxonomy for electronic negotiations. *Group Decision and Negotiation*, *12*, 143–164.
35. Cerquides, J., López-Sánchez, M., Reyes-Moro, A., & Rodríguez-Aguilar, J. A. (2007). Enabling assisted strategic negotiations in actual-world procurement scenarios. *Electronic Commerce Research*, *7*, 189–220.
36. Desai, N. J., Mallya, A. U., Chopra, A. K., & Singh, M. P. (2005). Interaction protocols as design abstractions for business processes. *Transactions on Software Engineering*, *31*(12), 1015–1027.
37. Su, S., Huang, C., & Hammer, J. (2000). A replicable web-based negotiation server for e-commerce. In *Proceedings of 33rd international conference on system sciences*, Hawaii.
38. Kumar, M., & Feldman, S. I. (1998). *Business negotiations on the Internet* (Technical Report). IBM Research Division, New York.
39. Koumoutsos, G., & Thramboulidis, K. (2007). Towards a framework for automated e-negotiations. In *International conference on e-business (ICE-B)*, Barcelona, Spain.
40. WfMC—XML Process Definition Language. http://www.wfmc.org/standards/xpdl.htm. Accessed 25 November 2008.

41. Kephart, J. O., & Das, R. (2007). Achieving self-management via utility functions. *IEEE Internet Computing*, *11*(1), 40–48.
42. Ragone, A., Straccia, U., Noia, T., Sciascio, E., & Donini, F. (2008). Towards a fuzzy logic for automated multi-issue negotiation. In *FoIKS* (pp. 381–396).

Giannis Koumoutsos received his diploma in 1992 and he is currently a PhD student in Electrical and Computer Engineering Department in University of Patras, Greece. He works mainly in the field of electronic negotiations, semantic web, expert systems, workflow technologies, knowledge representation and management, autonomic communications and network management. He has worked for 1 year in Greece Organization of Telecommunications (OTE) in the department of new applied network technologies and for 1 year in the General Secretariat of Research and Technology (GRNET). He has published several conference papers and 2 journal papers. He is currently working on an autonomic network management platform in Autonomic Internet (AUTOI) European Research Project.

Kleanthis Thramboulidis is a research and teaching staff associate professor in software engineering in the Department of Electrical & Computer Engineering at University of Patras, Greece, where he is leading the Software Engineering Group (http://seg.ee.upatras.gr/seg). He has authored seven books on programming and modeling (in Greek) and proposed a constructivism based approach to teach Object Oriented programming. He has extensive engineering experience working as an analyst and design engineer in many different application domains applying Object Technology with great success. He is the designer of REDOM, an OO Language to define and on-line manipulate regulations in the resource (re)scheduling problem used in the airline domain, and CORFU, a framework for the unified development of distributed control systems (http://seg.ece.upatras.gr/corfu/). He has proposed Model Integrated Mechatronics (MIM), a new paradigm for the model driven development of Mechatronic Manufacturing systems (http://seg.ece.upatras.gr/MIM/). He is currently working on a service-based development platform for distributed control and automation systems. Research areas cover object technology, model driven development, meta-modeling, distributed control and automation systems, embedded systems, CASE tools, component and service based development, service oriented architectures, semantic web and Mechatronics.