A Framework for Automated Negotiation

Claudio Bartolini, Chris Preist
Trusted E-Services Laboratory
HP Laboratories Bristol
HPL-2001-90
April 20th , 2001*

E-mail: claudio_bartolini@hp.com, chris_preist@hp.com

automated negotiation, electronic marketplaces, business-to-business e-commerce, negotiation protocols, auctions, software framework

Negotiation has been a central subject of study in disciplines such as economics, game theory, and management for decades. With the increasing importance of business-to-business electronic trading, the interest in automated negotiation and to the production of enabling software infrastructure has soared.

Existing approaches to architecting software enabling automated negotiation provide either ad-hoc software for particular market mechanisms or proprietary solutions. We want to take an approach that is general with respect to a wide variety of market mechanisms, from one-to-one negotiation to auctions and double auctions. At the same time we take an open system approach by defining a standard protocol for interaction among the participants to the negotiation process that is parametric with respect to the negotiation rules embodying the particular market mechanism that the negotiation host chooses to impose.

The generality of this approach allows us to provide value to all the actors in a negotiation process. Its value to the negotiation participants is that it provides support to the automation of the negotiation process, and hence makes the process more efficient. Furthermore, they can be confident that the basic rules of interaction in any negotiation are standardised, hence reducing the effort to automate many different kinds of business interactions. Moreover, the protocols provide the participants with trust guarantees, that no party has access to extra information or is able to forge false information. Its value to negotiation hosts such as auction houses and market makers is that it provides a standard framework that all potential customers can use to interact with them. However, it does not require a specific market mechanism, so allows the host to decide on an appropriate one. It provides standard off-the-shelf market mechanisms, but also allows custom mechanisms to be implemented for particular special needs.

1. Introduction
The increasing importance of business-to-business electronic trading has driven interest in automated negotiation to soaring heights. In particular, we foresee a need for a general software infrastructure that enables independent entities to interact using multiple forms of negotiation. This infrastructure would cover a variety of aspects, including defining a general protocol for negotiation (including the definition of the actors, roles and phases of negotiation), defining a taxonomy and a language for negotiation rules to cast the general protocol into one that embodies the desired market mechanism and defining a language to express negotiation proposals.

Negotiation has been for decades a central subject of study in disciplines such as economics, game theory, and management. When discussing negotiation, it is important to distinguish between *negotiation protocol* and *negotiation strategy*. The protocol determines the flow of messages between the negotiating parties, dictating who can say what and when and acts as the rules by which the negotiating parties must abide if they are to interact. The protocol is necessarily public and open. The strategy, on the other hand, is the way in which a given party acts within those rules in an effort to get the best outcome of the negotiation. For example, when and what to concede, and when to hold firm. The strategy of each participant is therefore necessarily private. In this document we concentrate on the requirements for architecting software enabling automated negotiation; we discuss protocols and not strategy.

Existing approaches to architecting software enabling automated negotiation provide either ad-hoc software for particular market mechanisms or proprietary solutions. We take an open approach by defining a standard protocol for interaction among the participants in the negotiation process. Our protocol is parametric with respect to the negotiation rules embodying the particular market mechanism that the negotiation host wants to impose. This approach is general with respect to a wide variety of market mechanisms, from one-to-one negotiation to auctions and double auctions.

The remainder of this paper is structured as follows. In section 2, we list the requirements for the framework. In section 3, we take the design one step forward, by presenting a conceptual abstraction over the various kinds of negotiation process and refining it down to the definition of the roles and responsibilities of the actors involved, taking a use-case driven approach. In section 4, we focus on the general negotiation protocol and in section 5 we work towards a taxonomy of types of negotiation rules to go with the protocol definition. Examples of market mechanisms supported by the negotiation protocol, along with the negotiation rules that are required to cast the general protocol to implement them are presented in section 6. In section 7 we discuss future work and conclude in section 8.

2. A General Approach to Designing Software for Automated Negotiation

2.1. Value Proposition
The framework aims to provide infrastructure that allows two or more independent entities to interact with each other over time to reach agreement on the terms and conditions of a service provision, or more in general of an exchange of economic value. This infrastructure is designed primarily, though not exclusively, as a means to reach trade agreements. It can be used both by automated entities and by users via appropriate software tools.

The value of such a framework to *negotiation participants* is threefold. First, the framework frees participants from having to develop their own negotiation infrastructure, providing support for the automation of the negotiation process. Second, the infrastructure enforces the standardization of basic interaction rules, allowing participants to be confident that basic rules of interaction in any negotiation will be followed. For example, participants will able to negotiate simple contracts, where only price is undetermined, as well as more complex contracts involving multiple complex and interdependent parameters. Third, the protocols provide the participants with trust guarantees, ensuring that no party has access to extra information or is able to forge false information.

The value to *negotiation hosts*, such as auction houses and market makers, is that by providing a standard framework that is independent of any specific market mechanism, they will increase the number of potential customers that can interact with them. Standardization will also ease system integration. The infrastructure would allow the hosts to select an appropriate market mechanism. It would provide standard off-the-shelf market mechanisms (e.g. English auction), and also allow custom mechanisms to be implemented for particular special needs (e.g. radio spectrum and football TV-rights auctions [Klemperer, 2000]).

2.2. Requirements

We identify the following requirements for a negotiation protocol that would meet our goals:

1. Be sufficiently formal that automated entities can interact using it.
2. Support negotiation about simple and complex objects.
3. Be sufficiently general that a variety of different market mechanisms (e.g. one-to-one negotiation, combinatorial auctions, exchanges) can be expressed as specific instances of it.
4. Support security mechanisms and protocols that enable participants to do business in a trusted way.
5. Allow, but not require, the existence of a third party to arbitrate a given negotiation (e.g. an auctioneer in an auction.)
6. Support existing ways of doing business, as well as permitting more radical approaches in the future.

2.3 What the Negotiation Framework Needs to Define

Technology-wise, the framework is based upon the assumption that messaging middleware will be available. For example, any implementation of the Java ™ Message Service (JMS) [Monson-Haefel, 2000; Giotta, 2001] would do. Industrial systems implementing JMS are available, such as Progress Software SonicMQ, IBM MQSeries and many others. Open source JMS systems such as JbossMQ from Jboss.org are available too.
Another possibility is to layer the framework on top of a message transport service such as the one described as part of the FIPA abstract architecture [FIPA, 2000].
In addition to that, our negotiation framework needs to define:

- a general protocol for negotiation that can support a wide variety of market mechanisms
- a taxonomy of rules of negotiation
- a language to define rules of negotiation
- a language to express negotiation proposals

2.3.1 A general protocol for negotiation that can support a wide variety of market mechanisms

A negotiation protocol provides a means of standardizing the communication between participants in the negotiation process by defining the pattern of interaction among actors.
We propose that the protocol should be general enough to support a wide variety of market mechanisms. Therefore the protocol should be based on the common aspects of the various market mechanisms that it wants to support. That is, it should define the negotiation process as an exchange of negotiation proposals followed by a phase of agreement formation. The two phases will often be intertwined for some market mechanisms.

Designing the protocol requires the definition of:

1. The roles played by the actors involved in negotiation processes
2. The phases of the negotiation process (e.g. admission, proposal submission, agreement formation)
3. The messages that are exchanges by the actors in each phase

2.3.2 A taxonomy of rules of negotiation

As noted above, the protocol itself will be parametric with respect to the negotiation rules embodying the particular market mechanism supported by the negotiation host. The rules for negotiation will then cast the general protocol into one that can be used to embody a particular market mechanism.
Examples of types of rules for negotiation would be rules for deciding on the well formedness of a negotiation proposal. Another example is rules regulating the alternation of participants in submitting proposals. Again, there will be rules that dictate the visibility aspect of proposals in many-to-many negotiation, i.e. who among the participants is entitled to see a submitted negotiation proposal, and so on.

2.3.3 A language to define rules of negotiation

The idea is to have a declarative language for expressing rules in a way that negotiation participants can reason over them. The declarative layer would then be mapped to reusable software components implementing the logic expressed by the rules. These components would be plugged into the orchestration infrastructure for the protocol to be cast to embody a desired market mechanism.

2.3.4. A language to express negotiation proposals

The requirements to be satisfied by a candidate language for negotiation proposals are:

- Support for ontology and namespaces
- High degree of expressiveness
- Ability to express less than fully bound specifications
- Ability to express constraints over ranges of possible values as well as definite values of a specification
- Loose support for types and some degree of inheritance
- Support for complex queries
- Support for complex matching

RDF Schema [Brickley, 2000] and DAML-OIL [Van Harmelen, 2001] are promising candidate languages.

In the remainder of this document, we will propose a framework for automated negotiation. The framework defines a general protocol to cater for a variety of market mechanisms. We will also describe a classification of types of negotiation rules, used to cast the general protocol into more specific ones, embodying different market mechanisms. We will not go so far as defining a language for expressing the rules and the negotiation proposals, but we will be able to list the requirements that our framework imposes on them.

3. The Negotiation Framework

We take a use-case driven approach to the specification of the negotiation framework. In our quest for generality, the scenarios that we are considering are derived from an abstraction of the commonalities of the negotiation process in various mechanisms defined in game theory.

3.1. Abstraction of the Negotiation Process

Negotiation is the process by which two or more parties interact to reach an agreement. Usually this will be about some business interaction such as the supply of a service in return for payment. However, the concepts described in this section are sufficiently general that they can be used to negotiate other forms of agreement.

Negotiation takes place by parties communicating through a *negotiation locale*. The negotiation locale is an abstraction over the messaging system that is used by negotiation participants to address each other. After admission to negotiation, a participant is given access to the negotiation locale. This locale may already exist, or may be created specifically for this new negotiation. Along with the access, the participant is given a *mailbox* where messages encoding negotiation proposals will be delivered. Each participant can send proposals by broadcasting them to the negotiation locale. Reliable delivery and security will be enforced by the underlying messaging infrastructure. Singling out a counterpart can be achieved by limiting the visibility of the broadcast message, in case the market mechanism rules allow it. That allows us to model one-to-one negotiation as a particular case of many-to-many.

To be able to negotiate with each other, parties must initially share an *agreement template*. This specifies the different parameters of the negotiation (e.g. product type, price, supply date etc). Some of the parameters will be constrained within the template (e.g. product type will almost always be constrained in some way) while others may be completely open. A negotiation locale has an agreement template associated with it. This defines the object of negotiation within the locale.

As part of the admission process of participants to the negotiation, they are requested to accept the agreement template. The admission step might be formalized by the specification of *admission policies*. The admission policies can specify what credentials (if any) are requested from participants for them to be admitted to the negotiation.

Depending on what parameters a party is willing to negotiate on, it will adopt more or less constrained agreement templates. For example, a party that is willing to negotiate nothing (such as a catalogue) will only advertise a fully instantiated agreement template, with a fixed price. A party willing to negotiate features of a product, such as colour, as well as price and delivery date, will leave these parameters unconstrained.

The process of negotiation is the move from an *agreement template* to an *agreement*, which the agreeing parties find acceptable. A single negotiation may involve many parties, resulting in several agreements between different parties and some parties who do not reach agreement. For example, a stock exchange can be viewed as a negotiation where many buyers and many sellers meet to negotiate the price of a given stock. Many agreements are formed between buyers and sellers, and some buyers and sellers fail to trade.

In the process of reaching agreements, the negotiation participants exchange *proposals* representing the deals that are currently acceptable to them. Each proposal will contain constraints over some of the parameters expressed in the agreement template. These proposals are sent to the negotiation locale, and can then be viewed by other negotiation participants. However, before a proposal is distributed by the locale, it must be *valid*. To be valid, it must satisfy two criteria:

1. It must be a more constrained form of the agreement template. That is the constraints over the parameters in the proposal must be tighter that the corresponding ones in the agreement template. The constraints represent acceptable values to the proposing participant. (Often, these constraints will be a single acceptable value of a parameter).

2. The proposal must be submitted according to the set of rules that govern the way the negotiation takes place. These rules specify (among other things) who can make proposals, when they can be made, and what proposals can be submitted in relation to previous submissions. (For example, auctions often have a 'bid improvement' rule that requires any new proposal to buy to be for a higher price than previous proposals). Such rules are specified and agreed at the admission stage.

An agreement will be formed according to the *agreement formation* rules associated with the negotiation locale. When the proposals in the locale satisfy certain conditions, they are converted by these rules into agreements, and returned to the proposers. The end of a negotiation is determined by *termination rules*.

This abstraction of the negotiation process can easily be mapped to a bargaining process (one-to-one negotiation). Participants take turns in exchanging proposals, the format of which is agreed before the bargaining process begins. The rules in this case are simple. Anything goes, as long as the proposals are consistent with the agreement template, and termination will occur when the same proposal is returned unchanged (which we take as declaration of acceptance). The agreement that is formed will be as specified in the last proposal that was transmitted. Termination occurs when the agreement is formed, or if one party leaves the negotiation locale.

In an English auction, the proposals would only specify the price of the good, every other parameter being given a specific value in the agreement template. Negotiation rules state that every new proposal (bid) will be valid only if it is an improvement over the current best proposal. Termination occurs at a deadline, and the agreement that is formed will contain the specification of the good as expressed in the agreement template, at the price specified in the winning bid.

We now present the negotiation framework in more detail. Firstly, we describe the roles involved in negotiation.

3.2. Roles in Negotiation

There are two main roles in negotiation – *participant*, and *negotiation host.* The participants are those who wish to reach agreement. The negotiation host is the role responsible for enforcing the protocol and rules of negotiation.
It is to be noted that the roles that are introduced at the analysis phase are to be considered as aggregations of responsibilities. They do not necessarily map one-to-one onto entities that will be introduced in the concrete design phase.
For example, the host is often a third party outside the negotiation. In the case of an auction, the host is the auctioneer. In the case of an exchange, the host is the market provider. However, the host may also be a participant; in one-to-one negotiation or catalogue provision, this is usually the case. In some cases, the role of negotiation host may alternate between different entities as the negotiation progresses.
In the negotiation framework, we break down the negotiation host role into sub-roles, in order to give a more precise definition of its responsibility. These sub-roles are gatekeeper, infrastructure provider, proposal validator, protocol enforcer, agreement maker, and information updater. We now briefly describe each role. In the following section, we will explain how they interact in the general negotiation protocol.

The *Negotiation Participant* participates in a negotiation by posting proposals according to the rules provided by the negotiation host.

The *Negotiation Host* is responsible for creation and enforcement of rules governing participation, execution, resolution and termination of a negotiation. It has the following sub-roles:

> *Gatekeeper*
> Role responsible for the enforcement of policy governing admission to a negotiation.
>
> *Infrastructure provider*
> Provider of the underlying communications infrastructure of the negotiation locale. Forwards proposals and information updates, according to the visibility rules defined by the negotiation host.
>
> *Proposal validator*
> Role responsible for ensuring that a proposal is well formed with respect to the agreement template.
>
> *Protocol enforcer*
> Role responsible for ensuring that participants' proposals are posted and withdrawn according to the negotiation rules.
>
> *Agreement maker*
> Role responsible for agreement formation.
>
> *Information updater*
> Role responsible for notifying participants of the current state of the negotiation, according to the display rules.

The responsibilities of each of the roles will be clarified starting in the next section, where we carry out a use cases analysis of the negotiation process and describe the general negotiation protocol that the framework supports.

3.3. Use cases

To create the definition of the general negotiation protocol, we carried out a use-case analysis, specifying how the roles interact during the negotiation process.

The notation that we use in this section is derived from the UML notation for use cases. We use it to define the interactions among the roles, rather than a system specification. Notice that no system boundaries are drawn, because our focus is on the definition of the protocol that the actors use to interact.

Fig. 1a-e shows the use cases that we developed for the negotiation framework.

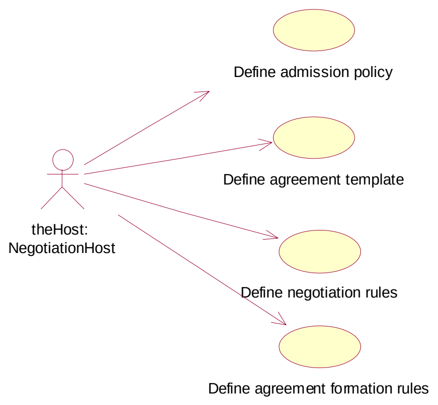


Fig 1a. Pre-negotiation use cases

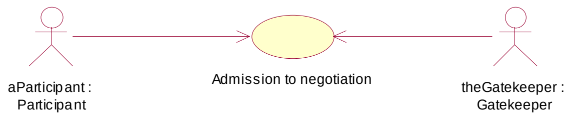


Fig 1b. Admission use case

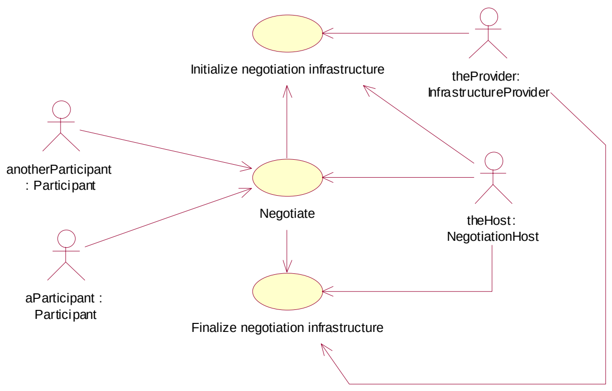

Initialize negotiation infrastructure
theProvider: InfrastructureProvider
anotherParticipant : Participant
Negotiate
theHost : NegotiationHost
aParticipant : Participant
Finalize negotiation infrastructure

Fig 1c. Negotiate use case

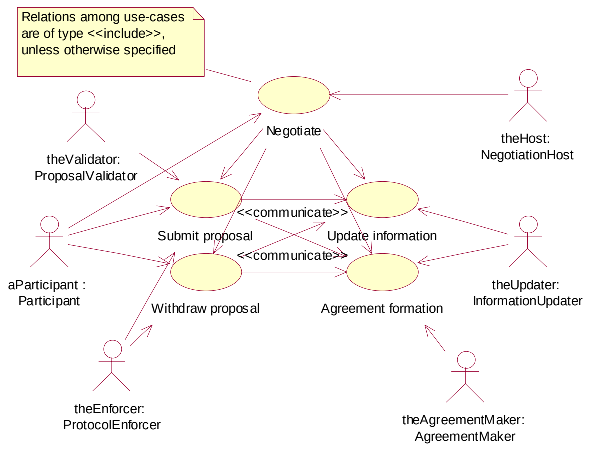

Relations among use-cases are of type <<include>>, unless otherwise specified
Negotiate
theHost: NegotiationHost
theValidator: ProposalValidator
<<communicate>>
Submit proposal
Update information
<<communicate>>
aParticipant : Participant
Withdraw proposal
Agreement formation
theUpdater: InformationUpdater
theEnforcer: ProtocolEnforcer
theAgreementMaker: AgreementMaker

Fig 1d. Negotiate use case and sub-use cases

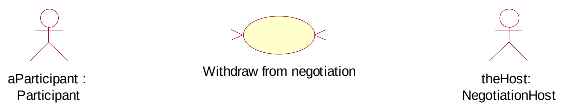


Fig 1e. Negotiation withdrawal use case

4. The General Negotiation Protocol

To specify each of the use cases, we used a template suggested in [Coleman, 1998]. See appendix A.

For the principal use cases (i.e. *Negotiate* and sub use cases), we specify the interactions of the roles using UML activity diagrams enhanced with *swimlanes*. The swimlanes in the diagrams are the views of each of the actors involved. Activity diagrams proved to be very useful in defining the negotiation protocol because they specify *activities* that the actors carry out at the same time as specifying the *states* that the process is in. Moreover, activity specification can be made modular by specifying an activity in a use-case as a sub-use-case. For instance, notice how we break up the definition of the *Negotiate* use case into the activities of *Submit proposal*, *Withdraw proposal* and *Agreement formation*, and how we specify each of those activities in turn with its own activity diagram.

We now present the *Negotiate* use case that constitutes the core of the proposed general negotiation protocol, together with its sub-use-cases. Appendix A provides full details.

Fig 2 shows the negotiate use case. We assume that a negotiation locale exists and is functional, and a negotiation template exists. The negotiation host declares the negotiation open and, from this moment on, participants can be admitted to the negotiation process if they meet the admission requirements.

The participants now exchange proposals until termination is reached. Agreement formation can occur at any time (for example when two proposals are matched in a continuous double auction). Agreement formation may trigger termination (e.g. one-to-one negotiation) or may not (e.g. continuous double auction).

Each time a participant submits a proposal (Fig. 3) by posting it to the negotiation locale, the negotiation host, in the role of *proposal validator*, validates the proposal against the agreement template. It checks that the proposal is a constrained form of the agreement template, and is syntactically well formed.

If the proposal is not valid, then it is rejected. If the proposal passes this first stage of validation, then the negotiation host (playing here the role of protocol enforcer) validates the proposal against the negotiation rules. The negotiation rules define the way in which the negotiation should take place, and may include restrictions on when a proposal can be made (e.g. participants must take turns to submit) and semantic requirements on valid proposals (e.g. requirements that a proposal must improve on previous ones).

If the proposal passes this second stage of validation, then the current set of proposals and associated data structures are updated accordingly and participants are notified. Who is notified, and the structure of the notification, is defined by visibility rules and information filtering rules.

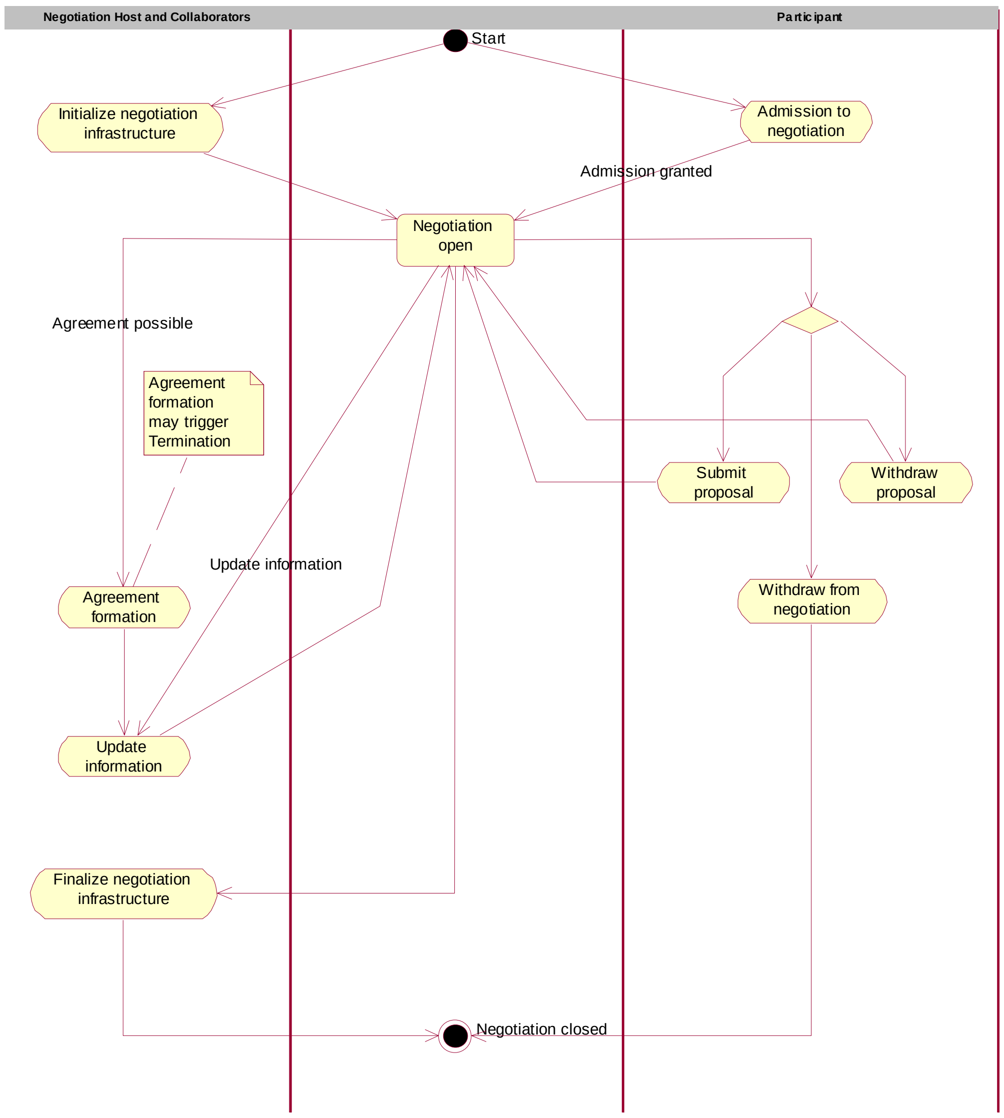


Fig 2. Negotiate activity diagram

Operations such as proposal validation against the negotiation template and agreement rule enforcement might be difficult to implement. However, these abstractions make it very easy to explain the general protocol. For a discussion of this issue, see the section on future work.
As Fig. 2 (negotiate activity diagram) shows, an agreement formation process can be triggered at any time during negotiation, according to the specification contained in the agreement formation rules. The negotiation host (this time in the agreement maker role) then looks at the current set of proposals to determine whether agreements can be made. Agreements can potentially occur whenever two or more negotiating parties make compatible proposals. In this case, agreement formation rules determine exactly which proposals are matched with each other, and the final instantiated agreement that will be used. Agreement rules may state, for example, that the highest priced offer to buy should be matched with the lowest priced offer to sell, and that the final agreement will take place at the average price. Often, ‘tie breaking’ agreement rules will be defined that will be used if the main agreement rules can be applied in several ways. For example, earlier posted offers may take priority over later ones.

When the agreement formation rules have been applied to determine exactly which agreements are made, the negotiation host (information updater) notifies the participants. Fig. 4 illustrates the agreement formation utility diagram.

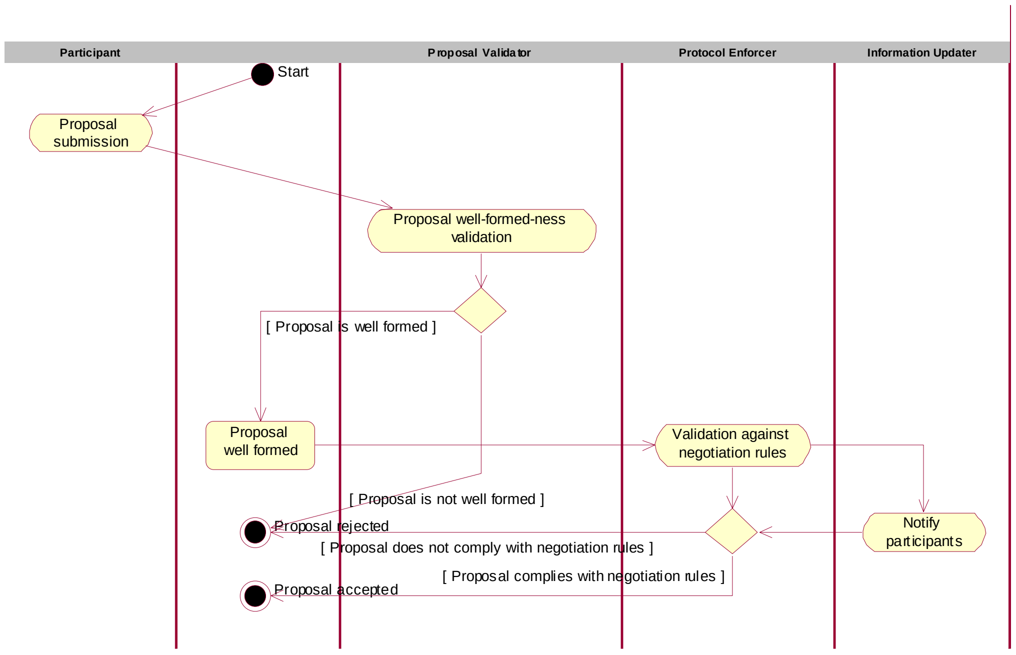

Participant
Proposal Validator
Protocol Enforcer
Information Updater
Start
Proposal submission
Proposal well-formed-ness validation
[Proposal is well formed]
Proposal well formed
Validation against negotiation rules
[Proposal is not well formed]
Proposal rejected
[Proposal does not comply with negotiation rules]
Notify participants
[Proposal complies with negotiation rules]
Proposal accepted

Fig 3. Proposal submission activity diagram

Some of the issues discussed above for negotiation rules apply to the agreement formation rules too. We will discuss this in the future work session.

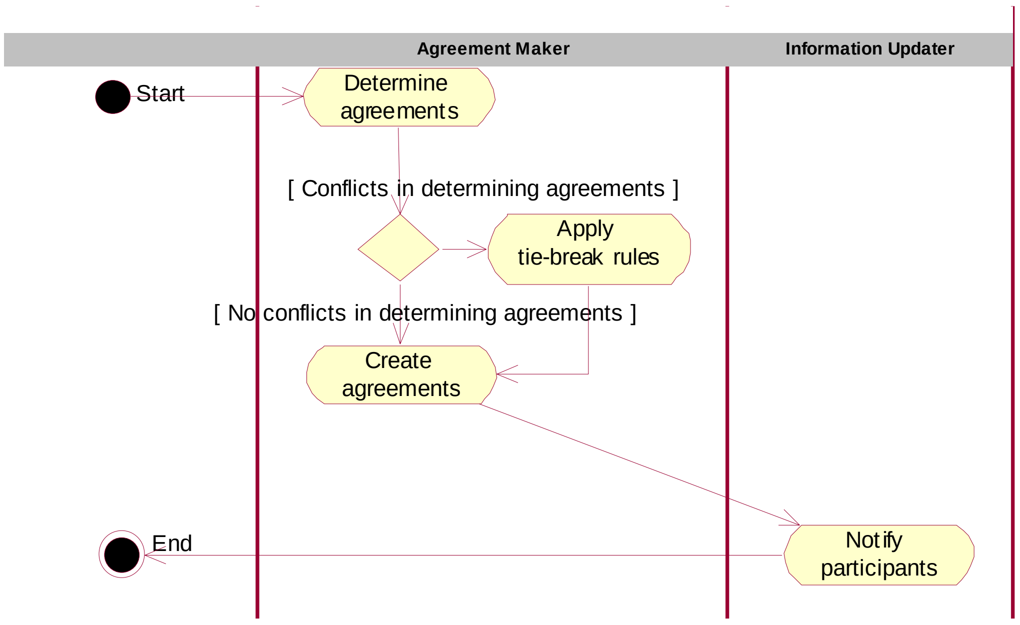


Fig 4. Agreement formation activity diagram

In this section, we have defined a general protocol for negotiation to take place between two or more parties. The protocol is parameterised by a set of rules. By choosing a specific set of rules to enforce, a negotiation host can create a specific market mechanism. In section 5, we identify the different types of rules a negotiation host must select, and in section 6 we give specific examples of market mechanisms, and the associated rules that are selected.

5. A Taxonomy of Rules for Negotiation

The separation that we have imposed between negotiation protocol and negotiation rules requires that we show how the software infrastructure must be customized to embody a market mechanism. Our approach can only be useful if there are at least guidelines on a series of issues. These are: how to express negotiation rules, what language to use to express them, how a particular set of rules is suited to a market mechanism and a sketch of a software architecture that eases the task of a negotiation host wanting to constrain the general protocol with the appropriate negotiation rules.

In this section we provide a first taxonomy of negotiation rules in order to provide an insight on what are the ways to cast the general protocol to specific ones embodying chosen market mechanisms. In the following section we give examples of how these rule types are instantiated and applied, for different kinds of negotiation.

Posting rule
Negotiation rule determining the circumstances in which a participant may post a proposal.

Visibility rule
Negotiation rule specifying who, among the participants, has visibility over a submitted proposal.

Display rule
Negotiation rule specifying if and how the information updater notifies the participants that a proposal has been submitted or an agreement has been made – either by transmitting the proposal unchanged or by transmitting a summary of the situation.

Improvement rule
Negotiation rule specifying, given a set of existing proposals, what new proposals may be posted.

Withdrawal rule
Negotiation rule specifying if and when proposals can be withdrawn from negotiation, and policies over the expiration time of proposals.

Termination rule
Negotiation rule specifying when no more proposals may be posted (e.g. a given time, period of quiescence, etc.).

Agreement formation rules
Rule responsible for determining, given a set of proposals at least one pair of which intersect, which agreements should be formed.

Tie-breaking rule
A specific agreement formation rule applied after all others.

6. Examples of Market Mechanisms Supported by the Negotiation Framework

In this section we propose a way of instantiating the rule types for a few specific examples of negotiation mechanisms. For each example, we give a short description of the negotiation process in terms of the roles that we introduced in section 3, and we refer to the use cases and phases of the protocol highlighted in section 4. In addition, we describe how the rule types described in section 5 will be instantiated. The market mechanisms that we describe are: a simple shop front negotiation, an English auction where a single item is for sale and a continuous double auction, similar to a stock exchange.

6.1. Simple Shop Front

The actors involved in the simple shop front scenario are the shopkeeper and one or more buyers. A buyer plays the participant role, whereas the shopkeeper plays both the participant and the negotiation host roles at the same time. The shop front is modelled by the negotiation locale abstraction.

Before negotiation begins, the shopkeeper decides the admission policy, negotiation template, negotiation and agreement formation rules.

Admission policy
This will usually be the null policy: anyone is admitted.

Negotiation template
The shopkeeper decides on templates of goods it is willing to sell. These will be fully defined, specifying all details exactly, including price. To be valid with respect to the negotiation template, a buyer's proposal must therefore be an exact copy of the seller's proposal (except it is 'buy' rather than 'sell').

Negotiation rules
The shopkeeper adopts standard 'shop front take it or leave it' negotiation rules. These state that:

a. [*Posting rule*]A buyer may post a proposal at any time, irrespective of posted proposals by other buyers. A seller may post or withdraw proposals at any time.
b. [*Termination rule*] Termination occurs when there are no seller proposals posted in the shop front

Agreement formation rules
The shopkeeper adopts standard shop front agreement formation rule:

a. [*Agreement formation rule*] Agreements are formed whenever a buyer posts a proposal identical to the seller's proposal.

After rules have been specified, negotiation can begin. The shopkeeper submits proposals for all goods it sells. If it expects high demand, it can place several identical proposals on the table for the same good. If all proposals for a given good are accepted, and the shopkeeper still has more in stock, it resubmits identical proposals. A buyer submits a proposal, an identical copy of the shopkeeper's proposal, when it wishes to purchase a given good. Agreement formation occurs as the shopkeeper – in the referee role – identifies valid buyer proposals and sends agreements to the buyers.

6.2. Single Item English Auction

Actors involved are a seller and various buyers in the role of participants. The auctioneer that auctions the item on behalf of the seller plays the negotiation host role. The auction room is modelled as a negotiation locale. Admission policy, negotiation template and negotiation and agreement formation rules are expressed as follows.

Admission policy
Auctioneer and seller decide the policy. This could be the null policy – anyone is admitted – or they could restrict admission to a number of invitees on presentation of an invitation certificate (participant's credential).

Negotiation template
The seller decides on the template of the good it is selling. This will be fully defined, specifying all details exactly, except for the price attribute, which will be constrained to be greater than an initial reservation price. To be valid with respect to the negotiation template, a buyer's proposal must therefore be an exact copy of the seller's proposal except for having a price that is higher than what is specified in the negotiation template as initial reservation price.

Negotiation rules
The auctioneer adopts standard English auction negotiation rules. These state that:

a. [*Posting rule*] A buyer may post a proposal at any time.
b. [*Improvement rule*] The price field of the buyer's proposal must be a certain increment above the value of all previously posted buyer proposals
c. [*Withdrawal rule*] It is not possible to withdraw a proposal that represent the currently highest bid.
d. [*Visibility rule*] The proposals that buyers submit are visible to all the participants.
e. [*Termination rule*] Termination occurs at a fixed time or after a period of inactivity

Agreement formation rules
The auctioneer adopts standard the English auction agreement formation rule, that states:

a. [*Agreement formation rule*] After termination, an agreement between the highest bidding buyer and the seller is formed for the item fully specified in the template to be sold to the buyer at the price specified in the highest bid.

When the negotiation is open, buyers submit proposals with the price instantiated to its bid value. At the deadline, the auctioneer identifies the highest bidding buyer, and forms agreement between it and the seller. It finally notifies both parties.

6.3. Multiple Item Continuous Double Auction (a.k.a. Exchange)

The actors are traders as participants and the market maker as the negotiation host. The negotiation locale is the exchange floor. Admission policy, negotiation template and rules are described below.

Admission policy
Either the null policy – anyone is admitted – or admission on presentation of credentials such as qualified trader.

Negotiation template
The market maker decides on the template of goods that are traded in the exchange. This will be fully defined, specifying all details exactly, except for the price attribute and quantity attribute, which will be open. To be valid with respect to the negotiation template, proposals must therefore be a copy of the proposal template, with price and quantity instantiated to specific values.

Negotiation rules
The market maker adopts standard continuous double auction negotiation rules. These state that:

a. [*Posting rule*] Buyers and sellers may post proposals at any time.

b. [*Improvement rule*] The price field of a buyer's proposal must be above the value of all currently posted buyer proposals. The price field of a seller's proposal must be below the value of all currently posted seller proposals.[1]
c. [*Withdrawal rule*] Any proposal can be withdrawn at any moment, before an agreement is formed that involves it.
d. [*Visibility rule*] Proposals are only visible to the market maker, in order to protect the participants from receiving too much information.
e. [*Display rule*] The market maker regularly updates the *order books*, containing information on proposals to buy and to sell, ordered by price.
f. [*Termination rule*] Termination occurs only when the auction ceases to be used.

Agreement formation rules
The market maker adopts standard continuous double auction agreement formation rules. These state that:

a. [*Agreement formation rule*] Agreement is formed between all overlapping buyers and sellers. The price is the midpoint of the overlap. Highest buyers and lowest sellers are satisfied first. When traders have different quantities, this may result in a single party having trades with several others (multiple agreements).
b. [*Tie breaking rule*] In case of ties, earlier proposals have priority.

During negotiation, the traders continuously exchange proposals. Agreement formation occurs whenever there is an overlap between buyers and sellers proposals, according to the rules above. Participants are notified of any agreements made.

7. Future Work

We intend to continue to work on the following aspects of the negotiation framework:

- Language for negotiation rules
- Architecture of the system supporting flexible plugging of rules
- Language for negotiation proposals
- Support for privacy of proposals
- Support for multi-party agreements

We have briefly touched on the language for negotiation rules in section 2. The idea is to have a declarative language for expressing rules in a way that participants in negotiation can reason about them. In this way, the participants take part in negotiation by implementing the same protocol, but can adjust their strategy based on their reasoning about the rules.

The declarative layer to express the rules would then be mapped to reusable software components implementing the logic expressed by them. These components would be plugged in the orchestration infrastructure for the protocol to be cast to embody a desired market mechanism. The protocol, as defined in section 4, presents a natural hook for the rule evaluation components where it requires the activity of *validation of rules against the negotiation rules* defined in the *submit proposal* use case. The same principle applies to agreement formation rules, as can be seen in the *agreement formation* use case.

In section 2, we also underlined the importance of defining a language for negotiation proposals. . We are currently investigating the applicability of languages such as RDF Schema and DAML-OIL.
These languages seem promising with respect to satisfying the requirements listed in section 2. They provide support for ontology and namespaces, present a high degree of expressiveness. Can express less than fully bound specifications and constraints over ranges

[1] This is referred to in the literature as the NYSE improvement rule.

of possible values as well as definite values of a specification. They loosely support types and some degree of inheritance, and offer support for complex queries and complex matching.

We are currently working on enhancing the negotiation framework with mechanisms for guaranteeing privacy of proposals and on other aspects related to security.

8. Conclusions

We have presented a framework for automated negotiation that covers a variety of aspects, including defining a general protocol for negotiation (with the definition of the actors, roles and phases of negotiation), defining a taxonomy and a language for negotiation rules to cast the general protocol into one that embodies a desired market mechanism and defining a language to express negotiation proposals. Among the aspect of the negotiation framework, we mainly concentrated our attention on the definition of the general negotiation protocol and working towards a taxonomy of types of negotiation rules to go with the protocol definition. Our protocol is parametric with respect to the negotiation rules embodying the particular market mechanism that the negotiation host chooses to impose. This approach is general with respect to a wide variety of market mechanisms, from one-to-one negotiation to auctions and double auctions.

We have highlighted the methodology that we have followed to design the negotiation framework. We started by stating the value proposition and listing the requirements of the framework. Then we took the design one step forward, by presenting a conceptual abstraction over the various kinds of negotiation processes and refining it down to the definition of the roles and responsibilities of the actors involved, taking a use-case driven approach. This led us to the definition of the general negotiation protocol and to a first pass at a taxonomy of types of negotiation rules. We then gave examples of market mechanisms supported by the negotiation protocol, along with the negotiation rules that are required to implement them.

The main issues that remain open are the definition of languages for negotiation rules and for negotiation proposals. Our next steps towards the completion of the specification of the negotiation framework will tackle these problems.

9. Acknowledgements

Kannan Govindarajan provided useful input and valuable feedback during the ideas inception phase.
Dorothea Beringer, Andrew Byde, Harumi Kuno, David Trastour and Mike Yearworth provided feedback on early drafts of the paper.
We'd like to thank David Bell, Naiem Dathi, Kemal Guler and Alan Karp for inspiring discussions.

10. References

[Klemperer, 2001] P. Klemperer, *What really matters in auction design*, available at http://www.nuff.ox.ac.uk/users/klemperer/design3aweb.pdf

[Monson-Haefel, 2000] R. Monson-Haefel and D. Chappell, *Java Message Service*, O'Reilly and associates Inc., 2001

[Giotta, 2001] P. Giotta, S. Grant, M. Kovacs, M. Kunnumpurath, S. Maffeis, K. S. Morrison, G. S. Raj, M. P. Kovacs, J. McGovern, *Professional JMS*, Wrox press, 2001

[FIPA, 2000] Foundation for Intelligent Physical Agents, *FIPA Abstract Architecture Specification*, available at http://www.fipa.org/specs/fipa00001/XC00001I.pdf

[Brickley, 2000] D. Brickley and R. V. Guha (editors), *Resource Description Framework (RDF) Schema Specification 1.0*, W3C Candidate Recommendation 27 March 2000, available at http://www.w3.org/TR/2000/CR-rdf-schema-20000327

[Van Harmelen, 2001] F. van Harmelen, P. F. Patel-Schneider and I. Horrocks, editors, *Reference description of the DAML+OIL, March 2001*, available at http://www.daml.org/2000/12/reference.html

[Coleman, 1998] D. Coleman, *A use case template*, available at http://www.bredemeyer.com/pdf%20files/use_case.pdf

URLs last checked on March 23, 2001

Appendix A: Use Cases

The use case are presented using the template suggested in [Coleman, 1998]. For the principal use cases, we present UML activity diagrams with swimlanes.

1. Define Admission Policy

Use Case	Define Admission Policy
Description	The Negotiation Host defines the policies that will be used to admit Participants to negotiation. Participants could be involved in the definition for negotiations such as auctions or RFQs when a participant plays a dominant role.
Assumptions	
Actors	Negotiation Host (primary) Participant
Steps	IF participants are involved in the process, Negotiation Host gathers participants input The Negotiation Host defines the admission policy
Variations	
Non-Functional	
Issues	Definition of admission policy. Language for admission policy

2. Define Agreement Template

Use Case	Define Agreement Template
Description	The Negotiation Host defines the agreement template that will be used as a reference during the negotiation. Participants could be involved in the definition.
Assumptions	
Actors	Negotiation Host (primary) Participant
Steps	IF participants are involved in the process, Negotiation Host gathers participants input The Negotiation Host defines the agreement template
Variations	
Non-Functional	
Issues	Definition of agreement template and relative operations. See document on agreement template

3. Define Negotiation Rules

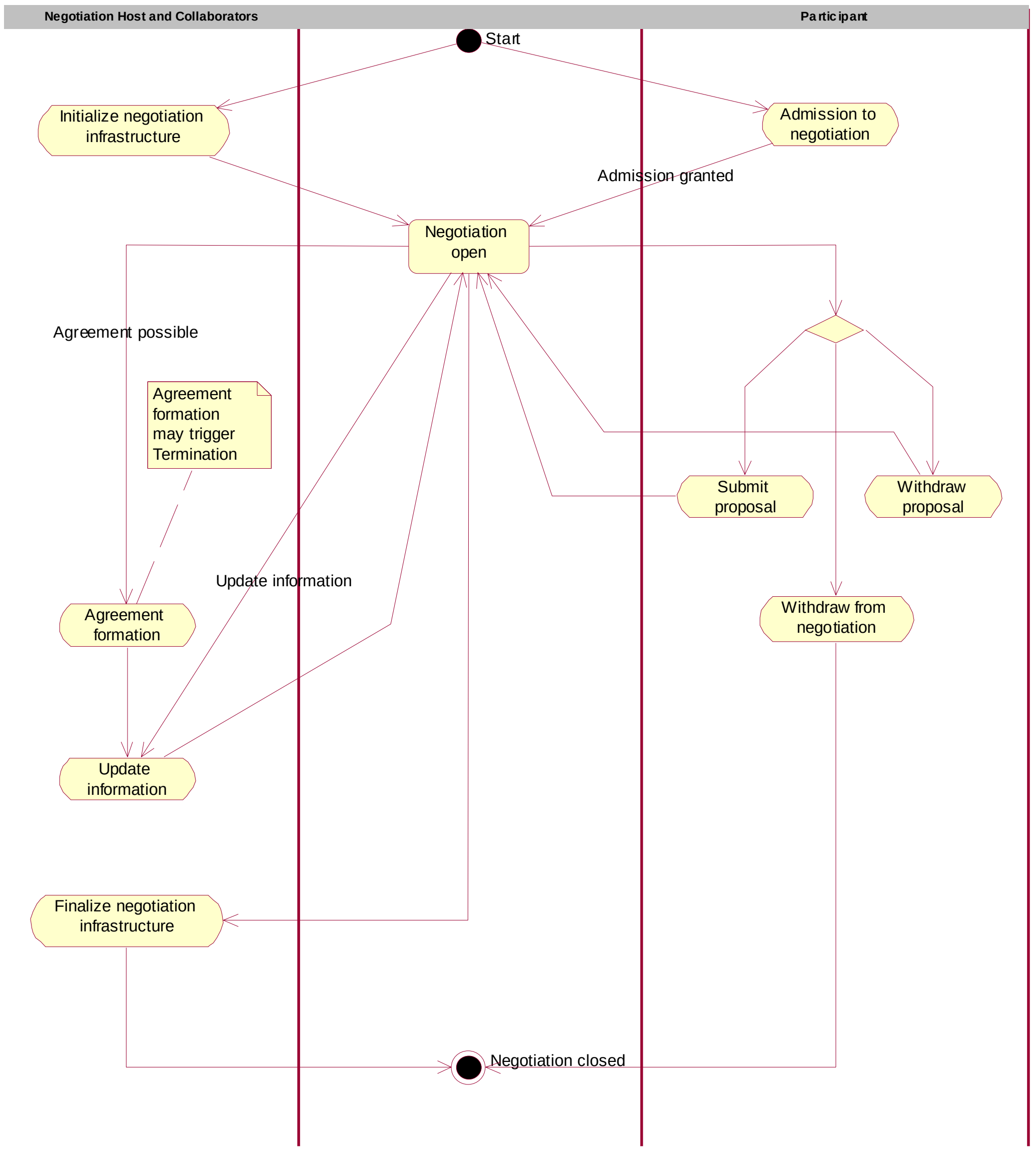
Use Case	Define Negotiation Rules
Description	The Negotiation Host defines the rules that Participants will have to comply with during negotiation. Participants could be involved in the definition for negotiations such as auctions or RFQs when a participant plays a dominant role.
Assumptions	
Actors	Negotiation Host (primary) Participant
Steps	IF participants are involved in the process, Negotiation Host gathers participants input The Negotiation Host defines the negotiation rules
Variations	
Non-Functional	
Issues	Definition of negotiation rules. 1.1 Language for negotiation rules

4. Define Agreement Formation Rules

Use Case	Define Agreement Formation Rules
Description	The Negotiation Host defines the agreement formation rules that will be used during the negotiation. Participants could be involved in the definition.
Assumptions	
Actors	Negotiation Host (primary) Participant
Steps	IF participants are involved in the process, Negotiation Host gathers participants input The Negotiation Host defines the agreement formation rules
Variations	
Non-Functional	
Issues	Definition of agreement formation rules 1.1 Language for agreement formation rules

5. Admission to Negotiation

Use Case	Admission to Negotiation
Description	The Gatekeeper admits Participants to the negotiation on verification of their credentials
Assumptions	
Actors	Gatekeeper (primary) Participant
Steps	
Variations	
Non-Functional	
Issues	Credentials Admission when negotiation has already started



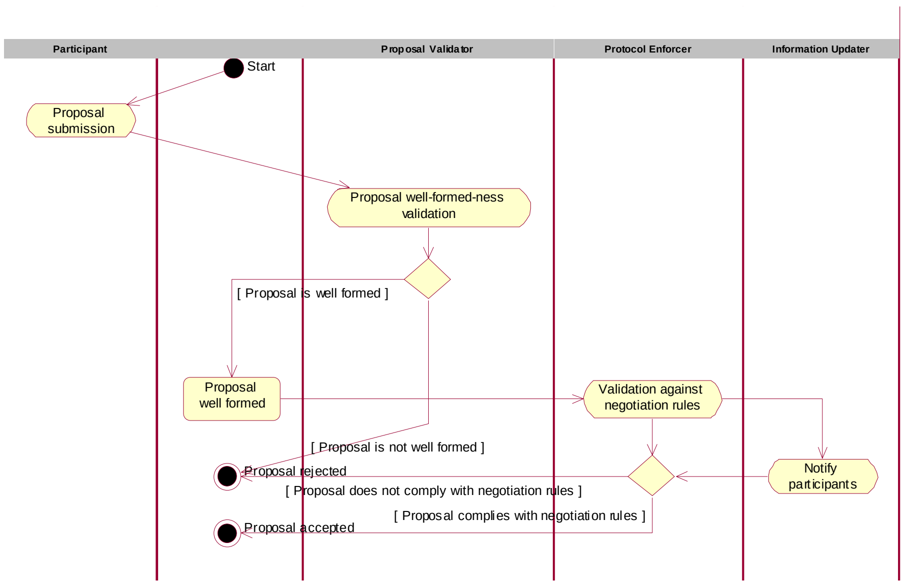
Activity diagram: Negotiate

6. Negotiate

Use Case	Negotiate
Description	Participants negotiate to get to the formation of agreements
Assumptions	A negotiation locale exists and is functional A negotiation template exists
Actors	Participant (primary) Negotiation Host
Steps	PERFORM *Initiate negotiation infrastructure* REPEAT 2.1 PERFORM *Submit proposal* 2.2 IF Agreement possible 2.2.1 PERFORM *Agreement formation* ENDIF UNTIL Termination PERFORM *Finalize negotiation infrastructure*
Variations	
Non-Functional	
Issues	Is it general enough to cater for any kind of negotiation? Might be interleaved with the *Admission to negotiation* use case, if that is allowed by the particular negotiation rules

6.1 Initialize negotiation infrastructure

Use Case	Initialize negotiation infrastructure
Description	Operations and communications preliminary to the negotiation process
Assumptions	
Actors	Negotiation Host (primary) Infrastructure Provider Participant
Steps	
Variations	
Non-Functional	
Issues	


Participant
Proposal Validator
Protocol Enforcer
Information Updater
Start
Proposal submission
Proposal well-formed-ness validation
[Proposal is well formed]
Proposal well formed
Validation against negotiation rules
Notify participants
[Proposal is not well formed]
Proposal rejected
[Proposal does not comply with negotiation rules]
[Proposal complies with negotiation rules]
Proposal accepted

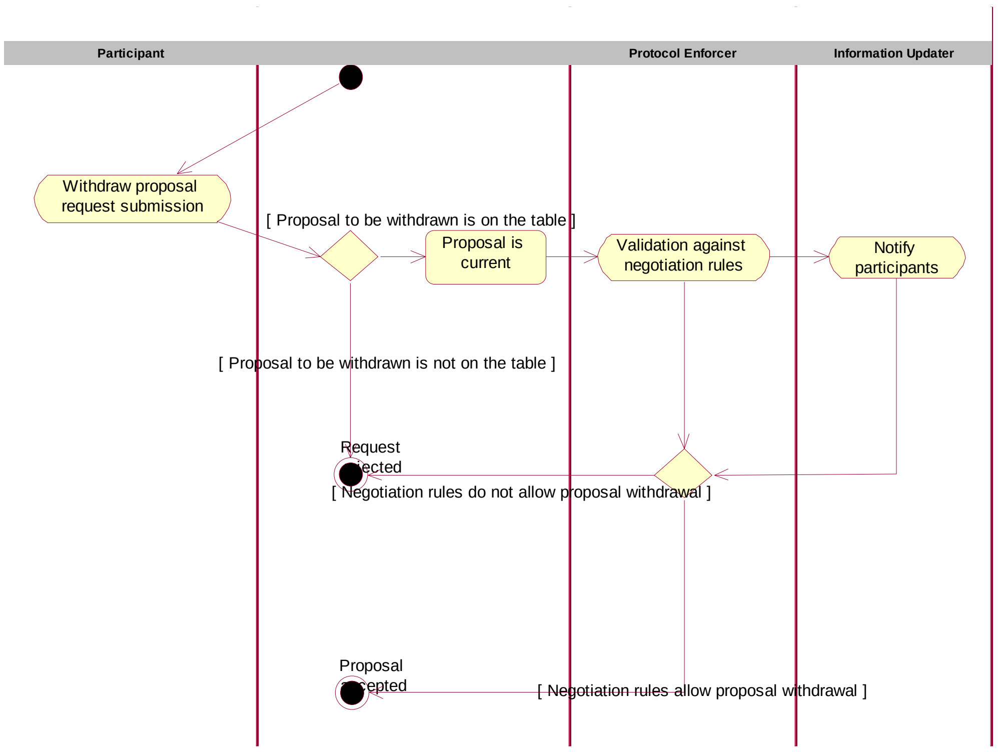
Activity diagram: Submit proposal

6.2 Submit proposal

Use Case	Submit proposal
Description	Participant submits a proposal that is validated against the agreement template and the negotiation rules
Assumptions	A negotiation locale exists and is functional An agreement template exists
Actors	Participant (primary) Negotiation Host
Steps	Participant sends a proposal to the negotiation table Negotiation Host (in the role of proposal validator), validates the proposal against the negotiation template (is the proposal relative to the object we are negotiating over? Is it well formed? Is it conforming to allowed expiration time? …) If the proposal is not valid, use case ends Negotiation Host (in the role of protocol enforcer) validates the proposal against negotiation rules (is it the submitter's turn? Does the proposal comply with the improvement rules? Is negotiation not terminated already?) If the proposal is valid, the current set of proposals and dependant data structures are updated accordingly and participants are notified, as defined by visibility rules and information filtering rules.
Variations	
Non-Functional	
Issues	Are proposal validator and protocol enforcer fully fledged roles or just responsibilities of the Negotiation Host? Operations such as proposal validation against negotiation template and agreement rules might be difficult to implement. The big advantage though is that this makes it very easy to explain the general protocol. Definition of negotiation template and relative operations. <Pointer here to document on negotiation template> Definition of negotiation rules. Language for negotiation rules

6.3 Withdraw proposal

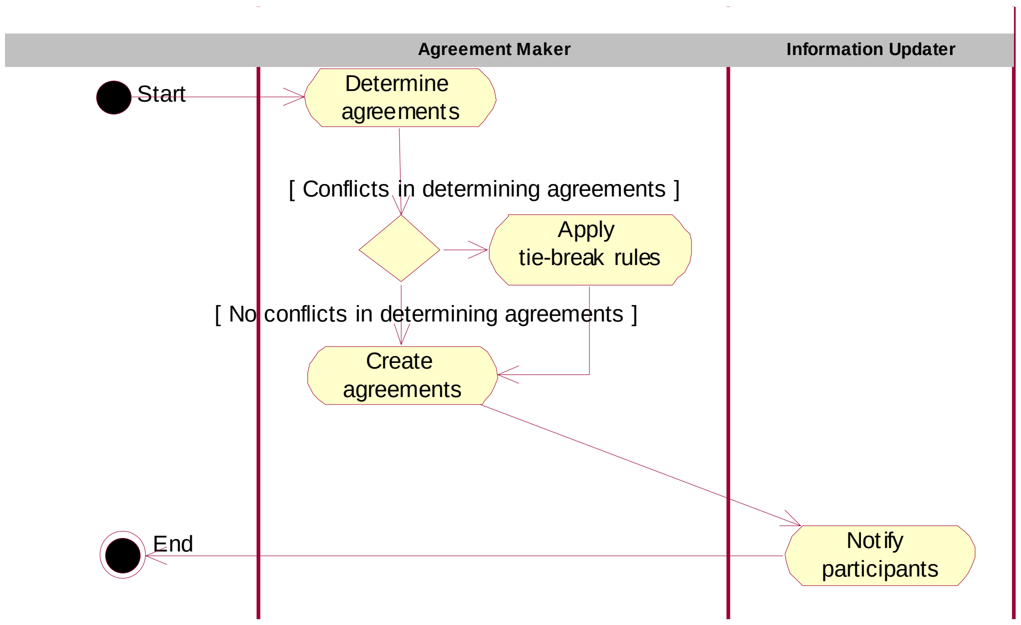
Use Case	Withdraw proposal
Description	Participant requests to withdraw a proposal
Assumptions	A negotiation locale exists and is functional An agreement template exists
Actors	Participant (primary) Negotiation Host
Steps	Participant sends a request to withdraw a proposal to the negotiation locale Negotiation host (playing the Proposal validator role) checks that the withdraw request refers to a proposal that is currently on the table If this is not the case, use case ends Negotiation host (playing the Protocol enforcer role) validates the withdraw proposal request against negotiation rules (is proposal withdrawal allowed in general? is it allowed to withdraw this particular proposal? Is negotiation not already terminated?) If the withdraw proposal request is accepted, the current set of proposals and depending data structures is updated accordingly and participants are notified. Also, depending on the negotiation rules, agreement formation can be triggered.
Variations	
Non-Functional	
Issues	Same issues as in submit proposal


Participant
Protocol Enforcer
Information Updater
Withdraw proposal request submission
[Proposal to be withdrawn is on the table]
Proposal is current
Validation against negotiation rules
Notify participants
[Proposal to be withdrawn is not on the table]
Request rejected
[Negotiation rules do not allow proposal withdrawal]
Proposal accepted
[Negotiation rules allow proposal withdrawal]

Activity diagram: Withdraw proposal

6.4 Agreement formation

Use Case	Agreement formation
Description	The Negotiation Host (in the agreement maker role) converts of a set of proposals, into a set of agreements.
Assumptions	A negotiation locale exists and is functional A negotiation template exists
Actors	Negotiation Host (primary)
Steps	Negotiation Host (in the agreement maker role) looks at the current set of proposals to determine whether agreements can be made Negotiation Host (in the agreement maker role) applies tie-breaking rules if that is the case Negotiation Host (in the agreement maker role) creates the possible agreements given the proposals on the table and the resolution rules Negotiation Host notifies the participants of agreements that have been made
Variations	
Non-Functional	
Issues	Is agreement maker a fully-fledged role or just a responsibility of the Negotiation Host? Definition of agreement formation rules Language for agreement formation rules



Activity diagram: Agreement formation

6.5 Finalize negotiation locale

Use Case	Finalize negotiation infrastructure
Description	Operations and communications posterior to the negotiation process
Assumptions	
Actors	Negotiation Host (primary) Infrastructure Provider Participant
Steps	
Variations	
Non-Functional	
Issues	

7. Withdraw from negotiation

Use Case	Withdraw from negotiation
Description	Participant requests to withdraw from negotiation, and negotiation rules permitting, Negotiation Host acts accordingly
Assumptions	The participant is taking part in the negotiation
Actors	Participant (primary) Negotiation Host
Steps	Participant requests to withdraw from negotiation If the participant has pending proposals, the negotiation host attempts to withdraw them If the proposals could be withdrawn, the participant is withdrawn from negotiation, otherwise the request is rejected
Variations	
Non-Functional	
Issues	