Towards Support Systems for Non-Monolithic Electronic Negotiations

The Contract-Group-Message Model

Willy Picard

Department of Information Technology
The Poznań University of Economics
ul.Mansfelda 4, 60-854 Poznań, Poland
picard@kti.ae.poznan.pl

ABSTRACT. *Traditionally, research in electronic negotiations support has concentrated on monolithic negotiations,* i.e. *negotiations in which parties are unitary decision entities. In this paper, a model for non-monolithic electronic negotiations is introduced. This model, named the contract-group-message model, integrates a multiversion contract model, and support for group dynamics and message exchange. Design of support systems for non-monolithic electronic negotiations based on the contract-group-message model is also presented in this paper.*

RÉSUMÉ. *Les travaux de recherche sur les négociations électroniques assistées par ordinateur ont traditionellement pour objet l'étude des négociations monolithiques,* i.e. *des négociations dans lesquelles les parties engagées sont considérées comme des entités décisionnelles unitaires. Dans cet article, un modèle pour les négociations électroniques non monolithiques est introduit. Celui-ci, appelé « contrat-groupe-message », intègre un modèle de contrat multiversionné et un support pour la dynamique de groupe et les échanges de messages. La conception de systèmes d'aide aux négociations électroniques non monolithiques basés sur le modèle contrat-groupe-message est également étudiée dans cet article.*

KEYWORDS: *electronic negotiations, non-monolithic negotiations, CSCW, group dynamics, multiversion contract, message exchange.*

MOTS-CLÉS : *négociations électroniques, négociations non monolithiques, travail coopératif, dynamique de groupe, contrat multiversionné, échanges de messages.*

JDS – 13/2004. Electronic Negotiations, pages 423 to 440

1. Introduction

In the field of electronic negotiations, many taxonomies of negotiation processes (Lomuscio *et al.,* 2001)(Wurman *et al.,* 2001)(Ströbel *et al.,* 2003) have been presented in the literature: single-attribute vs. multi-attribute, single-item vs. multi-item, two-parties vs. many-parties, etc. Little attention has been accorded – at least in the research area of electronic negotiations – to the distinction between monolithic and non-monolithic negotiations. Traditionally, research in electronic negotiations support has concentrated on monolithic negotiations. *Monolithic* negotiations are negotiations in which all parties are monolithic, *i.e.* each party behaves as a *unitary decision entity*. On the opposite, non-monolithic negotiations are negotiations in which some parties may be non-monolithic.

A non-monolithic party consists of many persons with various perceptions and goals. A non-monolithic party may be a family, an enterprise, a lobby group, or even a nation, depending on the negotiation process. An example of non-monolithic negotiations could be the negotiations concerning the conflict between Israel and the Palestine. The Camp David negotiations analyzed in (Raiffa, 1982) is another example of non-monolithic negotiations.

In non-monolithic negotiations, not only parties are negotiating with other parties (external negotiations), but members of a given party may negotiate with other members of the same party (internal negotiations), as their perceptions and goals are different. It may happen that individuals, not necessarily representatives, from various parties negotiate directly (cross-parties negotiations). Contrarily to external negotiations, the cross-parties negotiations cannot lead to a binding agreement that may end the negotiation process. Cross-parties negotiations usually take place when some issues involving a high level of expertise have to be solved. In this case, representatives may decide to speed up negotiations allowing their respective experts to negotiate directly.

Three categories of support for the negotiation process may be distinguished: process support, decision automation, and decision support (Rangaswamy *et al.,* 1997)(Yuan *et al.,* 2003)(Kersten *et al.,* Dec. 2001). Decision support systems provide tools to organize information, develop negotiation strategies, and evaluate negotiation offers. Agent-based Negotiation Support Systems (NSS) attempt to automate negotiation through the use of software agents over an electronic media. Process support systems provide communication means for negotiators. Process support systems are used in the place of a negotiation table. The format of exchange between parties, and the dynamics and procedures of the negotiation processes are the basis of process support system design.

In the case of non-monolithic negotiations, a decision support system would be difficult to design and use for two reasons. First, the complexity of the parties, as decision entities, is high and current decision support models are not adapted to non-monolithic parties. Second, interactions taking place during internal/external/cross-parties negotiations implies complex relationships between negotiators' perceptions,

goals, and decisions, which are difficult to model and deal with. Without decision support, decision automation is not possible. On the other hand, there is a need for process support systems to facilitate the negotiation process, providing negotiators with support for parallel internal/external/cross-parties negotiations.

For the reasons presented above, we focus on computer support for non-monolithic negotiations oriented towards negotiators' needs. Agent technology and automated negotiations are not addressed in this paper, although they may probably take advantage of the proposed model.

Two main approaches for process support systems may be distinguished (Schoop *et al.*, Jan. 2001b):

– a communication-based approach: in this approach, the negotiation process is considered as a complex communication process in which negotiators, *via* message exchange, are arguing to reach an agreement. Communication management systems manage the structure of messages that are exchanged, limiting ambiguity of interactions among negotiators;

– a document-based approach: in this approach, the negotiation process is considered as a collaborative authoring process in which negotiators are editing a contract in a collaborative manner. Traditional document management systems support the evolution of documents by keeping track of different versions.

As stated in (Schoop *et al.*, Jan. 2001b), "there can be no separate document and communication management for effectively supporting electronic negotiations". In the case of non-monolithic negotiations, a third aspect has to be taken into account: group dynamics. Therefore, we argue that three elements have to be taken into account for an efficient support for non-monolithic negotiations: document management, communication management, and group dynamics. These three elements are required for negotiation support systems for non-monolithic negotiations as:

– the document management element is required to model the object of the negotiation processes, *i.e.* the contract to be negotiated,

– the communication management element is required to model the meta-data about the negotiation processes, *i.e.* communication exchange about the negotiation processes,

– the group dynamics element is required to model social structures in which the negotiation takes place, *i.e.* negotiating groups.

The three elements presented above are not conclusive. Other components of a negotiation system, such as computer support for strategies, for proposition offers, may be included in the proposed model. However, it should be noticed that the three elements presented above are inherently related with the non-monolithic aspect of the negotiation processes, while other components are usually related with a negotiator's perceptions/goals concerning the negotiation process.

In this paper, a model for electronic non-monolithic negotiations is presented. This model, named the contract-group-message model, includes a contract multiversion scheme, support for group dynamics and message exchange. The proposed model does not rely on any assumption concerning the type of non-monolithic negotations, *e.g.* coalition formation in multi-lateral negotiations or moderated negotiations. The paper is organized as follows. In section 2, the contract-group-message model, which consists of a contract model, a group dynamics model, and a message exchange model, is presented. Then, the concept of group protocol, used to structure the interactions among negotiators within a group, is detailed. Next, the design of support systems for non-monolithic negotiations with the help of the proposed model is discussed in section 4. A presentation of existing related work is given in section 5. Section 6 concludes the paper.

2. Contract-Group-Message Model

Three aspects of the negotiation process may be distinguished in the case of non-monolithic negotiations: contract edition, group dynamics, and message exchange.

2.1. *Contract Model*

In non-monolithic negotiations, the available amount of information is high. Negotiators may feel difficulties to apprehend all available data. Negotiators cannot keep track of all the information generated during the negotiation process. The cognitive overload results from the fact that each individual's capacity to process information remains fix while a group generates more data than a single person.

The cognitive overload forces negotiators to forget information. However, forgotten information could be useful to prepare or modify negotiation strategies and tactics. Negotiators would benefit from mechanism giving them access to past information.

As a consequence, a contract being the object of non-monolithic negotiations should be a *multiversion* contract. In the proposed contract model, a multiversion contract consists of various versions of the contracts, some of them being offers or counter-offers, while other versions could be draft versions. This implies that contract versions must be associated with a *version type*. A version type can be for instance "draft", "offer", or "counter-offer".

Contract versions are organized as an directed acyclic graph. In the *contract version graph*, edges capture the "answers to" relationship between the source version and the target version. In Figure 1, contract version 1.1.1 is an answer to contract version 1.1.

The "directed acyclic" aspect of the contract version graph allows a contract version to answer to many versions. In Figure 1, contract version 1.1.2 is an answer to both contract versions 1.1 and 1.2.

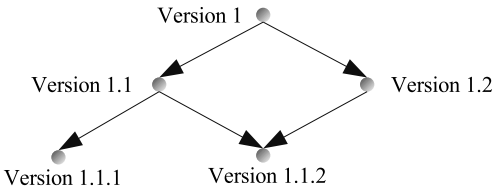

Figure 1. *Example of contract version graph*

More formally,

```
VersionGraph  = ({Version}, {Oriented_Arcs})
Version       = (VersionID, type)
Oriented_Arcs = (Version_src, Version_dest)
```

A contract version consists of a set of *parts*. A part is an atomic data unit. Its semantical and syntactical definition cannot be given as various contracts may use parts of different kinds. Examples of parts can be an XML element, an image, a link to a Web page, or some data capturing the structure of a contract as a list of references to other parts.

In the proposed contract model, a *multiversion contract* consists of *multiversion parts*, while a given contract version consists of given versions of these parts. It is assumed that all the versions of a contract are composed of the same set of parts. Differences between contract versions are reduced to differences between part versions. If a part is missing in a given contract version, the version of this part in this contract version is null. Adding a new part p to a given contract version v causes the addition of a multiversion part P to the whole multiversion contract. The value of a newly added multiversion part is p for contract version v, and it is $null$ for all the other contract versions.

Some multiversion parts may point to other parts to capture the structure of the contract. Such parts are called *composite multiversion parts*. A simple multiversion contract may consist of several multiversion parts, each one modeling a contract paragraph, and a composite multiversion part modeling the structure of the contract as a paragraph list. Another contract with additional semantics may consist of multiversion parts modeling various, semantically different contract parts, a multiversion part modeling the price, another multiversion part modeling the warranty, etc. A more complex contract may consist of composite multiversion parts to capture a tree structure of contract parts. The paragraph concerning the price may then be part of a section concerning offers, which is a part of the contract. At a higher level of abstraction, the structure of the contract may be complex, modeling semantics of various parts of a contract, *e.g.* addenda.

In the proposed contract model, a fixed structure of contracts is not assumed. The proposed contract model does not limit contracts with regard to their structure and allows new contract structures to be built on the top of the proposed multiversion contract model. Therefore, advanced contract structures (*e.g.* tree structured contracts) may be built using the concepts of multiversion members and multiversion composite members proposed in the multiversion contract model.

One-to-many relationships between part instances and contract versions are implemented as *association tables*. An association table associates each part instance with at least one contract version. An association table consists of rows, one row per part instance. Each row is a pair (`partInstanceID, set of contract versions associated with the given part instance`).

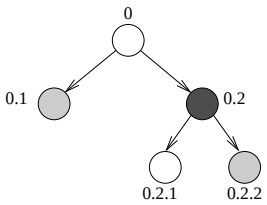

Figure 2. *Example of a version graph*

mvPrice

mvID: 11353427

svID	**contents**
sv_0	"price: 30€"
sv_1	"price: 25€"
sv_2	"price: 20€"

version parts

svID	**contract versions**
sv_0	$ver_0, ver_{0.2.2}$
sv_1	$ver_{0.1}, ver_{0.2.1}$
sv_2	$ver_{0.2}$

association table

Figure 3. *Example of a multiversion part for price*

Figures 3 and 4 illustrate the representation of two multiversion members of a multiversion contract. The contract version graph is assumed to be the one presented in Figure 2. The contract consists of a price and a warranty. The structure of the association table for the price is presented in Figure 3, while the structure of the association table for the warranty is presented in Figure 4. Analyzing simultaneously association tables for price and warranty, one may notice that price and warranty are dependent

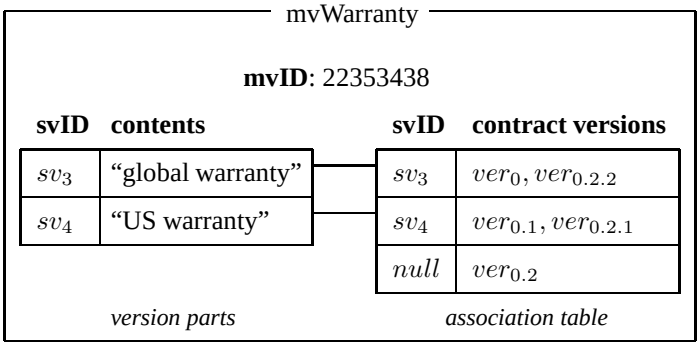

Figure 4. *Example of a multiversion part for warranty*

clauses, each new warranty corresponding to a new price and vice-versa. This result is obtained only from syntactic information, *i.e.* the relationships between negotiators proposals. No semantics concerning price or warranty is known. Therefore, association tables may be used as the basis for analysis of the negotiation process.

The contract model may be simply formalized as follows:

```
MvContract       = (VersionGraph, {MVPart})
MVParts          = ({PartInstance}, AssociationTable)
PartInstance     = (PartInstanceID, PartInstanceValue)
AssociationTable = {(PartInstanceID, {VersionID})}
```

Beside contract model, various actions may be executed. Two kind of actions on the contract may be distinguished. Modifications of the version graph take place with the help of *version actions*: creation of new versions of a given type and modification of existing version types may be examples of version actions. Modifications of contract versions take place with the help of *edition actions*. Edition actions may for instance allow for reading, editing, deleting, or adding a new part.

2.2. *Group Dynamics Model*

In non-monolithic negotiations, groups consisting of many negotiators, potentially from various parties, are the basic negotiation unit. Even when a single negotiator works alone on a proposal, it may be considered as a group consisting of only herself/himself. Therefore, it may be stated that *a group is a non-empty set of negotiators*.

Groups evolve: a negotiator may join or leave an existing group, a group may split in two or more groups, two or more groups may merge into a single group. Group

dynamics may be modeled by a set of *group actions*. The following group actions have been identified:

– `create` action: creates a new group;

– `join` action: adds a negotiator to the set of negotiators of an existing group;

– `merge` action: creates a new group consisting of the union of the set a of negotiators of at least two groups;

– `end` action: deletes an existing group;

– `leave` action: removes a negotiator from the set of negotiators of an existing group;

– `split` action: creates at least two groups from an existing group and the union of the sets of negotiators of the created groups equals the set of negotiators of the existing group.

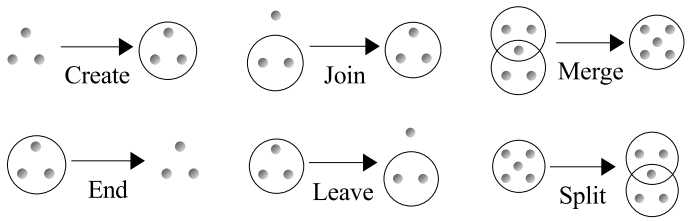

Figure 5. *Group actions*

Group actions are illustrated on Figure 5. Dots represent negotiators while circles represent groups. One may notice that, as shown on Figure 5 for the split and merge actions, a given negotiator may participate at a given time in many groups.

The introduction of the notion of *World* as the root-level group allows to reduce the set of basic action related to group dynamics. The *World* contains all negotiators and all groups are its descendants. Using the *World* concept, a `create` action may be seen as a `split` action on the *World*, while the `end` action may be seen as a `merge` action with the *World*.

2.3. *Message Exchange Model*

In a communicative approach, negotiations can be modeled as a structured message exchange. In this approach, negotiators are exchanging messages related to the negotiation process, *e.g.* counter-offers or additional information concerning a given offer. The structured aspect of the message exchange may come from the introduction of message types (Schoop *et al.*, 2001a). The following message types have been identified: offer, request, counter-offer, accept, reject, confirm, and information. Message

types are used to specified the intentions of negotiators and to limit misunderstanding that may occur during communication.

Moreover, some constraints may be set on sequences of message types to avoid communication non-senses. It makes no sense to answer a request for additional information by a counter-offer. A set of possible sequences of message types constitutes a *communication protocol*.

In non-monolithic negotiations, message exchange is not limited to messages concerning the contract. Messages may for instance explain why a negotiator is leaving a group. Therefore, message exchange should be extended to deal with dynamics of groups. Such an extension implies additional message types. These message types should reflect actions related with group splitting, merging, exclusion, etc.

In non-monolithic negotiations, a message consists of:

– some contents, usually a non semantically formalized text;

– a message type, used to structure message exchange;

– potentially an action: the action can be version action, edition action, or group action.

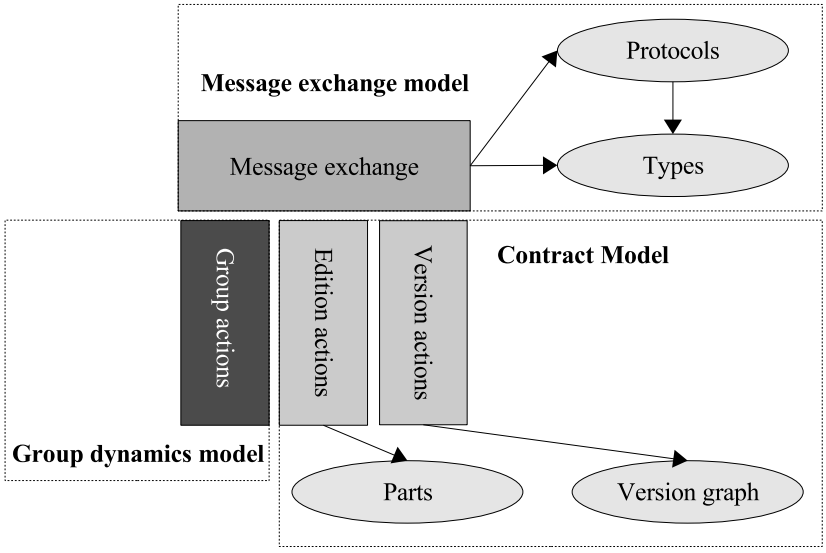

Figure 6. *Contract-group-message model*

The whole contract-group-message model is illustrated in Figure 6. It integrates all three models: contract model, group dynamics model, and message exchange model. Any kind of action that triggers an external program, such as *e.g.* a proposal evalution tool or a brainstorming framework, may be integrated with the proposed model.

3. Group Protocol

Based on the contract-group-message model, the interactions among negotiators within a given group may be structured by a *group protocol*.

As stated in (Kersten *et al.*, 2004), "the protocol is a formal model, often represented by a set of rules, which govern software processing, decision-making and communication tasks, and imposes restrictions on activities through the specification of permissible inputs and actions".

In the case of non-monolithic negotiations, potential activities, inputs, and actions may be different in every group. Therefore, the interations among negotiators in non-monolithic negotiations may potentially be structured in a different way within each group. Thus, each group may potentially be ruled by a different protocol.

3.1. *Behavioral Elements and Roles*

Let's first define the concepts of *behavioral element* and *roles*, which are the fundamental bricks of group protocols based on the contract-group-message model.

The concept of *behavioral element* is required to model relationships between actions and message types.

Behavioural element. A *behavioral element* is defined as a couple `(MessageType, Action)`.

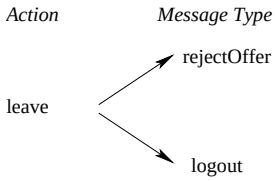

Figure 7. *Example of 1-to-n relationship between action and message types*

The relationship between actions and message types is a *n-to-n* relationship. Indeed, a 1-to-n relationship between action and message types may exists, as illustrated in Figure 7. In the presented example, one action – the `leave` action, triggered by a negotiator which wants to leave an existing group – may be associated with various message types, such as `rejectOffer`, when a negotiator wants to send a message concerning offer rejection, or `logout` when a negotiator wants to log out the system.

Reciprocally, a 1-to-n relationship between message type and actions may exists, as illustrated in Figure 8. In the presented example, a message type – the `rejectOffer` message type, triggered by a negotiator which does not accept a previously proposed offer – may be associated with various actions, such as a `leave` action to leave an existing group or `end` action to delete an existing group.

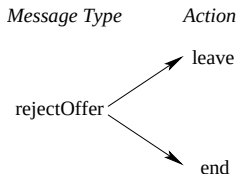

Figure 8. *Example of 1-to-n relationship between message type and actions*

The concept of *behavioral element* enables n-to-n relationships between actions and message types.

Role. A *Role* r consists of a set of behavioral elements, denoted r_{be}, and a role name, *e.g.* `buyer`, `seller`, or `moderator`.

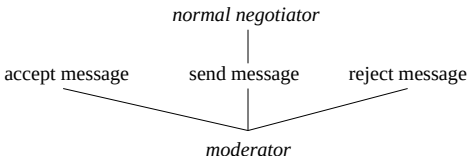

Figure 9. *Example of n-to-n relationship between behavioral elements and roles*

The relationship between behavioral elements and *roles* is a n-to-n relationship, as presented in Figure 9. First, in the presented example, one role – the `moderator` role – may be associated with various behavioral elements, such as "send message", "accept message", or "reject message". Second, on the same example, the behavioral element "send message" is associated with many roles, such as "moderator" or "normal negotiator".

Social Behavioural Element. For a given role r, the set of pairs $(role\_name, be)$, with $be \in r_{be}$ is denoted r_{seb}. A pair $(role\_name, be)$ is called a *social behavioral element*.

Using the example presented in Figure 9, the social behavioral element `(moderator, accept message)` is defined, while the social behavioral element `(normal user, accept message)` is not defined. For the role `moderator`, r_{seb} is a set containing the following social behavioral elements: `(moderator, accept message)`, `(moderator, send message)`, `(moderator, reject message)`.

3.2. *Group Protocol Definition*

In the proposed model, a group protocol is a formal model, represented by a finite state machine, which govern negotiation processes within a group, and imposes

restriction on activities through the specification of permissible actions depending on both the state of the negotiation process and social roles of acting negotiators.

The base concepts of the proposed model are the same as the base concepts underlying the finite state machine model: a group protocol consists of a set of *states*, a set of permitted *inputs*, a *transition function*, a *start state*, and a set of *end states*. The transition function returns `true` if a given input may cause a transition from one state to another state, `false` otherwise.

In the proposed model, the notion of input has been extended to take into account social aspects of negotiation processes. An input is defined as a social behavioral element. Therefore, the set of possible transitions depends both on social roles of negotiators and triggered actions.

It should also be noticed that the notion of role is associated with both a group and a negotiator, *i.e.* a negotiator may play various different roles in different groups. Moreover, a given role may be played by various negotiators within a given group, *e.g.* many buyers may negotiate within a group in which an auction process is taking place.

Group Protocol. A group protocol p consists of:

– a set of states S,

– one starting state $s_0 \in S$,

– a set of ending states $S_{ending} \subset S$, with $s_0 \notin S_{ending}$,

– a set of roles R,

– a transition function t from $(S - S_{ending}) \times S \times \bigcup_{r \in R} r_{seb}$ to $\{true, false\}$.

3.3. *Group Protocol Validity*

The former definition specifies the basic requirements for group protocols. It links the notions of states, roles, and behavioral elements. However, the former definition does not ensure that such protocols are valid, neither structurally, nor semantically.

Structural validity. A protocol is *structurally valid* iff:

1) there is a path from the starting state to all states,

2) there is a path from every state to an ending state,

3) from a given state, there is only one transition associate with a given social behavioral element.

The two first conditions ensure that no state are neither non-accessible (and therefore should be suppressed from the protocol), nor leading to a lock in the negotiation process, *i.e.* forbids the negotiation process to be finished.

The third condition ensures that no ambiguity exists in a group protocol. An ambiguity may occur when two or more transitions associated with one social behavioral

element be may lead to two or more states from a single state s. In this case, it is not possible to decide to which state the social behavioral element be leads to.

Formally, the last condition for structural validity may be formulated as follows:

$$\forall s \in (S - S_{ending}), \exists (s_i, s_j) \in S^2, seb \in \bigcup_{r \in R} r_{seb} \\ t(s, s_i, seb) = t(s, s_j, seb) = true \Rightarrow s_i = s_j$$

Semantical validity. A protocol is *semantically valid* iff:

1) all transitions to ending states are associated with social behavioral elements containing an ending action,

2) social behavioral elements containing an ending action are associated only with transitions leading to ending states.

An action a is an *ending action* iff the life of the group g ends when action a is called in group g. The life of a group g ends when no more message can be sent to group g. The set of ending actions of a given protocol is denoted $EndingActions$. One may notice that the group action end is obviously an ending action.

The first condition ensures that a transition leading to an ending state really ends the life of the group. The second condition ensures that the group protocol cannot be "interrupted" by a transition associated with an ending action.

Formally, the first condition for semantical validity may be formulated as follows:

$$\forall (s_i, s_j, seb_k) \in (S - S_{ending}) \times S_{ending} \times \bigcup_{r \in R} r_{seb}, \\ t(s_i, s_j, seb_k) = true \Rightarrow a_k \in EndingActions$$

where a_k is the action associated with the social behavioral element seb_k.

The second condition for semantical valitidy may be formulated as follows:

$$\forall (s_i, s_j, seb_k) \in (S - S_{ending}) \times (S - S_{ending}) \times \bigcup_{r \in R} r_{seb}, \\ t(s_i, s_j, seb_k) = true \Rightarrow a_k \in EndingActions$$

The concepts of structural validity and semantical validity are illustrated on Figure 10. The case a) is an example of a valid group protocol, while the cases b) and c) are respectively examples of structurally and semantically invalid group protocols. In the case b), the group protocol is structurally invalid for two reasons: first, there is no path from the starting state to the state s_2. Second, there is no path from the state s_1 to the ending state s_4. In the case c), the group protocol is semantically invalid for two reasons: first, the transition t_1, which is not associated with an ending action, leads from the state s_1 to the ending state s_4. Second, the transition t_2, which is associated with an ending action, leads from the state s_2 to the state s_3, which is not ending state.

4. Designing Support Systems for Non-Monolithic Electronic Negotiations

The proposed contract-group-message model may be used to design support systems for non-monolithic negotiations. Following the model, specification of a support

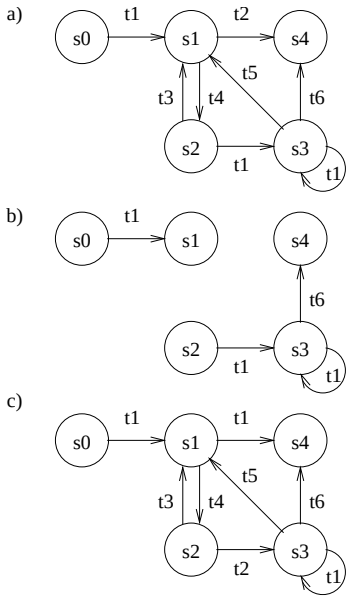

Figure 10. *Examples of a) valid, b) structurally invalid, c) semantically invalid group protocols, where $s_0, s_1, \ldots, s_4$ are states, s_4 being the only ending state, and $t_1, t_2, \ldots t_5$ are transitions, t_2 and t_6 being the only transitions associated with an ending action*

system for non-monolithic negotiations involves three areas: contract specification, message exchange specification, and role specifications. The last element to be specified during the design of a support system for non-monolithic negotiations is the protocol definition.

4.1. *Contract Specifications*

Before the negotiation process starts, the object of the negotiation and the way it is represented and accessed as a document (contract) have to be specified. This specification aspect, named contract specifications, implies the specifications of contract parts, as well as specifications of methods to access the contract, *i.e.* version and edition actions.

4.2. *Message Exchange Specifications*

Before the negotiation process starts, the way negotiators will exchange messages, as well as the messages they may exchange, have to be specified. This specification

aspect, named message exchange specifications, implies the specifications of available message types, as well as specifications of negotiation protocols (cf. Section 3.1).

4.3. *Roles Specifications*

An individual may play various roles during the negotiation process. Roles are used to define and control the prerogatives of negotiators during the negotiation process. A role is defined by:

– a role identifier, such as “buyer”, “seller”, or “moderator”;

– a set of available version actions: used to control access to the version graph (not everybody should be able to create a counter-offer version);

– a set of available editing actions: used to limit access to the document contents (not everybody should be able to modify the price clause);

– a set of available group actions: used to control the capabilities of negotiators to influence the group dynamics (not everybody should be able to create new groups or to end a group);

– a set of available message types: used to control message exchange (not everybody should be able to post a binding message).

4.4. *Protocols Specifications*

When roles have been specified, the next step consists in specifying various group protocols which may be used during the negotiation process to structure interactions among negotiators within multiple co-existing groups.

The protocols specifications may be summarized to the following set of tasks:

– identification of groups that may be constituted during the whole negotiation process,

– for each group, identification of roles that may be played within the group,

– for each group, identification of all potential states of the group,

– for each group, specification of transitions leading from one state to another, *i.e.* specification of the transition function.

5. Related Work

Three approaches to negotiation support systems are of particular interest for non-monolithic negotiations, as they address partially some of the three previously presented elements: the *Agora* approach, the *Doc.Com* approach, and the *NeSSy* approach.

In the first approach, *i.e.* the *Agora* approach (Cellary *et al.,* 1998), negotiators are exchanging messages in virtual negotiation rooms while coediting a contract. The most interesting aspect of this approach is the idea of combined message exchange and contract edition. However, it lacks support for group dynamics. Moreover, both message exchange model and contract edition model are too simple for non-monolithic negotiations: the message exchange is modeled as a list of plain text messages; the contract model allows negotiators to work simultanously on various clauses of a given version but does not track relationships among versions.

The second approach, *i.e.* the *Doc.Com* approach (Schoop *et al.,* 2003)(Schoop, 2001)(Searle, 1969) (Weigand *et al.,* 2003), is based on a communicative approach and models the negotiation process as message exchange on a multiversion contract. The most interesting aspect of this approach is the idea of structured message exchange with the introduction of message type and message exchange protocol. However, the lack of support for group dynamics and a too simple contract versioning scheme – only one version may be modified at a given time – are a major obstacle to an application to non-monolithic negotiations.

The third approach, *i.e.* the *NeSSy* approach (Picard, 2002)(Picard, 2003)(Picard *et al.,* 2002), addresses the problem of mass electronic negotiations. In this approach, the negotiation process is modeled as a multiversion contract and analysis tools are provided to build synthetic views of the negotiation process. Even if the contract versioning scheme proposed in this approach may be used for non-monolithic negotiations, this approach lacks support for both group dynamics and message exchange.

6. Conclusion

Non-monolithic negotiations have received little attention from the electronic negotiations research community, even if negotiators are often facing this kind of negotiations. The proposed contract-group-message model aims at providing a base for process support systems for non-monolithic negotiations, integrating three elements: collaborative edition of the negotiated contract, message exchange and group dynamics. To our best knowledge, it is the first model for electronic support to non-monolithic negotiations.

It integrates two approaches to negotiation process support systems – document-based and communication-based approaches – which were usually isolated, although complementary. The proposed multiversion contract model allows negotiators to work in parallel on various contract versions, which corresponds to the reality that negotiators are facing in non-monolithic negotiations. The message exchange support allows negotiators to argue on the negotiation process in a more efficient way as the introduction of message types may be used to limit ambiguities which may arise during the communication. One may also notice that, as actions may be executed only *via* message exchange, the edition process is linked with the communication process. There-

fore, not only the contract versions may be retrieved but also the messages associated with the various contract modifications.

The introduction of the group dynamics leads to introduction of new actions. As for collaborative contract edition, the group dynamics is closely coupled with the message exchange, as group dynamics always take place *via* messages. Therefore, negotiators have to justify their group "behavior", as they have to send a message to other negotiators to *e.g.* join, split or leave a group.

The notion of protocols based on roles and behavioral elements enables the design process support systems that would take into account both social characteristics of negotiators – *via* roles – and features offered by a given system – *via* actions.

A prototype is currently developed to estimate the pertinence of the contract-group-message model for non-monolithic negotiations. A complex task is the development of the graphical user interface, as in the case of non-monolithic negotiations, negotiators have to face a high amount of data which should be presented in a comprehensible way. This implies that the graphical user interface should take into account the notion of "social context", which is related with the notions of group and action: a negotiator may like to check who is working/has left/has join in her/his group, another may like to have only access to all messages she/he has exchanged with other negotiators from the beginning of the negotiation.

Among future works, as explained above, the notion of "role" should be extended to take into account not only behavioral elements but also the "social context". Moreover, further modeling to ensure coherence of message types and available actions would be a valuable contribution to the presented model.

7. References

Cellary W., Picard W., Wieczerzycki W., « Web-based Business-to-Business Negotiation Support », *Int. Conference on Electronc Commerce TrEC-98*, dpunkt.verlag, Hamburg, Germany, p. 80-89, 1998.

Kersten G. E., Strecker S. E., Lawi K. P., « Protocols for Electronic Negotiation Systems: Theoretical Foundations and Design Issue », *The 5^{th} Conference on Electronic Commerce and Web Technologies, EC-Web04*, IEEE Computer Society, Sarragoza, Spain, 2004.

Kersten G., Lo G., « Negotiation Support Systems and Software Agents in E-business Negotiations », *The First International Conference on Electronic Business*, Hong Kong, Dec. 2001.

Lomuscio A. R., Wooldridge M., Jennings N. R., « A Classification Scheme for Negotiation in Electronic Commerce », *Agent Mediated Electronic Commerce*, Springer LNAI, p. 19-33, 2001.

Picard W., Multi-facet Analysis of e-Negotiations, Ph.d. thesis, École Nationale Supérieure des Télécommunications, Paris, France, 2002.

Picard W., « NeSSy: Enabling Mass E-Negotiations of Complex Contracts », *4th Int. Workshop on Negotiations in Electronic Markets – Beyond Price Discovery, associated to the DEXA Conference*, IEEE Computer Society, Prague, Czech Republic, p. 829-833, 2003.

Picard W., Cellary W., « Electronic Negotiations in a Highly Concurrent Environment », *2nd IFIP Conference on E-Commerce, E-Business, E-Government, I3E*, Kluwer Academic Publishers, Lisbon, Portugal, p. 525-536, 2002.

Raiffa H., *The Art and Science of Negotiation*, The Belknap Press of Harvard University Press, Cambridge, Massachusetts and London, England, 1982.

Rangaswamy A., Shell G. R., « Using Computers to Realize Joint Gains in Negotiations: Toward an "Electronic Bargaining Table" », *Management Science*, vol. 43, n° 8, p. 1147-1163, 1997.

Schoop M., « An Introduction to the Language-Action Perspective », *SIGGROUP Bulletin*, vol. 22, n° 2, p. 3-8, 2001.

Schoop M., Jertila A., List T., « Negoisst: A Negotiation Support System for Electronic Business-to-Business Negotiations in E-Commerce », *Data and Knowledge Engineering*, vol. 47, n° 3, p. 371-401, 2003.

Schoop M., Quix C., « DOC.COM, a framework for effective negotiation support in electronic marketplaces », *Computer Networks*, vol. 37, n° 2, p. 153-170, 2001a.

Schoop M., Quix C., « DOC.COM, Combining document and communication management for negotiation support in business-to-business electronic commerce », *Proceedings of the 34th Hawaiian International Conference On System Sciences (HICSS-34)*, IEEE Press, Maui, Hawaii, Jan. 2001b.

Searle J., *Speech Acts – An Essay in the Philosophy of Language*, Cambridge University Press, Cambridge, United Kingdom, 1969.

Ströbel M., Weinhardt C., « The Montreal Taxonomy for Electronic Negotiations », *Journal of Group Decision and Negotiation*, vol. 12, n° 2, p. 143-164, 2003.

Weigand H., Schoop M., De Moor A., Dignum F., « B2B Negotiation Support: The Need for a Communication Perspective », *Group Decision and Negotiation*, vol. 12, n° 1, p. 3-29, 2003.

Wurman P., Wellman M., Walsh W., « A Parameterization of the Auction Design Space », *Games and Economic Behavior*, vol. 35, p. 304-338, 2001.

Yuan Y., Head M., Du M., « The effects of multimedia communication on Web-Based Negotiation », *Group Decision and Negotiation*, vol. 12, n° 2, p. 89-109, 2003.