Negotiation processes within inter-organizational alliances

Adina CRETAN
Faculty of Cybernetics, Statistics and Economic Informatics
Academy of Economic Studies, Bucharest, România
badina20@yahoo.com

This paper describes the negotiation component of E-Alliance, a software infrastructure defined for supporting negotiation activities in concurrent inter-organizational alliances. The E-Alliance's main intent is to preserve the autonomy of organizations grouped in an alliance. The purpose of this work is to offer support for small and medium enterprises which cannot or do not want to fulfill a big contract alone. This approach is illustrated by a sample scenario where partners are printshops grouped into an alliance to better accomplish customers' demands.

Keywords: *negotiation, middleware, virtual enterprises, multi-agent systems, interaction protocol*

1 Introduction

Nowadays, the number of virtual enterprises is increasing, in particular thanks to the development of Internet. In the context of virtual enterprises, B2B interactions need to be explicitly constrained by general rules of behavior agreed upon by the participants.

We therefore seek to achieve an intermediate solution to provide support to the collaborations within an alliance of organizations and we propose *negotiation* as a fundamental mechanism for these collaborations. The E-Alliance infrastructure allows an organization to dynamically join or leave an alliance and make autonomous decisions to progress in its collaborations. Negotiation-based collaborations may occur asynchronously following very different patterns, but of course do not preclude prior agreement between the alliance members as to how negotiation should execute. An *e-alliance* is more structured than a shared dataspace for collaborative work, but less structured than a workflow assigning precise roles to participant organizations and tightly scheduling their interactions.

In this paper we describe the current status of E-Alliance, showing how organizations participate to and control the status of the negotiations and how the alliance life-cycle is managed. To illustrate our approach, we use a sample B2B scenario (Sec. 2) where autonomous printshops form an alliance to better accomplish their customers' requests.

2. Scenario

We consider a scenario (Andreoli et al., 2000) of collaborations within an alliance of distributed autonomous printshops. The alliance is a dynamic entity where new printshops may join or leave. A printshop manager interested in joining an alliance fills in an adhesion contract with information on his printshop competencies and preferences. If the alliance committee accepts it as a new partner, the new member commits to respecting the rules of the alliance and the adhesion contract and introduces itself to the other partners.

Each printshop autonomously manages its contracts, schedules etc. When a print request reaches a printshop, the manager analyses it to understand if it can be accepted, taking into account job schedules and resources availability. If the manager accepts the print request, he may decide to perform the job locally or to (partially) outsource it, given the printshop resource availability and technical capabilities. If the manager decides to outsource a job, he starts a negotiation within the alliance with selected participants. The manager may split the job into slots, notifying the partners about the outsourcing requests for the different slots. If the negotiation results in an agreement, a contract is settled between the outsourcer and the insourcer printshops, which defines an inter-organizational workflow enacting the business process fulfilling the outsourced jobs and a set of obligation relations among participants.

3. E-Alliance Requirements and Goals

The printshops alliance scenario shows a typical example of the e-alliances targeted by E-Alliance: virtual alliances where partner organizations may *a priori* be in competition with each other, but may want to cooperate in order to be globally more responsive to market demand. A lot of flexibility and coordination among the partners is needed to publish selected information, reach agreements on how and when to accomplish customers' requests, execute and monitor contracts, handling changes. E-Alliance main goal is to provide a software support for inter-organizational alliances enabling:

1. management of an alliance's life-cycle, including services for information publishing, partners authentication, joining/leaving the alliance;
2. collaborative activities among alliance partners, through services enabling the partners to negotiate, execute and monitor contracts.

E-Alliance should flexibly support negotiation activities in the alliance respecting the autonomy of the partners without statically attaching each negotiation participant a role according to a strict protocol. Also, the mechanisms supporting such collaborations should be generic enough to adapt to any B2B context.

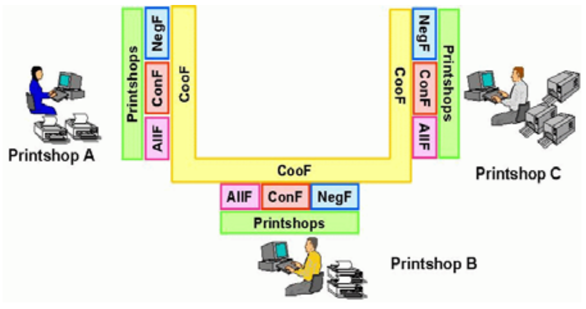

Fig.1. E-Alliance software infrastructure

Moreover, E-Alliance should help the partners to augment their efficiency and ability to react to unforeseen situations, thus improving their market competitiveness. These issues have not been completely explored yet, although a lot of work has been done on other aspects (e.g. how to define payment mechanisms). We focus instead on how to:
(1) represent decentralized organizations; (2) model the coordination of different concurrent interactions; (3) formalize negotiations; (4) deploy and maintain an alliance during its lifecycle; (5) create and administrate contracts. Next section describes how the E-Alliance approach takes into account the discussed requirements.

4. E-Alliance Approach

The E-Alliance infrastructure proposes a multi-level architecture for providing services to assist alliance partners along their collaborative concurrent activities taking into account the aspects discussed in Sec. 3. The infrastructure (Fig.1) is organized in three layers. A first, application dedicated, layer specializes the generic mechanisms provided by the other two layers according to the specific domain, e.g. the printing domain. A second layer is dedicated to the support of job insourcing/outsourcing within an alliance and comprises three facilities: *AllF* (alliance life-cycle management), *ConF* (contract management), and *NegF* (negotiation). The third, middleware and coordination, layer (*CooF*) offers generic mechanisms to enact negotiations in a distributed environment. The *CooF* is shared across the partner sites, while the two other layers are replicated on each partner site, enabling a decentralized negotiation and preserving the autonomy of the partners.

5. Alliance Facility

The goal of the *AllF* facility is to support an alliance life-cycle including: new members subscriptions and members departures, modifications of adhesion contracts, of members preferences, of the global rules of the alliance. This addresses two major issues: (1) what kind of software architectures cope at once with autonomy, openness and evolution requirements of alliances; (2) what processes to put in place in order to specify, enact, deploy such alliances. In order to maintain the global state of the alliance and to provide managers with the appropriate information, an *AllF* supervises the activities of the *NegF* and *ConF* to check whether the rules of the alliance are respected or not. It also gathers information in order to build a global history of the system. If an event in the life of the alliance has an impact on ongoing negotiations and contracts, the *AllF* interacts with the concerned facility in order to maintain the global coherence of the system. The information manipulated within the alliance includes *global* information, e.g. adhesion contracts, and partner *local* information, e.g. its representation of the others. An adhesion contract expresses the engagements between the alliance and a member, e.g. services the member will provide.

E-Alliance provides a software environment offering the users means for dynamically
adding / retracting / replacing software components, without interrupting the system execution. The underlying approach relies on modeling an alliance life-cycle using the Zeta (Alloui and Oquendo, 2001) architecture description language and generating an executable code from the description into a target implementation environment called ProcessWeb (PML, 1996).

The resulting software environment allows the *AllF* to communicate with both *NegF* and *ConF* facilities, e.g. to provide the *ConF* with rules to apply to a contract or to record in the history a new contract or negotiation.

6. Contract Facility

The *ConF* facility of a partner of the alliance manages the execution of the contracts in which that partner is involved in. The management of a contract is organized around three main steps: (1) *creation*; (2) *execution*; (3) *closing*.

During the creation, the *ConF* of the principal contractant defines a contract from the terms of the agreement reached during the negotiation by its *NegF* and the *NegFs* of the other participants in the negotiation. A lot of B2B contract models have been proposed in the literature (Grefen and Angelov, 2001). In E-Alliance a contract is composed of a business process, fulfilling the negotiation agreements, and of a normative and policy structure, ruling the participants' behavior. Using the *M*OISE+ model (Hubner et al., 2002), we define the structure of a contract as a set of *roles* linked with each other with authority and communication *links*. This structural schema sets the authority structure that governs the contract. The responsibilities of the *ConF* of each contract participant are defined by linking the *roles* it can play with the part of the business process that it has to execute. These links are expressed as obligations or permissions, and are qualified by penalties in case a participant cannot fulfill a task it is responsible for.

Executing a contract consists of the distributed execution and enactment of an inter-organizational workflow between the participants and it is supported by the *CooF*. Different events may stop the execution of a contract and imply modifications of it. These events may be communicated by the *AllF* as a consequence of a change in the alliance itself. The *ConF* will interact with the *NegF*, if a new negotiation is needed, and with the *AllF* to make it aware of the penalties for the participants.

7. Negotiation Facility

The *NegF* agent of a partner manages the negotiations the partner is involved in. Fig. 2 shows the architecture of a *NegF*, which assists its manager at a global level (negotiations on *different jobs*) and at a specific level (negotiation on the *same job* with different participants) by coordinating itself with the *NegF* of the other partners through the *CooF*.

A negotiation is organized in three main steps: initialization; refinement of the job under negotiation; and closing. The initialization step allows to define what has to be negotiated (*Negotiation Object*) and how (*Negotiation Framework*). A selection of negotiation participants can be made using history on passed negotiation, available locally or provided by the *AllF*. Following the approach

in (Vercouter, 2000), each participant has its own representation of the other participants and uses it to build a network of dependence relations (Sichman et al., 1994). In the refinement step, which relies on a set of speech acts (Carron et al., 1999), participants exchange proposals on the negotiation object trying to satisfy their constraints.
The manager may participate in the definition and evolution of negotiation frameworks and objects. Decisions are taken by the manager, assisted by his *NegF* agent. Decision functions operate in the "Reasoning" box (Fig. 2), totally or partially automating the negotiation. For each negotiation, a *NegF* manages one or more negotiation objects, one framework and the negotiation status, detailed in Sec. 8.

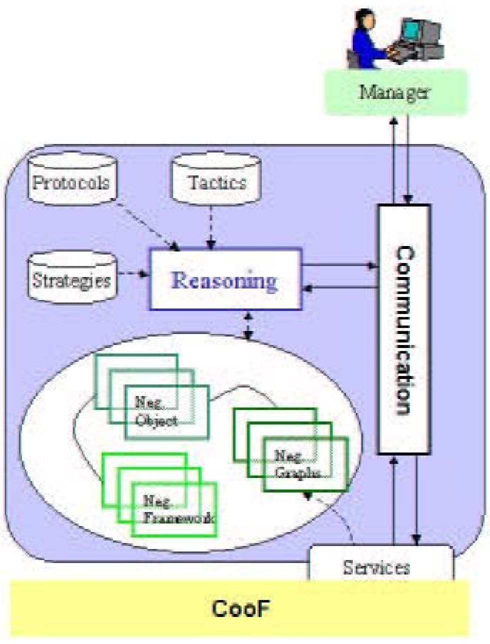

Fig.2. The architecture of the *NegF* facility

A negotiation object can be composed of several negotiation objects, which can be interdependent thus expressing interdependencies among concurrent negotiations.
Negotiation Frameworks gather requirements of managers on negotiations, formalizing plans for the interaction process and the degrees of autonomy in decisions and actions of the *NegF*. A negotiation object has a unique negotiation framework, but a negotiation framework can cover several negotiation objects.

8. Negotiation Middleware

The *CooF* is the negotiation oriented coordination middleware that supports the different processes provided by the facilities in the second layer of the E-Alliance infrastructure. It is an extension of the CLF middleware (Andreoli et al., 1999) aiming at enriching its negotiation support capabilities (Andreoli and Castellani, 2001). In CLF all the components are viewed as resource managers.
Resources can be "tangible" elements, e.g. a printer, as well as more virtual entities, e.g. a print task. CLF components make visible their resources through interfaces that define abstract services through which operations on resources are made possible. The interaction with a CLF component through one service of its interface follows a specific protocol, defined by eight "interaction verbs", similar to speech acts, which have a meaning in terms of resource manipulations (discovery, selection, insertion and destruction). The coordination of these components, considered as resource managers, is expressed by means of high-level rule-based scripts hiding the communication protocol and directly expressing the desired resources manipulations. Specific CLF components, called *coordinators*, translate scripts into invocations of the protocol on different components, realizing the abstract resource manipulation prescribed by the script. Thus, the coordinators can be considered as generic clients of the middleware platform and the client side of a CLF application can be expressed as a set of scripts. As an example, a printshop in the alliance scenario could be represented by a CLF component offering services for outsourcing/insourcing jobs (Andreoli et al., 2000). The resources held by this component are decisions to outsource or insource a job.
From CLF to a Negotiation Middleware
The CLF protocol allows to perform multiple dependent searches for resources held by several components, but it enables only a "uni-directional" propagation of the information through the involved components. Indeed, each response sent by a server in the search phase must be a "complete" specification of the *actual* resource (e.g. a given print

job) to be used in the enactment phase if and when it is performed. The server cannot return "partial" answers describing a set of *potential* resources and then letting the client refine that set in order to converge towards the resource to be used in the enactment, if any. The *CooF* is an extension of CLF supporting a multi-party, "multi-directional", multi-attribute negotiation in the search phase of the execution of coordination scripts. It allows the resources that trigger a rule to be negotiated by successive refinements between the components involved in the negotiation.
The search tree thus becomes a "negotiation graph", which captures the dependencies between the negotiation interactions.

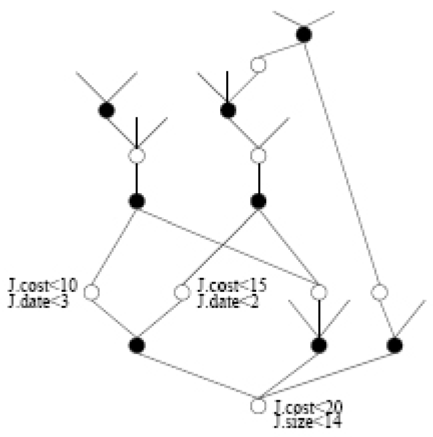

Fig.3. An example of negotiation graph

A negotiation graph is a directed bi-colored graph expressing the topological structure of a negotiation: white nodes characterize the contexts in which decisions are taken; black nodes characterize alternatives in a decision. Each context (white) node in the graph contains constraints on different issues for the parameters of the service execution that is being negotiated. For example, for the services outsrc(job) and insrc(job), the (here unique) parameter of the negotiation (job) is a print job and an issue can be the price which can assume a range of possible values. Different branches of negotiation can be created in the negotiation graph to explore alternatives in terms, say, of price. The partners may then refine each branch specifying different delays.
The interaction specifying the delay would occur in the context of one or the other branch created by the interaction concerning the price. Fig. 3 shows an example of a negotiation graph.
A negotiation process is modeled as the collaborative construction of a negotiation graph among the negotiation participants. For example, a proposal made by the printshop who initiated the negotiation is represented in the graph copy visible by the outsrc service and the proposals made by printshops involved in the negotiation are represented in the graph copies visible through their insrc services. The purpose of the *CooF* protocol is to allow the synchronization of the different copies (Andreoli and Castellani, 2001). So, the partners do not communicate directly, but only via a set of operations, that they perform on their negotiation graph copies and which are appropriately propagated by the *CooF* infrastructure.

9. Conclusions

This approach proposes a decentralized multi-issue negotiation model in which a set of agents can conduct several one-to-one conversations in a concurrent manner according to the same middleware protocol. So, each agent is able to distinguish between its possible multiple bi-lateral negotiations that characterize a negotiation state and to act according to them. This paper aims at modeling the negotiation process at least at three levels (middleware, multi-agent, and human).
All three levels must cooperate and interact in a coherent way to build the negotiation process. The middleware level manages generic negotiation mechanisms such as the propagation of alternatives, enactment and transaction primitives.
The multi-agent level provides the Manager with semi-automatic negotiation mechanisms, given the specification of protocols, tactics. Finally, the Manager, at the moment, takes the decision and makes the negotiation progress via his(er) NegF.

References

[1]. I. Alloui and F. Oquendo. Software Process Architecture = Process Components

+ Intentionbased Interactions. In *Proc. of PDPTA*, Las Vegas, Nevada, 2001.
[2]. J.M. Andreoli, D. Arregui, F. Pacull, M. Riviere, J.Y. Vion-Dury, and J. Willamowski. CLF/Mekano: a Framework for Building Virtual-Enterprise Applications. In *Proc. Of EDOC*, Manheim, Germany, 1999.
[3]. J-M. Andreoli and S. Castellani. Towards a Flexible Middleware Negotiation Facility for Distributed Components. In *Proc. of "E-Negotiations" (DEXA)*, Munich, Germany, 2001.
[4]. J.M. Andreoli, S. Castellani, and M. Munier. AllianceNet: Information Sharing, Negotiation and Decision-Making for Distributed Organizations. In *Proc. of EcWeb*, Greenwhich, U.K., 2000.
[5]. T. Carron, H. Proton, and O. Boissier. A Temporal Agent Communication Language for Dynamic Multi-Agents Systems. In *Proc. of 9th MAAMAW*, LNAI 1647, 1999.
[6]. A. Cretan – "Virtual Alliances among Autonomous Organizations". In *Digital Economy* – The sixth International Conference on Economic Informatics, Bucharest, Romania, Ed. Economica, May 8-11, 2003.
[7]. A. Cretan, "Intelligent Solutions for Virtual Negotiations", in *Information & Knowledge Age* – The seventh International Conference on Informatics in Economy, Bucharest, Romania, Ed. Economica, May 19-20, 2005.
[8]. P. Grefen and S. Angelov. B2B eContract Handling – A Survey of Projects, Papers and Standards. TR-01-21, CTIT, University of Twente, 2001.
[9]. J. Hubner, J.S. Sichman and O. Boissier. A model for the structural, functional and deontic specification of organizations in multiagent systems. In *Proc. of SBIA'02*, LNAI 2507, pp. 118128, Porto de Galinhas, PE, Brazil, 2002.
[10]. ProcessWise Integrator PML Reference Manual. ICL/PW/635/0, Versions 4.1-4.5, April 1996.
[11]. J.S. Sichman, R. Conte, Y. Demazeau and C. Castelfranchi. A social reasoning mechanism based on dependences networks. In *Proc. of ECAI*, Amsterdam, The Netherlands, 1994.
[12]. L. Vercouter. A distributed approach to design open multi-agent systems. In *Proc. of ESAW*, 2000.