Negotiation Coordination Model for Supporting Enterprise Interoperability

A. Cretan, C. Coutinho, B. Bratu and R. Jardim-Goncalves

Abstract The scientific evolution of negotiation in collaborative working environments allowed companies to benefit from new interoperability standards. The proliferation of SMEs led to a highly competitive environment, in which the various partners rely on interoperability and collaboration to be efficient. This paper highlights the role of negotiation in solving interoperability issues by proposing a distributive coordination model in order to manage multiple parallel negotiations. The research results will be validating within the European research project H2020 C2NET.

Keywords Enterprise interoperability · Negotiation · Coordination rules · Model-driven · Product design

A. Cretan
"Nicolae Titulescu" University, 185 Calea Văcăreşti, District 4,
040051 Bucharest, Romania
e-mail: badina20@yahoo.com

C. Coutinho (✉)
Caixa Mágica Software, Rua Soeiro Pereira Gomes, Lote 1-4 B,
1600-196 Lisbon, Portugal
e-mail: carlos.coutinho@caixamagica.pt

B. Bratu
Atos Big Data and Security—R&D, Rue Jean Jaurès B.P. 68,
78340 Les Clayes-Sous-Bois, France
e-mail: bbratu@yahoo.com

R. Jardim-Goncalves
CTS, Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa,
UNINOVA, Lisbon, Portugal
e-mail: rg@uninova.pt

K. Mertins et al. (eds.), *Enterprise Interoperability VII*,
Proceedings of the I-ESA Conferences 8, DOI 10.1007/978-3-319-30957-6_7

1 Introduction

In order to survive in a global economy, enterprises, especially Small and Medium Enterprises (SMEs), must collaborate by exchanging information. In this respect, Enterprise Interoperability (EI) is defined as the capacity of an enterprise to interact and to exchange information with others within a collaborative networked environment [1]. In this environment, any change in any network partner affects the others, leading to system interoperability breaking. In this context, sustainable EI (SEI) is defined as the ability of maintaining interoperability along the enterprise systems and applications [2].

This paper highlights the role of negotiation in resolving these discrepancies, and proposes a generic coordination negotiation model by describing coherent sets of rules that manage multiple bilateral negotiations to support the interoperability within the collaborative working environments.

2 Related Work

In order to support sustainable interoperability, the proposed approach states that one important aspect is the designing of intelligent software components. These components are able to support the communication, coordination and collaboration activities at all levels within the networked environment. This paper tackles two main issues related to low-level business decision support and interoperability breaking among different structures of the company. Regarding this, our approach proposes the use of intelligent agents for decision support at business level. For example, it is presented in [3] an interesting approach by creating different agents with Expert Systems. Previous papers [4], have used Artificial Intelligence (AI) and algorithms in the supervised learning process, achieved through Restricted Bolzmann Machines (RBM) and employed as latent factor analysis application [5].

However, these procedures providing flexible ways of making inferences have not been truly exploited in Multi-Agent Systems (MAS). In this regard, this paper proposes MAS models for supporting the coordination of negotiation process within the dynamic environment. In addition, our approach does not promote the entire substitution of human decision, but proposes the agent technology helping to communicate results among the different software systems. In this context, rule-based systems are the main AI technique that has been added to software agents. As first objective, the proposed negotiation coordination model based on MAS serves to break boundaries within a negotiation environment represented by the contracting authority, several contractors and subcontractors. In second, the proposed solution should be as generic as possible in order to be successfully employed in heterogeneous business environments addressing dynamical changes in the product offers or the business processes, workflows, policies and rules of an enterprise or group of enterprises.

3 Negotiation System with MAS

There are three main steps to the negotiation process: Preparation; Refinement of the job under negotiation; and Closure. The first step, Preparation, establishes the *Negotiation Object* (i.e., the task that will be negotiated) and the *Negotiation Framework* (i.e., the manner in which the Negotiation Object will be negotiated). In the second step, Refinement, both parties exchange proposals and counter proposals with the goal of meeting their constraints. The third step, Closure, finalizes the negotiation.

We have described in detail the architecture of the negotiation system in our previous papers [6]. In order to describe the complex types of negotiation scenarios, we have proposed in our previous work [7] seven different services: *Outsrc; Insrc; Block; Split; Broker*; *SwapIn/SwapOut; Transport.*

Our coordination approach proposes two different classes of services: (i) *Coordination services in closed environment*—refer to the services that manage the coordination constraints among multiple valid proposals only based on information extracted from a single negotiation; (ii) *Coordination services in an open environment*—refer to the services that manage the coordination constraints among valid proposal extracted from several or all ongoing negotiations into the system. The different coordination services in open or closed environment highlight the main features implemented in this negotiation process: distributive and parallelism.

In [8] we have proposed a formal model to settle and to manage the coordination rules of one or more negotiations, which can take place in parallel.

In this paper we are proposing a negotiation method based on coherent sets of coordination rules that can be easily instantiated and triggered on top of a communication middleware level. Then using those rules, we will present an example of a negotiation tactic as a coherent set of coordination rules.

4 Coordination Rules

Before presenting the rules, in the next section will detail our proposed constraint model and several definitions.

4.1 Fundamental Concepts

The fundamentals of the negotiation model are given by the following basic concepts:

A *Negotiation Model* is defined as a quintuple $\mathrm{M} = <\mathcal{T}, \mathcal{P}, \mathcal{N}, \mathcal{R}, \mathcal{O}>$ where: $\mathcal{T}$ denotes *the time of the system*, $\mathcal{P}$ denotes *the set of participants* in the negotiation framework, $\mathcal{N}$ denotes *the set of negotiations* that take place within the negotiation

framework, $\mathcal{R}$ denotes *the set of policies of coordination* of the negotiations (a coordination policy is a set of rules establishing dependencies between several negotiations) and $\mathcal{O}$ denotes *the common ontology* that consists of the set of definitions of the attributes that are used in a negotiation.

A *negotiation* is described at a time instance through a set of negotiation sequences. Let $\mathcal{S} = \{\text{si} \mid \text{i} \in \mathbb{N}\}$ denote the set of *negotiation sequences*, such that $\forall$si, sj $\in \mathcal{S}$, i $\neq$ j implies si $\neq$ sj. A *negotiation sequence* si $\in \mathcal{S}$ such that si $\in$ N(t) is a succession of negotiation graphs that describe the negotiation N from the moment of its initiation and up to the time instance *t*. The function *view()* returns the participant, the negotiation and the coordination policy described by a negotiation sequence: *view*: $\mathcal{S} \rightarrow \mathcal{P} \times \mathcal{N} \times \mathcal{R}$.

The negotiation graph created at a given time instance, *G = (A,E)* where *A* is the *set of nodes* and *E* is the *set of edges*, is an oriented graph in which the nodes describe the negotiation proposals that are present at that time instance and the edges express the precedence relationship between the negotiation proposals.

The *Status* (*Status* $\in$ *{initiated, undefined, success, failure}*) is the possible state of a negotiation, with *initiated* defining the sequence in which the negotiation has just been initiated; *undefined* defining the sequence with ongoing negotiation proposals; and *success* and *failure* are defining the sequence in which an agreement has been reached or the negotiation has been stopped with a denial.

Issues is the set of attributes with associated values that describe the proposals made in a negotiation.

The functions *status* and *issues* return, respectively, the state of a negotiation proposal and the set of the negotiated attributes.

We define *Role* as the *set of participant roles* such that *Role = {initiator, guest}*; with *initiator* being the participant initiating a negotiation N and *guest* being the participant invited in the negotiation N.

The functions *role()*: $\mathcal{T} \times \mathcal{P} \times \mathcal{N} \rightarrow$ *Role* and *role_s*: $\mathcal{T} \times \mathcal{S} \rightarrow$ *Role* returning the role of the participant p involved in negotiation N or in a particular sequence s, with the property that a participant has only one role in a negotiation and this role does not change in time.

4.2 Constraints Definition Model

We use these coordination rules as basic rules to describe the complex links among negotiations. We consider the coordination rule as the implementation of a dependency relation among several sequences of negotiation. A coordination rule has the following structure:

<Rule_Definition>':'[<Parameter_Definition>]* ';'<Graphs_conditions> <Condition><Relation><Result>

- The first part of the coordination rule (**<Rule_Definition>**) is used to define the name of the rule and its parameters—ex.: *name_rule(v1,v2,..,vn)*, with $v_i \in \mathcal{T} \cup S$.

- In **<Parameter_Definition>** each variable is related to its field of possible values—ex.: *synchronize*(T_1,T_2,*s1*,*s2*) : $T_1 \in \mathcal{T}$, $T_2 \in \mathcal{T}$.
- **<Graphs_conditions>** sets the conditions related to graphs and, in particular, related to the nodes of the negotiation sequences involved in coordination.
- The second part of the coordination rules is composed of a left part named *condition* **<Condition>** and a right part named *conclusion* **<Result>** and a relationship between both named (**<Relation>**).

The proposed model establishes two types of relationships between condition and conclusion: (i) *hard relationship* "→•" and (ii) *soft relationship* "→".

Hard dependency relations (denoted → •) ensures the fact that if there is a time instance t_1 where conditions are met, then the conclusion of the relation will be satisfied to the next time instance ($t_1 + 1$).

Soft dependency relations (denoted →) ensures the fact that if there is a time instance t_2 where the conclusion is met, then the conditions have been met at the time instance $t_1 < t_2$ and they have been remained true until the instance time $t_2 - 1$.

In other words, even if at some point the conditions of the *soft relationship* are satisfied, there is no guarantee that the result of the relation will be obtained. The conditions of a soft relation are necessary but not sufficient to obtain the result.

Our negotiation-centric set of rules can express constraints between the execution time of negotiations and their corresponding states (status dependencies), between the tasks and the attributes negotiated (attribute dependencies), and between the participants involved in the negotiation process (role dependencies). As an example, the *status dependences* establish the constraints between two or more states of the negotiation sequences involved in the same negotiation or in different negotiations.

We have defined the function *status*(*t*,*s*,*a*,*s*) with values in the set *Status* $\in$*{initiated, undefined, success, failure}; status*: $E_{\Phi 4} \rightarrow$ *Status* where $E_{\Phi 4}$ is the graph of Cartesian product $\mathcal{T} \times \mathcal{S} \times \mathcal{P}h \times \mathcal{S}$ that meets the relation Φ_4 such that:

$\forall(t,s_i,ph,s_j) \in \mathcal{T} \times \mathcal{S} \times \mathcal{P}h \times \mathcal{S}$, *exists* $\Phi_4(t,s_i,ph,s_j)$ *if and only if* $s_i, s_j \in \mathcal{S}$ *and ph initiated in* s_i

For example: *status*($t,s_0,2,s_2$) returns the state of the negotiation proposal in the negotiation proposal 2 initiated in sequence s_0 and visible in s_2 at the time instance *t*.

5 Coordination Pattern

At a certain point, for a participant involved in several negotiations, the infrastructure should handle many coordination rules, which can be very different. The proposed coordination model is not defined as a centralized process, managed by a single module, but it is distributed on several coordination modules. Therefore, we do not need to describe all the coordination rules and all possible actions, but we

want to set out coherent sets of coordination rules managed by the coordination process that may have an isolated execution.

Coordination Pattern is a set of coordination rules applicable within a given context and according to the chosen negotiation tactic. The coordination rules are expressed both by *Program Formula* and by the sequences involved in defining the coordination rules:

PattCoor = (TriggerSet, RulesSet, ProgramFormulaSet)

TriggerSet is the set of trigger conditions which must be met at the same moment of time, in order to apply the coordination pattern. These are descriptive conditions of negotiation sequences involved in a set of coherent coordination rules.

TriggerSet is described by three types of expressions:
(<Exp(t,N,p) >) ';' (<Exp(s)>)* ';' (<Exp(t,s)>)**
where

- *<Exp(t,N,p)>* are expressions that identify negotiations and participants as being the object of one or more dependence relations. These expressions are functions defined on the Cartesian product $\mathcal{T} \times \mathcal{P} \times \mathcal{N}$. These functions establish the involvement and role of the participants in the negotiations (ex. *role() function*);
- *<Exp(s)>* are expressions that identify punctually the sequences that will be involved in the dependence relations. These expressions are functions by type *view()* that establish the link between the sets $\mathcal{P}, \mathcal{N}, \mathcal{R}$ and the set $\mathcal{S}$;
- *<Exp(t,s)>* are expressions that establish the conditions on the characteristics of negotiation visible in sequences previously identified to trigger execution of the coordination rules. According to our model, the characteristics of negotiation are returned by the functions *status(), issues()* şi *role_s ()*.

RulesSet is the set of coordination rules that the coordination pattern is committed to synchronize.

First, the coordination rules are set and globally represented (visible in one or multiple negotiations and their corresponding sequences). Then, using the proposed negotiation model, these rules are splitted into coordination policies locally represented (visible in a single sequence), in order to be handled by a single negotiation sequence.

The implementation of a coordination pattern will correspond to a negotiation tactic that can be activated for a set of proposals in a negotiation (i.e., negotiation sequence), for the entire negotiation (i.e., negotiation graph) or for multiple negotiations (i.e., dependent negotiation graphs). We have proposed in [8] a modelling solution based on IAM but other rules based coordination frameworks can be employed (e.g., JESS or Drools).

Next we will detail an example of negotiation tactics using our coordination rules model.

5.1 Coordination in a Closed Environment

The two main characteristics of the coordination in a closed environment are the following: (i) The defined model refers only to dependences among bilateral negotiations of a single negotiation (i.e., one negotiation initiator with multiple negotiation participants); (ii) The evolution of a negotiation is carried out without taking into account other negotiations involving the same participant.

The proposed model will manage the coordination of exchanged proposals among partners on the attributes of the negotiation object considered in the concurrent bilateral negotiations.

As an example, a tactic stating that a task has to be outsourced as a block shall be described using our coordination rules model in a closed environment.

5.1.1 Negotiation as a Block

The block tactic is used in the negotiations where the task must be executed in its totality by a single participant of the negotiation process. The interactions take place between the enterprise that initiated the negotiation and all the other enterprises invited in the negotiation.

The following scenario provides the constraints of the negotiation process. A manager of a SME (participant P1) initiates a negotiation with the goal of establishing a contract regarding the execution of the entire outsourced task by a single participant. The negotiation ends when the participant P1 reaches an agreement with one of the partners (e.g., participant P2) regarding the set of attributes that describes the task being negotiated. At the same time, participant P1 ends all bilateral negotiations with the other partners.

These constraints can be described by the coordination pattern detailed below:

TriggerSet

The conditions of coordination pattern refer mainly to the role of the enterprises involved in negotiation. The coordination pattern manages the constraints on the participant p1 proposing a task within the collaborative working environments. Thus, he has the role of *initiator*.

$(p_1 \in participants(t,N))\ (role(t, p_1,N) = initiator);view(s_1) = (p_1, N, R_1);$
$(\exists\ a \in s_1(t) : status(t,s_1,a,s_1) = undefined)$

RulesSet

For the negotiation N, this coordination pattern manages dependences between different bilateral negotiations on the current task. These dependences refer to the status of negotiations. The participant P1 can independently manage each bilateral negotiation proposals. In other words, if the participant P1 reaches an agreement within a bilateral negotiation, then he must stop all bilateral negotiations for the same task. Therefore, the coordination should manage dependences when the

proposal is accepted. These dependences between bilateral negotiations are established by the coordination rule *(**competition**)*.

If the task is contracted by a single participant the coordination module should ensure that the execution of tasks was entirely accepted.

By defining rule *(**block**),* the coordination pattern ensures that the negotiation considers only the proposals on overall task.

***competition**(T_1, T_2,s1,Si) $T_1 \in \mathcal{T}$ $T_2 \in \mathcal{T}$ Si∈ negotiation(T1,N)-{s1};*

$\exists a \in s1(T_1)$; $\forall b \in s1(T_2)$ cu *$b \neq a$; $\exists a' \in s1(T_2)$* cu *$a' = a$*

status(T_1,s1,a,s1) = success ∧ status(T_1,Si,a,Si) ∧ issues(T_1,s1,a,s1) = issues(T_1, Si,a,Si) ∧ role_s(T_1,Si) = guest → • status(T_2,s1,b,s1) = failure;

In other words, if in the sequence s1 of the participant p1 there is a negotiation proposal with the status *success* and, also, the sequence *si* of the participant pi (guest) has the same status *success* and, if the sets of attributes (Issues) are equal, then the negotiation N stops all the active proposals present in s1.

***block**(T_1, T_2,s1,Si) : $T_1 \in \mathcal{T}$ $T_2 \in \mathcal{T}$ Si∈ negotiation(T1,N)-{s1};*

$\exists a \in s1(T_1)$; $\exists a' \in s1(T_2)$ cu *$a' = a$ ¬(status(T_1,s1,a,s1) = failure) ∧ ¬(issues (T_1,s1,a,s1).size = issues(T_1,Si,a,Si).size) ∧ role_s(T_1,Si) = guest → status(T_2,s1, a',s1).failure;*

In other words, the negotiation stops in the proposals where the size of the task is not the one specified by the participant p1.

These global rules can be interpreted using coordination policies composed of visible local rules. These policies model the behavior of a particular type of negotiation service (named *Block)* that negotiates for the participant p1. Next we are presenting the policy allowing a participant p1 to integrate a new negotiation sequence in an ongoing negotiation with a tactic Block. Other policies can be defined similarly for the guest participants' tactics.

policy for sequence *sk*

In order to comply with the rules defined by coordination pattern, this new sequence must be able to observe and act upon exchanged proposals between the participant p1 and other participants (guests). Therefore, this sequence should manage the active proposals that participant p1 shares through sequence *s*1, with other participants:

$\exists$ sk∈N(t1) : sk ⊂ s1

The following two rules represent the transcription of global rules defined at the beginning of the coordination pattern, according to local rules rules visible in sequence *sk.*

***competition**(T_1, T_2,sk,s1,Si) : $T_1 \in \mathcal{T}$ $T_2 \in \mathcal{T}$ Si∈ N(T_1)-{s1,sk}; $\forall a \in sk(T_1)$ ∧ $\forall b \in sk(T_2)$* cu *$b \neq a$ ∧$\exists a' \in sk(T_2)$* cu *$a' = a$*

status(T_1,sk,a,s1) = success∧status(T_1,sk,a,Si) = success∧issues(T_1,sk,a, s1) = issues(T_1,sk,a,Si) ∧ role_s (T_1,Si) = guest → • status (T_2,sk,b,sk) = failure

***block**(T_1, T_2,sk,s1,Si) : $T_1 \in \mathcal{T}$ $T_2 \in \mathcal{T}$ Si∈ N(T_1)-{s1,sk}; $\forall a \in sk(T_1)$ $\exists a' \in sk$ (T_2)* cu *$a' = a$*

¬(status(T_1,sk,a,sk) = failure)∧¬(issues(T_1,sk,a,s1).size = issues(T_1,sk,a, Si).size) ∧ role_s (T_1,Si) = guest → status(T_2,sk,a',sk) = failure

ProgramFormulaSet

Further, the policies presented are translated through different *Program Formula*, in methods modelling the structure and the content of the corresponding negotiation graphs.

The corresponding *Programs Formula* are described in our previous work [8].

Therefore, using the proposed coordination model we can manage the coordination among several bilateral negotiations on a task that must be fully accepted by a single contractor.

6 Application to the Manufacturing Segment

The outcomes of this research are being applied to the European Project C2NET. The goal of this project is the creation of cloud-enabled tools for supporting the supply network optimization of manufacturing and logistic assets, based on collaborative demand, production and delivery plans.

Particularly for the development of the Data Collection Framework in C2NET, which is responsible for a seamless integration among the developed components, modules and the cloud, the project proposes to complement the data collection framework with methods for interoperability decisions and steps that lead to adequate business process activities between them. To enforce the proper interoperability between systems, supported by a maturity environment, the set of negotiation mechanisms proposed in this paper are being implemented by technologies including Multi-Agent Systems (MAS) and the inference of reasoning rules. Another application of the negotiation services here developed will naturally be in the C2NET collaboration tools suite, for generic negotiation purposes. With respect to this, a possible scenario for collaborative working environment is the interaction between the plants and their suppliers. Each plant has its supply chain, which is used to satisfy any needs of the factory, and according to the requirements for each need, multiple suppliers may propose solutions that can be used, and which can make use of the negotiation system.

In the implemented scenario using the proposed framework, the manufacturing domains are interconnected by a set of services that support the development and maintenance of the interoperable enterprise collaborative environment.

Whenever the changes occur in the interoperable space, the negotiations will take place in order to find the most suitable solution mutually accepted by all parties.

The C2NET environment is the proof-of-concept that is under development by the authors to prove the framework.

7 Conclusions and Future Work

This paper proposes a generic coordination model based on a coherent set of rules and patterns that can be applied in order to model a rapid solution delivery in response to changes in a business process environment. The proposed solution allows handling several negotiations in parallel satisfying the possible dependencies among them. Managing parallel and dependent negotiations based on the users' constrains and negotiation tactics empowers the users to change their business workflows, policies and to cope with a continuous changing business processes. In the current work we have described the coordination of negotiations in a closed environment where the coordination is achieved only through bilateral negotiations that compose the same negotiation. It is presented an example of B2B interactions with the goal to outsource the entire task to a single participant. A negotiation process may end with a contract and in that case the supply schedule management and the well going of the contracted task are both parts of the outsourcing process. In the sequence of our research we will complete our model with the coordination of negotiation in an open environment that allow the coordination of constraints among several different negotiations in parallel.

The proposed model allows to coordinate negotiations of systems and applications' changes towards interoperability in enterprise interactions, supporting a sustainable interoperability networking environment along its life cycle.

Acknowledgments This work has been partly funded by the European Commission through the Project H2020 C2NET: Cloud Collaborative Manufacturing Networks (Grant Agreement No. 636909).

References

1. Panetto, H., & Cecil, J. (2013). Information systems for enterprise integration, interoperability and networking: Theory and applications. *Enterprise Information Systems, 7*(1), 1–6.
2. Jardim-Goncalves, R., Sarraipa, J., Agostinho, C., & Panetto, H. (2011). Knowledge framework for intelligent manufacturing systems. *Journal of Intelligent Manufacturing, 22*(5), 725–735.
3. Kadar, M. (2013, July 16–19). Assessement of enterprise interoperability maturity level through generative and recognition models. In *Proceedings of the 2013 International Conference on Economics and Business Administration (EBA 2013)*, Rhodes Island, Greece, ISBN 978-1-61804-200-2.
4. Wooldridge, M., & Jennings, N. R. (1995). Intelligent agents: Theory and practice. *The Knowledge Engineering review, 10*(2), 115–152.
5. Sycara, K., & Dai, T. (2010). Agent reasoning in negotiation. In *Advances in Group Decision and Negotiation* (Vol. 4, pp. 437–451). Handbook of group decision and negotiation, Part 4.
6. Cretan, A., Coutinho, C., Bratu, B., & Jardim-Goncalves, R. (2012). NEGOSEIO: A framework for negotiations toward sustainable enterprise interoperability. *IFAC Journal Annual Reviews in Control, 36*(2), 291–299.

7. Coutinho, C., Cretan, A., & Jardim-Goncalves, R. (2012, June 18–20). Cloud-based negotiation for sustainable enterprise interoperability. In *Proceedings of the 18th International ICE Conference on Engineering, Technology and Innovation* (ICE'12), Munich, Germany.
8. Coutinho, C., Cretan, A., & Jardim-Goncalves, R. (2013, October). Sustainable interoperability on space mission feasibility studies. *Computers in Industry—Special Section on Interoperability and Future Internet for Next-Generation Enterprise*, *64*(8), 925–937.