Interoperability in Collaborative Processes: Requirements Characterisation and Proof Approach

Matthieu Roque[1], Vincent Chapurlat[1],

[1] LGI2P- Laboratoire de Génie Informatique et d'Ingénierie de Production
Site EERIE de L'Ecole des Mines d'Alès, Parc Scientifique Georges Besse 30035 Nîmes cedex 1 – France
{Matthieu Roque, Vincent Chapurlat}@ema.fr

Abstract. Interoperability problems which can occur during the collaboration between several enterprises can endanger this collaboration. Consequently, it is necessary to become able to anticipate these problems. The proposed approach in this paper is based on the specification of properties, representing interoperability requirements, and their analysis on enterprise models. Due to the conceptual limits of existing modeling languages, formalizing these requirements and intending to translate them under the form of properties need to add conceptual enrichments to these languages. Finally, the analysis of the properties on enriched enterprise models, by formal checking techniques, aims to provide tools allowing to reasoning on enterprise models in order to detect interoperability problems, from an anticipative manner.

Keywords: Enterprise modeling, interoperability, model checking.

1 Introduction

The interoperability concept is started from a pure software problem in the middle of 90's where it is defined as "*the ability of two or more systems or components to exchange information and to use the information that has been exchanged*" [1]. Then, even if some efforts have been made to develop enterprise interoperability concepts, especially in Europe under various projects from FP5 and FP6, there is still no an overall satisfactory solution on interoperability. For example, [2] defines interoperability as "*the ability of a system or a product to work with other systems or products without special effort from the customer or user*". Interoperability is then analyzed by considering simultaneously different levels of detail of the pointed out enterprise (business, process, service and data), three kinds of barriers (conceptual, technological and organizational) i.e. three kinds of 'incompatibility' or 'mismatch' obstructing sharing and exchanging data and three different approaches (integrated, unified, federated). These three dimensions represent the interoperability framework. According to this, the classification of some related works and solutions for interoperability issues become possible. A lot of research and development works have been done concerning the conceptual barrier such as UEML [3] or PSL [4]. The goal is then to provide solutions to solve syntax and semantic problems. In the same

way, [5] proposes and interesting approach in order to design a Mediation Information System dedicated to deal with exchanged data, shared services and collaborative processes. This approach covers the technological and organizational barrier but considering only information system point of view. Other approaches are focused on the definition of maturity interoperability models. Let us cite for example LCIM [6], LISI [7], OIM [8] or EIMM [9] in order to evaluate the level of maturity of enterprises concerning their abilities to collaborate with other enterprises. However, they do not propose tools to measure and to evaluate interoperability itself. To solve this problem [10] proposes three kinds of enterprise interoperability measurements: interoperability potentiality, interoperability compatibility and interoperability performance. However, all these works do not provide a relevant solution in order to detect interoperability problems from an anticipative manner taking into account the different enterprise objects and their relationships within a network in which various enterprises must work together. Moreover, they do not allow identifying, in a formal way, what are the causes of interoperability problems. Thus, the research work presented in this paper aims to provide concepts and formal supports for reasoning on enterprise models in order to formalize and to detect interoperability problems as proposed in another domain by [11]. According to interoperability framework presented before, we focus, in this paper, on organizational interoperability problems by considering the Data and Service level. However, the Data level is extended in order to consider other natures of exchanged objects i.e. material, energy, financial, information or human objects. The interoperability approaches are not considered here because the goal of this work is only to detect where interoperability problems can occur by identifying their causes and not to provide solution to solve them.
Finally, the paper is structured respecting the following proposed approach:
Enterprise and Interoperability modeling: formalization of concepts, existing modeling language conceptual enrichment and property modeling.
Enterprise model re-writing: from enriched model of collaborative process to formal model allowing reasoning mechanisms.
Checking technique and mechanisms: proving properties in order to check the interoperability requirements.

2 Formalization of Interoperability Requirement

Interoperability is a crucial requirement having to be verified by systems when being in relationship (cooperation, collaboration, exchange) with other systems in order to assume a common mission. In this case, considered systems are enterprises or parts of enterprises which have to interact in a collaborative and common process with other enterprises or parts of enterprises in order, for example, to design a new product, to produce and integrate different part of a given product, etc. Formalizing what kind of relationships can exist in this area allows us to define more precisely this interoperability requirement. Thus, the relationship may be punctual or may exist during more or less long periods. Thus, all along the relationship life cycle, systems must being able to: (1) continue to fulfill their own missions, respecting the common mission and (2) remain independent of other systems and thus able to resume its

autonomy when relationship will stop. In another words, the relationship must be totally reversible i.e. differs to the integration [2]. In the following, any enterprise, process, activity of an enterprise in relation with another one will be simply considered as a processor inspired by [12]. A processor is a point where an object carried by a flow (concretizing the relationship) is processed i.e. transformed. Indeed, one or several object's characteristics (time – duration, delay, ... –, space – position, speed, acceleration,... – or form – geometry, color, ... –) change during a processor execution under the action of entities considered as resources of the processor and respecting some constraints and rules. Moreover, any of these processors use resources (human actor, organizational unit, machine, tool, or software application) as means necessary to transform the inputs into outputs. According to the system modeling framework called SAGACE [13], three types of relationships between two processors can be considered: transaction, coupling and interaction. Each relationship induces a set of requirements (functional and not functional [11] in order to assume that concerned processors are interoperable when it is needed. All these requirements have then to be checked in order to detect and to avoid interoperability problems. So, enterprise parts (processor, resources, flows, etc.) and interoperability requirements must be modeled. The next part intents to formalize the interoperability requirement corresponding to relationship typology.

Transaction is the basic relationship and only focuses on the flow of exchanged objects between two processors (supplier to customer).The flow can carry material, energy, financial, information or human objects. The customer processor can use this flow as an input to process or as a resource which support its execution. Transaction concerning objects of nature information induce, for example, the well known problems of the syntactic and semantic (form) of the exchanged information. It can also be related to the organizational aspects (time, form and/or space) i.e. the rules indicating how the different entities in the enterprise are structured and organized in order to fulfill the processor mission. For example, "is the actor in charge of a given processor must dispose of the required and updated information (about environment context, other processors and abilities for controlling the processor execution)?".

Coupling represents a reciprocal influence of a processor P1 named then controller processor to another processor P2 named operating processor: the controller processor P1 controls or constraints the execution of the operating processor P2 which have to provide reporting information and data to P1 as a feedback loop. This relationship corresponds typically to the link between decision and operating systems in system theory. The interoperability requirements are then, in addition to the ones of the transaction (based on form, state and time attributes), more related to the objectives or constraints provided by the controller processor to the operating processor. Thus, for example, the "production objectives" have to be clearly defined and well understood by all the resources involved in the processor "Reach production objectives" which receive the production objectives. Moreover, these production objectives have to be reachable in order to not induced interoperability problems between the two processors. Concerning the feedback loop, the requirements are the same ones of the transaction.

Interaction represents an influence of a processor to another processor requiring an intermediate processor which plays the role of interface between the two processors. This interface remains required because some change of one or more attributes of

time, space and / or form of the object carried by the flow cannot be done by one of the two connected processors. However the interface processor cannot be controlled by one of the two processors. For example, the interface processor can be a service provided by external entity and the enterprise cannot intervene during its execution. An interaction is then defined as a 3-uple {Event, Processor, Condition} where:
Event: event from which occurrence is required to execute the intermediate processor corresponding to an interruption of the normal running of the processor e.g. a machine failure or simply the end of the processor,
Processor: description of the intermediate processor as an input/output function,
Condition: condition under which the processor has fulfilled its mission. In case of the condition is not valid, the processor cannot provide its output and this can generates a hazard which can produce or not another interaction.
So, interaction can have a stochastic behavior taking into account the event occurrence, the condition validity but also of external constraints. In this case, interoperability requirements focus essentially on the intermediate processor. Indeed, neither of the two processors can have an influence on the behavior of the intermediate processor. The requirements consist then to prove that the processors are simultaneously aware about the possible risks associated to the fluctuations of interface processor behavior and able to adapt their own behavior, structure or functioning modes in order to anticipate these risks occurrences. In other words, are the processors able to find alternative in case of dysfunction of interface processor? The relationships between processors and this interface processor can be considered as a kind of Transaction relationship. So, the interoperability requirements concerning transaction have then to be checked to detect other interoperability problems.

3 From Interoperability Formalisation to Property

The requirement formalization consists on a representation under the form of a causal and constrained relation between two sets. This relation is called a property [11] defined by *as a requirement or a characteristic that have to be checked on each model of a pointed out system*. The first set models the condition called here the cause under which the requirement has to be checked. The second set describes the resulting situation of the studied part of the collaborative process i.e. the condition called here the effect under which the requirement have to be checked. The relation can be a logical (implication, equivalence or influence), temporized of not taking into account the requirement. If cause and effect are verified by the collaborative process, then the requirement is itself respected. This indicates that no prejudice can be induced to the collaborative process behavior regarding this requirement. Last, any effect can be considered as a new cause of other problem, so the interoperability requirements can be defined by using a recursive approach. For example, if given partners (i.e. part of enterprises) intents to be involved during a given activity A, they have to check all the abilities required by this activity A. In this case, if all partners check a set of abilities, they must be able to find internal resources able and available for supporting really the activity. At this stage, each interoperability requirement is formalized by (a set of) properties by experts from the domain. First, these properties are expressed by using

natural. The Fig. **1** illustrates some other properties which can be written. However, due to the conceptual limits of existing modeling languages, formalizing all the requirements need to add conceptual enrichments to the enterprise modeling languages. Thus, the meta-model of a modeling language (enriched BPMN language [14]), allowing to represent a model of the collaborative process to interoperability analysis issue, has been developed (not presented here). It has been implemented by using Graphical Modeling Framework of the Eclipse Platform [15]. This allows first to provide a modeling tool which is used in order to represent the collaborative process and enterprise models, second to develop the property proof mechanisms presented in the next part. Then, each property is translated into a formal language. Conceptual graphs are chosen [16]. A conceptual graph is a finite, connected, directed bipartite graph. It is defined as a graph with only two kinds of nodes: the concepts and the relations. The translation is performed by using interpretation mechanisms (considering the concepts and relations extracted from the modeling language which are described later in this paper). These ones are now under development and use the tool COGITANT [17].

Property 1	Each activity which provides an information or a product to another activity have to receive an acknowledgment receipt. [18]
Property 2	All shared information have to be periodically updated.
Property 3	The person who has the responsibility to update information has to be clearly defined.
Property 4	Partners of the collaboration have to be able to continue to achieve their own objectives.
Property 5	Partners have to remain independent of other partners and thus be able to resume its autonomy when relationship will stop (the relationship must be totally reversible).

Fig. 1. Example of properties

4 Enterprise Model and properties Re-Writing

The approach proposes to re-write enterprise models based with our enriched BPMN language in others models based on a formal language. The objective is to obtain models without sense ambiguity in order to check formal properties describing interoperability requirements. Thus, the enriched enterprise model is translated into Conceptual Graphs by using formal rules. The re-writing procedure starts from the meta-model of the enriched BPMN language, established in UML. This UML diagram is analyzed and formalized in order to provide all the needed concepts and relations of the Conceptual Graph. All concepts are obtained by considering all the modeling entities which will be used in the checking task. Thus, each class of the meta-model (but also its attributes) is translated into concepts. Then, the relations are obtained by translating each association between classes into a relation between concepts. Then, the defined concepts and relations (described in hierarchical structures called concepts and relations lattices) allow transforming the enterprise network model build with the enriched BPMN language into a conceptual graph. To

do this transformation, each marker (which refers to specific instances of concepts) has to be extracted from the model in order to produce a unique conceptual graph G. Thus, G gathers all the knowledge described in the model. Moreover, according to the concepts and relations lattices which have been defined, it is also possible to re-write the properties written in natural language, in a formal way. The Fig. 2 illustrates the re-writing of the property 1 which has been presented in the chapter 3.

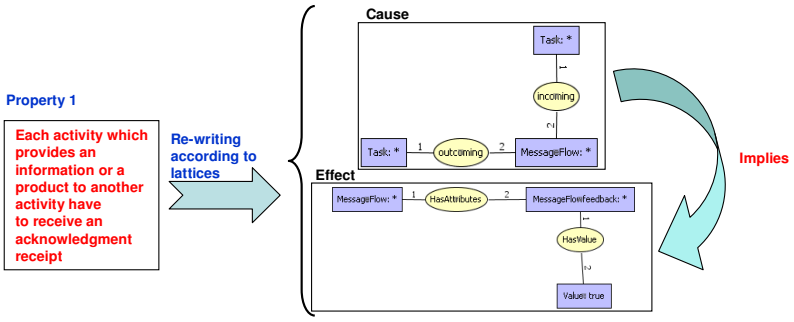

Fig. 2. Example of property re-writing

5 Checking Technique and Mechanisms

The checking technique is inspired by [19] which use analysis mechanisms allowed by conceptual graphs. These analysis mechanisms are:

Projection: This involves comparing the obtained conceptual graph coming from the translation of the model with another one translating the property. If the projection fails, then the modeled property cannot be verified and the causes are highlighted.

Constraint: a property describes what the links and/or constraints are between facts. In this case, the property is translated on a positive or negative conceptual graph constraint. A positive constraint between two facts A and B must be interpreted as: "If A is true, then B must also be true". Conversely, a negative constraint must be interpreted as: "If A is true then B must be false".

Dynamic and static rules: A property is directly modeled as a property composed of a cause and an effect. If the graph corresponding to the causes match with a part of the conceptual graph translating the system models, then the effect must be checked in the same way.

The Fig. 3 illustrates two examples of property proofs by using the projection mechanism. The Fig. 3a represents the studied model built by using our modeling tool. This model describes the exchange of data between two partners in order to find a common available day for organizing a meeting. The Fig. 3b illustrates a part of this model translated in conceptual graph taking into account the exchange of data between the activity "contact partner" and the activity "To check availability". The concepts and relations which are in black are the ones which allow to verify (by using the projection mechanism) the property 1 (defined in the chapter 3). Then, the

responsible of the activity "To check availability" has to check his availability by accessing to his enterprise online agenda (linked to a common database concerning all employees of the enterprise). This agenda is normally updated regularly. The objective of the proof of the property 2 (illustrated in the Fig. 3c) is to check if the data in the agenda corresponds effectively to the last updated version. Thus, as for the property 1, the concepts and relations which are in black are the ones which allow to verify this property. These two properties concerns requirement of the transaction relationship as defined in the chapter 2.

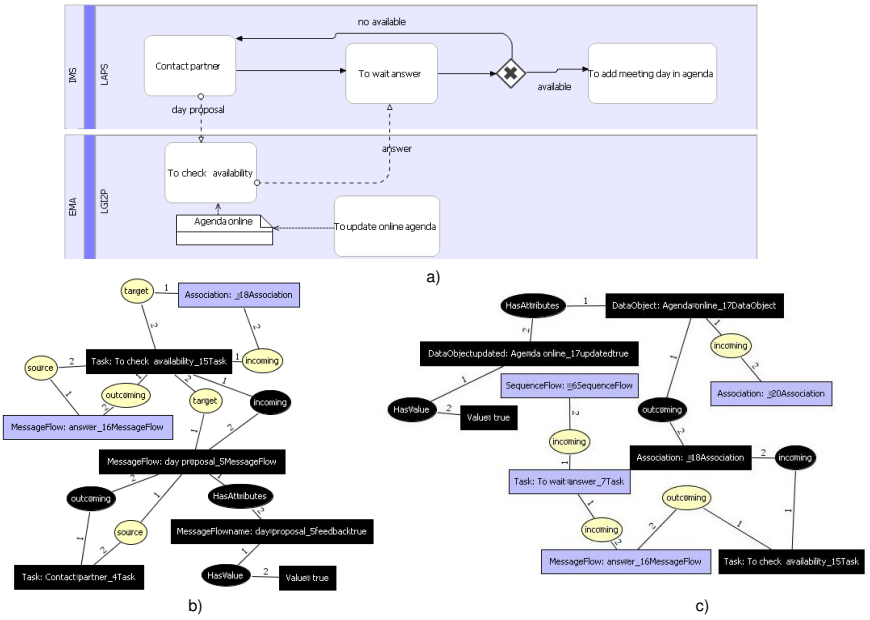

Fig. 3. Example of projection

6 Conclusions and perspectives

This article presents the first results of our research. A formal model of interoperability requirement in collaborative processes context is introduced. A set of modeling and formal proof mechanisms is then described in order to analyze from a static point of view the network model. The main perspectives of this work are the following. First, a reference properties data base and rewriting mechanisms have to be developed in order to help actors to analyze more rapidly the collaborative processes and then become able to anticipate interoperability problems. Second, rewriting mechanisms from model and properties database have to be implanted in order to be interfaced with model checkers such as UPPAAL. Third, other works in progress intents to make the gap between network model and an enriched multi agents system

allowing to simulate the behavior of the different parts of the enterprises involved in the collaborative process. The goal these two last works is then to assume dynamic properties can be then checked.

References

1. IEEE, A compilation of IEEE standard computer glossaries: standard computer dictionary, New York, (1990)
2. INTEROP, Enterprise Interoperability-Framework and knowledge corpus - Final report, INTEROP NoE, FP6 – Contract n° 508011, Deliverable DI.3, May 21st (2007)
3. Berio G., UEML 2.0, Deliverable 5.1, INTEROP project UE-IST-508011 (2005)
4. Schlenoff, C., Gruninger M., Tissot, F., Valois, J., Lubell, J., Lee, J., The Process Specification Language (PSL): Overview and Version 1.0 Specification, NISTIR 6459, National Institute of Standards and Technology, Gaithersburg, MD (2000).
5. Bénaben F., Touzi J., Rajsiri V., Truptil S., Lorré J.P., Pingaud H., Mediation Information System Design in a Collaborative SOA Context through a MDD Approach. MDISIS'08: Model Driven Interoperability for Sustainable Information System. Montpellier, France (2008)
6. Tolk A., Muguira J.A., The Levels of Conceptual Interoperability Model, Fall Simulation Interoperability Worshop (2003)
7. C4ISR, Levels of Information Systems Interoperability (LISI), Architecture Working Group ,United States of America, Department of Defence (1998)
8. Clark T., Jones R., Organisational Interoperability Maturity Model for C2, Australian Department of Defence (1999)
9. ATHENA, Framework for the establishment and management methodology, Integrated Project ATHENA, deliverable A1.4 (2005)
10. Daclin N., Contribution au développement d'une méthodologie pour l'interopérabilité des entreprises (in French), PhD Thesis, University Bordeaux1, December, (2007)
11. Aloui S., Chapurlat V., Penalva J.-M., Linking interoperability and risk assessment: A methodological approach for socio-technical systems, Proceedings of INCOM2006, 12th IFAC Symposium on Information Control Problems in Manufacturing, Information control: a complex challenge for the 21st century, A. Dolgui, G. Morel and C. Pereira Eds., ISBN: 978-0-08-044654-7, Saint Etienne, France, hal-00354778 (2006)
12. Le Moigne, J.L.: La modélisation des systèmes complexes, Paris, Bordas, Dunot (1990)
13. Penalva, J-M., La modélisation par les systèmes en situation complexe (in French), PhD thesis. Paris Sud university (1997)
14. BPMN, Business Process Modeling Notation, V1.2 (2009). http://www.bpmn.org/
15. GMF, Graphical Modelling Framework (2008), http://www.eclipse.org/modeling/gmf/
16. Chein M., Mugnier M-L., Conceptual graphs: fundamental notions, Revue d'intelligence artificielle, vol.6, n°4 (1992)
17. Cogitant, CoGITaNT Version 5.2.0: Reference Manual (2009) (http://cogitant.sourceforge.net)
18. Blanc S., Contribution à la caractérisation et à l'évaluation de l'interopérabilité pour les entreprises collaboratives (in French), PhD Thesis, University Bordeaux1, December, (2006)
19. Chapurlat V. and Aloui S., How to detect risks with a formal approach? From property specification to risk emergence, in proceedings of Modeling, Simulation, Verification and Validation of Enterprise Information Systems (MSVVEIS), Paphos, Cyprus (2006)