Cross-organizational Business Processes Modeling Using Design-by-Contract Approach

Malik Khalfallah, Nicolas Figay, Parisa Ghodous, Catarina Ferreira da Silva

Cross-organizational Business Processes Modeling Using Design-by-Contract Approach

Malik Khalfallah, Nicolas Figay, Parisa Ghodous, and Catarina Ferreira Da Silva

European Aeronautic Defence and Space Company (EADS), Paris, France
Lyon 1 University, Lyon, France
{malik.khalfallah,nicolas.figay}@eads.net
{ghodous,catarina.ferreira-da-silva}@liris.cnrs.fr

Abstract. Reaching agreements between organizations in a collaborative environment is a way to ensure interoperability between these organizations at all levels. For business processes interoperability this agreement can be reached by well defining the cross-organizational process. However most BPM frameworks have used meta-models centered on flows of activities, with the data manipulated by these activities seen as second-class citizens. For business analysts (for example in complex product design collaborative environments) data plays a major role. In this paper, we propose a methodology backed by a conceptual framework to model the cross-organizational process relying on the product model. This framework defines the evolution of the product model through a finite number of states, and then automatically generates executable artifacts to support the collaboration during run-time phase. This approach is being implemented in the living laboratory provided by EADS in the context of the European project IMAGINE.

Key words: design-by-contract, interoperability, product model, business process, model driven architecture, UML

1 Introduction

The aeronautic and aerospace industries are seeing a rapid shift to the extended enterprise strategy. Companies specialized in this domain, including EADS, are increasingly externalizing activities for subcontractors while focusing on their core activities that are specific to their disciplines.

Ensuring interoperability between organizations in the extended enterprise is a complex problem and several studies have been conducted and reported in the literature. While many of these proposals focus on the static extended enterprise, where partners do not leave the network and are not replaced by new ones, ensuring sustainable interoperability for a dynamic extended enterprise has not been addressed well. We have recently seen the events of the volcano's eruption in Iceland and the nuclear disaster in Fukushima and their impact on enterprises. These events have reaffirmed the need for greater flexibility and maintaining organizations' interoperability and to cope with the dynamic nature

of collaborative environments. To address this problem, the concept of Dynamic Manufacturing Network (DMN) has emerged [1].

In the context of product design collaborative environments, standards such as ISO-10303 (STEP) have been proposed in order to ensure partners' interoperability. The overall objective of STEP is to provide a mechanism that describes a complete and unambiguous product definition throughout the life cycle of a product. STEP provides both broadly useful data modeling methods and data models focused on specific industrial uses [2]. Even though STEP is an accepted standard in today's industry, it is still not sufficient to ensure a satisfactory level of interoperability in the extended enterprise. Indeed, relying on the classification of interoperability levels provided by Lewis et al. [3], we can say that STEP is limited to data interoperability of the first three levels (machine, syntactic, and semantic) but does not address data interoperability at the organizational level, as illustrated in figure 1, which remains the most complex one [3]. To fill this gap, EADS has participated to the European project CRESCENDO [4] that defined the concept of Behavioral Digital Aircraft Business Object Model (BOM). The BOM is a data model built with UML and defines high level concepts to be used by engineers pertaining to different organizations involved in the design and development of aircrafts. The BOM and its mapping to STEP concepts are the building blocks of the agreement necessary to ensure data interoperability at all levels.

In addition to data interoperability, another issue in the extended enterprise is the interoperability of organizations' business processes [5]. Cross-organizational processes aim to achieve an agreement between organizations at the process level. They specify what messages each organization should send and receive as well as their sequences. To model the cross-organizational process languages such as BPMN/XPDL (Business Process Modeling Notation/XML Process Definition Language) can be used. As illustrated in figure 1, XPDL covers the first three levels but not the fourth one. The reason is that when modeling the cross organizational process using XPDL, the business expert of each organization defines the exchanged messages based on their representation using STEP concepts (serialized in XML). Consequently, the high level concepts defined by the BOM are not usable and the business expert needs to understand the concepts defined by STEP. Moreover, the constructs provided by process modeling languages do not fit well for DMN processes due to the dynamic nature of DMN. They do not support well the changes on-the-fly that could occur [6]. This analysis raised the following challenges: (i) how to define an agreement between participants that will ensure their business processes interoperability at the organizational level while using the concepts defined by the BOM? (ii) How to make this agreement flexible enough to maintain a sustainable interoperability of the whole DMN?

Contributions. Since the BOM defines the concepts used to design the whole aircraft from different point of views (i.e. physical, functional, system), we start from this data model and then, define the evolution of the product model properties along the collaboration until reaching the final configuration. The evolution is modeled using temporal logic relations between product model states' speci-

fications. Each state describes what constraints should the product model fulfill and which organizations should ensure these constraints.

Using the proposed framework to model the cross-organizational process is promising because it decreases the coupling between organizations' processes. In addition it provides the business expert the means to naturally define the cross-organizational process based on high-level concepts, since existing business process modeling approaches fail to capture the business intent underlying the interactions [7].

The rest of the paper is organized as follow: in section 2, we provide motivations regarding the proposed approach. In section 3 we give an overview of the methodology accompanied with an example. Section 4 elaborates on the conceptual framework that supports this methodology. Section 5 gives an overview of the implementation. Section 6 discusses the related work. Finally, section 7 concludes the paper and out-lines the perspectives of the current work.

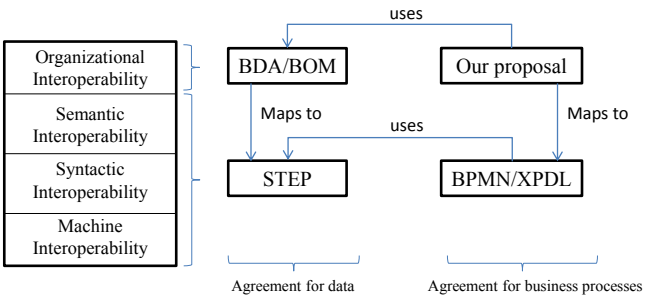

Fig. 1. Proposal overview

2 Background

In this section we give a brief description of the building blocks of the proposed framework.

2.1 Model Driven Architecture

At the conceptual level, MDA is a holistic approach to improving the entire IT life cycle specification, architecture, design, development, maintenance and integration based on formal modeling [8]. Today's MDA is less about generating code per se and much more about precisely capturing requirements, enforcing architectural standards, maintaining traceability, and facilitating effective communication between the business and IT.

From the analysis above, we figured out that we are facing the same issues that software developers faced in the domain of software engineering. One major issue is the lack of abstraction. Since MDA has proved to be successful as

illustrated by industrial case studies demonstrated by [8], or even internally in EADS, where it has been used in multiple projects, we carry on in this direction by using it. Our aim is not limited to generate executable code but also to generate more complex artifacts as explained below. Such an approach provides engineers with high level concepts when modeling the cross-organizational process without caring how this process will be implemented using workflow engines.

3 Design By Contract

Design by Contract (DbC) is an approach to building reliable software that focuses on making the contract of a software module explicit and formal [9]. DbC is not limited to software design but can be used for high-level modeling [10]. DbC involves writing two kinds of formal constraints:

- Preconditions and post-conditions that are assertions about operations;
- Invariants that are assertions about the system state that must be true, except during the execution of an operation.

In this paper we extend the core idea of DbC - software development through elaboration of type signature with logical predicates - to the design and modeling of cross-organizational process by linking constraints' specifications on product model properties using temporal logic relationships. This approach allows us to formalize an agreement between all organizations involved in the collaboration based on the product model. This is an important element to ensure interoperability at organizational level as noticed by Lewis et al. [3].

3.1 Business Rules

A business rule is "a statement that defines or constrains some aspect of the business. It is attended to assert business structure or to control or influence the behavior of the business" [11].

Business rules are executable by rules engine, thus they bridge the gap between contract constraints expressed at the organizational level and their implementation in the execution platform. A typical pattern of business rules is the ECA (Event Condition Action) notation [12]:

- The event component specifies when the rule has to be executed;
- The condition component indicates a condition to be checked before any action is triggered;
- The action states what has to be done depending on the result of the evaluation of the condition component. In general an action terminates by raising one or more relevant events.

We elaborate on the usage of business rules in the implementation overview in section 5.

3.2 Problem Statement

The general problem addressed in this paper is the need to develop a product-based technique to improve the change management and maintainability of cross-organizational business processes for DMN. Such a method should allow the product architect to predict and manage the impact of change on the processes used to develop the product, while decreasing the time needed to implement changes.

4 Methodology of Design by Contract for DMN

In this section, we present a methodology that defines how our approach can be used to model the cross-organizational process. This is illustrated in figure 2.

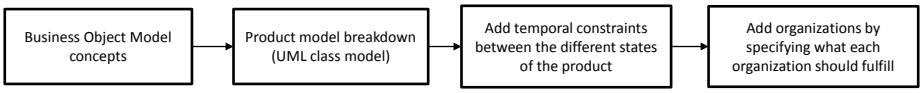

Fig. 2. Proposal overview

Step 1. The Business Object Model is an ontology that defines the concepts used by all organizations in the DMN. It ensures semantic interoperability between all participants. It defines the concepts used to develop all product components.

Step 2. The product model breakdown is a static model that defines the structure of the product. It structures the concepts defined in the BOM and establishes the relationships between these concepts. The product model is the building block of the contract, because all participants agree on this structure. However it remains not sufficient because it does not cover the behavior of the collaboration.

Step 3. Adding generic constraints to the classes' attributes of the product break-down allows the architects to formally define what values the properties should take. The objective of the collaboration is to iterate over the cross-organizational process until finding the optimal values of all product and sub products properties and defining the final configuration of the product. Relying on the classical relationships between constraints (e.g. and, or) is not sufficient to define the cross-organizational process. We need to add temporal relationships between constraints in order to define the succession of the product model states until reaching its final state.

Step 4. The logic-based model of the product model evolution is stable and it is in-dependent from: the organizations involved in the DMN, the processes enacted by these organizations and the IT systems that support the collaboration. Therefore, the last step adds organizations to the model and specifies what constraints each organization should fulfill.

4.1 Example

The following is an example of applying the proposed methodology using the results of the project CRESCENDO [4]:

Step 1. The concepts defined in the real BOM are complicated for non-aircraft engineers to understand. In this example we use usual concepts and simple attributes in order to demonstrate our solution.

Step 2. The product model of an aircraft is a tree-like structure. For a single aircraft, we can have several views that generate multiple decompositions (physical, functional, system etc.). Figure 3 illustrates this decomposition.

Step 3. Relying on the product model, the aircraft architect develops the cross-organizational process by defining the constraints on the aircraft functions (i.e. what functions the aircraft should fulfill). Additionally, he adds relationships between these constraints to indicate their dependencies as illustrated in figure 4. For this specific example, since a function is realized by physical components, the aircraft architect sets constraints on these components as well. This is illustrated in figure 5.

Step 4. Finally, each constraint is attached to two actors the requester, who sets the values of the attributes (the aircraft architect in our case) and the supplier (e.g. the wing designer) who tries to design the sub-product using these values.

At this stage the design of the contract is complete. The next step consists in iteratively running the collaboration between the involved partners until satisfying all constraints.

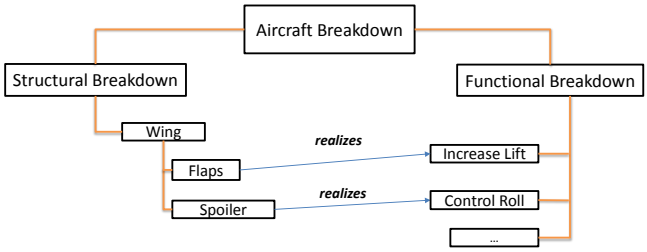

Fig. 3. Proposal overview

4.2 Handling Network Dynamicity

Using the proposed methodology, backed by a conceptual framework presented in the next section, reveals some advantages regarding the way of modeling the cross-organizational process and how to maintain interoperability between organizations.

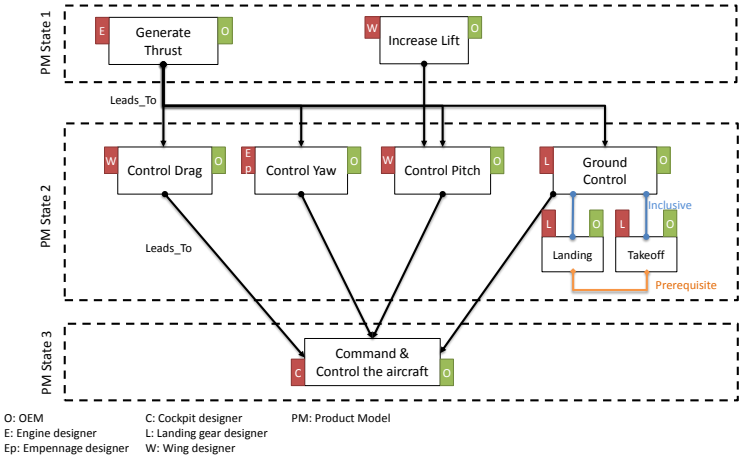

Fig. 4. Proposal overview

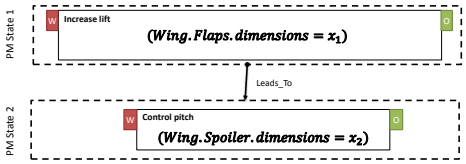

Fig. 5. Proposal overview

Product architects define the desired evolution of the product along the collaboration by defining constraints and their relationships. Consequently, the focus is put on the product model which remains stable even if some organizations quit the network. This decreases the coupling between organizations' processes which ensures a sustainable interoperability of the whole collaborative environment.

For example, in figure 5, if the wing designer quits the DMN and he should be replaced by two separate designers: one to design the spoiler and another to design the flaps. This will have no impact neither on the evolution of the product nor on the remaining organizations in the DMN.

5 A Conceptual Framework for DbC

The objective of this framework is to back the presented methodology by formal basis. For this purpose, we define the concept of *Obligation*.

Definition 1. *An obligation defines both what one party is obliged to guarantee and, dually, what other parties can rely on.*

More specifically, each product component designed by an external organization needs to respect predicates over attributes representing its properties. These

predicates are obligations for the component designer and a guarantee for all other organizations (e.g. the aircraft architect decides that the length of the wing shall be 20m. This is a guarantee for the aircraft architect and an obligation for the wing designer).

5.1 Contract Formalization

It is essential to have a tractable and rigorous representation of obligations. Formal representation of obligations will provide accurate and unambiguous specifications. For this reason we use logic formulae to formalize obligations. Logic formulae are a general, rigorous and flexible tool to describe constraints. Indeed using logic formulae, the product architect specifies the properties of the subcomponents composing the product and the relationships between these properties.

Obligations are the building block of the cross-organizational process. Relationships between obligations are important as well. Using temporal logic provides the means to define how the collaboration should evolve through different product model states.

Definition 2. *A product model state is a set of obligations specifying constraints on a subset of objects defined in this model.*

Two states are interconnected by a temporal logic relationship *LeadsTo*, as illustrated in figure 4, to express the precedence relationship between these states.

In this framework we purposely use temporal logic relationships patterns to inter-connect obligations and states. For instance, the knowledge required to use formal models and their complexity remain a significant obstacle for their widely adoption by business experts (i.e. product architects) [13]. Using patterns supports shielding the complexity of formalisms from business experts and facilitate their specification in the abstract.

[14, 15] already identified temporal logic patterns that we use to formalize the relationships between obligations and states. Given two obligations:O_1, O_2 and two states: S_1, S_2:

Pattern	Description
$O_1 inclusive O_2$	The fulfillment of O_1 mandates the fulfillment of O_2
$O_1 exclusive O_2$	The fulfillment of O_1 mandates the non-fulfillment of O_2
$O_1 prerequisite O_2$	The non-fulfillment of O_1 mandates the non-fulfillment of O_2
$O_1 mutexchoice O_2$	Either O_1 or O_2 are fulfilled but not any of them or both of them
$S_1 LeadsTo S_2$	State S_1 must always be followed by state S_2

Table 1. A set of patterns to interconnect obligations and states

Besides states and obligations, we add the concept of role. There are two types of roles:

– The requester: this is the partner who instantiate the obligations and waits to see whether the physical component can be designed under these obligations.

– The supplier: this is the partner who fulfills the obligations set by the requester.

5.2 Local Projections of the Contract

Modeling the cross-organizational process using obligations and product model states makes the local projections easy to generate. There are three kinds of artifacts to generate:

The requester process. The process of the organization that relies on obligations is illustrated in the right hand side of figure 6. For example, the aircraft architect and the wing designer collaborate in order to build an optimal model of the wing. The aircraft architect starts by giving a value to the flaps dimension. After that the wing designer executes its business process, the aircraft architect receives the value of the new flaps dimension. This value can be equal to the already set value in the constraint which means that the wing designer has fulfilled its obligation in this case the aircraft architect carries on the execution of his internal process. Otherwise, the aircraft architect may give a new value to the constraint and repeat the process.

The supplier process. The process of the organization that fulfills a particular obligation has the pattern illustrated in the left hand side of figure 6. For example, the wing designer and the aircraft architect collaborate in order to ensure state 1 in figure 5. The wing designer receives the value of the flaps dimension, carries out an internal process to design the flaps following the set conditions, and then he notifies the aircraft architect of the new value.

The business rules that manage the update of the product shared model. Since in our implementation we are using a shared product model of the aircraft that is updated during the collaboration until reaching the final state, we decided to generate business rules that handle the updates of attributes' values and that generate notifications. Notice that even if this shared product model was absent, the generated artifacts would have been different but this remains an implementation detail and it has no impact on the way the cross-organizational process is modeled. processes interoperability issues in this case has been addressed in our previous paper [16]. Figure 7 gives an excerpt of the generated business rules of the given example. We rely on the business rules of type ECA as defined in [12].

6 Implementation

To illustrate our proposal, we have started to develop a prototype to test it with the results of the CRESCENDO project. The following steps illustrate how the proposed architecture (figure 8) implements the presented methodology with the conceptual framework. We should notice that there are two different phases:

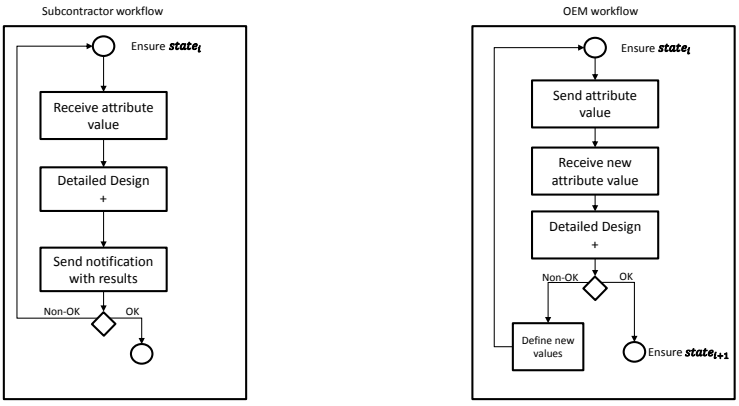

Fig. 6. Process Pattern for both sides of collaboration

ON *Ensure_state1_event*
IF **State1 = true**
RAISE *Ensure_state2_event*
ON *Ensure_state2_event*
IF **State1 Leads_To State2 = true**
RAISE *Ensure_state3_event*

Fig. 7. Generated business rules

the design-time phase where the aircraft architect defines generic constraints and assigns them to organizations. This defines the cross-organizational process (steps: 1, 2 and 3) and the run-time phase, where the cross-organizational process is executed until reaching the final configuration of the product model (step4):

Step 1. We assume that a product model is built and is being shared through a collaborative platform.

Step 2. The aircraft architect sets the (temporal) constraints on the shared product model through the collaborative platform. He uses an extension of the language OCL (Object Constraints Language) with temporal properties to define constraints on the classes in the UML model of the product. Additionally, he assigns corresponding organizations to each constraint. This step can be further enhanced by using advanced graphical user interfaces. At this stage we use an extension of OCL that supports temporal logic constructs and organizations assignment to constraints.

Step 3. Once the cross-organizational process is ready, the workflow generation module generates the workflows of the organizations and deploys them in their work-flow engines. Actually, a generated workflow is the public view on the complete organization's process. More specifically, it contains activities involved in the exchange of data with other organizations or with the collaborative platform. The detailed design sub-process (illustrated in figure 5) is the internal process of each organization that contains private activities and is not shared with the

external world. Additionally, in this step, the workflow generation module generates the business rules that handle update events that occur on the shared product model and verify the fulfillment of every obligation of every state.

Step 4. The aircraft architect collaborates with the sub-products designer through an iterative process until reaching a satisfactory configuration of the product.

We can notice that in the cross-organizational process model illustrated in figure 4, there is no exception handling (i.e. it models only the nominal process). This is purposeful, because we noticed that it is very difficult for an aircraft architect to identify all possible exceptions that can occur during a product design at the design time phase of the cross-organizational process. These exceptions are handled during the run-time phase.

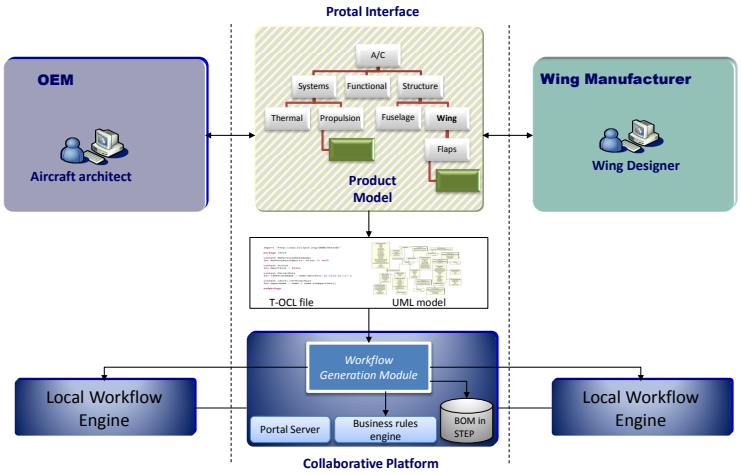

Fig. 8. Implementation overview

7 Related Work

Our objective in this paper is to provide a framework supported by a formal theory that allows product engineers to model the cross-organizational process (i.e. the con-tract) using business concepts that are more abstract than low-level concepts such as data and control flows modeling constructs.

Modeling cross-organizational processes and defining their local projections is a widely studied subject: standardized languages WS-CDL, BPMN as well as Decserflow for choreography [19], Scribble [20], and Bocchi et al. proposal [10] focus on formally defining the cross-organizational process and propose a formal definition of the projection to generate participating organizations processes'

public and private views. The common limits of these proposals regarding our problematic are twofold:

- They still use constructs such as messages, activities, flows that were demonstrated to be low-level concepts Telang et al. [7].
- The coupling maintained between organizations that could be difficult to handle when an organization quits the network and be replaced by a new organization.

Business rules and business processes. Eijndhoven et al, and Charfi et al. [11], [21] Work aims to increase the process flexibility by separating between the procedural flow of the process and the business rules. Their proposal is limited to a single enterprise process, because they assume that business rules are internal to an enterprise and not shared with the external world.

Product model and business processes. Van der aalst et al. [22] proposed a method to design the optimal workflow based on the product model. The proposed methodology is limited to decision-making processes. They gave the example of a product decomposition of a candidate that wants to become a pilot. Relying on multiple criteria, they generated the optimal workflow that can decide whether a given candidate is able to become a pilot or not.

Declarative modeling of business processes. Van der aalst et al. [23] already pointed out the limits and the rigidity of procedural modeling languages such as BPMN and proposed a declarative language to model business processes. This work was used by Montali et al. [19] in order to model the cross-organizational process. The limit of this proposal is the possible ambiguity when interpreting these models. While Van der aalst et al. [24] pointed out the mandatory nature of a process model to be unambiguous, these studies lack this property. Indeed, the modeled process is a temporal logic formula interconnecting activities. Inherently, they assume that the activity is true when it finishes, while there is a possibility to give a different interpretation: an activity is true when it starts. These two possible interpretations generate two different processes, the first one where two successive activities are executed in sequence and the second interpretation is that two successive activities are executed in parallel.

Business processes and web services. The interoperability of services' processes is also a widely studied subject and several researches are conducted in the domain of web services. Basically, these approaches generate adapters either automatically or semi automatically that could resolve both structural and behavioral mismatches between two communicating business processes. We have already defined an ontology-based approach to resolve structural mismatches that could occur due to the difference of STEP application protocols used by two communicating organizations [25], [26]. For the behavioral mismatches we have decided to use the concept of DbC because mediation generation is an organization-dependent approach. If an organization quits the network we are

obliged to regenerate a new mediator and redeploy it in the collaborative platform, which can be time consuming and expensive. Munusamy et al. [27] provide a thorough state of the art on the techniques that generate adapters to resolve mismatches between two communicating business processes.

8 Conclusion

Imperative languages have proven to be non-trivial to support changes of the pro-cesses on-the-fly [6]. This is a major obstacle to ensure business process interoperability in Dynamic Manufacturing Networks (DMN). In this paper we have presented a methodology backed by a conceptual framework to ensure business processes interoperability for DMN. We have used the Design by Contract (DbC) approach. DbC provides a declarative specification of what each organization in the DMN should accomplish (obligations) while maintaining this specification independent from organizations' specificities and the IT platform that supports the collaboration. Using the MDA architecture, we were able to project the DbC specification on the execution platforms and run the collaboration in order to fulfill the contract.

The results presented in this paper are being implemented in the living lab of EADS provided in the context of the European project IMAGINE [1].

References

1. Innovative end-to-end Management of Dynamic Manufacturing Networks (IMAGINE Project), september 2012, [Online; http://www.imagine-futurefactory.eu/]
2. Kramer, Thomas, and Xu, Xun: STEP in a Nutshell, Advanced Design and Manufacturing Based on STEP, Springer London, 122, Eds: Xu, Xun, and Nee, Andrew Y. C., 2009
3. Lewis, Grace A., Morris, Edwin J., Simanta, Soumya, and Wrage, Lutz: Why Standards Are Not Enough to Guarantee End-to-End Interoperability, ICCBSS, 164173, 2008
4. Collaborative and Robust Engineering using Simulation Capability Enabling Next Design Optimization (CRESCENDO Project), september 2012, [Online; http://www.crescendo-fp7.eu/]
5. Papazoglou, Mike P., and van den Heuvel, Willem-Jan: Service oriented architectures: approaches, technologies and research issues, VLDB J. 16(3), volume 16, 389415, 2007
6. van der Aalst, W. M. P., and Jablonski, S.: Dealing with workflow change: identification of issues and solutions, International Journal of Computer Systems Science and Engineering 15(5), volume 15, 267276, September 2000
7. Telang, Pankaj R., and Singh, Munindar P.: Business Modeling via Commitments, SOCASE, 111125, 2009
8. Guttman, Michael, and Parodi, John: Real-Life MDA: Solving Business Problems with Model Driven Architecture, Morgan Kaufmann Publishers Inc., 2007
9. Frankel, David: Model Driven Architecture: Applying MDA to Enterprise Computing, John Wiley and Sons, Inc., 2002

10. Bocchi, Laura, Honda, Kohei, Tuosto, Emilio, and Yoshida, Nobuko: A Theory of Design-by-Contract for Distributed Multiparty Interactions, CONCUR, 162176, 2010
11. van Eijndhoven, Tim, Iacob, Maria-Eugenia, and Ponisio, Mara Laura: Achieving Business Process Flexibility with Business Rules, EDOC, 95104, 2008
12. Knolmayer, Gerhard, Endl, Rainer, and Pfahrer, Marcel: Modeling Processes and Workflows by Business Rules, Business Process Management, Models, Techniques, and Empirical Studies, Springer-Verlag, 1629, 2000
13. Kharbili, Marwane, and Keil, Tobias: Bringing Agility to Business Process Management: Rules Deployment in an SOA, Emerging Web Services Technology Volume III, Birkhuser Basel, 157170, Eds: Binder, Walter, and Dustdar, Schahram, 2010
14. Dwyer, Matthew B., Avrunin, George S., and Corbett, James C.: Property specification patterns for finite-state verification, Proceedings of the second workshop on Formal methods in software practice, ACM, 715, 1998
15. Tretken, Oktay, Elgammal, Amal, van den Heuvel, Willem-Jan, and Papazoglou, Mike P.: Enforcing compliance on business processes through the use of patterns, ECIS, 2011
16. Khalfallah, Malik, Figay, Nicolas, Barhamgi, Mahmoud, and Ghodous, Parisa: Product-based Business Processes Interoperability, ACM Symposium on Applied Computing, 2013
17. van der Aalst, Wil M P, Lohmann, Niels, Massuthe, Peter, Stahl, Christian, and Wolf, Karsten: Multiparty Contracts: Agreeing and Implementing Interorganizational Processes., Comput. J. 53(1), volume 53, 90106, 2010
18. Zaha, Johannes Maria, Barros, Alistair P., Dumas, Marlon, and ter Hofstede, Arthur H. M.: Let's Dance: A Language for Service Behavior Modeling, OTM Conferences (1), 145162, 2006
19. Montali, Marco, Pesic, Maja, van der Aalst, Wil M. P., Chesani, Federico, Mello, Paola, and Storari, Sergio: Declarative specification and verification of service choreographiess, TWEB 4(1), volume 4, 2010
20. Honda, Kohei, Mukhamedov, Aybek, Brown, Gary, Chen, Tzu-Chun, and Yoshida, Nobuko: Scribbling Interactions with a Formal Foundation, ICDCIT, 5575, 2011
21. Charfi, Anis, and Mezini, Mira: Hybrid web service composition: business processes meet business rules, ICSOC, 3038, 2004
22. Reijers, Hajo A., Limam, Selma, and van der Aalst, Wil M. P.: Product- Based Workflow Design, J. of Management Information Systems 20(1), volume 20, 229262, 2003
23. van der Aalst, W. M. P., and Pesic, M.: DecSerFlow: Towards a Truly Declarative Service Flow Language., Lecture Notes in Computer Science : Web Services and Formal Methods, Volume 4184, 2006, 123, 2006
24. Van Der Aalst, Wil M. P., Hofstede, Arthur H. M. Ter, and Weske, Mathias: Business process management: a survey, Proceedings of the 2003 international conference on Business process management, Springer-Verlag, 112, 2003
25. Khalfallah, Malik, Barhamgi, Mahmoud, Figay, Nicolas, and Ghodous, Parisa: A Novel Approach to Ensure Interoperability Based on a Cloud Infrastructure, ISPE CE, 11431154, 2012
26. Figay, Nicolas, Ghodous, Parisa, Khalfallah, Malik, and Barhamgi, Mahmoud: Interoperability framework for dynamic manufacturing networks, Computers in Industry 63(8), volume 63, 749755, 2012
27. Munusamy, Kanmani, Selamat, Harihodin Bin, Ibrahim, Suhaimi, and Baba, Mohd Sapiyan: A comparative study of process mediator components that support behavioral incompatibility, CoRR abs/1110.2258, volume abs/1110.2258, 2011