A Tool for Model-Driven Development of Collaborative Business Processes

Jorge Roa[1], Verónica Castañeda[1], Pablo Villarreal[1], Omar Chiotti[1,2]

[1]CIDISI, Universidad Tecnológica Nacional-FRSF, Lavaise 610, 3000, Santa Fe, Argentina
{pvillarr}@frsf.utn.edu.ar
[2]INGAR-CONICET-UTN, Avellaneda 3657, 3000, Santa Fe, Argentina
{chiotti}@santafe-conicet.gov.ar

Abstract. New management models encourage enterprises setting up collaborative relationships with their partners through Business-to-Business (B2B) solutions. This requires the modeling and specification of collaborative business processes that define the global view and behavior of the inter-enterprise collaboration. In order to provide a development environment that enables the development of B2B solutions, in this work we propose a tool that supports a model-driven development method for modeling, verification and specification of collaborative business processes. The tool consists of a set of plug-ins for the Eclipse platform. On one hand, we discuss the development of a set of graphical editor plug-ins that support the visual modeling of collaborative processes by using the UP-ColBPIP Language. On the other hand, we discuss the implementation of a model transformation machine that supports the model transformations required to generate Petri Net specifications for verifying the correctness of collaborative process models.

Keywords: Collaborative Business Process, Business-to-Business, Model-Driven Development, Development Environment.

1 Introduction

New management models, such as those for the supply chain management, encourage enterprises setting up collaborative networks with their trading partners through Business-to-Business (B2B) solutions. A central issue is the design of collaborative business processes, which define the global view and behavior of B2B collaborations.

The development of a B2B Collaboration requires the definition of a business solution and a technological solution. The *business solution* refers to model collaborative processes from a business perspective in an independent way of the implementation technologies; i.e. to define: partners' roles, common business goals to be achieved, collaborative processes that fulfill the goals, business documents to be exchanged, and the collaboration behavior in these processes. The *technological solution* consists of generating specifications of collaborative processes in a technology-dependent language in order to execute processes through business process management systems. Example of a B2B standard language for specifying

XXXIV Conferencia Latinoamericana de Informática (CLEI 2008), pp. 900-909, 2008.

collaborative processes based on Web Services is the Business Process Execution Language for Web Services (BPEL) [1].

In [2], [3] a model-driven development method for collaborative business processes, which is based on the Model-Driven Architecture (MDA) [4], has been proposed to support the development of B2B collaborations. The main benefits of this method are: increase of the abstraction level, since the focus is on the design of technology-independent collaborative processes; reduction of development costs and times and guarantee of alignment of a business solution with a technological solution, since process specifications are generated automatically from process models.

The proposed MDA-based method for collaborative business processes fulfills the above requirements by providing: the UP-ColBPIP modeling language [2], [3] for the design of collaborative processes independent of the idiosyncrasies of particular B2B standards; and a set of model-to-model and model-to-code transformations for the automatic generation of B2B specifications and Petri Net [5] specifications from collaborative process models. Petri Net specifications are generated for enabling the verification of collaborative processes modeled with the UP-ColBPIP language.

To support the language and model transformations proposed by the MDA-based method for collaborative business processes, a specific tool is required in order to provide a development environment so that business analysts and system designers can develop B2B collaborations.

In this work we present a tool for the model-driven development of collaborative business processes. The tool was built on the Eclipse open development platform [6]. The tool consists of: a set of Eclipse-based plug-ins that support the visual modeling of collaborative processes with the UP-ColBPIP language, and transformation machines that execute model transformations and generate Petri Net specifications and B2B specifications based on BPEL.

This paper is organized as follows. Section 2 describes the MDA-based method for Collaborative Business Processes. Section 3 describes: the architecture of the Eclipse-based tool for Collaborative Business Processes; the design, development and functionalities of the Eclipse plug-ins for the UP-ColBPIP language; and the implementation of the model transformation engine for generating Petri Net specifications. Section 5 discusses related work and section 6 presents conclusions.

2 The MDA-based Method for Collaborative Business Processes

The MDA-based method for collaborative processes [3], [7] provides a development process consisting of three phases: *analysis and design of collaborative processes, verification of collaborative processes and generation of B2B specifications.*

The analysis and design of collaborative processes requires to model these processes from a business perspective using concepts that are less bound to the implementation technology and are closer to the B2B collaborations domain. In this phase business analysts and system designers focus on the business aspects of B2B collaboration.

We have proposed the UP-ColBPIP language, which is a UML Profile that enables the modeling of technology-independent collaborative processes from a business perspective and fulfills the requirements of B2B collaborations, such as decentralized

management, enterprise autonomy, and focus on business interactions. This language encourages a top-down approach and supports the modeling of four views:

- *B2B Collaboration View*, which defines the participants (trading partners and their roles) of the B2B collaboration, the collaborative agreement's parameters and the common business goals to be fulfilled.
- *Collaborative Processes View*, which identifies the collaborative processes required to achieve the common business goals.
- *Interaction Protocols View*, which defines the explicit behavior of the collaborative processes through the use of interaction protocols. UP-ColBPIP extends the semantics of the UML2 interactions to model interaction protocols.
- *Business Interfaces View*, which defines the business interfaces of the trading partners, which contain the business services (operations) that support the exchange of messages of the interaction protocols.

Through the Interaction Protocols View the behavior of collaborative processes is defined. An *interaction protocol* describes a high-level communication pattern through a choreography of business messages between trading partners playing different roles. The message choreography describes the global control flow of the peer-to-peer interactions between trading partners, as well as the responsibilities of the roles they fulfill. Figure 1 shows a sequence diagram of the interaction protocol that realizes the *Demand Forecast Request* collaborative process.

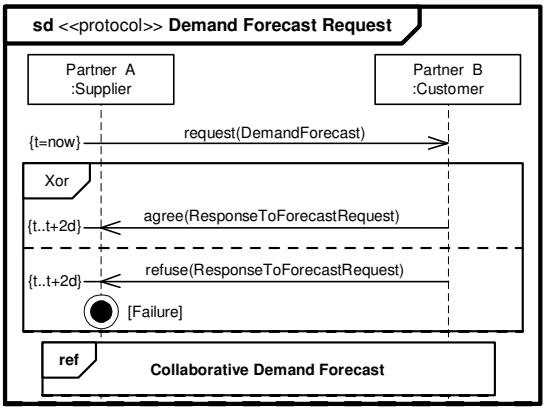

Fig. 1. Sequence Diagram of the *Demand Forecast Request* Interaction Protocol

The second phase consists of verifying the correctness of collaborative processes. To do that, interaction protocols are mapped into Petri Net specifications, which can be verified by using Petri Net tools. According to the results of the verification, it is possible to go back to the first phase in order to correct the processes and several cycles of analysis/design and verification can occur. Finally, the third phase consists of selecting the target implementation technology (i.e. the B2B standards and languages to be used) and generating the B2B specifications that fulfill the collaborative processes defined in the first phase.

To produce the output artifacts of the second and third phases, a generic pattern of transformations is proposed based on the MDA principles (Figure2a). Input artifacts are technology-independent collaborative process models defined with the UP-

ColBPIP language. Intermediate artifacts are technology-specific models and output artifacts are XML-based specifications of the target technology-specific languages.

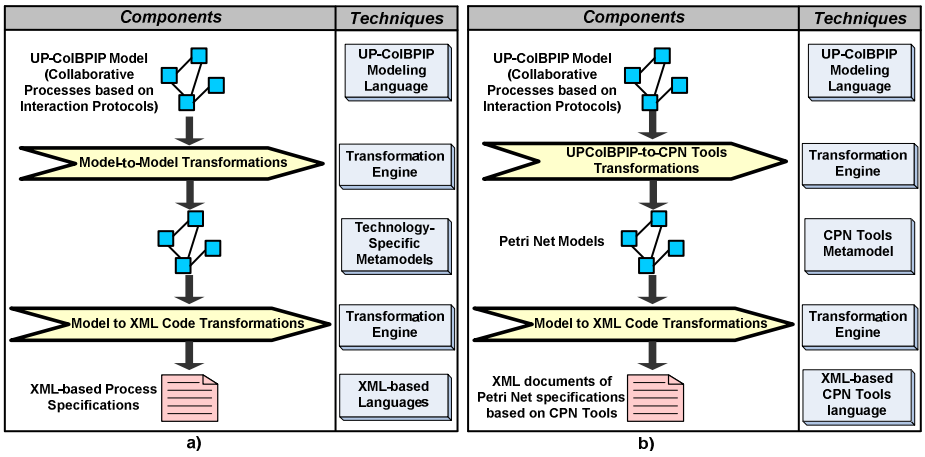

Fig. 2. Pattern of Transformations to generate B2B Specifications

Two types of transformations are supported: transformations of technology-independent models into technology-specific models, and transformations of technology-specific models into XML-based process specifications. Both types of transformations are supported by a transformation engine. It also requires the building of metamodels corresponding to the target technology-specific languages.

Figure 2.b shows the pattern of transformations proposed for generating Petri Net specifications. The target language is CPN Tools [8], which supports the verification of properties of Petri Nets. Taking as input a UP-ColBPIP model, model-to-model transformations are applied to generate Petri Net models for each protocol. These models are defined according to the language used by CPN Tools to store XML-based specifications of Petri Nets. To do that, the CPN Tools metamodel is defined for building CPN Tools models. Finally, from these models, XML-based CPN Tools documents containing Petri Net specifications of protocols are generated.

3 Eclipse-based Tool for Collaborative Business Processes

In order to provide a development environment for the MDA-based method for collaborative business processes, we have developed a tool that supports the language and model transformations proposed in this method. Several requirements were considered in the development of the tool: implement a set of editors for the UP-ColBPIP language, implement the metamodel of this language to manipulate UP-ColBPIP models, provide extension mechanisms to allow the addition of new editors and transformation machines, management of projects of B2B collaborations containing UP-ColBPIP models and diagrams where model files must be separated from diagram files. The developed tool is based on the Eclipse open development platform [6]. Thus, we use a well-known development environment for business

analysts and system developers, since there are several modeling and development tools based on the Eclipse platform.

3.1. Architecture of the Eclipse-based Tool

The Architecture of the Eclipse-based Tool for Model-Driven Development of Collaborative Processes consists of the following components (Figure 3):

- A set of Eclipse-based plug-ins, which are graphical editors that support the definition of UP-ColBPIP diagrams and models. They were built by using the Graphical Modeling Framework (GMF) [9], which provides an infrastructure for developing graphical editors based on the Eclipse Modeling Framework (EMF) [10] and the Graphical Editing Framework (GEF) [11].
- A Transformation Machine for Petri Net specifications. This machine was built using the model transformation language and toolkit ATLAS Transformation Language (ATL) [12], which is based on the Eclipse platform and enables the definition and execution of model transformations. This machine takes as input a UP-ColBPIP model and by applying model transformations produces a Petri Net specification for each interaction protocol defined in the input model.
- A Transformation Machine for BPEL specifications. This machine is also built using ATL. This machine takes as input a UP-ColBPIP model and by applying model transformations produces: BPEL specifications of the partner roles for each protocol of the input model, and one WSDL specification for each partner role defined in the input model. This machine is out of the scope of this work.

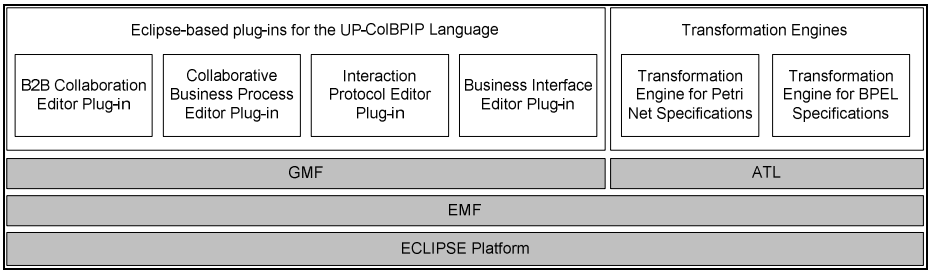

Fig. 3. Architecture of the Eclipse-based Tool for Collaborative Processes

3.1 Eclipse Plug-ins for the UP-ColBPIP Language

The Eclipse plug-ins for the UP-ColBPIP language were developed by using GMF. They have the purpose of supporting the UP-ColBPIP language in order to business analysts can define B2B collaborations and model collaborative business processes.

The first step in the development of the plug-ins was the implementation of the UP-ColBPIP metamodel, which defines the syntax and semantics of the language. The UP-ColBPIP metamodel provides the conceptual elements to model the views supported by the language, which are defined in four packages: *B2B Collaborations*, *Collaborative Processes*, *Interaction Protocols and Business Interfaces*.

Figure 4 shows the conceptual elements of the Interaction Protocols View. A collaborative process can be realized through an interaction protocol, which defines a message choreography through an ordered sequence of: business messages, control flow segments, interaction paths, protocol references and terminations. UP-ColBPIP allows the association of a speech act to business messages. A speech act means the intention that a trading partner has with regard to the business document sent in the message. Thus, decisions and commitments done by the trading partners can be known from the speech acts. This also enables the definition of complex negotiations.

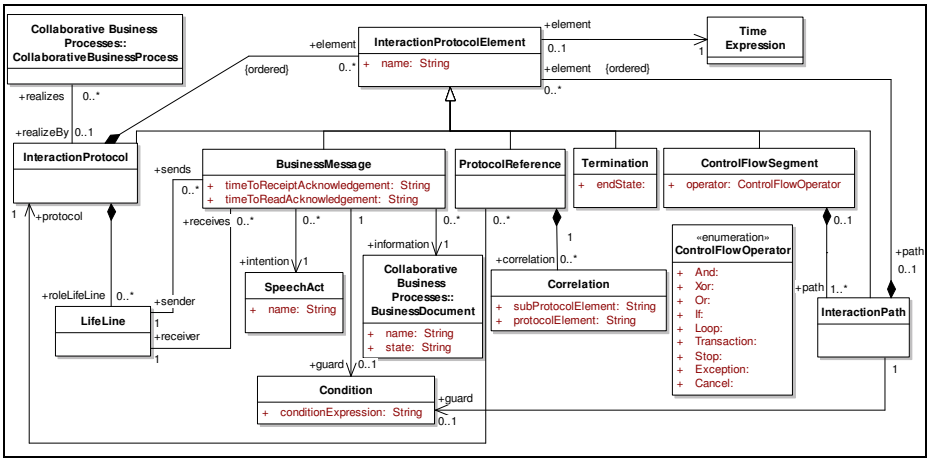

Fig. 4. Interaction Protocol package of the UP-ColBPIP metamodel

The UP-ColBPIP metamodel was implemented by using EMF and some modifications are applied to its conceptual version to solve implementation issues. The EMF implementation of the metamodel contains a main class UPColBPIPModel in the root package, which is composed of a B2B Collaboration. Thus, a UP-ColBPIP model can contain only one B2B Collaboration with its corresponding collaborative processes. Also, the root package contains the four packages of the metamodel.

The next step was the creation of the graphical editors required by the UP-ColBPIP language. For each UP-ColBPIP view, a graphical editor was generated as an Eclipse plug-in that enables the definition of the diagram corresponding to the view that it supports. The development of each graphical editor plug-in with GMF consists of: (1) create graphical representations for nodes/links of the diagram, (2) develop the tool definition, (3) define the mapping of the conceptual elements with their graphical representations and tool definition, (4) create the generator model and adjust the generation parameters, (5) and generate the plug-in code through the generator model.

In this way, four plug-ins were generated to support the UP-ColBPIP language: the B2B Collaboration editor, the Collaborative Business Process editor, the Interaction Protocol editor and the Business Interface editor (Figure 5).

A UP-ColBPIP model is created when a new B2B Collaboration diagram is generated with the B2B Collaboration Editor plug-in. Then collaborative process and interaction protocol diagrams can be created by using the Collaborative Business Process Editor and the Interaction Protocol Editor. Through these editors, the UP-

ColBPIP model is updated by adding the collaborative processes and interaction protocols. Each diagram is stored in a file separated from the file containing the UP-ColBPIP model. Thus the model is clearly separated from its graphical representation.

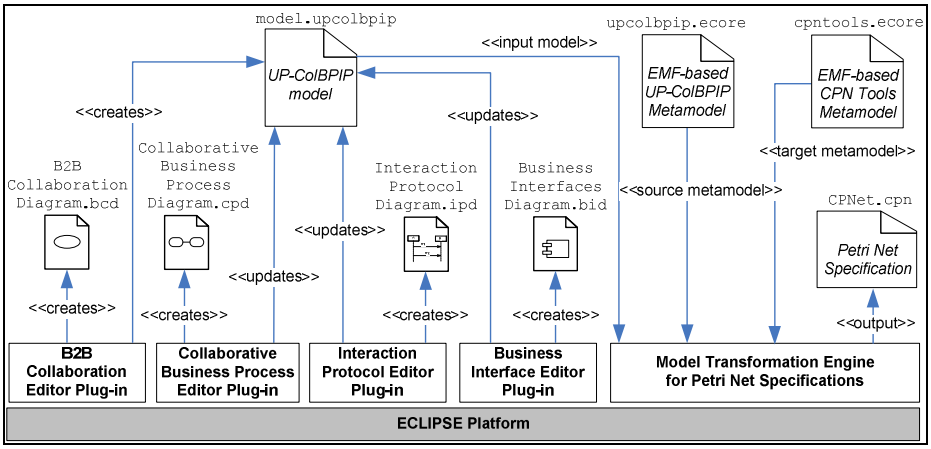

Fig. 5. Eclipse-based plug-ins for UP-ColBPIP along with the artifacts they create and update

Figure 6 shows the Eclipse-based tool with an example of a UP-ColBPIP project that contains a B2B Collaboration called *Vendor Managed Inventory (VMI) Collaboration*. The structure of the UP-ColBPIP project is shown in the Package Explorer view (on the left side). This contains a folder for each type of UP-ColBPIP diagram and another folder that contains the EMF-based UP-ColBPIP model.

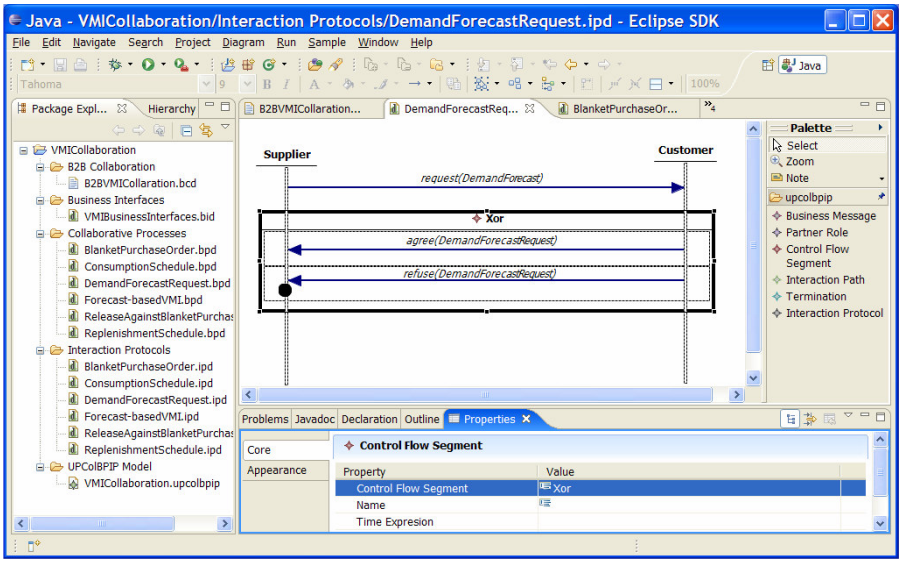

Fig. 6. Example of a UP-ColBPIP project created with the Eclipse plug-ins for UP-ColBPIP

The main edition area shows the tabs that contains the diagrams. In particular, the Interaction Protocol editor plug-in is shown which contains the interaction protocol *Demand Forecast Request*. The right side shows the tool palette with the elements used to model an interaction protocol. Through the drag and drop of palette elements into the diagram, a protocol can be created and modified. Finally, the properties view can be used to set the attributes of the model elements defined in the diagram.

3.2 Model Transformation Engine for Petri Net Specifications

In order to enable the verification of the correctness of collaborative processes defined in a business solution, we propose the use of Hierarchical Colored Petri Nets (CP-Nets) to formalize and verify interaction protocols. The conceptual mapping of interaction protocols into Hierarchical CP-Nets is described in [13]. This mapping is based on a set of predefined CP-Net patterns. Each CP-Net pattern represents and formalizes a UP-ColBPIP conceptual element. Then, CP-Net modules are built from these patterns and are composed into hierarchical CP-Nets for representing an interaction protocol to be verified. A CP-Net is generated by each interaction protocol. Figure 7 shows the CP-Net pattern used to represent business messages.

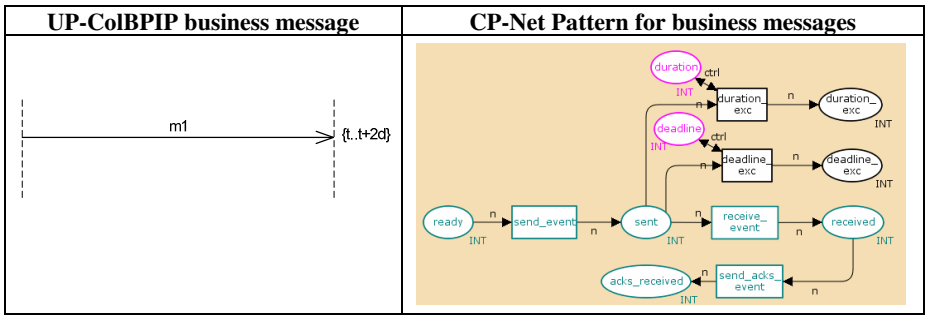

Fig. 7. CP-Net pattern for Business Messages of Interaction Protocols

Therefore, in order to generate CP-Net specifications from UP-ColBPIP models defined with the Eclipse plug-ins described above, we have developed a model transformation engine (see Figure 5) using the tool and language ATL [12]. The engine takes as input a UP-ColBPIP model and generates CP-Net specifications.

The engine is defined as a set of ATL model transformation rules that implement the mappings of interaction protocols into CP-Nets. As an example, Figure 8 shows the transformation rule that generates a CP-Net for each business message found in an interaction protocol. The source pattern of this rule (keyword *from*) looks for a business message of a UP-ColBPIP model and the target pattern generates the *PageType* element of a CPN model, which contains the required places, transitions and arcs according to the CP-Net pattern for a business message.

CP-Nets specifications of interaction protocols generated by this model transformation engine can be read and analyzed with the software CPN Tools [8]. In this way, analyzing CP-Net properties such as home marking, dead markings, liveness and so on, we can interpret the results about those properties and indicate if a protocol

is correct, i.e., all messages and paths defined on the protocol can be executed and protocol is free from deadlocks and livelocks [13].

```
rule BusinessMessage2CPNetBusinessMessage {
from p : upcolbpip!BusinessMessage
to
      tipoPageTypeMensaje:Cpn!PageType(pageattr <- tipoPageattrType,
           place <-p_ready, place <-p_acks_received, place <- p_duration,
           place <- p_deadline, place <- p_sent, place <-p_received, place <-
           p_duration_exc, place <-p_deadline_exc,
           trans <- t_send_event, trans <- t_receive_event, trans <-
           t_deadline_exc, trans <-t_duration_exc, trans <- t_send_acks_event,
           arc <- a_pr2tse, arc <-a_tse2ps, arc <- a_ps2tre, arc <-a_tsae2par,
           arc <-a_ps2tde, arc <-a_tde2pdue, arc <-a_tde2pde, arc <-a_tre2pr,
           arc <-a_pr2tsae, arc <-a_pd2tdue, arc <-a_pd2tde, arc <-a_ps2due)
```

Fig. 8. Example of the ATL transformation rule for business messages

4 Related Work

Although UML Case tools can be used to model collaborative processes based on the UP-ColBPIP language, they have some disadvantages such as: poor support to define constrains among elements of UML Profiles, poor support to implement model transformations, and a lack of an environment for domain-specific languages.

There are open and commercial tools for modeling business processes with the BPMN language [14]. However, BPMN is more suitable to describe private or public processes from the viewpoint of one partner [3]. Instead, the Eclipse-based plug-ins for UP-ColBPIP enable the modeling and verification of collaborative processes from a global viewpoint using concepts closer to the B2B collaborations domain.

A tool for modeling web services interactions from a global viewpoint is proposed in [15]. The tool is based on a proprietary platform, instead of an open source platform as Eclipse. This tool implements a language with particular notations. Instead, UP-ColBPIP uses the well-known UML notations in order to facilitate the learning of the tool by business analysts and system designers.

Finally, an Eclipse-based tool is proposed to specify and verify collaborative processes as web services choreographies based on the WS-CDL language [16]. However, for design purposes, the use of a XML-based language to model processes is less useful than a graphical modeling language such as UP-ColBPIP.

5 Conclusions and Future Work

In this work we have presented a tool based on the Eclipse platform, which supports a MDA-based method for collaborative business processes. In this way, we propose a tool based on a well-known development environment for business analysts and system designers can model, verify and specify collaborative processes.

This tool consists of a set of graphical editors implemented as Eclipse plug-ins that support the UP-ColBPIP language to model collaborative processes. The editors were

developed by using Eclipse frameworks that enable the building of model-driven development tools. GMF was used to build and generate automatically the code of the graphical editors for the UP-ColBPIP language. EMF was used to implement the UP-ColBPIP metamodel and generate automatically the code that manipulates UP-ColBPIP models. Thus, we have developed editors for the UP-ColBPIP language reducing development costs and times and reusing open source tools. Also, the use of the Eclipse platform enables to link the tool with other current and future Eclipse plug-ins contributions. However, due to these Eclipse frameworks are oriented to any application domain, some particular extensions and behaviors have been added and implemented in the interaction protocol editor plug-in.

Finally, to generate Petri Net specifications for verifying interaction protocols, we have developed a model transformation engine with the ATL language. Thus, we use a declarative and imperative approach to implement model transformations that can be integrated with the Eclipse plug-ins for the UP-ColBPIP language.

Our ongoing work is about the implementation of the transformation engine that supports the generation of B2B specifications based on the BPEL language and the incorporation of another plug-in that supports the simulation of interaction protocols.

References

1. OASIS, Web Services Business Process Execution Language (2007, May), Available: http://www.oasis-open.org/committees/download.php/23964/wsbpel-v2.0-primer.htm.
2. Villarreal, P.: Method for the Modeling and Specification of Collaborative Business Processes. PhD Thesis. Universidad Tecnológica Nacional, Santa Fe, Argentina (2005).
3. Villarreal, P., Salomone, H.E. and Chiotti, O.: Modeling and Specifications of Collaborative Business Processes using a MDA Approach and a UML Profile. In: Rittgen, P. (eds): Enterprise Modeling and Computing with UML. Idea Group Inc, 2007.
4. Object Management Group: MDA Guide V1.0.1, 2003. http://www.omg.org/mda.
5. Girault, C., Valk, R.: Petri Nets for System Engineering: A Guide to Modeling, Verification, and Applications, Springer-Verlag New York, Inc, 2001.
6. Eclipse Org. Eclipse Platform. www.eclipse.org
7. Villarreal, P., Salomone, E, Chiotti O.: A MDA-based Development Process for Collaborative Business Processes. Proceedings of the European Workshop on Milestone, Models and Mappings for Model-Driven Architecture (3M4MDA), Bilbao, España, 2006.
8. CPN tools. http://www.daimi.au.dk/CPNtools/
9. Eclipse Org. Graphical Modeling Framework. http://www.eclipse.org/modeling/gmf/
10. Eclipse Org. Eclipse Modeling Framework. http://www.eclipse.org/emf/
11. Eclipse Org. Graphical Editing Framework. http://www.eclipse.org/gef/
12. Jouault, F, and Kurtev, I: Transforming Models with ATL. In: Satellite Events at the MoDELS 2005 Conference. LNCS 3844, Heidelberg, pages 128—138. 2006.
13. Villarreal, P., Roa, J., Salomone, H.E. and Chiotti, O.: Verification of Models in a MDA Approach for Collaborative Business Processes. Proceedings 10th Ibero-American Workshop of Requirements Engineering and Software Environments (IDEAS 2007).
14. OMG/BPMI (2006). Business Process Modeling Notation 1.0. http://www.bpmn.org.
15. Decker, G.; Kirov, M.; Zaha, J. M.; Dumas, M.: Maestro for Let's Dance: An Environment for Modeling Service Interactions. BPM Demo Session at the 4th International Conference on Business Process Management (BPM 2006). Vienna, Austria, 2006.
16. Foster, H., Uchitel, S., Magee, J., Kramer, J. Model-Based Analysis of Obligations in Web Service Choreography. IEEE IC on Internet & Web Applications and Services, 2006.