An Agent-Based B2B Collaboration Platform for Executing Collaborative Business Processes

Edgar Tello-Leal[1,2], Omar Chiotti[2,3], and Pablo D. Villarreal[2]

[1] Universidad Autónoma de Tamaulipas,
Matamoros entre J.B. Tijerina y C. Colón, 87000, Victoria, Tamaulipas, México
etello@uat.edu.mx
[2] CIDISI, Universidad Tecnológica Nacional-FRSF,
Lavaise 610, S3004EWB, Santa Fe, Argentina
pvillarr@frsf.utn.edu.ar
[3] INGAR-CONICET, Avellaneda 3657, S3002GJC, Santa Fe, Argentina
chiotti@santafe-conicet.gov.ar

Abstract. Nowadays, organizations establish Business-to-Business (B2B) collaborations with their business partners. Inter-organizational collaboration is carried out through the execution of collaborative business processes. Organizations are requiring and undergoing the setting up of dynamic B2B collaborations, instead of conducting face-to-face negotiations and agreements for executing collaborative processes. This implies that business partners, maybe without a previous relationship, agree dynamically on the execution of collaborative processes based on predefined models of these processes. In this work, we propose an B2B collaboration platform which provides agent-based systems and interaction mechanisms in order to enable organizations to establish dynamic agreements with their partners and carry out the decentralized execution of collaborative processes. Agents use models of collaborative processes to enact them in a dynamic way. The role an organization performs in a collaborative process is translated into a Petri Net model that a collaboration agent interpret to execute the process.

Keywords: Collaborative Business Process, Business-to-Business, Software Agents, Model-Driven Architecture.

1 Introduction

Nowadays, organizations are focusing on the setting up of Business-to-Business collaborations with their business partners in order to manage inter-organizational collaborations and improve their performance and competitiveness. A B2B collaboration entails a process-oriented integration among heterogeneous and autonomous organizations which must be achieved at a business level and at a technological level [1]. Inter-organizational collaboration is carried out through the execution of collaborative business processes. A collaborative business process defines the global view of the interactions among enterprises to achieve common business goals [1], [2]. The design and implementation of collaborative business processes (CBPs) implies new

W. Cellary and E. Estevez (Eds.): I3E 2010, IFIP AICT 341, pp. 40–50, 2010.

challenges, such as participants autonomy, decentralized management, peer-to-peer interactions, negotiation, and alignment between the business solution and the technological solution [3], [4].

To fulfill the above issues, we proposed a methodology based on a model-driven architecture (MDA) for the design [5], verification, and implementation of CBPs [5], [6]. CBPs are modeled with the UML Profile for Collaborative Business Processes based on Interaction Protocols (UPColBPIP) [5], [6] from which business process specifications can be generated B2B standards such as BPEL or WS-CDL [3]. By using the UPColBPIP language, the behavior of CBPs is modeled through interaction protocols to represent the communicative aspects of B2B interactions.

However, organizations are also requiring and undergoing the setting up of dynamic B2B relationships, instead of conducting face-to-face negotiations and agreements, for executing CBPs. This type of dynamic B2B collaborations implies that business partners find each other and agree on the execution of CBPs based on predefined CBP models. This requires B2B platform and systems that enable organizations to create relationships with their partners and instances of CBPs in a dynamic way, based on predefined CBP models that organizations provide or find either in a partner or public repository. Traditional approaches are only focused on the execution of predefined CBPs based on static agreements among partners [7], [8].

Since the features of software agents such as autonomy, heterogeneity, decentralization, coordination and social interactions are also desirable for organizations involved in B2B collaborations [1], the use of this technology can be considered as appropriate for this domain. The use of software agents to perform CBPs helps to improves process integration, interoperability, reusability and adaptability [9], [8], [10].

Therefore, in this work we propose an agent-based B2B collaboration platform which provides agents and interaction mechanisms in order to enable organizations to establish dynamic B2B collaborations with their partners, and carry out a decentralized execution of CBPs. From a CBP model defined with the UPColBPIP language, a Petri Net model that represents the role a partner performs in the collaborative process is generated for each partner involved in the CBP. Then, collaboration agents representing each partner execute their Petri-Net models to coordinate and manage CBPs in a decentralized way. Thus, all the existing UP-ColBPIP models can be used to enact CBPs in a dynamic way. In addition, our proposal is consistent and integrated with the proposed model-driven development approaches for CBPs and B2B information systems [5], [1].

The remainder of this paper is organized as follows. Section 2 describes an MDA-based methodology for collaborative processes and the modeling approach based on the UP-ColBPIP language. Section 3 describes the architecture and agents that make up the proposed B2B collaboration platform. Section 4 describes the implementation of this platform. Section 5 presents conclusions and future work.

2 MDA-Based Methodology for Collaborative Processes

A methodology, which is based on a model-driven architecture (MDA) [11], was proposed to support the design and implementation of CBPs [6]. This methodology

consists 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* consists in the modeling of these processes from a business perspective, i.e. using concepts that are less bound to the implementation technology and are closer to the B2B collaboration domain. The UPColBPIP language [1], [6] is used to model technology-independent CBPs.

The second phase consists in verifying the correctness of CBPs defined in a UPColBPIP model. The purpose is to support the verification of these processes at an early stage of the development, when most of the fundamental decisions of a B2B collaboration are carried out, i.e. previous to the generation of the technological solution. The verification is essential to allow partners to make sure the behavior of collaborative processes is well-defined. To support this, the MDA-based method for generating Petri Net specifications from a UPColBPIP model is applied [12]. Then, interaction protocols are formalized, transformed and mapped into Colored Petri Net (CPN) [13] specifications, which are then verified with CPN Tools.

Finally, the third phase consists in selecting the target implementation technology (i.e. the B2B standards) and generating the B2B specifications (i.e. the business process specifications and interfaces of the partners' systems) that fulfill the CBPs defined in the first phase. The input is a UPColBPIP model that contains CBPs based on interaction protocols and partners' business interfaces. From this model, technology-specific business process models and technology-specific partners' interface models are made. Previous work describes the application of MDA-based methods to generate technological solutions based on the widely used B2B standards: ebXML [1], WS-BPEL [1], and WS-CDL [3].

In this work, we focus on the generation of technological solutions based on software agents in which process models are interpreted by software agents instead of process specifications based on a B2B standard.

2.1 The UPColBPIP Modeling Language

The UPColBPIP language supports the definition of behavior of CBPs through the modeling of interaction protocols. UPColBPIP extends the semantics of UML2 Interactions to model interaction protocols in UML2 Sequence Diagrams. An *interaction protocol* describes a high-level communication pattern through a choreography of business messages between organizations who play different roles. The message choreography describes the global control flow of peer-to-peer interactions between organizations as well as the responsibilities of the roles they fulfill. This also enables the representation of the decentralized management of the interactions between partners.

The conceptual elements used to define interaction protocols are: (a) *Partners* and the *Role* they fulfill, which are represented through lifelines. (b) *Business Messages* that define an interaction between two roles. They contain a *business document* whose semantics is defined by its associated *speech act*, which represents the sender's intention with respect to the exchanged business document. They also indicate that the sender's expectation is that the receptor acts according to the semantics of the speech act. (c) *Control Flow Segments* (CFS), which represent complex message sequences

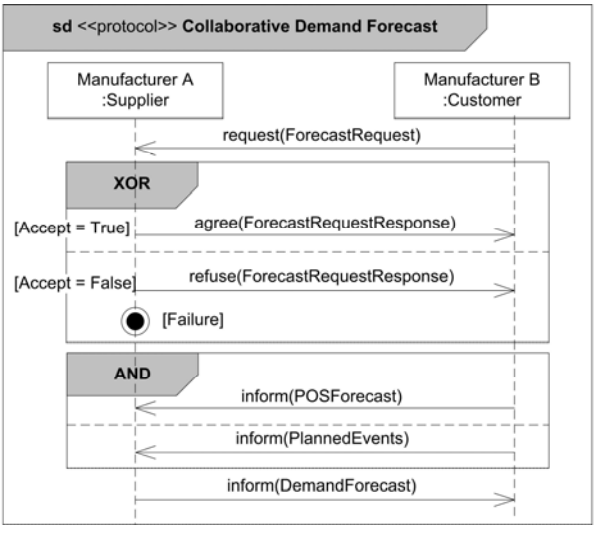

Fig. 1. Collaborative Demand Forecast interaction protocol

They contain a *control flow operator* (such as Xor, And, Or, Loop, etc) and one or more interaction paths. An *interaction path* can contain any protocol elements messages, termination events, protocol references and nested control flow segments. (d) *Protocol Reference*, which represents a sub-protocol or nested protocol. When the sub-protocol is called, the protocol waits until the sub-protocol ends. (e) *Termination event*, which represents an explicit end of a protocol. Termination events are: *success*, which implies the successful termination; and *failure*, which implies that the protocol business logic ends in an unexpected way; and (f) *Time Constraint*, which denotes a duration or deadline that can be associated with: messages, control flow segments or protocols. It represents the available time limit for the execution of such elements.

As an example of an interaction protocol, Figure 1 shows the sequence diagram of the protocol that represents the *Collaborative Demand Forecast* process. This protocol supports a negotiation process between a customer and a supplier to agree on a demand forecast. The process begins with the customer, who requests a demand forecast. The supplier processes the request and may respond by accepting or rejecting it, as it is indicated by the *Xor* control flow segment. If it is accepted, the supplier undertakes to realize the required forecast; otherwise, the process finishes with a business failure. If the supplier accepts the request, the customer informs, in parallel, a sale forecast of its points of sales (POS) and its planned sales, as it is indicated by the *And* control flow segment. Finally, with this information, the supplier generates a demand forecast and sends it to the customer. Then, the process ends.

3 An Agent-Based B2B Collaboration Platform

In this section we propose a B2B collaboration platform based on software agents in order to enable organizations to establish collaborations in a dynamic way and execute

CBPs. Since the communication among software agents is based on interaction protocols, a straightforward implementation of UPColBPIP collaborative process models based on interaction protocols is provided by this platform. Two main functionalities are supported. One is the management of dynamic agreements on collaborations among organizations to execute one or more CBPs. This implies that mechanisms for organizations can instantiate and execute CBPs whose models are provided by an organization or stored in a public repository. The second functionality refers to the capabilities to execute and monitor CBPs that organizations agreed to carry out.

The Agent-based B2B collaboration platform consists of two types of agents to support the above functionalities: the *administrator agent* and the *collaboration agent* (see Fig. 2). An *administrator agent* represents an organization or participant of a B2B collaboration and is responsible to establish communications with other administrator agents in order to agree, in a dynamic way, on the CBPs to be executed. Also, it is responsible for instantiating collaboration agents that execute CBPs.

A *collaborative agent* executes the role an organization fulfills in a CBP, and thus, it is the responsible for the jointly execution of a CBP along with the other collaborative agents from the partners involved in the CBP. Thus, at a specific moment or time, an organization can have many collaborative agents as CBPs being executed on the B2B collaboration (see Fig. 2). The following sections further describe both agents.

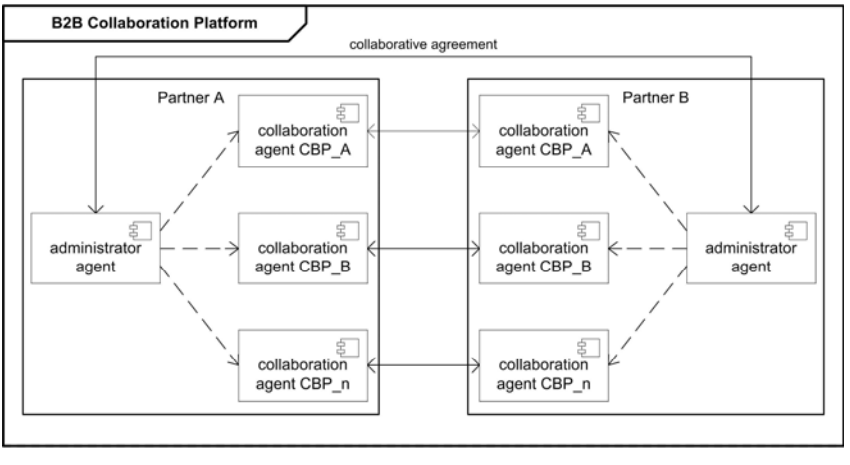

Fig. 2. Agents of the B2B Collaboration platform

3.1 Administrator Agent

Administrator agents are used to create dynamic collaboration agreements and instantiate the collaboration agents that execute CBPs. When an organization wants to establish a dynamic agreement with another organization to collaborate and execute a CBP, its *administrator agent* is used to initiate a conversation with the administrator agent of the another organization. This conversation among agents is carried out by executing a predefined CBP *Request for Collaboration*. Figure 3 shows the UPColBPIP model that represents the interaction protocol of this CBP.

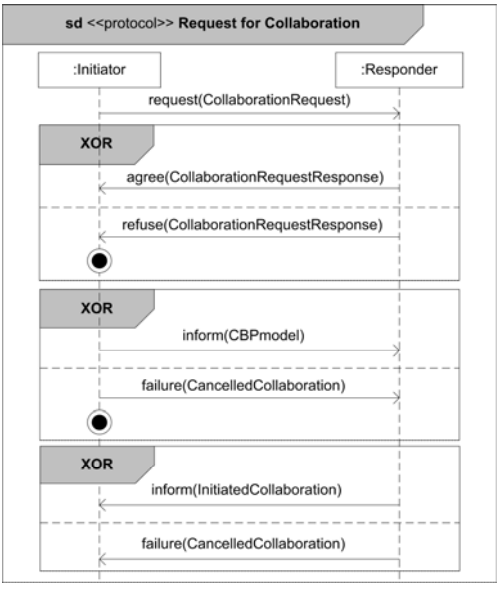

Fig. 3. Request for Collaboration Interaction Protocol

The *initiator* role is performed by the administrator agent of the organization that initiates the conversation. It sends a *request* for collaboration to a *responder* role, which is performed by the administrator agent the other organization which receives the request. This means that an administrator agent can perform the initiator or responder role according to the way that the organization engages in a collaboration agreement.

The request sent by the initiator conveys a *CollaborationRequest* document which contains the UPColBPIP model of the CBP that the *initiator* wants to execute with a responder. The *responder* evaluates the request and responds with a successful *agree* message or a *refuse* message. In the last case, it implies that the responder does not want to establish a dynamic collaboration agreement to execute the CBP, and the protocol finishes. If the responder sends an *agree* message, it waits for the process model that contains the activities that the role is going to execute in the CBP. When the initiator receives the *agree* message, it creates an instance of a collaboration agent, and if it is successful, it sends an *inform* message with the responder's process model. Otherwise, the initiator sends a *failure* message indicating that it could not instantiate its collaboration agent.

Once the responder receives the process model, it immediately instantiates a collaboration agent, and if it is successful, it sends an *inform* message indicating that the collaboration was established, and the protocol finishes. Otherwise, if an error occurs in the instantiation of the collaboration agent, the responder sends a *failure* message indicating that the collaboration was not established, and the protocol finishes. Thus, by executing this protocol administrator agents establish a dynamic agreement to execute a CBP. In case the CBP involves several organizations, the initiator executes the *Request for Collaboration* protocol with each organization that can fulfill the roles

of the CBP. Administrator agents can also remove their collaboration agents in case of the execution of a CBP needs to be interruption.

3.2 Collaboration Agent

A *collaboration agent* is responsible for performing the role an organization fulfills in a CBP. Collaboration agents carry out the jointly and decentralized execution of CBPs from an established dynamic agreement (which was achieved by administrator agents) among the organizations involved in the CBP. To perform the role of an organization, a collaboration agent takes a process model containing the behavior of the organization's role, which is known as integration process or abstract process model [5], [1]. An integration process contains the public and private activities and logic that support the role the organization performs in the CBP. Thus, a CBP is executed by the enactment of the integration process of each organization by means of the collaboration agents of the organizations.

In order to interpret and execute an integration process model by a collaboration agent, this model is defined in terms of a High-Level Petri Net (PN) model, which is derived at design-time from the UPColBPIP model of the CBP to be executed. A collaboration agent has an embedded *process machine* that interprets the PN model and executes the organization's role in a CBP (see Fig. 4). By executing this PN model, the behavior of the collaboration agents is generated at run-time. The activities or transitions defined in a PN model represent the actions (i.e. send a message to an other agent, wait for the reception of a message from another agent, execute an internal action) that a collaboration agent has to perform to support the message exchange defined in the interaction protocol of the CBP that is executed, according to the CBP role that the agent performs.

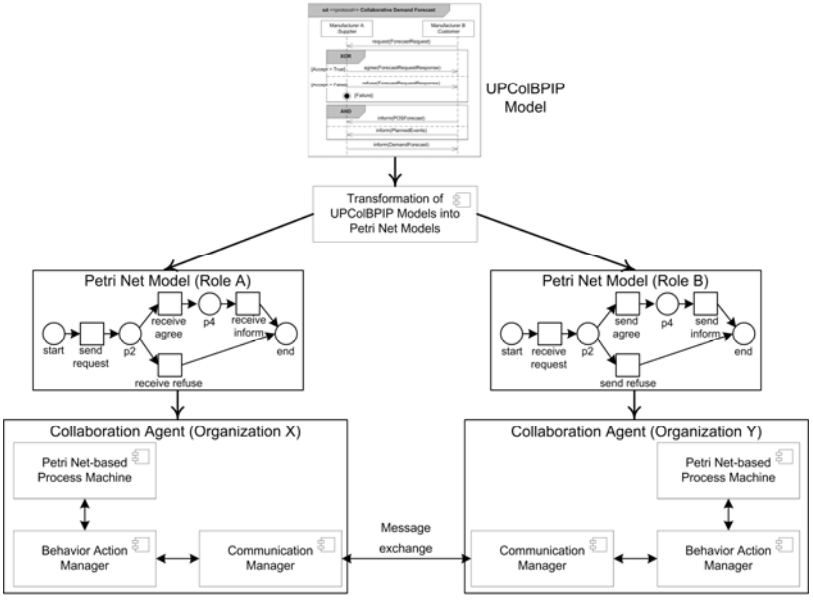

Fig. 4. Collaboration Agents executing a CBP

The behavior of this agent is driven by the *process machine* according to a PN model and it invokes the *behavior action manager* component which is responsible for planning and executing the agent's actions derived from the transitions of the PN model. The sending or receiving of a message from another agent is carried out by the *communication manager* component of the collaboration agent when the *behavior action manager* executes a sending or receiving action message. Tokens in the execution of a PN model contain the *business documents* that are exchanged between the collaboration agents. Thus, *business documents* move through the organizations' collaboration agents when the actions for the sending of messages are executed.

In this way, collaboration agents do not have actions defined at design-time, and they do not require to have the behavior of predefined interaction protocols implemented. The transitions of the PN models execute actions. Such actions are used to activate the behavior of the collaboration agent according to the type of the transition to execute. Different types of transitions (such as send message, receive message, invoke an internal process) were defined to plan and assign the behavior to be executed by the agent. Tokens maintain information about the process in which they were generated (process ID) and business documents which are sent, received, or generated internally. Therefore, the concrete actions (speech acts and business documents) of these agents are obtained at run-time according to the logic defined in PN models. This enables collaboration agents to execute any interaction protocol representing the CBP that organizations want to execute as part of a dynamic collaboration agreement. To generate the PN model that each collaboration agent uses to execute a CBP, a process transformation approach is applied to derive the PN models of the organizations from a UPColBPIP model. This is supported by applying an MDA-based method for generating PN models, as it was proposed in the second phase of the MDA approach for collaborative business processes (see Section 2). However, details of this transformation approach are out of the scope of this work.

4 Implementation of the Agent-Based B2B Collaboration Platform

In this section we describe the implementation of the Agent-based B2B collaboration platform. The software agents were built by using the Java Agent DEvelopment Framework (JADE) [14], which is a physical multi-agent development framework which complies with FIPA specifications and aims at simplifying the development and implementation of multi-agent systems [14]. Therefore, the administrator and collaboration agents are executed on the JADE platform and they use ACL [15] messages to communicate among them.

The *process machine* component of the collaboration agent was implemented by using the Java-based Petri Net framework (JFern). JFern provides an object-oriented Petri Net simulator [16]. It consists of a lightweight Petri Net kernel, providing methods to store and execute Petri Nets in realtime, and for simulation. Further, JFern supports XML based persistent storage of Petri nets and markings. The tokens of PNs contain *business documents*, which are in XML format and they are transported as parts of the content of ACL messages that agents exchange.

The agent-based B2B Collaboration platform includes an administration tool based on Eclipse Rich Client Platform. This tool has to be deployed in each organization

and provides the support to: (a) instantiate the JADE platform (b) instantiate the administrator agent of the organization, (c) search, select and retrieve the UPColBPIP and Petri Net models stored in the local repository of the organization; and (d) administrate and monitor the collaboration agents instantiated.

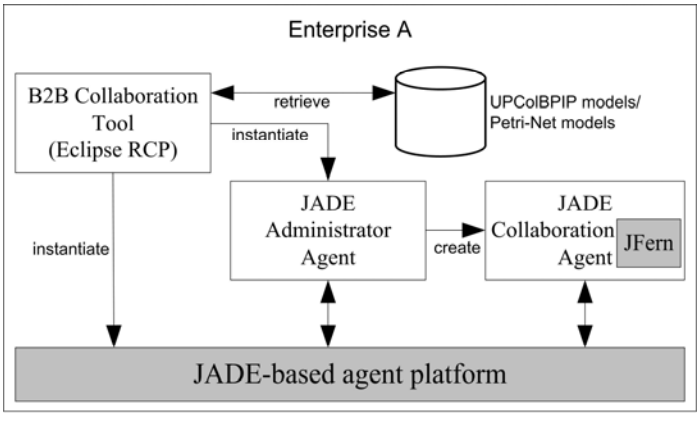

Fig. 5. Agent-based architecture for B2B collaboration

5 Conclusions and Future Work

In this work we have proposed an agent-based B2B collaboration platform which provides agent-based systems and interaction mechanisms in order to enable organizations to establish dynamic B2B collaborations with their partners, and carry out a decentralized execution of collaborative business processes (CBPs). By using this platform, UPColBPIP models can be used directly to enact CBPs in a dynamic way, instead of using process specifications based on a specific technology or B2B standard. Thus, we provide a platform to carry out a model-driven execution of CBPs, which is consistent with model-driven development approaches for CBP and B2B systems proposed in previous works.

The proposed B2B collaboration platform consists of two types of B2B agents: an administrator agent and a collaboration agent. Administrator agents representing the organizations involved in a B2B collaboration establish a collaboration agreement with other organizations by executing the proposed *request for collaboration* interaction protocol. Collaboration agents execute CBPs. We apply High-Level Petri Net models to represent the required behavior of a collaboration agent to perform the role an organization fulfills in a CBP. Petri Net (PN) models are derived from UPColBPIP models of CBPs. Therefore, process models based on Petri Nets, which collaboration agents use, are in compliance with the UPColBPIP models of the CBPs that are executed. Hence, the decentralized execution of a CBP is achieved by the enactment of a PN model carried out by each collaboration agent representing the role of an organization in the CBP. In addition, due to the behavior of collaboration agents is driven by the Petri Net-based process machine, their actions are generated in run-time to support the execution of any interaction protocol representing a CBP. This makes more

flexible the architecture of agents for executing CBPs. This is different from the traditional development of agents where actions and interaction protocols that can be executed are beforehand implemented and defined in design-time.

Finally, an implementation of the agent-based B2B platform is based on the JADE agent platform. Also a Petri Net simulator was used to implement the process machine of the collaboration agent. In addition, a distributed B2B collaboration management tool based on the Eclipse platform was developed that enables organizations to manage their agents and their own repository of CBP models.

Future work is about the addition of mechanisms and tools to discover CBP models either in a public or organizations' repositories. In the current implementation of the collaboration agent the private activities or processes of the organizations, which generate or processes the exchanged messages, are simulated. However, a support for a real integration and execution will be developed by integrating Web service technology with agents. Another future work is to establish a mechanism based on CBP *Request for Collaboration*, in which the administrator agent determines if *agree* or *refuse* to establish a dynamic agreement with another organization to collaborate and execute a CBP.

References

1. Villarreal, P.D., Salomone, E., Chiotti, O.: Modeling and Specifications of Collaborative Business Processes using a MDA Approach and a UML Profile. In: Rittgen, P. (ed.) Enterprise Modeling and Computing with UML, pp. 13–45. Idea Group Inc., USA (2007)
2. Roser, S., Bauer, B.: A Categorization of Collaborative Business Process Modeling Techniques. In: 7th IEEE International Conference on E-Commerce Technology Workshops, pp. 43–54 (2005)
3. Villarreal, P.D., Salomone, E., Chiotti, O.: Transforming Collaborative Business Process Models into Web Services Choreography Specifications. In: Lee, J., Shim, J., Lee, S.-g., Bussler, C.J., Shim, S. (eds.) DEECS 2006. LNCS, vol. 4055, pp. 50–65. Springer, Heidelberg (2006)
4. Weske, M.: Business Process Management. Concepts, Languages, Architectures. Springer, Heidelberg (2007)
5. Villarreal, P.D., Salomone, E., Chiotti, O.: A MDA-based Development Process for Collaborative Business Processes. In: European Workshop on Milestone, Models and Mappings for Model-Driven Architecture (3M4MDA) (2006)
6. Villarreal, P.D., Lazarte, I., Roa, J., Chiotti, O.: A Modeling Approach for Collaborative Business Processes based on the UP-ColBPIP Language. In: Rinderle-Ma, S., et al. (eds.) BPM 2009 Workshops. LNBIP, vol. 43, pp. 318–329. Springer, Heidelberg (2010)
7. Liu, C., Li, Q., Zhao, X.: Challenges and opportunities in collaborative business process management: Overview of recent advances and introduction to the special issue. Information Systems Frontiers 11(3), 201–209 (2009)
8. Zinnikus, I., Hahn, C., Fischer, K.: A Model-driven, Agent-based Approach for the Integration of Services into a Collaborative Business Process. In: 7th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2008), pp. 241–248 (2008)
9. Trappey, C.V., Trappey, A.J.C., Huang, C.-J., Ku, C.C.: The design of a JADE-based autonomous workflow management system for collaborative SoC design. Expert Systems with Applications 36(2), 2659–2669 (2009)

10. Guo, L., Robertson, D., Chen-Burger, Y.H.: A Novel Approach for Enacting the Distributed Business Workflows Using BPEL4WS on the Multi-Agent Platform. In: The 2005 IEEE International Conference on e-Business Engineering (2005)
11. OMG. MDA Guide V1.0.1, http://www.omg.org/cgi-bin/doc?omg/03-06-0103-06-01.pdf
12. Villarreal, P., Roa, J., Salomone, H.E., Chiotti, O.: Verification of Models in a MDA Approach for Collaborative Business Processes. In: 10th Ibero-American Workshop of Requirements Engineering and Software Environments (2007)
13. Girault, C., Valk, R.: Petri Nets for System Engineering: A Guide to Modeling, Verification and Applications. Springer, New York (2001)
14. Bellifemine, F., Caire, G., Greenwood, D.: Developing Multi-Agent Systems with JADE. Wiley, Chichester (2007)
15. FIPA. FIPA Agent Communication specifications deal with Agent Communication Language (ACL), http://www.pa.org/repository/aclspecs.html
16. Nowostawski, M.: JFern - Java-based Petri Net framework (2003)