A Conceptual Model for Business-Oriented Management of Web Services

Jyhjong Lin/Yu-Ying Hsu/Chun-Yu Hsu
Department of Information Management
Ming Chuan University
Kweishan, Taoyuan County, Taiwan 333
E-mail: jlin@mcu.edu.tw, s2750234@ss24.mcu.edu.tw, s2750040@ss24.muc.edu.tw
Fax: 886-3-3593875

Abstract

Web services have been developed in recent years as a fundamental technique for the new generation of business-to-business (B2B) or enterprise application integration (EAI) applications. As perceived, the current development research about them is focusing on their underlying infrastructures such as SOAP, UDDI, WSDL, WSCL, BPEL, BPML, and among others. However, once such technical issues get matured and more Web services become available, the attention will naturally shift from deploying these services to managing them. From the perspective of business management, this means that these services are monitored and controlled for fulfilling business objectives. In this paper, we propose an object- oriented modeling approach that addresses this issue by dividing required mechanisms into three layers: business objective, service agent, and service composition ones. With this architecture, Web services are managed via the recognition of a business objective, the employment of a service agent that arranges a composition of demanded Web services for achieving the objective, and the confirmation of interactions/coordination among these services in achieving the objective. For specification, an object-oriented model is presented for each layer that describes the working detail of that layer. To illustrate, these models are applied in the fulfillment of a business travel plan that involves a set of business objectives to be achieved by various Web services offered by different providers.

Keywords: Web service, business management, object-orientation, conceptual modeling

1 Introduction

Conceptual modeling is an important technique for representing (part of) a complex situation in an abstract manner with concise notations. It has been commonly used, for example, in analyzing and specifying user requirements of a computer-based application, as well as collecting and representing information required for dealing with complex technical and/or managerial issues to be resolved. In general, conceptual modeling can be achieved by using function-, data-, or object-oriented ways where the development of object-oriented ones is particularly motivated by the drawbacks and problems in the other two kinds: the significant features and benefits of object- oriented approaches would make the resultant models more abstract and hence easier to be understood, maintained, and reused.

For the rapid advances of Internet technologies in recent years, Web services have been developed as a fundamental technique for the new generation of business-to-business (B2B) or enterprise application integration (EAI) applications. As perceived, the current development research about them is focusing on their underlying infrastructures such as XML [1,2], SOAP [3], UDDI [4], WSDL [5], WSCL [6], BPEL [7], BPML [8], and among others. However, once such technical issues get matured and more Web services become available, the attention will naturally shift from deploying these services to managing them. From the perspective of business management, this means that these services are monitored and controlled for ensuring the fulfillment of a business objective (or goal used interchangeably in the literature [9]). In our knowledge, this managerial issue is needed in order to specifically deal with such a dynamic and changeable environment on the business/Internet nowadays. As stated above, in order to address this complex issue with an abstract conceptual modeling mechanism, it is not uncommon to think of the powerful object-oriented paradigm that possesses such features as encapsulation of an object's specifics and interacted/coordinated nature of its behaviors with other objects; these features make an object-oriented approach easier to be configured for an extensive support of addressing this issue. To account for this, we propose in this paper such an object-oriented method for modeling and specification of the business management issue of Web services.

As clarified in [10], business management of Web services refers to what service clients really care about that includes the recognition of a business objective and how the objective is specified and achieved by required Web services under a commitment mechanism (i.e., engaging the achievement of these objectives through the executions of these Web services). A traditional way to deal with these needs includes specifying/directing the executions of these services with such languages as BPEL [7] and WSCL [6], and then mapping the execution effects into meaningful metric values that are inspected for checking the satisfaction of the business objective. As one may see, this approach does not address on the mapping with a holistic manner from what objective is expected to how services collaborate to support it; instead, focus is put by an ad hoc code that maps the execution descriptions into business metrics.

For this limitation, the authors in [10] proposed a systematic approach with both a metric model that describes business expectations (i.e., objectives) and a Service model that depicts how Web services collaborate to achieve these expectations. Although this

approach supports well a holistic mapping between business-level expectations and service-level collaborations, it has still some deficiencies: (1) its service model is based on BPEL that describes how services collaborate, e.g., being composed and interacted with each other, in a rather statically structured manner such that the compositions and interactions among services cannot be easily extended/modified for reusing these services in achieving various but related business objectives; and (2) similarly, its metric model for describing business objectives is specified structurally such that the possible relationships, e.g., extensions, combinations, and associations, among business objectives cannot be easily maintained for reusing these objectives in dealing with different business situations; in our view, making these relationships maintainable would specifically benefit for keeping an enterprise competitive by easy adjustment, e.g., extensions or modifications, of her business objectives to respond to the dynamic and changeable business environment nowadays. To overcome these limitations, our approach takes advantage of the object-oriented paradigm, together with the use of visual notations and formal mechanisms, to specify business-level objectives and their corresponding service-level collaborations. It employs three layers of constructs: business objective, service agent, and service composition ones; with this architecture, the business management of Web services for an enterprise is accomplished by recognizing a set of related business objectives where each objective is engaged by a service agent that arranges a composition of Web services offered by various providers for achieving the objective. For specification, an object-oriented model is presented for each layer that describes the working detail of that layer: (1) a business objective model that specifies the desired business objectives and their relationships; (2) a service agent model that presents the agents responsible for these objectives and the compositions of Web services these agents arrange for achieving these objectives; and (3) a service composition model that describes the compositions and interactions among those Web services within a composition.

With these three models, our specifications start from a higher-level of business objective descriptions and end at a lower-level of Web service compositions. It should be particularly noted that our service composition model imposes formal constructs based on Petri nets [11-13] such that verification of objectives-compliance of the service compositions can be conducted; we believe this formality is very important for the purpose of business management, since what service clients really care about is the achievement of objectives by demanded Web services. For illustration, the three models are applied in the fulfillment of a business travel plan that involves a set of business objectives to be achieved by various Web services offered by different providers.

This paper is organized as follows. Section 2 overviews the background and motivation of the proposed approach. Section 3 presents the three models in the approach. Finally, section 4 has the conclusions and future work.

2 Background and motivation

For an open environment as on the Internet, any business objective that requires Web services offered by different providers needs to be monitored and controlled for ensuring its fulfillment. For the specification of this issue, some approaches have been proposed as those stated in [10] and the discussions about their limitations have already been presented in the previous section. To address these limitations, the author in [14] proposed a 'Web Service Componentization' concept that describes in a (object-oriented) class definition what a service composition comprises and how its constituent Web services interact with each other such that the interactions and compositions of these services can be easily amended via reuse and specialization for reusing these services in achieving different business objectives. In general, based on its object-oriented structures, this concept provides a sound mechanism for easy maintenance of the specification of a service composition. Nonetheless, by using a textual representation for specifying only the structural aspect of the composition, it lacks a visual formalism for specifying and verifying its dynamic aspect such as how constituent services collaborate and how they satisfy desired objectives; as commonly recognized, however, such a visual formalism for behavioral specification and verification is a critical conduit for comprehension and reasoning about the composition.

In addition to the issue about service-level compositions, for the purpose of business management, the specification of business-level objectives that provides a systematic mapping between objectives and compositions is also needed such that what (how) different objectives are achieved by what (how) different services, and vice versa, can be easily captured. Explicitly, this would help an enterprise in keeping competitive by proposing critical objectives and monitoring their accomplishments via demanded Web services. As stated in the previous section, the approach in [10] specifically addresses this issue by employing a metric model that provides a holistic view between objectives and services. However, from our observation, its metric model is rather statically structured such that the possible relationships, e.g., extensions, combinations, and associations, among different objectives cannot be easily maintained in order to reusing these objectives in dealing with different business situations; this would still make it difficult to adjust, e.g., extensions or modifications, these objectives to respond to the dynamic and changeable business environment nowadays (note that many existing approaches that describe business/software objectives such as those surveyed in [9] actually suffer from the same limitations).

Our method is proposed to supplement the abovementioned deficiencies in current approaches by providing a visual formalism for easy specification and maintenance of business objectives and their corresponding service compositions. In order to deal with the complexity of required mechanisms, it supports the specification in a top-down fashion. As results, a higher-level business objective model is created first that describes desired business objectives

and their possible relationships without considering detailed specification. That is, the detailed specification

via service agent and service composition models starts after all related business objectives have been described in an abstract level. We think this provides better understanding about critical objectives before proceeding too early to formally specify their accomplishments using some complex notations. Finally, due to its formal semantics of the service composition model, behavioral verification of satisfying the desired objectives can be conducted via formal analysis of the model [15]. Note that due to its enhanced modeling constructs for an extensive support of the objective, agent, and composition issues, our object-oriented model is different from other existing ones, including the most well-known UML [16-18]. Although these models can also be modified/extended to support the same specification as ours does, for space limitations, we do not address herein how such modifications/extensions may be conducted.

3 Modeling constructs

The modeling constructs of our approach include three models: (1) a business objective model that specifies the desired business objectives for an enterprise and their possible relationships; (2) a service agent model that presents the agents responsible for these objectives and the compositions of Web services they arrange for achieving these objectives; and (3) a service composition model that describes the compositions and interactions among those Web services within a composition.

3.1 The business objective model

In the literature, many classifications for objectives have been proposed as those discussed in [9] where a distinct is made between soft (non-functional) ones whose satisfaction cannot be established in a clear-cut sense and hard (functional) ones whose satisfaction can be established through verification techniques. Among other types of classification, in our knowledge, this distinct is most often referenced such that our model focuses on the specification of business objectives with soft and hard object types (classes). Figure 1 shows an example model that specifies by proper object types a 'travel plan' objective that is extended as 'recommended' and 'un-recommended' ones: to say, a customer would enjoy a planned travel either through a computer- recommended process: recommending possible travel plans, evaluating these recommended plans, booking a selected travel plan, and finally giving suggestions after the travel, or through a self-organized process: booking directly a preferred travel plan and then giving suggestions after the travel. In these two processes, however, keeping flexibility on recommending possible travel plans and booking a travel plant(i.e., adjusts those plans recommended and/or booked) is an enhanced objective for making the customer more satisfied. As shown in the figure, a (soft or hard) objective object is specified with (1) attributes such as objective priority and scope; (2) extensions into more specialized sub-types or compositions with AND/OR/XOR constituent objects [19,20]; and (3) associations with other objective objects [21] such as 'sequential' that denotes an achievement sequence from *source* to *destination*, and 'contribute' that denotes the contribution of an achievement for *source* toward that for *destination*. Further, it is noticed that an object that is composed of one or more constituent soft objects is specifically classified as a soft one. This is because an objective that is composed of one or more constituent soft sub-objectives should be classified as a soft one due to its satisfaction depending on those of these constituent sub-objectives.

3.2 The service agent model

With a business objective model, the service agent model is used to specify more detail about the desired agents that arranges demanded Web services for achieving those objectives specified (note that the reader is referred to [22-24] for employing agents for the achievement of objectives). Its description includes the compositions of Web services these agents arrange and how these services may participate in achieving various objectives (i.e., a Web service may be demanded for achieving more than one objective). The modeling constructs of the service agent model include four kinds of object type: soft/ hard objective, agent, and service ones. In particular, each agent object is specified for realizing a desired agent that arranges a composition of Web services for achieving a soft/hard objective; its specification includes a name, required properties (e.g., the effective period of its responsibility), and a set of public interface operations that are purposed for engaging the achievement of the objective through invoking the operations of its constituent service objects (that is, in our means, the execution of each interface operation would result in those of its constituent service operations that collaboratively produce a final result as the output of the interface operation). In turn, each constituent service object is specified for modeling a Web service demanded for achieving an objective with a description about its provider, port type exposed, and associated operations.

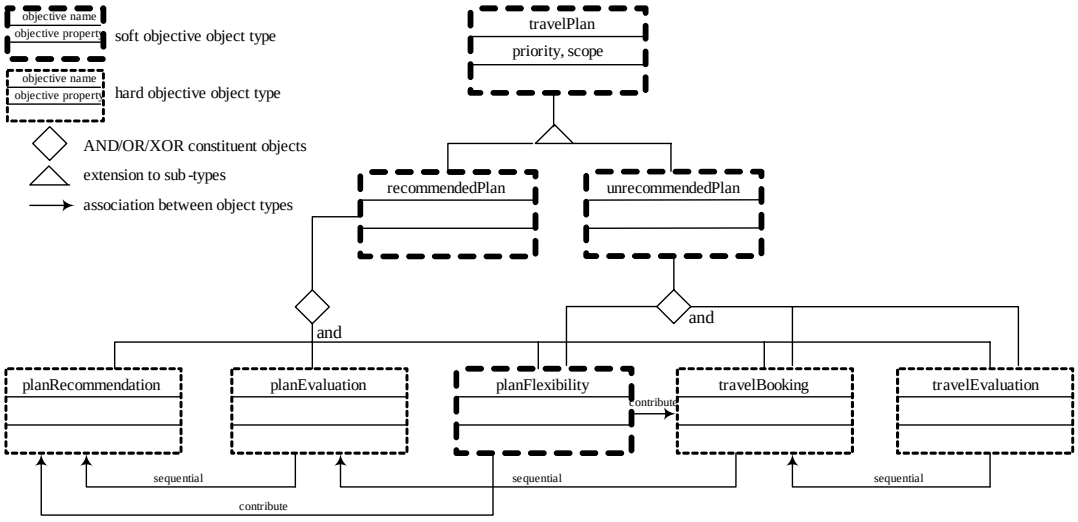

Figure 1: desired sub-objectives for a travel plan objective

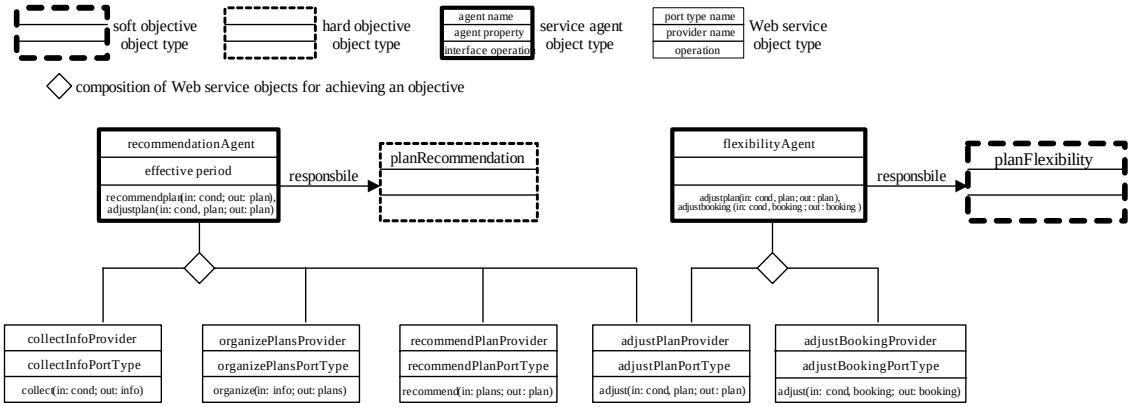

Figure 2: agents responsible for achieving desired objectives

As shown in Figure 2, two agents are identified that are responsible for achieving respectively the two 'planRecommendation' and 'planFlexibility' sub-objectives under the 'travel plan' one identified in Figure 1. Specifically, the 'recommendation' agent object is specified with an 'effective-period' property and two interface operations, 'recommendplan(in: cond; out: plan)' and 'adjustplan(in: cond, plan; out: plan)' for achieving the 'planRecommendation' sub-objective. For the 'recommendplan(..)' operation, in particular, its 'cond' input parameter is received at the start of its execution that in turn invokes some operations of the four constituent service objects; its 'plan' output parameter results at the end of its execution from the executions of those constituent operations invoked. The specification of how those constituent operations invoked collaborate to get the 'plan' output parameter produced will be presented in the service composition model below.

3.3 The service composition model

With a service agent model, the service composition model is finally used to present in detail how the operations of a service agent engage the achievement of an objective by invoking those of its constituent service objects that collaborate through various sequences, e.g., sequential, alternative, and exclusive. In general, its modeling constructs are based on Petri nets [11-13] with a set of (normal/ control) transitions and places. Normal transitions specify the operations that are executed for achieving desired objectives, while control transitions impose the control flows for those executions of normal transitions. Likewise, places are divided into two kinds: normal places that hold entity objects for the executions of transitions, and control places that hold control objects for controlling the executions of transitions. Each transition is specified with a name, a set of interaction places that its execution accesses, and a pre/post-condition that its execution satisfies. With this specification, a transition is executable if and only if each of its input places contains an object that together makes its pre-condition true. Once executed, objects in its input places are consumed by the transition, and objects in its output places are produced that make its post-condition true.

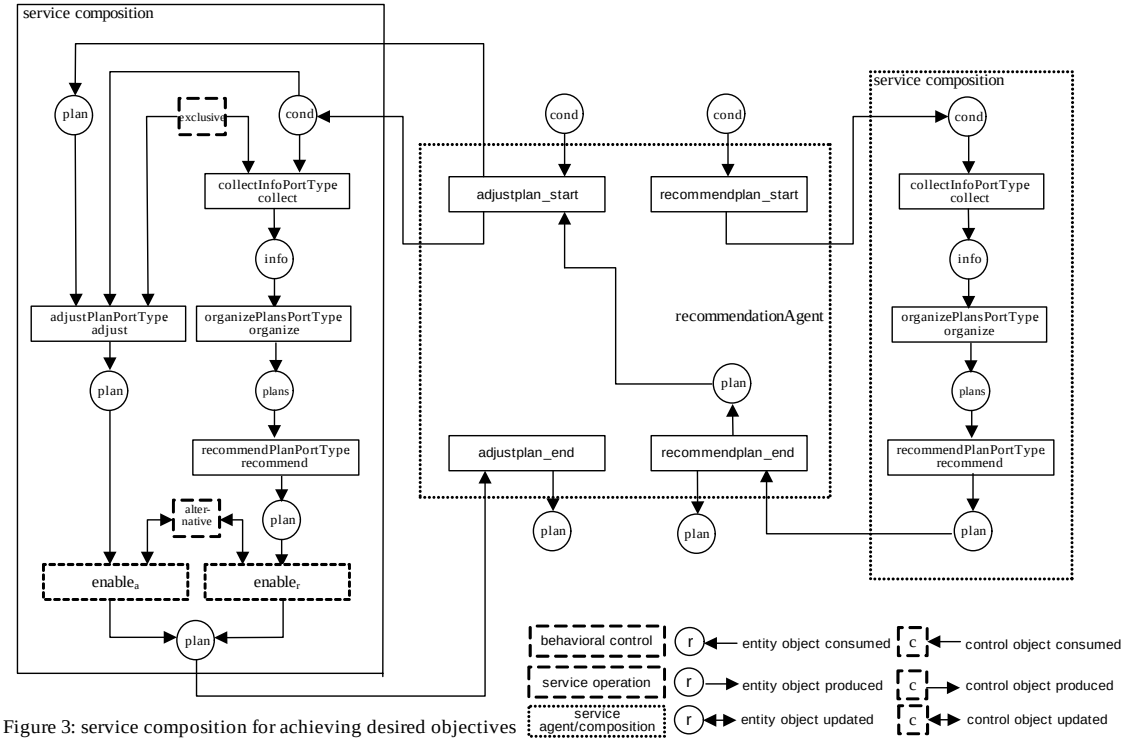

Figure 3: service composition for achieving desired objectives

In Figure 3, a service composition model is presented that describes in detail how the executions of the two interface operations, 'recommendplan(..)' and 'adjustplan(..)', of the 'recommendation' agent object result in those of the operations of four constituent service objects. As shown in the figure, at the start of the execution of the 'recommendplan(..)' operation, some predefined conditions, contained in a 'cond' entity object, are input and then forwarded to the 'collect( )' constituent service operation that bases on these conditions to collect desired travel information into a 'info' entity object; the information is then transmitted to the 'organize( )' operation for organizing adequate travel plans into a 'plans' entity object; finally, the 'recommend( )' operation evaluates these organized plans and recommends some suitable ones in a 'plan' entity object that is forwarded as the output at the end of the execution of the 'recommendplan(..)' operation. Thereafter, once some travel plans are recommended, it is however possibly needed to adjust these plans due to some conditions changed. Hence, the 'adjustplan(..)' operation is then executed in case some new conditions in another 'cond' entity object are provided. In this situation, the start of the execution results in the execution of either the 'collect( )' constituent service operation for re-recommending some new travel plans or the 'adjust( )' operation for simply adjusting those recommended plans. It is noticed that the two alternative paths are controlled via the access of a 'exclusive' control object by these two operations; in addition, for the two sets of resultant plans from these two paths, only one of them is actually available, via the alternative access of a 'alternative' control object by the two behavioral control operations, '$enable_r$()' and '$enable_a$()', as the output at the end of the execution of the 'adjustplan(..)' operation.

Finally, with the service composition model, one may see that since the model is based on Petri nets, its formal semantics can then be applied for behavioral verification of how the two interface operations of the 'recommendation' agent object engage the achievement of the 'planRecommendation' sub- objective by various collaborations of the four constituent service operations (e.g., their input/output is consistently forwarded to/eventually derivable from the service composition). This can be achieved via decision procedures that traverse the reachability graph derived from the service composition. The reader is referred to [15] for more detail about this issue.

4 Conclusions

Software requirements specification is a key activity in developing a computer-based application. Motivated by the problems in other methods, object-oriented specification methods are developed in order to produce software more understandable and maintainable. The method proposed in this paper is based on the object-oriented paradigm for formal specification about business management of Web services. In order to deal with the modeling complexity for the achievement of business objectives by demanded Web services, business objectives, service agents, and Web services are identified and specified in a top-down fashion. As results, a higher-level business objective model is created first that describes effectively desired business objectives and their possible relationships without considering detailed specification. That is, the detailed specification with service agent and composition models starts after all of related business objectives have been described in an abstract level. We think this provides better understanding about desired business objectives before proceeding too early to formally specify their achievement using some complex notations. Finally, due to its formal semantics of the service composition

model, behavioral verification of satisfying those desired objectives can be conducted via formal analysis of the model.

The work for business management of Web services has already become a new discussion. Although some researches about it have been done, but none of them provides a complete mechanism for supporting all about a holistic view between objectives and Web services, a flexible reusing of these objectives and services, and a visual formalism for their specification. Our method presented herein provides an effort on these issues by using object-oriented visual models for specifying business objectives and their possible extensions and/or constituents, employing service agents for engaging the achievement of these objectives, and imposing verifiable service compositions for achieving these objectives under the arrangement of these service agents. In our knowledge, these models are much helpful for identifying and specifying those important requirements about business objectives and their achievement by demanded Web services.

As the technical issues about Web services are getting rapidly matured in these years, more Web services are expected to be available in the near future and hence a comprehensive mechanism for full supports of their business management will certainly become much more desirable. Thus, the development of such a mechanism is a desired field. In our view, using object-oriented techniques together with sound modeling constructs is a promising approach for an effective construction of the mechanism. In our future work, we will explore further some other key issues that our models have not addressed yet, including effective registration and selection of Web services before creating a business level agreement for Web services, and desired manipulations (e.g., create, delegate, assign, cancel, and release) on the agreement during its lifecycle. As stated in [25,26], these issues are critical for keeping an agreement flexible to achieve managerial purposes. Therefore, how to specify them by using our models' constructs will be carefully explored. Meanwhile, we will construct a tool to facilitate practical application of our models. These include a design environment for building the abstract business objective model and then deriving the detailed service agent and composition models. The specification method presented in section 4 will be integrated with the tool when constructing the three models.

References

[1] Extensible Markup Language (XML), http://www.w3.org/TR/xml11.
[2] C. Goldfarb and P. Prescod, The XML Handbook, Prentice-Hall, 1998.
[3] Simple Object Access Protocol (SOAP), http://www.w3.org/2002/ws.
[4] Universal Discovery, Description, and Integration (UDDI), http://www.ibm.com/services/uddi/ standard.html.
[5] Web Services Description Language (WSDL), http://www.w3.org/TR/wsdl.
[6] A. Banerji, et al., Web Services Conversion Language (WSCL) 1.0, W3C note, March 2002.
[7] T. Andrews, et al., Business Process Execution Language for Web Services (BPEL) 1.1, May 2003.
[8] Business Process Modeling Language (BPML), http://www.bpmi.org.
[9] A. Lamsweerde, "Goal-Oriented Requirements Engineering: A Guided Tour," Proc. of 5th IEEE Int'l Conf. on Requirements Engineering, Aug. 2001, pp. 249-262.
[10] F. Casati, et al., "Business-Oriented Management of Web Services," CACM, vol. 46, Oct. 2003, pp. 55-60.
[11] J. Peterson, "Petri Nets," ACM Computer Surveys, vol. 9, no. 3, Sep. 1977, pp. 223-252.
[12] J. Peterson, Petri Net Theory and The Modeling of Systems, Prentice-Hall, 1981.
[13] E. Yiannis and L. Thomas, "Specification and Analysis of Parallel/Distributed Software and Systems by Petri Nets with Transition Enabling Function," IEEE Transaction on Software Engineering, vol. 18, March 1992, pp. 252-261.
[14] J. Yang, "Web Service Componentization," CACM, vol. 46, no. 10, Oct. 2003, pp. 35-40.
[15] J. Lin, et al., "Object-Oriented Specification and Formal Verification of Real-Time Systems," Annals of Software Engineering, 1996, vol. 2, pp. 161-198.
[16] G. Booch, et al., The Unified Modeling Language User Guide, Addison Wesley, 1999.
[17] M. Fowler and K. Scott, UML Distilled: Applying the Standard Object Modeling Language, Second Edition, Addison Wesley, 2000.
[18] J. Rumbaugh, et al., The Unified Modeling Language Reference Manual, Addison Wesley, 1999.
[19] A. Dardenne, et al., "Goal-Directed Concept Acquisition in Requirements Elicitation," Proc. of 6th Int'l Workshop on Soft. Spec. and Design, 1991, pp. 14-21.
[20] A. Dardenne, et al., "Goal-Directed Requirements Acquisition," Science of Computer Programming, vol. 20, 1993, pp. 3-50.
[21] R. Darimont, et al., "GRAIL/KAOS: An Environment for Goal-Driven Requirements Engineering," Proc. of 20th Int'l Conference on Soft. Eng., April 1998, vol. 2, pp. 58-62.
[22] A. van Lamsweerde, et al., "Managing Conflicts in Goal-Driven Requirements Engineering," IEEE Trans. on Software Engineering, Nov. 1998.
[23] A. van Lamsweerde and L. Willemet, "Inferring Declarative Requirements Specifications from Operational Scenarios," IEEE Trans. on Software Engineering, Dec. 1998, pp. 1089-1114.
[24] E. Letier and A. van Lamsweerde, "Agent-Based Tactics for Goal-Oriented Requirements Elaboration," in Proc. of 24th Int'l Conf. on Software Engineering, May 2002.
[25] K. Jain, et al., "Agents for Process Coherence in Virtual Enterprises," Communications of the ACM, vol. 42, no. 3, March 1999, pp. 62-69.
[26] K. Jain and M. Singh, "Using Spheres of Commitment to Support Virtual Enterprises," in Proc. of 4th ISPE International Conference on Concurrent Engineering: Research and Applications (CE), International Society for Productivity Enhancements (ISPE), Aug. 1997, pp. 469-476.