A framework for business rule driven service composition

Orriëns, B.; Yang, J.; Papazoglou, M.

Published in:
Proceedings of the 4th International Workshop on Conceptual Modeling Approaches for e-Business Dealing with Business Volatility

Publication date:
2003

A Framework For Business Rule Driven Web Service Composition

Bart Orriëns, Jian Yang, Mike P. Papazoglou

Tilburg University, Infolab
PO Box 90153
5000 LE, Tilburg
Netherlands
{b.orriens,jian,mikep}@uvt.nl

Abstract. With web services emerging as a promising technology for supporting open and dynamic business processes, it is witnessed that standards for business process specification in the context of web services composition have been fast developed in recent years, e.g. WSFL, XLang, BPEL. However, none of the proposing specifications really address the issues of dynamic business process creation, e.g. a vast service space to search, a variety of services to compare and match, and different ways to construct business processes. One of the assumptions these standards make is that the business process is pre-defined. Obviously this assumption does not hold if the business needs to accommodate changes in applications, technology, and organizational policies. We believe business processes can be dynamically built by composing web services if they are constructed based on and governed by business rules. In this paper we analyze the basic elements in business modelling and how they relate to the web service composition process. As a result a rule driven mechanism is developed to govern and guide the process of service composition in terms of five broad composition phases spanning abstract definition, scheduling, construction, execution, and evolution to support on demand and on the fly business process building.

1 Introduction

The Web has become the means for organizations to deliver goods and services and for customers to search and retrieve services that match their needs. Web services are self-contained, Internet-enabled applications capable not only of performing business activities on their own, but also possessing the ability to engage other web services in order to complete higher-order business transactions. The platform neutral nature of web services creates the opportunity for building composite services by combining existing elementary or complex services, possibly offered by different enterprizes. For example, a travel plan service can be developed by combining several elementary services such as hotel reservation, ticket booking, car rental, sightseeing package, etc., based on their WSDL descriptions. We use the term composite service to signify a service that employs and synthesizes other services. The services that are used in the context of a composite service are called its constituent services. Some standards are emerging, e.g., BPEL [3], which specifies business processes and business interaction protocols in the context of web service composition.

Unfortunately, composite web service development and management is currently a manual activity that requires specific knowledge about composing web services in advance and takes a lot of time and effort. This even applies to the applications that are currently being developed on the basis of available standards, such as BPEL. The difficulty is that service composition is simply too complex and too dynamic to handle manually. With a vast service space to search, a variety of services to compare and match, and different ways to construct composed services, the only alternative capable of facilitating dynamic service composition development and management is an automated process of service composition governed by rules and administrated by rule engines. In this paper we investigate a rule based approach for service composition that combines best practices from rule base systems and software engineering to support parameterization, dynamic binding, and flexible service compositions.

Rules are logical statements about how a system operates. Some of these rules may be expressed in the language of the business, referring to real-world business entities, and are therefore called Business Rules. Business rules can represent among other things typical business situations such as escalation ("send this document to a supervisor for approval"), managing exceptions ("make sure that we deal with this within 30 min or as specified in the customer's service-level agreement"). Our conviction is that business rules can be used in the context of service composition to determine how the composition should be structured and scheduled, how the services and their providers should be selected, and how run time service binding should be conducted.

In our framework a rule driven mechanism is used to steer the process of service composition in terms of five broad composition phases spanning *definition*, *scheduling*, *construction*, *execution*, and *evolution*. Based on these phases we analyze and classify business rules and determine how they impact service composition. Although previous work on business rules, such as that in [14], introduces a simple classification scheme for business process that classifies rules as relationship, constraint, authorization, choice, and action rules, these are of a general nature and do not consider service composition requirements. It is not clear for example how we can use these types of rules for the specification of constraints on scheduling, the criteria and conditions of task and resource selection, run-time constraints for service execution, and time, cost and quality concerns regarding the selection of service providers. Nevertheless, they can form a sound basis for extension and application to the service composition life cycle phases. Service composition and business rule evolution can also be handled in accordance with the phased approach we are developing. Once we know how business rules affect the various service composition phases, we can then define transformation rules to handle changes in a consistent manner.

The paper is structured as followed: we begin in Section 2 with analyzing how we can realize business processes using service composition. Next, in Section 3 we introduce a business rule driven framework for service composition. We define a classification for business rules in the context of service composition in section 4. We present our conclusions and discuss future research in Section 5.

2 Service composition for business processes

We mentioned in Section 1 that service composition can be used to describe and realize business processes. In this section we identify the basic elements

of a service composition. Subsequently we introduce an architecture to realize business processes through web service composition.

2.1 Service composition elements

The nature of Business Processes varies in terms of complexity and scope. There have been some definitions developed in the past to analyze business processes [6, 9, 7, 10, 1]. However, none of them provided a full coverage of a business process. In our view the best way to design a business process is to analyze its role in business. Business can be viewed from various perspectives, as is done for example in [12], where the data, function, organization and resource view are distinguished. In [13] a similar distinction is made, although the data view is referred to as the informational perspective. [4] differentiates between a functional, organizational and informational view. It also includes a behavioral perspective.

To find the basic elements for defining and analyzing a Business Process we adopt the framework used in [18], which includes *how*, *why*, *when*, *who*, *what* and *where* aspect. The *how* aspect explains how things are done in the business, the *why* aspect provides the rationale, the *when* aspect tells us when things happen, the *who* aspect gives information about the people and resources involved, the *what* aspect addresses the impact on the informational structures, whereas the *where* aspect describes the geographical location of the departments involved.

By studying the current standard business process specification languages (e.g., BPEL, BPML) we identify the following composition elements representing these aspects: activity, condition, event, flow, message, provider and role elements. We briefly discuss each composition element as followed:

Activity
An activity represents a well-defined business function and is part of the *how* aspect. For example, the booking of a flight is an activity in the travel plan business process. Each activity is associated with a role that is responsible for its execution. Also, an activity may be related to messages, defining its data prerequisites.

Condition
The behavior of business processes is governed by business rules. They are statements that define or constrain some aspect of the business, which are intended to assert business structure or to control or influence the behavior of the business [2]. These rules as such provide the rationale behind the business process, representing the *why* aspect. Business rules are expressed in service composition in the form of conditions. Examples include pre -and post conditions for activities, and message integrity constraints.

Event
Events in a service composition represent business events, thus originating from the *when* aspect. A business event is an occurrence of some sort. An example of an event is "No seat available on flight". Events have an impact on business processes, influencing their behavior. They can trigger new activities or change the result of running activities.

Flow

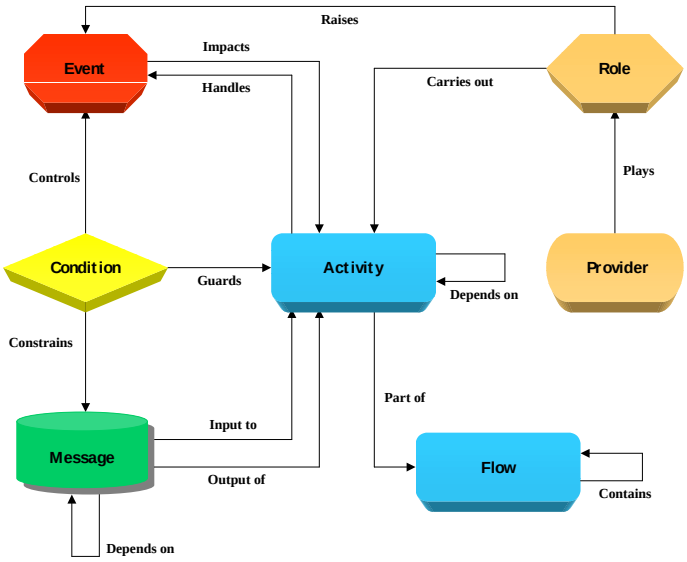

Fig. 1. Architecture for a service composition

Flow expresses the *how* aspect and is used to express the choreography of complex activities (such as business processes). Possible types include sequential, parallel, conditional and iterative flow patterns. Depending on the style of control flow modeling, e.g. block [1] or flow-transition ([3], [15]) based, a service composition can contain one or more control flow elements respectively. In both cases control flows can be nested at different levels of granularity, allowing the specification of arbitrarily complex structures.

Message
To represent the information exchanging behavior of a business process in a composition we utilize messages. Messages represent the *what* aspect and are associated with the composition as a whole, expressing the interactions of the business process with the outside. They are also linked to activities to model the distribution of information within the process. Finally, messages may be correlated to express data dependencies between activities.

Provider
The people and resources participating in a business process are depicted in a composition as providers. A provider belongs to the *who* and *where* aspect and describes a concrete web service.

Role

Roles are part of the *who* aspect and define the expected behavior of participants in the business in an abstract manner, i.e. without providing any specifics resource or person responsible for actually performing the task. In the context of service composition these provide abstract descriptions of the services involved in the composition.

2.2 Business process realization with service composition

In the previous subsection we identified and discussed a set of elements with which we can represent business processes in terms of service composition. Another important aspect related to service composition addresses the question of how we can use it to realize business processes. For this purpose we adopt a phased approach to service composition, developed in [17] and [16]. The purpose of these phases is to first describe services in the abstract and then generate executable service processes from these abstract specifications using business rules. In this approach five broad phases are distinguished, spanning composition *definition*, *scheduling*, *construction*, *execution*, and *evolution*. These phases, which together constitute the service composition life cycle, are supported by the architecture, depicted in Fig. 2.

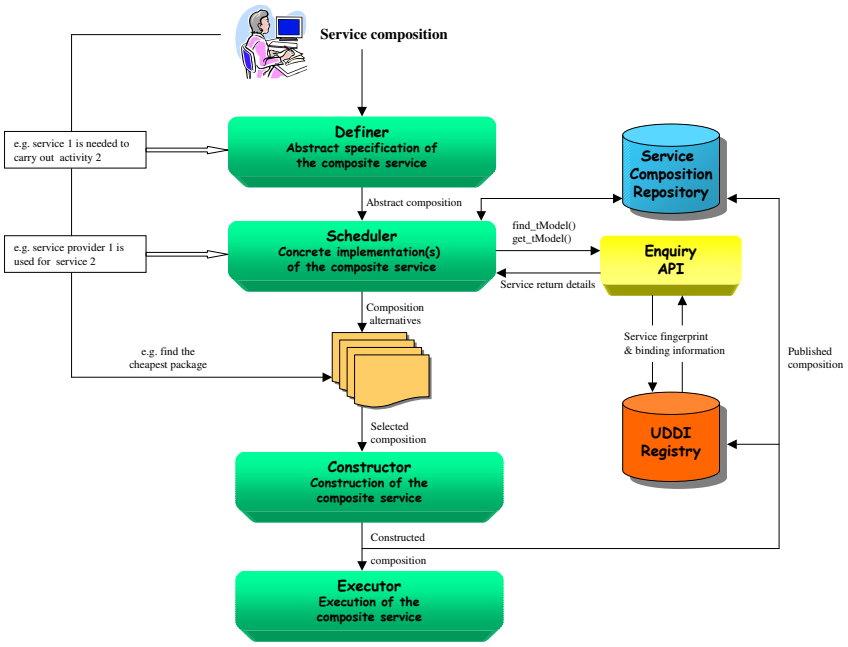

Fig. 2. Service composition life cycle architecture

There are four main components in the architecture that will be addressed as followed:

- **Definer:** it focuses on the specification of service composition definitions in an abstract manner. These definitions employ WSDL in conjunction with an orchestration language for web services to express business processes, e.g. the earlier mentioned BPEL. The tasks of the Definer include activity, constraint, event, message and role specification.
- **Scheduler:** the result of the Definer, the abstract composition, is passed on to the Scheduler. The task of the Scheduler is to concretize it by replacing the defined roles with concrete providers. For this purpose potentially available and matching service providers are located for each role through the interaction with an UDDI registry via the Enquiry API. Subsequently, the user selects a provider from the retrieved set. Note that it is possible to derive multiple implementations, allowing the user to choose between alternatives.
- **Constructor:** once the user has selected an alternative, it is passed on to the Constructor. The Constructor utilizes this composition to set up the execution environment. This is done through the generating of executable software. Optionally the composite service can be published as a new service in an UDDI registry to make it available to others. Alternatively, it may also be stored in the Service Composition Repository for reuse.
- **Executor:** the Executor is responsible for executing and managing the constructed composition. Management means monitoring of the composition behavior during execution as well as dealing with changes, e.g. when the governing rules or constituent services are subject to change. In the latter case the composition needs to be transformed to incorporate the change.

The above clearly demonstrates the idea behind the phased service composition development approach, being to start with an abstract definition and gradually make it concrete and executable.

3 Business rule driven service composition

In the previous section we analyzed the influential aspects for business process and how they can be expressed in the context of service composition. As a result we identified a set of *composition elements*. It is our conviction that interactions between these elements can be expressed in business rules. Examples include rules specifying how activities should be structured and scheduled, which roles are played by which providers and how run time service binding should be conducted.

The notion of using rules (referred to as *composition rules*) to link composition elements, enables us to define a rule mechanism to drive service composition. This is shown in Fig. 3, which shows an overview of our framework for business rule driven service composition. The advantage of the framework is that it makes the definition, scheduling, construction and execution of compositions flexible and controllable, since composition elements can be combined in a plug and play manner by using composition rules. This makes it possible to generate compositions on demand and on the fly in response to a user request. Furthermore, the framework allows service composition to become more

dynamic, because changes in the business can be easily incorporated and effectuated through the redefinition of the appropriate composition elements and rules. Additionally, plug and play service composition significantly reduces the time and effort involved in composition development and management, whilst at the same time increases the quality of the service composition process (e.g. in terms of syntactic and semantic composition correctness).

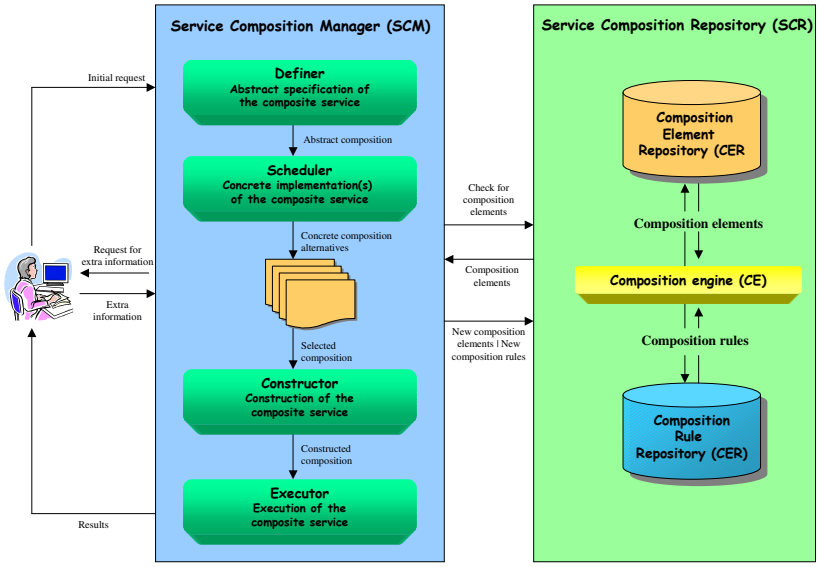

Fig. 3. Framework for business rule driven service composition

As can be seen Fig. 3 the framework consists of two main components:

- **Service Composition Manager (SCM):** it is responsible for assisting the user in developing, executing and managing service compositions. Its internal structure covers the service composition life cycle architecture discussed in Section 2.
- **Service Composition Repository (SCR):** is responsible for maintaining composition elements and rules. The **Composition Engine (CE)** facilitates storage and retrieval of these elements and rules, which are contained in the **Composition Element Repository (CER)** and the **Composition Rule Repository (CRR)** respectively.

We briefly outline how these components interact with one another to drive the service composition life cycle.

1. The User sends a request to the **SCM**. This request gives a brief description of the business activities that the User wants to perform, for example to arrange a travel plan.
2. The **SCM** receives the User request, determines what functionality is required and subsequently enters the service composition life cycle. The **Definer** begins with the definition of an abstract composition. We illustrate how this is done using the specification of activities for a travel plan as an example.

   (a) The **Definer** sends a check request to the **CE**. The purpose of this request is to find out whether there is any information available on activities for a travel plan. The **CE** receives this request and searches the **CRR** to see if there are rules specifying travel activities. If such a composition rule exists, then the **CE** uses it to retrieve the specified activities from the **CER**, which are then send to the **Definer**. Otherwise, no activity elements are returned.

   (b) In case relevant activities are found, the **Definer** will ask the User whether to use them in the composition. The User can then make a selection of which activities should be included in the composition. Optionally the User can also define and add other activities to the composition. In this case the **Definer** will not only add these activities, but also stores them in the **CER** as well as updates the composition rules in the **CRR** concerning travel plan activities.

   (c) When the User does not want to use the proposed activities (or if no relevant activity elements were found), then the User is obliged to indicate which activities should be included in the travel plan. The **Definer** subsequently uses the provided information to add activities to the composition. Similarly as in the previous step this new information is stored in the **SCR**.

   The other activities in the *Definition* phase, such as the specification of task constraints, message exchanges, runtime behavior and resource use, are performed in a similar fashion as described above.
3. When the **Definer** has developed an abstract composition, it is passed on to the **Scheduler**. The task of the **Scheduler** is, as explained in Section 2, to concretize the composition. This is done in a similar manner as the **Definer** performs its tasks. For each role in the composition the **Scheduler** contacts the **CE** to locate relevant rules regarding to the use of existing provider elements. If they are found, the specified providers are retrieved and proposed to the User. In case none are suitable according to the User or when no providers were found, the **Scheduler** initiates a search in e.g. an UDDI registry to locate suitable service providers. The User can then select a provider, which is subsequently added to the composition. The provider element is also stored in the **SCR**.
4. After the **Scheduler** has developed a set of concrete alternatives, the user is approached to select one. The selected composition is passed on to the **Constructor**, which generates executable software to enable execution. The composition is then executed by the **Executor**. The **Executor** monitors the running composition until it has completed its activities. The **SCM** subsequently presents the results to the User, for example a flight ticket and a hotel room reservation for a travel plan.

4 Business rule classification for service composition

In Section 3 we outlined a business rule driven framework for service composition. In the context of this framework it is useful to determine the types of business rules that will be required to facilitate the different phases in the service composition life cycle. There has been substantial work on business rule classification, e.g. [14] [2] [5]. The scheme in [8] provides a high-level classification, distinguishing between terms, facts and rules. Terms and facts are statements that contain sensible, business relevant observations, whereas rules are statements used to discover new information or guide the decision making process.

The problem with this (and other) classification is that it is generic and cannot be directly applied to service composition. It is not clear for example how the distinguished types of business rules can be used to specify scheduling constraints, criteria and conditions for task and resource selection, run-time constraints for service execution, and time, cost and quality concerns regarding the selection of service providers.

Therefore, we introduce our own classification scheme for business rules. In this scheme we classify business rules along two dimensions. The first dimension specifies whether we are dealing with a composition element or a composition rule, analog to the distinction between terms and facts, and rules. The second dimension positions the elements and rules in terms of to which aspect of a business process they belong, resulting in structure, role, message, event and constraint related business rules.

Structure related rules
These rules address the *how* aspect of a business process and facilitate the specification of the way in which the service composition is to be carried out in terms of activities. As discussed in Section 2 activity and flow elements are used to express this information. To combine these elements we identify *flow grouping* and *activity dependency* rules as relevant composition rules. These rules indicate respectively how activities are to be grouped in a composition and what dependencies exist between activities. To illustrate let us look at the following activity dependency rule (in pseudo code)

```
if (FlightBookingActivity and HotelActivity depended) then
(HotelActivity performedAfter FlightBookingActivity)
```

which specifies a prioritization rule relevant for a travel plan, stating that first a flight must have been booked before an attempt should be made to reserve a hotel room.

Role related rules
These rules govern the participants that are to be involved in the service composition, thus controlling the *who* and *where* aspect of the business process. These aspects are represented using and provider elements (see Section 2). The interactions of these elements with other composition elements include the assignment of a role to an activity, the raising of an event and the binding of a role to a provider. We can use *role assignment*, *event raiser* and *role binding* rules respectively to create these interactions. For example, suppose we have an activity for flight ticket booking. Then we may specify the role assignment rule

```
if (FlightBookingActivity is performed) then (Role type is
airline)
```

depicting that only airlines are capable of performing this activity. This will ensure that when concrete providers are selected only airlines are taken into consideration.

Message related rules
These rules regulate the use of information in the service composition, as such governing the *what* aspect of a business process. To express this aspect we defined message elements in Section 2. Please recall that messages are associated with a composition as a whole, depicting its use of information. Also, the information is internally distributed to the different activities. We can govern this distribution using *message distribution* rules. The actual assignment of messages to an activity we can do through *message assignment* rules. To derive message dependencies we utilize *message dependency* rules. We can illustrate using the following example

```
if (FlightBookingActivity has Input) then (Message contains
departureDate,arrivalDate,from,to)
```

showing a message assignment rule for the input of the flight booking activity, expressing that the input of the activity must include a departure date, arrival date, and from and to where the flight is to take place.

Event related rules
These rules govern the behavior of service compositions in reaction to (un-) expected events. They thus regulate the *when* aspect of a business process. Which events affect which activities is regulated by *activity influence* rules. To handle an event usually some sort of activity is performed. Which activity this is exactly we can depict in *event handler* rules, which specify the activities that should be performed in case events occur. An example is the following

```
if (SeatUnavailableEvent occurs) then (Stop the composition)
```

indicating that if there is no seat available (something which can occur e.g. in the context of a flight booking activity), the composition should be stopped.

Constraint related rules
These rules steer the use of constraints in a business process, represented by conditions in service composition. Conditions are associated with activities, specifying pre -and/or post-conditions. To specify the latter we utilize *pre-condition assignment* and *post-condition assignment* rules respectively. They can also control event occurrences and effectuate integrity constraints using *event control* and *message constraint* rules. The following example illustrates a post-condition assignment rule

```
if (FlightBookingActivity completed) then (Seat must be reserved)
```

constraining the result of the flight booking activity by specifying that after the activity has been completed a seat must have been reserved. Otherwise, the performance of the activity cannot be considered to have been successful.

In the previous we have briefly outlined how business rules can be used to govern the different aspects of a business process in the context of service composition. It should be noted that the above is preliminary and that further work is required to identify the exact rules that we require to steer the service composition development process.

5 Conclusions and future research

It is clear that current standards in service composition, such as BPEL, are not capable of dealing with the complex and dynamic nature of developing and managing composite web services. The challenge is therefore to provide a solution in which dynamic service composition development and management is facilitated in an automated fashion.

In this paper we showed how business processes can be expressed in terms of service composition through the use of various types of composition elements. Subsequently we explained our phased approach to service composition, in which five broad phases are distinguished to realize a business process, spanning composition definition, scheduling, construction, execution, and evolution. In order to cater the need for flexible and dynamic service composition, we introduced a rule driven approach that describes how business rules, referred to as composition rules, can be used to steer these five composition phases, e.g. to specify scheduling constraints for activities, criteria and conditions for task and resource selection, run-time constraints for service execution, and time, cost and quality concerns regarding the selection of service providers. We also defined a classification scheme for business rules that details how they can be applied in the context of service composition, an issue that has not been addressed in previous work on business rule classification.

We argue that the approach we presented in this paper not only makes service composition more flexible and dynamic compared to current standards, but also reduces time and effort involved in composition development and management. Nevertheless, the work reported in this paper is at its very early stage. Future work will be focused on the following issues:

- the specification and formalization of composition elements and rules;
- the design of a rule mechanism to manage and apply composition rules, e.g. rules as components, as services or as specifications;
- the architecture for the rule framework, e.g. centralized versus decentralized architecture; and
- the development of a change management system to manage the evolution of composition elements and rules and defined service compositions.

References

1. Business Process Modelling Initiative; "Business Process Modeling Language", June 24, 2002, http://www.bpmi.org
2. Business Rules Group; "Defining business rules, what are they really?", July 2000, http://www.brcommunity.com
3. F. Curbera, Y. Goland, J. Klein, F. Leymann, D. Roller, S. Thatte, S. Weerawarana; "Business Process Execution Language for Web Services", July 31, 2002, http://www-106.ibm.com/developerworks/webservices/library/ws-bpel/

4. B. Curtis, M.I. Kellner, J. Over; "Process Modeling", Communications of the ACM, Vol. 35, No. 9, pp. 75-90, 1992
5. C.J. Date; "What Not How: The Business Rule Approach to Application Development", Addison & Wesley Longman Inc, 2000
6. T.H. Davenport, J.E. Short; "The new industrial engineering: Information technology and business process redesign", Sloan Management Review , Vol. 31, No. 4, pp. 11-27, 1990
7. U. Dayal, M. Hsu, R. Ladin; "Business Process Coordination: State of the Art, Trends, and Open Issues", Proceedings of the 27th VLDB Conference, Rome, Italy, 2001
8. B. von Halle; "Business rules applied: Building Better Systems Using the Business Rule Approach", Wiley & Sons, 2002
9. H.J. Harrington; "Business Process Improvement: The Breakthrough Strategy for Total Quality, Productivity, and Competitiveness", McGraw-Hill, New York, NY, USA, 1991
10. M. Koubarakis, D. Plexousakis; "A Formal Framework for Business Process Modeling and Design", Information Systems, Vol. 27, pp. 299-319, 2002
11. T. Morgan; "Business Rules and Information Systems: Aligning IT with Business Goals", Addison & Wesley, 2002
12. A.W. Scheer; "Architecture for Integrated Information Systems, Foundations of Enterprise Modeling", Berlin, Germany, 1992
13. F. Vernadat; "CIMOSA, A European Development for Enterprise Integration Part 2, Enterprise Modelling", Proceedings of the First International Conference on Enterprise Integration Modelling, Austin, TX, USA, 1992
14. R. Veryard; "Rule Based Development", CBDi Journal, July/August 2002
15. F. Leymann; "Web Service Flow Language", May 2001 http://www.ibm.com/software/solutions/ webservices/pdf/WSFL.pdf
16. J. Yang, M.P. Papazoglou; "Service Component for Managing Service Composition Life-Cycle", Information Systems, June, Elsevier, 2003 (forthcoming)
17. J. Yang, M.P. Papazoglou; "Web Components: A Substrate for Web Service Reuse and Composition", Proceedings of the 14th International Conference on Advanced Information Systems Engineering, Toronto, Canada, 2002
18. J.A. Zachman; "A framework for information systems architecture', IBM Systems Journal, Vol. 26, no. 3, pp. 276-292, 1987