CASIA: A Context-Aware Service Identification Aligned to Business

Michel Antunes da Silva, Leonardo Guerreiro Azevedo, Flávia Maria Santoro

Applied Informatics Department (DIA), Post-Graduate Program in Informatics (PPGI), Federal University of the State of Rio de Janeiro (UNIRIO), Rio de Janeiro, Brazil

{michel.silva, azevedo, flavia.santoro}@uniriotec.br

Abstract. *Organizations use services to support the implementation of their business process. Service requirements are identified at design time. However, due to changes in the business processes, a service might no longer comply with those requirements. Hence it is needed to adapt the application that implements the process to consume other services which best fit its needs. This paper describes an approach for automatic and flexible identification of services that are more appropriate to meet the context of a specific process instance. An example of application of the proposal is conducted in order to demonstrate its applicability.*

Resumo. *Organizações usam serviços para apoiar a implementação do seu processo de negócio. Os requisitos para os serviços são identificados em tempo de projeto. No entanto, devido ao dinamismo que mudanças nos processos acarretam, serviços poderão deixar de atender os requisitos em tempo de execução. Dessa forma, é necessário adaptar a aplicação que implementa o processo para consumir outros serviços, que melhor se adaptem às suas necessidades correntes. Este artigo descreve uma abordagem para identificação automática e flexível de serviços mais adequados para atender o contexto de uma instância do processo. Um exemplo de uso da proposta é apresentada a fim de demonstrar a aplicabilidade da mesma.*

1. Introduction

The Service-Oriented Architecture (SOA) is a computing paradigm that uses services as fundamental elements for developing applications. Services are pieces of self-contained features that have exposed interfaces, and are invoked through messages (Marks and Bell, 2006). The development of services is directly linked to business process. Xin (2009) presents that a service is a concept about how business logic is structured to yields business agility. Besides, the use of service increases development speed. Services expose the features that are relevant to the business.

Companies face continuous changes in their environments, where systems are accessible through their own infrastructure, as well as the availability of services from distinct providers. Therefore, they need to be flexible to adapt their systems to changing aspects of the environment without losing compliance with aspects considered to be stable (Najar *et al*., 2009). However, we have not identified proposals of dynamic discovery of services at runtime, when applications that support a particular business

process must be adapt to support a change in requirements. This requires making explicit the semantics of both the service provider and the service consumer.

Truong and Dustdar (2009) indicate that context information may help in service adaptation. Besides, Brézillon and Pomerol (1999) define context as constraints that limits something without explicitly intervention, *i.e*., context is any information that can be used to characterize the situation of an actor, which may be a person, place or object. Thus, it is necessary to improve the context characteristics shared between business process and services. The increase of the ability to understand service and business contexts makes it possible to automate service discovery in equivalent contexts.

This work presents an approach for service discovery aware of context. It includes a framework that considers context information for alignment between business process and service. Besides, we propose the use of the aspect-oriented programming (AOP) paradigm (Kiczales *et al*., 1997) to capture the business context that will support the service discovery in service repositories. The proposed framework goal is to allow the application to use a new service (the better choice) without code maintenance.

The paper is organized as follows. Section 2 discusses related work. Section 3 presents our approach. Section 4 describes our context model. Section 5 describes an application scenario. Finally, Section 6 presents the conclusion and future work.

2. Related Work

Recent works aim at developing mechanisms to combine business perspectives with available software and hardware assets. Table 1 summarizes the works described.

Table 1 – Summary of related work

Source	Brief description	Comments
Keidl and Kemper (2004)	A framework for context aware web services	The focus is mobile computing, and does not consider the business process that is supported by the service
Prezerakos *et al*. (2007)	An extension of the ContextUML metamodel to generate code in an aspectual programming language	It does not define which attributes are required for service identification aligned to business process
Martin (2006)	General guidelines concerning web services contextualization such as: context representation, security etc	It does not address service discovery
Han *et al*. (2008)	Guidelines to structure process context related to business scope and characteristics to discover best fit services	It does not present a contextual representation that guarantees the alignment of business process and supporting services
Yang *et al*. (2008)	A framework to consider context from the service consumer and the web services perspectives	It does not mention any relation with business processes
Zhang *et al*. (2008)	An approach for handling non-functional requirements for web services using concepts from AOP	The focus of this work is on protocol layer in the server side
Singh *et al*. (2005)	An architecture for aspect-oriented web services	Their approach yields a big change in the patterns which can compromise their use

Keidl and Kemper (2004) present a framework for context aware web services. This framework uses the header of web service SOAP (*Simple Object Access Protocol*) message to transmit context information to a service repository. SOAP is a protocol

used for message exchange in a distributed platform based on XML. Keidl and Kemper (2004) concerns are for mobile computing requirements, and the main context attribute is the position of service consumer. This work does not consider the business process that is supported by the service. Prezerakos *et al*. (2007) propose to separate the logic implemented in a service from context features using a model-driven approach. Their proposal is based on an extension of the ContextUML metamodel (Sheng and Benatallah, 2005) and generates code in an aspectual programming language. The logic and context separation is designed in models and then passed to generated code. The use of an aspectual language allows handling the contextual layer in design and execution time. Nevertheless this work presents a solution for code generation, it does not present which attributes are required for service identification aligned to business process.

As a way of eliciting and grouping contextual elements, Martin (2006) proposes to separate the elements concerning web services by categories, such as: organizational arrangements, service types, user characteristics, user state, transaction history, resource state and transaction state. Martin (2006) considers that service discovery and selection may depend on all proposed categories, except the transaction state. Moreover, he proposes general guidelines about architectural issues concerning web services contextualization such as: context representation, security and privacy issues, scalability etc. He presents important categories for context definition, but his proposal does not address service discovery using these categories.

Han *et al*. (2008) propose the use of technical process semantic and business process semantic of a registered service for service discovery. The semantic of technical process is the service scope, while semantic of business process is related to enterprise business domain. They suggest automatic creation of process context by business process as soon as a function requires a web service execution. They present guidelines to structure process context related to business scope and characteristics to discover best fit services. Besides, they consider that a business model is composed by a domain that includes business rules and roles. Process context is composed by a set of criteria and a weight vector. Criteria are defined as properties used to improve service discovery precision and the vector of weights (in a range of 0 to 1) is used to sort the discovered services. The weight vector is defined by web service input and output parameters related to a set of priorities and a set of Quality of Services (QoS) attributes, such as, latency, cost, confidence etc. Besides, Han *et al*. (2008) propose to extend the Universal Description, Discovery and Integration (UDDI) in order to support aware comparison of business context. They use a simple process context model, and therefore improvements are required, such as OWL-S use to increase the semantic. Their proposal is similar to our proposal; however, they do not present a contextual representation that guarantees the alignment of business process and supporting services. For instance, their proposal does not consider the context of the user that started business process execution.

Yang *et al*. (2008) present a framework that considers context from the perspective of service consumer and the perspective of the web services. Their framework is structured on an ontology-based context model. They have considered that consumers are person and do not mention any relation with business processes. Zhang *et al*. (2008) present an approach for handling non-functional requirements for web services using concepts from AOP. They called advices as Aspectual web services to address implementation of non-functional requirements. On this research, the service

request is intercepted by their Weaver layer. The information contained in the SOAP protocol is parsed and validated against the aspectual services specification file (this file defines the AOP behavior). If a concern is identified, the aspectual service is triggered at runtime in accordance with the definitions described in the configuration file. The focus of this work is on protocol layer in the server side. On the other hand, Singh *et al*. (2005) propose an architecture for aspect-oriented web services. The most interesting highlight of this approach is that they propose an adaptation of SOA patterns. In their architecture, the WSDL and UDDI were changed to support concepts inherent from the paradigm of AOP. Their approach yields a big change in the patterns which can compromise their use.

As presented in this section, there are several research works aiming at developing solutions to achieve alignment between business processes and services in service discovery. However, there are not yet conclusive definitions around that alignment. In the next section, we present a proposal to address those problems.

3. CASIA: Context-Aware Service Identification Aligned to Business

In a Service-Oriented Architecture approach implementing services using web services standard, service information is stored on repositories structured following the UDDI OASIS standard[1] (Josuttis, 2007). Conventional UDDI server provides a service discovery through the pair key and value (Han *et al*., 2008). However, UDDI does not provide enough mechanisms to handle complex requests for service discovery. Some examples are: "Find a service that supports the activity *A* and fits current business process scope"; "Find a service that supports the activity *A* and is concerned by rules defined by the official *F*". Therefore, we propose to include context attributes in UDDI to ensure that the service will be available together with its proper context, making it possible to find out the best service for a business process need.

The goal of our proposal is to ensure the discovery of the appropriate service to support a process during its execution. We aim at improving the quality of service identification through including the match of process context and service context. Therefore, the main requirement is to specify the relevant contextual elements for process context and service context definition and the relationship between those elements. Besides, it is necessary an efficient and not intrusive mechanism to intercept a service request during process execution in order to discover the appropriate service.

In our approach, contextual elements are designed and grouped based on the categories proposed by Martin (2006). Besides, the semantics proposed by Han *et al*. (2008) were analyzed according to those categories in order to define inhibitors and to improve them to meet business and service alignment. Han *et al*. (2008) state that OWL-S (Ontology Web Language for Services) is able to express service context in the most effective way. Martin (2006) presents that OWL-S is a language for describing services in a standard way. Its vocabulary can be used combined with other aspects of the OWL (Web Ontology Language) language in order to create service descriptions. As a result, in our proposal, ontology is adopted as a model to represent the context.

[1] http://uddi.xml.org/

We propose a flexible framework to capture the contextual elements inherent to business process and make the comparison against the service context. Considering the non-intrusive form of implementation offered by AOP languages, it was chosen the *AspectJ* AOP language. AspectJ implementation offers the static and dynamic aspectual weaving feature, *i.e.*, the code combination at compile time and execution time (Laddad, 2009). The dynamic weaving enables the aspectual module to deal with the service discovery in a modular way, independent from the implementation that supports process execution. This ensures low coupling and transparency between the business process and the layer that handles its context. It makes possible to add the process context in the service invocation during process execution, i.e., without stopping it.

The logical architectural model is presented in Figure 1. Following the numbers in Figure 1 the architecture is composed by: (1) Business Application Server: It corresponds to the environment where the application that implements enterprise business processes run. This application is the service consumer; (2) The Aspectual Module: This component is responsible for handling the service discovery aware of context. This component uses a context model to make a semantic discovery of services; (3) Business Functionality: It represents the functionality implemented in an application that calls a service during business process execution; (4) Aspectual Weaving: In this module is responsible to combine the service that best match the business requirement considering business requirement and service and process contexts; (5) Context Model: Describe the ontological elements to be used to store business and service contexts; (6) Business Context: Repository of data describing business context according to the Context Model; (7) Service Context: Repository of data describing service context according to the Context Model; (8) Service Repositories: They corresponds to environments available on the Internet storing information about services offered for consumption. Repositories can public (8a) (i.e., they offer public services to be consumed) or private (8b) (i.e., they belongs enterprises offering services to be consumed internally or by other enterprises, but using a specific contract).

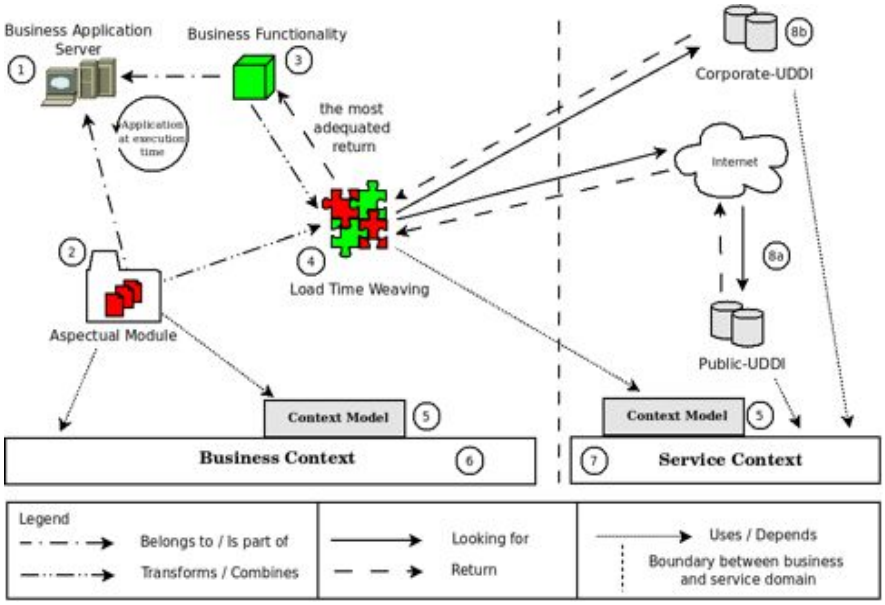

Figure 1. The logical model of the architecture

The operation of this architecture, when available in the enterprise environment (Fig. 1-1), will have its beginning when the implementation of business functionality (Fig. 1-3) runs (the target). When a specific part of the business requests a service, then the aspectual module (Fig. 1-2) intercepts this need to find out the most suitable service to support that business process (Fig. 1-4). In order to obtain this service, the business context information (Fig. 1-6) and the service context information (Fig. 1-7) following the Context Model (Fig. 1-5) is used. It increases the possibilities to meet the alignment between process and service. To select one service to support the process (Fig. 1-8), it is necessary to group and rank the identified services (Fig. 1-4). Then, the most suitable service is returned to the original instance of the process.

4. The Context Model

According to the context definition (Brézillon and Pomerol, 1999) and based on others context models (Han *et al*. 2008; Martin, 2006; Najar *et al*. 2009; Nunes *et al*., 2009; Rosemann *et al*., 2008; Yang *et al*., 2008), we propose a context model to align elements from service and business process.

Nunes *et al*. (2009) propose an ontology model for managing context-based knowledge of activities of process. From this work we add to our context model the elements business rules and glossary of terms. The approach of Najar *et al*. (2009) contributes to our work considering its statement that the relevant context information differs according to a domain and depends on the effective use of them.

Rosemann *et al*. (2008) describe the importance of business processes contextual information, and propose four categories to classify them. The first one is the *immediate context* that includes those elements on the pure control flow and that directly facilitate the execution of a process. The second one is the *internal context* which covers all information on the internal environment of an organization that impacts the process. The third one is the *external context*. It captures those elements outside the organization whose design and behavior is beyond its control sphere. The fourth is the *environmental context*. This last category is inherent in the environment which an organization is embedded on and, somehow, affects business process. So, we use this categorization to better group the context elements and to facilitate the understanding of them.

Yang *et al*. (2008) propose three ways to acquire context: form-filling, context detection and context extraction. They present a context model which express service requester context and web service context. From this work the following elements were considered: service profile (which includes identifier, name, description, precondition, input, output and effect) and QoS attributes (as aforementioned by Han *et al*., 2008).

We organized our context model to concern the alignment between IT services and business processes. On our context model (Figure 2), Entity is a central element. It is specialized in processes, activities, procedures and services. Entities are characterized by context elements (such as Immediate, Internal, External, and Environmental – in a first context level). The contextual properties describe the situation involved with the Entity at the moment of the process execution. In next figures, we present mind maps that expose each context layer and elements of our context model.

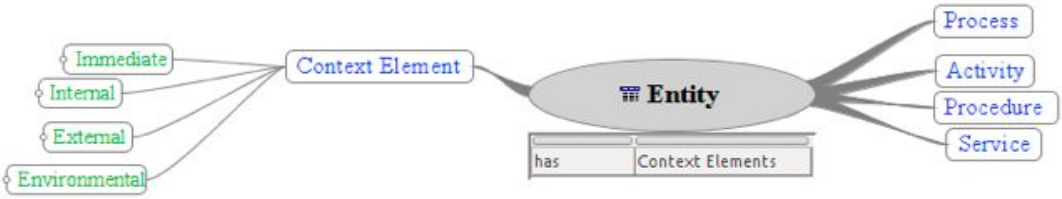

Figure 2. The mind map of the core of our Context Model

Figures 3, 4, 5 and 6 describe the elements based on Rosemann *et al*. (2008) proposal. Figure 3 presents elements inherent at principal group of element from process and service environment. It considers the immediate context. Figure 4 describes internal context that affects directly on each sides, business and service. We consider actors also a resource from process. Figure 5 considers external agents that affect business process, or service, indirectly. Figure 6 determines environmental conditions to help making the adequate decision on the business situation, e.g., when weather is not good and there are some disasters on the delivery route, the system will have to calculate another adequate route to delivery all products on time.

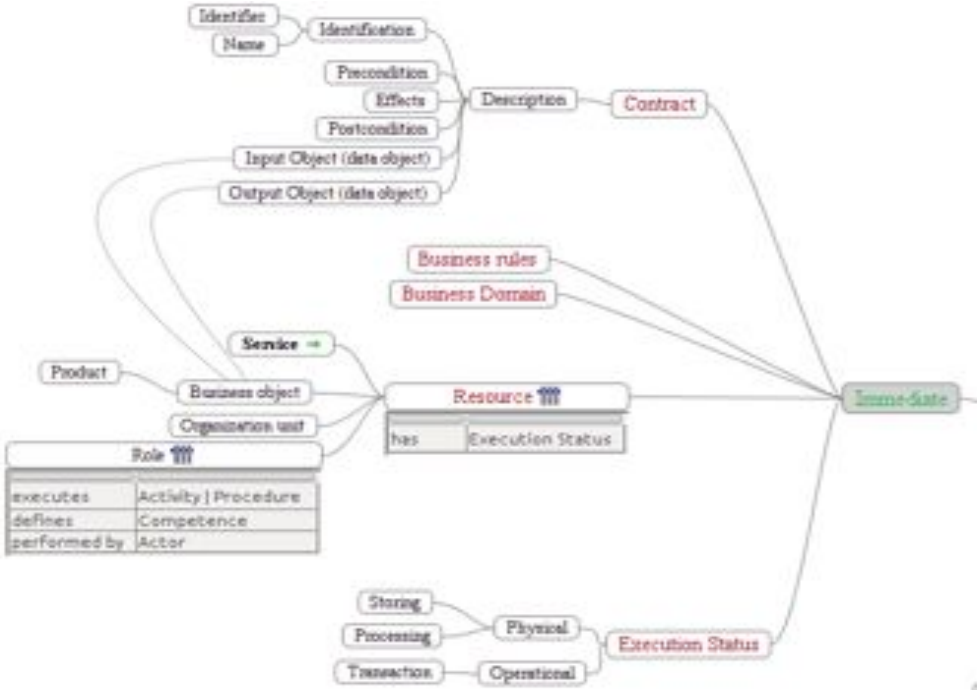

Figure 3. Immediate Context Layer

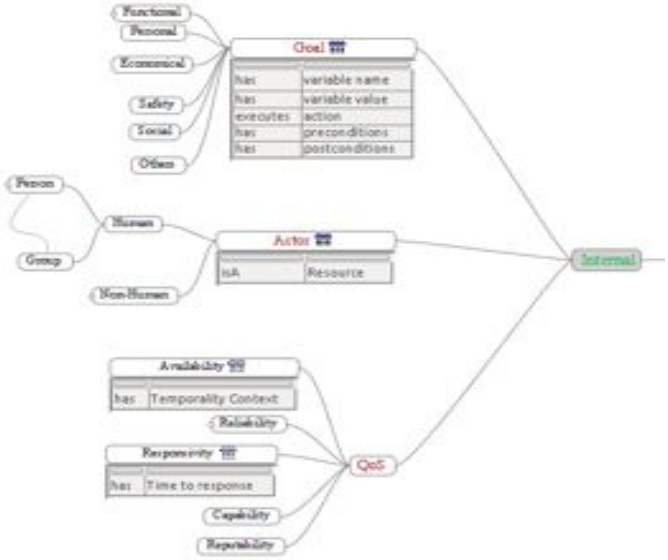

Figure 4. Immediate Context Layer

Figure 5. External Context Layer

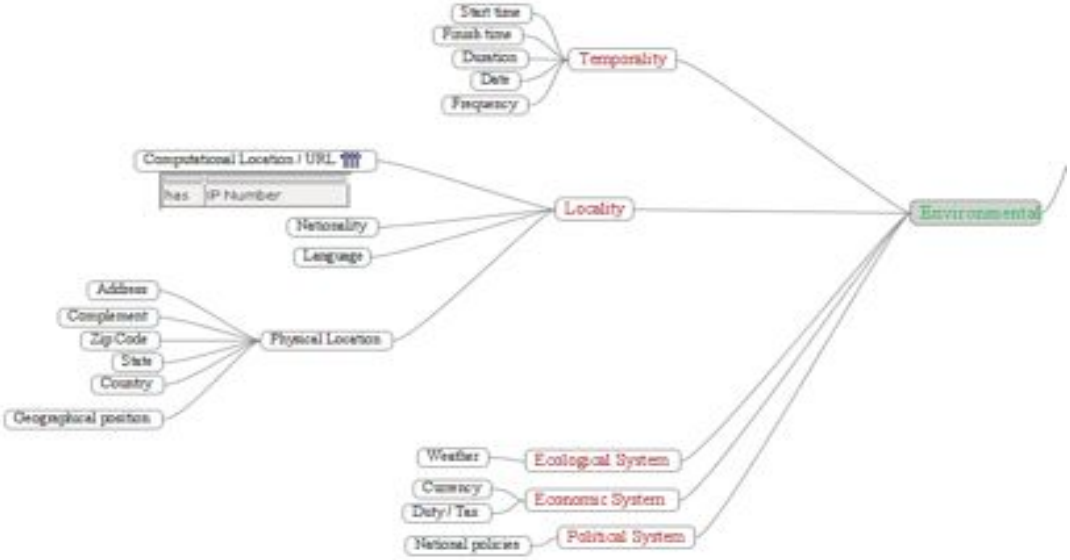

Figure 6. Environmental Context Layer

5. Application Scenario

This section discusses an application of the proposal in a fictitious scenario. The scenario is related to the sales process of an online bookstore that do not have physical stock. It acquires books from other suppliers and sells it to its customers.

The online bookstore has an on-demand logistic process that requires high level of controlling and optimization. Lots of information is required, e.g., customers information and their goals, quality of service, environmental status, weather conditions etc. Some of this information is available only when bookstore application runs, such as, weather and traffic situations, for example, to calculate a delivery route.

In order to obtain business context, user's information and enterprise's information have to be registered into business data repositories. This information helps finding the services that complies with a business process.

The process starts when a customer wants to buy a book. They search for a new book in the website and select it to buy. Afterwards, the system generates the invoice. Then, the customer pays, and the order is delivered to his address.

In this scenario, the logistic processes are performed by services, *i.e.*, there are services supporting the purchase process and the delivery process. These services are provided by other bookstores and carrier companies, respectively. When a customer starts the payment, the system is aware of the business context to discover the adequate services to get the requested book and to deliver it to the customer address. These services are selected according to customers' necessities and enterprise contextual situation. At the end of the process, the customer is informed about an expected delivery date and the invoice is generated. Afterwards, the customer pays the order. Then, the order is delivered to the customer address and the process finishes.

5.1 Application of the Proposed Architecture

The proposed architecture finds the service that supports adequately the business process. Business process runs on a Business Application Server (Fig1-1). The process is composed by functionalities (business functionalities). When business functionality (Fig1-3) tries to invoke a service, the aspectual module (Fig1-2) intercepts this event at runtime (Fig1-4) and finds the most appropriate service to support the business process.

The aspectual module reads the Business Context (Fig1-6) from the business environment and uses it to search on service repositories (Fig1-8) the services that carry out the process functionality. Services are selected based on their context (Fig1-7), structured following the Context Model (Fig1-5).

As an example, consider a library environment, where John, which is a logistic manager, needs to purchase some books for the enterprise XPTO, which he works for. In this scenario, he creates an account on the online bookstore system filling information about him and XPTO. He fills his *personal data* (e.g., name, address, telephone, email, nationality), his *role* when using the online bookstore (in this case Buyer), his *preferences* (e.g., payment preferences, address for delivery, preferred language for the books, media type of products and other information related to process domain). In addition, he fills XPTO information (e.g., *organization unit*, the *local* context dimension, invoice information). Besides, he fills the information needed to meet his requirement for the purchase, such as: price, expected date for the delivery, security concerns etc.

After the registration step, John starts the acquisition process. So, John selects all books and quantities that XPTO needs. Afterwards, he requests an estimate price for the selected books. The bookstore system searches for the suppliers that supply the books. Then, the system analyses the context associated with the book purchase activity. Suppose that John requested two copies of the book "*Java Web Services: Up and Running*" (*Martin Kalin, O'Reilly Media, first edition, 2009, English*) and three copies of "*Java Soa Cookbook*" (*Eben Hewitt, O'Reilly Media, first edition, 2009, English*).

The system search for services that support all requirements related to the scenario context. The context includes, e.g., an object of the bookstores' domains (the book), which is as an *input object* on the services *contract* of distinct suppliers, other context gathered from the contractual document, such as *language*, *supplier information*, *domain constraints*. The bookstore aims to provide books with lowest prices and fast deliver time. Then, based on our model, we need to find partner services from book suppliers and delivery companies that meet these requirements.

However, XPTO aims at meeting the Buyer requirements. The XPTO *goals* are defined according to the Buyer personal context - his *goals* and *preferences*. Thus, John can define, e.g., that it is more important to him the *delivery time* than *book prices*. He can also define that, considering delivery time restriction and price, a book value cannot exceed 10% of the value quoted by the lowest price.

John has a limited time to buy the books considering the other tasks he has to complete during the day. So, he established as a restriction that the response from the online bookstore cannot exceed 60 seconds. Hence, for the online bookstore, it restricts the response time of each service provider: each candidate service should have a total execution time less than 60 seconds.

A similar process is adopted to deliver the order. Thus, after the order payment confirmation, the package is sent to the selected carrier. In this case, the carriers, whose will meet the *expectations* of the customer, will use information about the *traffic conditions* and its *historical data* and the *weather conditions* to estimate the deliver time. The *historical information* includes, e.g., data of *climatic factors* and *natural phenomena* of distinct *locations*. Hence, the aspectual module uses both context models to rank and select the more adequate service to be consumed by the business process. This ranking is performed according to common rules for both contexts. After that, when the best service is selected, the aspectual module invokes it, and returns a response message to the business process instance.

The service ranking is based on QoS (Quality of Service) values, *e.g.*, time to response, compliance with service level agreement, confidence rate of the response message and so on. Even when there is only one candidate service to be ranked, its context must be analyzed to evaluate if it can support the business process meeting the requirements.

As an example of service ranking, the purchase service could be identified by the aspectual module from Amazon, Barnes&Noble and Booksite bookstores. The Booksite's purchase service, e.g., could have not a good “time to response” (time required by the service to process the request and return an answer).The Amazon bookstore, e.g., could not have an adequate “time to delivery” (time to ship the book to the customer). The Barnes&Noble purchase service could fit the requirements, and so, be selected. The second service to be invoked is performed by the delivery activity, and a flow similar to the purchase one is executed.

5.2 Discussion

Our proposal has some benefits. The first one is related to the flexibility to deal with different business domain; since it works at runtime level and no changes is required on

the business process flow, as well in application code to invoke different services. Besides, due to the adoption of an ontological domain definition, rules are modeled in a high semantic level that produces a good match between service and business process needs. However, it has also challenges. First it requires service repositories to be semantically structured, which is not easy to accomplish. Second, service composition is not addressed in this proposal, but it is going to be handled in future work.

Despite of most discussions around context-aware applications, they are focused on mobile environments. Our proposal considers some elements of them. Mobile can be considered one element of business process, but to attend business needs, we argue that it is necessary much broader view, which is conducted in this work.

6. Conclusions

The development of services is directly associated with business process. Service features (such as self-contained, low coupling, high cohesion, interoperability etc.) provide flexibility and agility to business to attend market requirements. Besides, the use of service speeds development time, and decreases maintenance costs (Erl, 2005; Josuttis, 2007). As demonstrated throughout this work, business processes and consumed services must be aware of their contexts.

This work proposed an approach to perform the discovery of context-aware services, using AOP to ensure flexibility in capturing business context. We showed a context model and an application scenario from which it is possible to conclude that the framework is flexible enough to identify (new) services aligned to business without code maintenance providing agility to time to market. In addition, the context improvements in services repositories make suitable for semantic manipulation of its components. We emphasize that a collection of context elements may vary according to the domain where it will be used.

As future work, we suggest performing a case study using benchmarking tools for web services, and compare the proposal to the context model proposed by Han *et al*. (2008).

References

Brézillon, P., Pomerol, J-Ch. (1999) “Contextual knowledge sharing and cooperation in intelligent assistant systems". In: Le Travail Humain, v. 62, n. 3,pp. 223 – 246.

Erl, T. (2005), Service-Oriented Architecture: concepts, technology, and design. Prentice Hall, Englewood Cliffs.

Han, W., Shi, X., Chen, R. (2008) “Process-context aware matchmaking for web service composition”. In: Journal of Network and Computer Applications, v. 31, n. 4, pp. 559-576.

Keidl, M. Kemper, A. (2004) “Towards context-aware adaptable web services”. In: Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters, New York, NY, USA.

Kiczales, G., Lamping, J., Mendhekar, A., Maeda, C., Lopes, C. V., Loingtier, J.-M., Irwin, J. (1997) "Aspect-oriented programming". In: European Conference on Object-Oriented Programming, pp. 220 – 242.

Josuttis, N. M. (2007), SOA in practice: The Art of Distributed System Design. O'Reilly.

Laddad, R. (2009), AspectJ in Action: Enterprise AOP with Spring Applications. Second edition, Greenwich, CT: Manning Publications Co.

Marks, E. A., Bell, M. (2006). Service Oriented Architecture (SOA): a planning and implementation guide for business and technology, Wiley.

Martin, D. (2006) "Putting web services in context". In: *Electronic Notes in Theoretical Computer Science*, v. 146, n. 1, pp. 3 – 16.

Najar, S., Saidani, O., Kirsch-Pinheiro, M., Souveyet, C., Nurcan, S. (2009) "Semantic representation of context models: a framework for analyzing and understanding". In: Proceedings of the 1st Workshop on Context, information and ontologies (Heraklion, Greece, June 01 - 01, 2009). J. M. Gomez-Perez, P. Haase, M. Tilly, and P. Warren, Eds. CIAO '09. ACM, New York, NY, pp. 1 – 10.

Nunes, V. T., Santoro, F. M., Borges, M. R. S. (2009) "A context-based model for Knowledge Management embodied in work processes". In: *Information Sciences*, vol. 179, issue 15, pp. 2538 – 2554.

Prezerakos, G. N., Tselikas, N. D., Cortese, G. (2007) "Model-driven composition of context-aware web services using ContextUML and aspects". In IEEE International Conference on Web Services, ICWS, pp. 320 – 329.

Rosemann, M., Recker, J. C., Flender, C. (2008) "Contextualisation of business processes". In: *International Journal of Business Process Integration and Management*, vol. 3, issue 1, pp. 47 – 60.

Sheng, Q. Z., Benatallah, B. (2005) "ContextUML: a UML-based modeling language for model-driven development of context-aware web services". In: Proceedings of the International Conference on Mobile Business, pp. 206 – 212.

Singh, S., Grundy, J., Hosking, J., Sun, J. (2005) "An Architecture for Developing Aspect-Oriented Web Services". In: European Conference on Web Services, ECOWS, pp. 72 – 82.

Truong, H., Dustdar, S. (2009) "A survey on context-aware web service systems". In: *International Journal of Web Information Systems*, v. 5, n. 1, pp. 5 – 31.

Xin, C. (2009) "Service-oriented architecture in business". In International Colloquium on Computing, Communication, Control, and Management, v. 4, pp. 521 – 524.

Yang, S. J. H., Zhang, J., Chen, I. Y. L. (2008) "A JESS-enabled context elicitation system for providing context-aware web services". In: *Expert Systems with Applications*, v. 34, issue 4, pp. 2254 – 2266.

Zhang, J., Meng, F., Liu, G. (2008) "Research on SOA-based applications based on AOP and web services". In: International Conference on Computer and Electrical Engineering, pp. 753 – 757.