Hypergraph of Services for Business Interconnectivity and Collaboration

Alida Esper, Youakim Badr, Frédérique Biennier
INSA-Lyon, LIESP, F-69621, Villeurbanne, France
{alida.esper, youakim.badr, frederique.biennier}@insa-lyon.fr

Mis en forme : Français France

Abstract. Due to the impacts of structural market evolution (globalization, sustainable growth, mass customization, product-service development…) enterprise are more and more focusing on their core business, developing outsourcing and collaborative strategies to support value-added customized product-service for the customers. This involves developing agile and interoperable information system. To achieve this goal, Service Oriented Architecture has been introduced to support systems interconnection by mean of service composition. Nevertheless, this approach do not integrate service contextual configuration so that different services must be defined according to the context, leading to un-consistent systems. To overcome this limit, we propose a Model Driven Engineering approach to support contextual service refinement. Thanks to an hypergraph organization of the different partial models, services can be contextually instantiated and contextual information can be either inherited from the global model or propagated through the service chain.

Keywords: SOA, Interoperability, dynamic context, service modeling, graph, collaboration.

1 Introduction

The need for increased customization and service-oriented products has forced firms to adapt their organizational strategy. While focusing on their core business competencies, outsourcing and collaborative strategies are developed making an heavy use of ICT. Unfortunately, enterprise information systems consist in several support systems devoted to different business areas (ERP for the management part, CRM for customer management, MES at a workshop level…), exhibiting poor interconnection and agility abilities.

To overcome these limits, the Service-Oriented Architectural style (SOA) [1] has bee introduced. Thanks to standardized component interface definition and publication, processes can be built by service selection and composition mean [2] to provide a basic technologically interoperable IT support.

Despite these intrinsic openness, SOA infrastructures are mostly designed to support intra-enterprise processes as they use only mono-contextual business processes without taking into account actor preferences, underlying resources, service delivery channels or business agreements.

As far as collaborative processes are concerned, a multi-contextual service environment is required, paying attention on information mediation, access rights management, business rules adaptation and user preferences. For example, different actors (final client, transportation firms, hotels or travel agencies) may use the same flight booking service but each of these actors will execute it in a given context, requiring different information, billing policies...

Our solution is based on a model-driven architecture: services are associated to contextualized models organized in an hypergraph so that model selection and service instantiation is achieved dynamically depending on the context: services properties and contextual parameters are either propagated among the service chain or inherited from higher levels models, taking advantage of the object-oriented paradigms [3].

After stating the context and current works, (section 2), we propose our solution globally before describing more precisely the propagation and inheritance mechanisms.

2 Business Collaboration

The growth of the internet appears as a driving force for enterprises to develop direct collaborations with their "professional" partners and customers. Several companies have already moved their operations onto the Web to collaborate with each other, where collaboration between enterprises means the interconnection and coordination of their business processes.

Corporate processes interconnection has been studied for several years. The old-fashion EDI standards ([4], [5], [6]) have been worthy introduced to support inter-organizational application-to-application transfer of business documents (e.g., purchase orders, invoices, shipping notices). As this approach is associated to interchange contracts, it is well suited for formal and administrative exchanges but it involves complex support systems and lacks of agility. More recently, Web Services have been introduced to support technological interoperability and seem to be the most popular implementation of the service oriented architecture. Web services are defined as a "business function made available via the Internet by a service provider, and accessible by clients that could be human users or software applications" [7] and are associated to a set of standards so that technological interoperability requirements can be fulfilled: WSDL (Web Services Description Language) [8] is an XML-based language for describing operational features of Web services, UDDI (Universal Description, Discovery, and Integration) [9] is used to support service publishing and discovery features, SOAP (Simple Object Access Protocol) [10] also messaging abilities between services….

Organising collaboration process involves taking into account the way tasks and activities are organised and coordinated as well as defining the actors involvement (role played…). Workflow process models provided by the Workflow Management Coalition (WfMC) and the Workflow Management Systems (WfMSs) provide convenient frameworks. Based on a predefined activities organisation and on a centralised organisation, they lack of agility. As far as distributed systems are concerned, another strategy consists in focusing on messaging flows. Both of these

approaches have bee taken in the web-service environment. On one hand, WSFL [11][12] is based on a flow model, specifying data exchanges as the execution sequence between component services. As WSFL exposes a WSDL interface, recursive composition is allowed. On the other hand, XLANG[13][12] supports a behavioural description of Web services. It also provides features to combine those services to build multi-party business processes and to support message exchange among services. Lastly, BPEL4WS [14][12] combines both WSFL and XLANG features for defining business processes, consisting in different activities. BPEL4WS defines a collection of primitive activities (such as invoking a Web service operation.) that can be combined into more complex primitives. It includes the ability to: (1) define an ordered sequence of activities (sequence); (2) have branching using the now common "case-statement" approach (switch); (3) define a loop (while); (4) execute one of several alternative paths (pick); and (5) indicate that a collection of steps should be executed in parallel (flow).

Nevertheless, these works do not provide multi-contextual execution support. Moreover, they lack taking into account environmental requirements (as security or other non functional requirements for example). To overcome these limits, service description, selection, composition and orchestration must be enriched to take into account environmental and contextual descriptions.

3 Contextual Collaborative Process organisation

To support collaborative process enactment, we propose to enrich the traditional service architecture to manage multi-context service execution. Our solution is based on the Model Driven Engineering approach to generate dynamically contextual services. Each process is defined by a set of views, related to enterprise policies such as security issues, management strategy and mediation constraints. These models are gathered in the Enterprise Meta-Model Architecture (EMMA), providing classes used to generate the convenient contextual service, linking the core-process services to technological services supporting security or mediation functions (Figure 1).

Mis en forme : Police :10 pt, Anglais Royaume-Uni

Mis en forme : Police :10 pt, Anglais Royaume-Uni, Vérifier l'orthographe et la grammaire

Supprimé : Figure 1

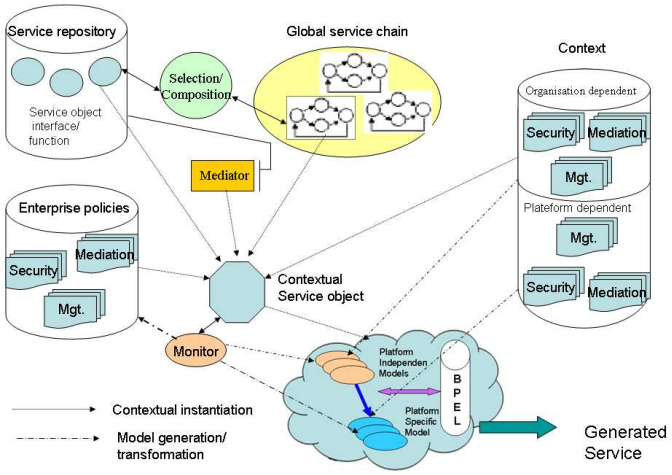


Figure 1: Principle of the Dynamic Service Generation

The Enterprise Meta-Model architecture we propose gathers different kinds of models:

Conceptual service models: these models are associated to generic conceptual activities (ordering, billing,…). They are used to set generic classes description, focusing on common functional properties (namely data and operations) and can be defined recursively as a combination of other generic conceptual models.

E-services models are instances from the previous models. They can inherit functional properties from the class they belong to so that the interface benefits from a global consistency.

Preferences oriented models are used to store in a similar way actors preferences and contextual policies. Generic models are used to define classes so that models that will be applied during the generation process will be instantiated according to the context.

Each enterprise publishes its conceptual and real models in its own service repository. Then, the services that can be used in inter-enterprise collaborative processes are also published in a common repository as well as pre-defined collaborative processes (Figure 2)

Supprimé : . Figure 2

This approach allows to organise a service-chain according to the following steps:

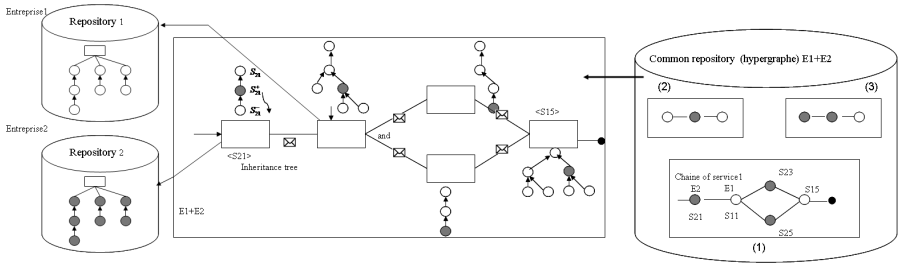
Conceptual services are selected depending on the activities involved in a generic workflow

Actors preferences and contextual information is used to identify both e-services and contextual non-functional models

The convenient models are extracted from the repositories and are used to "instantiate" the global service chain. This is achieved thanks to a service mediator in charge of selecting and generating the convenient service depending on the context. For example, while buying a train ticket, different billing services can be instantiated ("internal billing service" for a ticket bought at the station, on-line e-card billing for Internet based transactions or phone-card based billing…). This leads to a hierarchical

organisation of the billing activities in a tree where the conceptual “billing service” is a root and the different billing e-services are the instantiation.

–



–

Figure 2: presentation repertory common

In order to integrate the different models involved in the service generation in a common repository, we use an hypergraph structure:

The hierarchical service model organisation is used to support the model / service instantiation mechanism by applying specialisation rules. Inheritance relationships are used to support consistent interface definition

Different “horizontal relationships are introduced”:

- Equivalence relationships are used to link models from different enterprises offering the same “conceptual service”. By this way, context-dependant partnership selection can be improved by developing “service substitution” mechanisms
- Context relationships are used to combine different kinds of models (for example security policies coupled with conceptual services) so that context application can be simplified
- Service-chain relationships are used to store well-identified service chain so that already defined service chains can be reused more efficiently.

Due to this hypergraph organisation, classical inheritance mechanisms can not be implemented directly. Consequently, we’ll detail in the next section the inheritance mechanism.

4 Service Refinement and Constraint Propagation

The inheritance relationship favors reusing abilities between class and subclass, allowing the transmission of properties (attributes and methods) from a super class to its subclasses. Subclasses may re-define an attribute or change a method by “overloading“.

As far as functional properties are concerned, the inheritance mechanism allows the transfer of properties (attributes and methods) of the object which are in the super class to the objects that are subclasses. In our case when the service S_2 (object o_2) inherits of service S_1 (object o_1) we can keep the parameters or we can add another parameter, for example if we have both service (consult account) and (consult account

in another country) the inheritance between S_2 and S_1 impose to take into account a new parameter (country) (Figure 3).

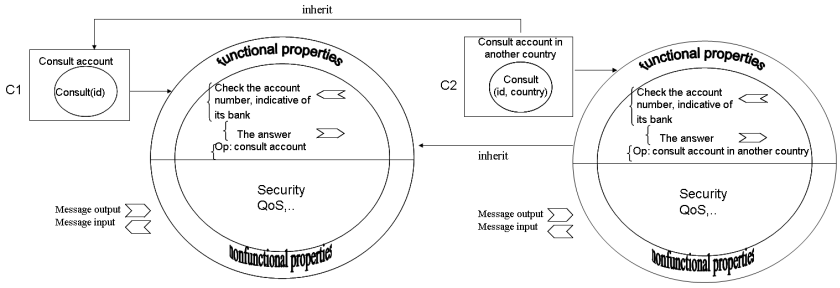


Figure 3: Contextual inheritance

Taking into account non-functional properties can provide additional information on the service. These attributes include security, reliability, messaging facilities, response time, availability, accessibility… [15] defines the Quality of Service as "quality is expressed referring to observable parameters, relating to non-functional property", ,including runtime quality and business quality [16]. By developing a late-binding process, quality of service parameters (including both business oriented parameters (price, delay, performance level) and technical parameters (execution delay, security requirements, resources required…) can be worthy used to select (and then instantiate) the best service to fit the contextual user's needs.

To interconnect the different services in a consistent service chain, we define the following inheritance algebra:

Each conceptual model is associated to a tree h_c. The classes (associated to the different e-service models) c_i are gathered in a set c= $\{c_i\}$. Hierarchical links between classes $({}_{ci}a_{cj}=(c_i,c_j)$ from c_i à $c_{j)}$ are gathered in a set a=$\{{}_{ci}a_c\}$. We call (A_{ip},p=1,2, ….n,) the set of the attributes of the class c_i. Each class c_i inherits the attributes from the preceding classes in the hierarchy (Figure 4).

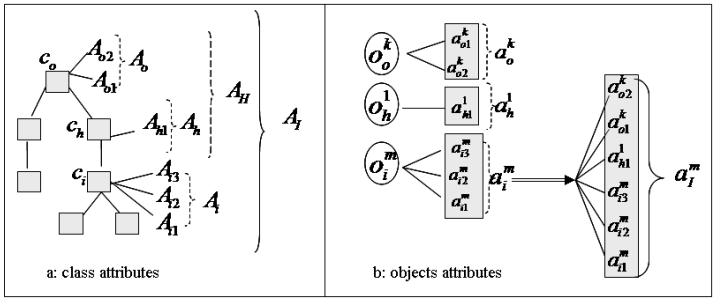


Figure 4: Class and object attributes organisation in the hypergraph structure

We call o_i^m an object of the class c_i, $o_i=\{ o_i^m \}$ is the set of objects of class c_i, $o_i^m \in o_i$. We call a_{ip}^m the attributes of the object o_i^m. Instantiating the object o_i^m involves merging the attributes from the preceding classes (from set a) leading to a_i^m of the attributes of this object o_i^m .

After the instantiation process, the object is linked to le classes it has inherited from so that any change in a class will be achieved "on line" on this object. Consequyently, we establish an arc $a_{ci}^{o_i^m}{}_{cj}^{o_j^n}$ with the condition that the vertice o_j^n has the same set a_i^m comme the o_i^m, $O=\{ o_i^m \}$, $a_{ci}^{o_i^m}{}_{cj}^{o_j^n}=(o_i^m, o_j^n)$, $a=a_{ci}^{o_i^m}{}_{cj}^{o_j^n} \in o^2$, $h_o=<o,a>$,

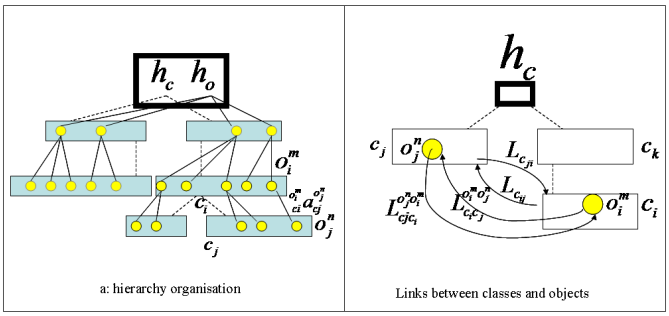


Figure 5: Object integration in the hypergraph organisation

"Horizontal" relationships between nodes from different hierarchies are organised in a "preceding list" so that predecessors can be found automatically and "horizontal inheritance" mechanism can be processes in a similar way (Figure 5).

Supprimé : Figure 5

5 Conclusion and Further works

In this paper, we presented an approach that allows dynamic enactment for inter-firms collaboratice process. Thanks to an hypergraph repository organisation, service composition can be achieved contextually. Inheritance mechanisms are used to provide a consistent support as objects are generated according to the context and inherits the attributes of both conceptual models (seen as super-class) and of preceding objects in the service chain.

Next steps will focus on business transaction orchestration in order to improve late-binding facilities.

References

[1] Thomas, E.: Service-Oriented Architecture: Concepts, Technology, and Design, Prentice Hall, (2005)

[2] Kuppuraju, S., Kumar, A., and Kumari, G.P.: Case Study to Verify the Interoperability of a Service Oriented Architecture Stack, IEEE International Conference on Services Computing, (2007)
[3] Esper, A., Sliman, L., Badr, Y., and Biennier, F.: Towards Secured and Interoperable Business Services, Enterprise Interoperability III, pp. 301-312,
(2008)
[4] X12EDI (Electronic Data Interchange), http://www.x12.org/
[5] Huemer, C., Quirchmayr, G., and Tjoa, A.M. : A Meta Message Approach for Electronic Data Interchange (EDI), Proceedings of the 8th International Conference on Database and Expert Systems Applications, pp. 377-386, Springer-Verlag, (1997)
[6] EDIFACT, http://www.edifact.fr/, last visited (2009)
[7] Casati, F., and Shan, M.: Models and Languages for Describing and Discovering E-services, Proceedings of the ACM SIGMOD International Conference on Management of Data, Santa Barbara: ACM, p. 626,(2001)
[8] W3C, Web Services Description Language (WSDL), (2003), http://www.w3.org/TR/wsdl
[9] W3C,Universal Description, Discovery, and Integration (UDDI), (2003), http://www.uddi.org.
[10] W3C , Simple Object Access Protocol (SOAP), (2003), http://www.w3.org/TR/soap."
[11] IBM ,Web Services Flow Language (WSFL), (2003), http://xml.coverpages.org/wsfl.html.
[12] Peltz, C.: Web Services Orchestration and Choreography, Computer, vol. 36, pp. 46-52, (2003)
[13] Thatte , S.: XLANG : Web Services for Business Process Design, Microsoft. (2001), http://xml.coverpages.org/XLANG-C-200106.html
[14] Business Process Execution Language for Web Services, http://msdn.microsoft.com/en-us/library/aa479358.aspx
[15] Ludwig, H.: Web Services QoS: External SLAs and Internal Policies or How Do We Deliver What We Promise?, Web Information Systems Engineering Workshops, 2003, Proceedings Fourth International Conference on pp. 115-120 (2003)
[16] Yu, Q., Liu, X., Bouguettaya, A., and Medjahed, B.: Deploying and Managing Web Services: Issues, Solutions, and Directions, The VLDB Journal, vol. 17,, pp. 537-572 (2008)