Towards a Disciplined Engineering of Adaptive Secured Service-oriented Business Processes

Khadir Bekki[1], Nasreddine Aoumeur[2], Hafida Belbachir[3]
[1]Département Informatique, Faculté des Sciences et Sciences de l'Ingénieur,
Université de Tiaret, Cite Zaaroura Tiaret, Algeria
Bekki_kh@Yahoo.fr
[2]ITI, FIN, Otto-von-Guericke-Universit¨at Magdeburg
Business Information group, D–39016 Magdeburg, Germany
E-mail: aoumeur@iti.cs.uni-magdeburg.de
[3]Département Informatique, Faculté des Sciences,
Université des sciences et technologie d'Oran (USTO) Oran, Algeria
H_Belbach@Yahoo.fr

Abstract

cross-organizations are increasingly coordinating their capabilities in the quest of dynamically adaptable and secured business process.

Towards a Disciplined Engineering of Adaptive secured Service-oriented Business Processes, we are proposing in this paper a stepwise approach. Firstly, at the domain-level, we are using profiled UML class-diagrams to intuitively capture structural aspects of service-driven applications. Secondly, to cope with security aspects and their agility, we are governing any business security with governing event-driven business rules. Each business activity is governed through interaction-centric Event-Conditions-Actions (ECA) based business rules. They are then smoothly conceptualized as transient ECA-driven architectural connectors, with roles playing service interfaces and glues capturing the service composition logic Our proposition is located in a multi-concern architectural approach that explicitly separate interaction concerns, security concerns from context-aware ones. Each concern is first independently conceived through tailored ECA architectural connectors. These concerns are then accordingly integrated at the fine-grained activity *level.*

Keywords: Dynamic Adaptability, UML, Business rules, Service-oriented, Security.

1 Introduction

Service-oriented architecture (SOA) with its enabling Web Services is currently offering best technological solutions to distributed and loosely-coupled cross-organizational business applications [7].

Web-services are explicit computational units, which can through their interfaces be universally described, published and more importantly (dynamically) composed using XML-based standards (e.g. WSDL, UDDI, BPEL4WS, WS-CDL) [1]. As these standards are maturing, more and more world-wide cross-organizations are opting for service oriented solutions. Consequently, all capabilities and limitations of this new paradigm are being at proof towards developing realistic service-driven applications . Adaptability and correctness, besides knowledge-intensiveness belong to the most challenging issues [8]. Indeed, whereas WSDL and BPEL are inherently static and manual, in face of the harsh competition and market globalization and volatility, realistic services are deemed to be highly adaptive and evolving. Whereas most of potential service-driven applications such as E-commerce and E-health and E-banking are getting mission-critical, BPEL and the other standards are only adhocly built, without any means to formally validate them. Last but least, whereas most of potential service-driven applications are knowledge-intensive (i.e. geared by business rules [6] and policies), in BPEL only basic variables and primitive conditions can be manipulating in static manner.

Towards address these challenging issues, an contribution given in [2] puts forwards an integrated engineering approach

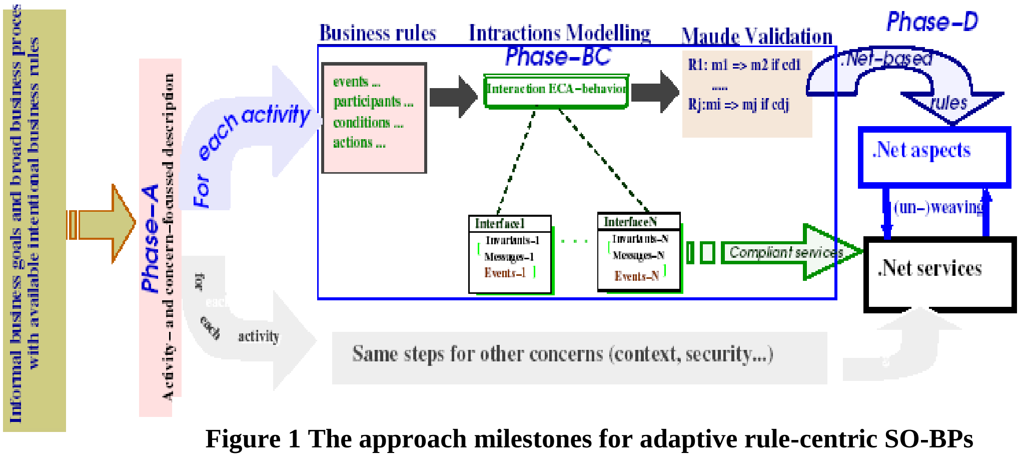

Figure 1 The approach milestones for adaptive rule-centric SO-BPs

as a *multi-concern architectural approach* with the following capabilities and features. Firstly, to intrinsically cope with rule-centricity in an agile and eventdriven manner, any business activity composing a service-oriented business process is governed with (Event-Conditions-Actions) ECA-based business rules [3]. Indeed, business rules are coined as policies, laws and know-how for doing business in any cross-organizations. Besides that they are intuitive and inherently sensitive to internal and external *changes*, and as such they represent main driving force for adaptability and competitiveness. Secondly, towards a disciplined conceptualization while boosting adaptability and bridging the gap to service-orientation, it is proposed to leverage the above intuitive business rules towards transient architectural connectors [SG96]. In such service-driven architectural conceptualization, ECA-driven connector roles will be playing the service interfaces, while the glues will be reflecting the behavioral and dynamic composition of involved service interfaces. Thirdly, towards a rigorous specification and validation of this service-driven architecture, they demonstrate how rewriting logic [5] and MAUDE language [4] can be straightforwardly adopted.

This approach is depicted in Figure 1.

In this paper, we propose a stepwise approach to govern the security concern. Firstly, at the domain-level, we use a profiled UML class-diagrams to intuitively capture structural aspects of service-driven applications. Secondly, to cope with security aspects and their agility, we govern any business security with governing event-driven business rules. Each business activity is governed through interaction-centric Event-Conditions-Actions (ECA) based business rules. The event-driven (business) rules [9] are mostly tailored for engineering multi-concern constraints.

The rest of this paper is organized as follows. In the second section, we detail the semi-formalmodelling of services using profiled UML diagrams. In the third section, we present event-driven business rules and the conceptualization of security aspect rules. In this paper is finally wrapped up by some concluding remarks and further required extensions of this work.

2 UML diagrams for service requirements

In its simplistic form, to stay competitive, banking systems are offering attractive packages for their customers, ranging from simple agreed-on contracts (e.g. different formulas for withdrawing / transferring moneys) over highly sophisticated complex offers (i.e. staged housing loans, mortgages, etc.) depending on their profiles, trust, experiences, etc. For simplicity, we consider a business process composed of (1) customer identification and (2) customer varied withdrawal agreements. A first withdrawal agreement concerns the ”away” case, where the amount should not be greater than a specific amount and where the customer is required to pay a charge. The second case we consider is the withdrawal with a credit, where privileged customers can go below their account balances. For be secured, the bank system require an authentification for their customer and an authorization on balancing their account or on transfer money when the amount is greater than a specific amount.

With the defacto standardization of UML diagrams for structural aspects (i.e. class- and object-diagrams), we argue that UML class-diagrams with slight profiled extensions allow capturing for each service, the operations and properties (attributes) structure.

The next Figure 2 goes in detail about the different classes (as services) and their interactions

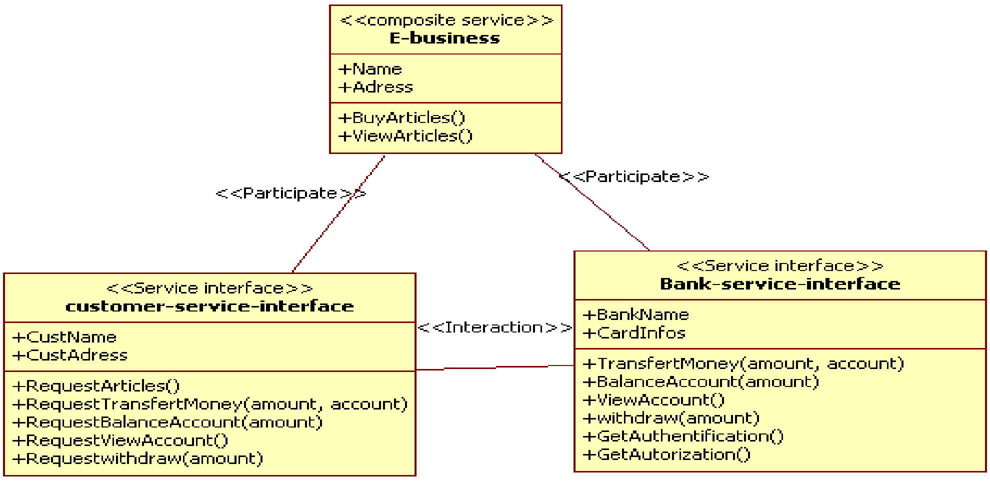

Figure 2 The E business with a SteroTyped UML Class-diagram for Services.

3 On Event-driven Architectural conceptualization

Generally speaking, *Business rules* are defined as: ”*the set of policies for regulating the whole business within and outside an organization*”. They are the *knowledge* on which stakeholders have to be abound by capturing the constraints, conditions and actions for doing business. They clarify, among others, what-to-do (which tasks, activities and / or actions) when internal / external *events* occur. These are the dominating Event-driven conditions actions (ECA) business rules. Since composition is essential in Web-Services business rules governing them, should be regarded instead as *agreements* (between business entities as services).

The ECA has as syntax:

ECA-behavioral interaction *<interaction-Identity>*
interface participants *<list-of-participants>*
invariant *<possible interaction constraints to respect>*
constants/attributes/operations
<extra-required elements for the interaction>
interaction rules: *<Rule-Name>*
at-trigger <(set-of-)events>
under <conditions>
reacting <set-of-actions>

For the interaction aspect, a withdrawal with credits (i.e. Crd-withd.) is given where a specific credit is given to the customer so that (s)he can go below the balance while withdrawing.

ECA-Interaction Crd-Withdraw
participants Acnt: Account;
Cust: Customer
attribute credit : Money
invariants Cust.own(Acnt) = True
interaction rule : VIP
at-trigger Cust.withdraw(M)
under Acnt.bal() + credit ≥ M)
acting Acnt.Debit(M)
end Crd-Withdraw

For security aspect, the bank system require an authentification for their customer and an authorization on withdraw, on balancing their account or on transfer money when the amount is greater than a specific amount (SpecifM for instance)

We govern this security rule for Crd-withdraw on business rule as fellow

ECA-Security Crd-Withdraw
participants Acnt: Account;
Cust: Customer
attribute credit : Money
invariants Cust.own(Acnt) = True
interaction rule : VIP
at-trigger Cust.withdraw(M)
under Accnt.Getauthentification() and M≥ SpecifM
acting Acnt.GetAutorization(M)
end Crd-Withdraw

Crd-Withdraw is governed on two Event Business rules, one for the interaction concern and the second for the security concern.

4 Conclusions and future work

We presented in the paper a stepwise approach for handling exogenous behavioral security in software-intensive component-based applications. This approach proposed an architectural conceptualization using ECA-driven transient connectors. We presented how ECA-security behaviors are to be defined and specified The approach has been thoroughly illustrated using a simplified banking example. For consolidating this promising and practical approach for dynamically and non-intrusively adapting software intensive applications at the architectural level, we will experimenting it with more complex case-studies from the E-commerce world (e.g. an E-shopping application).

For operational formalization, we aim to transform this service architectural modeling for security concern to rewriting logic and its efficient MAUDE language.

References

[1] G. Alonso, F. Casati, H. Kuno, and V.Machiraju. *Web Services*. Springer-Verlag, 2004.

[2] N. Aoumeur, G. Saake, and K. Barkaoui. Incremental Specifi-cation Validation and Runtime Adaptivity of Distributed Component Information systems. In *11th European Conference on Software Maintenance and Reengineering (CSMR'07)*, pages 123–136. IEEE Computer Society, 2007.

[3] M. Bajec and M. Krisper. A Methodology and Tool Support for Managing Business Rules in Organisations. *Information Systems*, 30(6):423–443, 2005.

[4] M. Clavel, F. Duran, S. Eker, P. Lincoln, N. Marti-Oliet, J. Meseguer, and C.L: Talcott. All About Maude - A High-Performance Logical Framework, How to Specify, Program and Verify Systems in Rewriting Logic. *Lecture Notes in Computer Science (springer)*, 4350, 2007.

[5] J. Meseguer. Conditional rewriting logic as a unified model for concurrency. *Theoretical Computer Science*, 96:73–155, 1992.

[6] B. Orrens, J. Yang, and M.P. Papazoglou. A Framework for Business Rule Driven Web Service Composition. In *Proc. of Conceptual Modeling for Novel Application Domains*, volume 2814 of *Lecture Notes in Computer Science*, pages 52–64. Springer, 2003.

[7] M.P. Papazoglou. *Web Service: Principles and Technology*. Prentice-Hall, Englewood Cliffs, 2007.

[8] M.P. Papazoglou, P. Traverso, S. Dustdar, and F. Leymann. Service-Oriented Computing: a Research Roadmap. *Int. J. Cooperativ Inf. Syst.*, 17(2):223–255, 2008.

[9] W.M.N. Wan-Kadir and P. Loucopoulos. Relating Evolving Business Rules to Software Design. *Journal of Systems Architecture*, 2003