Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 5 (2011) 610–617

Procedia Computer Science

The 8th International Conference on Mobile Web Information Systems (MobiWIS)

Towards context-adaptable Web service policies

H. Yahyaoui[a], L. Wang[b] A. Mourad[c], M. Almullah[a], Q.Z. Sheng[b]

[a]*Computer Science Department, Kuwait University, Kuwait, State of Kuwait*
[b]*School of Computer Science, University of Adelaide, Australia*
[c]*Department of Computer Science and Mathematics, Lebanese American University, Beirut, Lebanon*[1]

Abstract

We propose in this paper a new approach to achieve highly adaptable Web services through context-adaptable Web service policies. A Web service policy is a set of rules that defines the capabilities and requirements of a Web service as well as governs the runtime behavior, quality and result of the Web service. Our Web service policies are superbly adaptable in the sense that we specify context for both policies and rules to make the policies context-adaptable at both policy and rule levels. The main contributions of our approach include: i) an innovative WSPL (Web Service Policy Language) extension to allow context specification at both policy and rule levels; and ii) a tool that supports the development of aspect oriented policies, including an option to translate WSPL policies into aspect oriented policies. These policies can then be statically woven into composite Web services (e.g. a BPEL process). The synergy between context, policies and aspects allows to increase the level of adaptability of Web services at different levels of applications.

Keywords: Context, Policies, Web Services, Web Service Policy Language, Aspect-oriented Programming.

1. Introduction

Web services are the key technology behind the booming e-applications such as e-Business, e-Government and e-Science [1, 2, 3, 4, 5]. However, highly adaptable Web services, though most desirable, are rarely available, despite the worldwide research effort and achievements. During last few years, managing Web services through the specification of policies has become one of the active research areas [6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]. A Web service policy is a set of rules that defines the capabilities and requirements of the Web service as well as governs the runtime behavior, quality and result of the Web service. Two major languages were designed to specify theses policies, namely WS-Policy [17, 18] and the Web Services Policy Language (WSPL) [19, 7]. The first language is a W3C standard, while the second is based on the the OASIS XACML standard [20]. Our target language is any XACML-based language and particulary we selected WSPL for its simplicity and low learning curve. This language was used in [7, 21, 22] for specifying several kinds of policies such as business, security, and privacy policies. However, the presented approach can be easily applied to any XACML-based language.

Selection and/or peer-review under responsibility of Prof. Elhadi Shakshuki and Prof. Muhammad Younas.
doi:10.1016/j.procs.2011.07.079

In this paper, we go a step further by presenting a new approach to achieve highly adaptable Web services through adaptable Web service policies that are not only *policy context* adaptable but also *rule context* adaptable. We take an innovation to specify context at both policy and rule levels. In particular, we extend WSPL to specify context for both policies and rules[2], which results in context-adaptable WSPL policies. For instance, a `flightBooking` Web service policy contains a discount rule that offers the Gold members a 50% special discount rate for special occasions and a 20% regular discount rate in other times. This dynamic business rule can be defined elegantly in WSPL extension augmented with the rule context.

We also develop a policy tool that provides a framework for the specification and implementation of aspect oriented policies, including an option to translate WSPL policies into aspect oriented policies since WSPL policies can not be directly applied to a composite Web service (e.g. a BPEL process). In this way, we can achieve highly adaptable composite Web services. It's also worth noting that our highly context-adaptable Web service rules can be easily updated without any impact on the business logic of the BPEL process, which is a critical and useful feature in this fast changing world and adds a level of flexibility in developing composite Web services.

The rest of the paper is organized as follows. Section 2 gives some background information about context, WSPL and AOP as well as introduces our scenario - a `Flight Booking` system. Section 3 describes our new approach focusing on how context can be specified for WSPL rules and applied to composite Web services through AOP weaving. This section also presents the design and implementation of our aspect oriented policy tool. Finally, Section 4 briefly discusses some related work and concludes the paper with some future research directions.

2. Background

This section gives some background information about the concept of context, WSPL and AOP as well as introduces our scenario - a Flight System for the flight agency staff to book flights.

2.1. Context

Dey and Abowd have defined context—which is widely used in the literature today—as "*any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves*" [23].

In this paper, we focus on Web service rule context which refers to any information that affects a rule's guidance to the interaction between a Web service and its requester. For example, the "special" and "regular" occasions mentioned in the introduction are the context of the discount rule because the occasions affect the discount rate that a Gold member receives.

2.2. Web service policy language

WSPL is an XML language based on the the OASIS XACML standard [20], which is also known as XACML profile for Web services. WSPL is known to be simple and to have a low learning curve for Web service policies specification. WSPL has three top-level elements: `PolicySet`, `Policy`, and `Rule`. `PolicySet` is the container for policies and each policy is a sequence of one or more rules. Rules are listed in order of preference, with the most preferred choice listed first [7]. Using the *Apply* element, each WSPL rule defines a constraint that the service needs to abide. Predefined constraint operators include: *equals*, *greater than*, *greater than or equal to*, *less than*, *less than or equal to*, *set-equals*, and *subset*. However, current WSPL does not capture context at rule level, which seriously limits the adaptability of WSPL policies.

2.3. Aspect-oriented programming

The main objective of AOP is to have a clean separation between cross-cutting concerns which are the program parts that are not related to the program's primary function but tangled with (e.g. depended on) other program functions. In other words, AOP aims to isolate supporting functions from the main program's business logic. This is

[2]Current WSPL versions do not capture context at rule level.

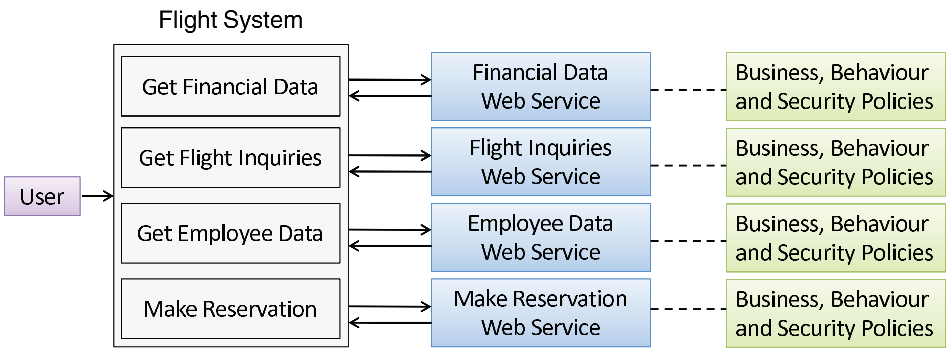

Figure 1: Flight system components

achieved through the definition of aspects. Each aspect is a separate module in which pointcuts are defined. A pointcut identifies one or more join points. A join point identifies one or more flow points (e.g. method calls) in a program (in our case a program is a BPEL process). At these join points, advices will be executed. An advice contains some code that can alter the process behavior before, after or around a certain flow point, hence the advice can be called *before advice*, *after advice* or *around advice* respectively. The integration of aspects within an application code is called weaving and is performed through one of the weaving technologies (e.g., AspectJ [24]).

2.4. A scenario: flight booking system

To better illustrate our approach, we will use a `Flight Booking` system as a running example. The `Flight Booking` system is composed of four distinctive Web services, a BPEL process and a graphical user interface that allows users to invoke the four Web services as depicted in Figure 1.

The `Financial Data` service allows a staff member to request the revenues and expenses of the flight agency for a given month. The `Flight Inquiry` service returns a list of the available flights including their names, time tables (departure and arrival time and date), available seats and ticket prices. The `Employee Information` service allows the user to view staff information including the employee's full name, phone number, email address, post and his office number. Finally, the `Make Reservation` service enables the user to reserve a seat on a certain flight.

Each staff member has an ID and a password stored in the database, in addition to other personal information. Each time a user wishes to access any of the four flight system services, both the `Authentication` and `Access Control` services may be invoked depending on the context in which the Web service is running.

3. Our approach

This section describes our new approach in details, focusing mainly on specifying context for rules in WSPL extension and applying context-adaptable WSPL policies to composite Web services through AOP weaving.

3.1. An overview

As shown in Figure 2, our approach involves four entities and three operation methods. The four entities are *Rule Context*, *WSPL Policy*, *AOP Aspect* and *BPEL Process*. The three operation methods are *Context Specification Method*, *Policy Translation Method* and *Policy Integration Method*. *Context Specification Method* includes operations for specifying Web service policy context such as the WSPL Extension that is used to specify rule context for WSPL policies in our case. *Policy Translation Method* describes operations for translating an exiting policy to a required format such as the Translation Tool that is used to translate a WSPL policy to an AOP aspect in our scenario. *Policy Integration Method* represents operations for applying context-based policies to Web services such as AOP Weaving that weaves an AOP aspect to a BPEL process in our example.

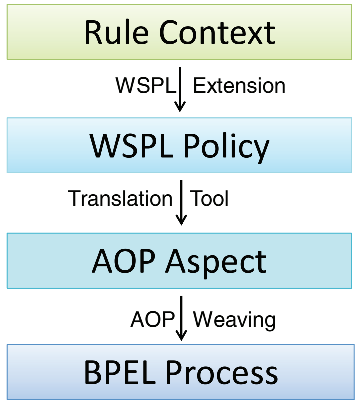

Figure 2: Approach Architecture

```
Policy(Aspect = "Business")
{
Rule(Context = "Special") {
Member-level = "Gold"
Rate = 50%}

Rule(Context = "Regular") {
Member-level = "Gold"
Rate = 20%}
}
```

```
Policy(Aspect = "Behavior")
{
Rule(Context = "Off peak") {
Reliability >= 90%
Throughput = 50 invocs/sec}

Rule(Context = "peak") {
Reliability < 60%
Throughput = 100 invocs/sec}
}
```

```
Policy(Aspect = "Security")
{
Rule(Context = "Authentication Required"){
Authentication-Algorithm = "Login/Password"
Trials = 3}

Rule(Context = "Authentication Not Required"){
Authentication-Algorithm = "None"
Trials = Null}
}
```

Figure 3: Specify Contexts for Business, Behavior and Security Rules

3.2. *Specify context for WSPL rules*

In our scenario, we define the following three Web service policies: i) the *Business* policy identifying the rules that govern the operation of a Web service (e.g. the discount rate for certain occasion), ii) the *Behavior* policy that defines the quality attributes of a Web service (e.g. the response time for service request), and iii) the *Security* policy that describes the rules of security measurement and algorithms (e.g., the authentication algorithm for certain interaction). The detailed policies, rules and their respective context are as specified in the pseudo notations in Figure 3.

The Business policy (left) specifies a discount rule that offers the Gold members a 50% *special* discount rate in special occasion but a 20% *regular* rate in other times. In other words, the Gold members get either a *special* or a *regular* discount rate depending on the occasions when they use the `Flight Booking` service.

The Behavior policy (middle) describes a behavioral policy that also depends on contexts: Peak and Off Peak time. In an Off Peak time, the reliability of the Web service is more than 90% when the service request is 50 invocations per second. On the other hand, in a Peak time, the reliability of the Web service is less than 60% when the service request is 100 invocations per second.

The Security policy (right) provides a security policy that again depends on contexts: Authentication Required and Not Required. If authentication is required, the user needs to provide a correct password in no more than 3 trials in order to access the flight service. If authentication is not required, the user can invoke a service without being asked for a password.

3.3. *Apply context-adaptable WSPL policies to composite Web services*

A policy would be meaningless if it could not be applied to its targeting Web service. Unfortunately there isn't a straight way to apply WSPL policies directly to a composite Web service. Our approach gets round this problem from two directions: 1. specifying the policies in the way that can be directly applied to composite Web services. For example, AOP aspects can be directly weaved into a BPEL process through AOP technologies; and 2. translating WSPL policies into AOP aspects.

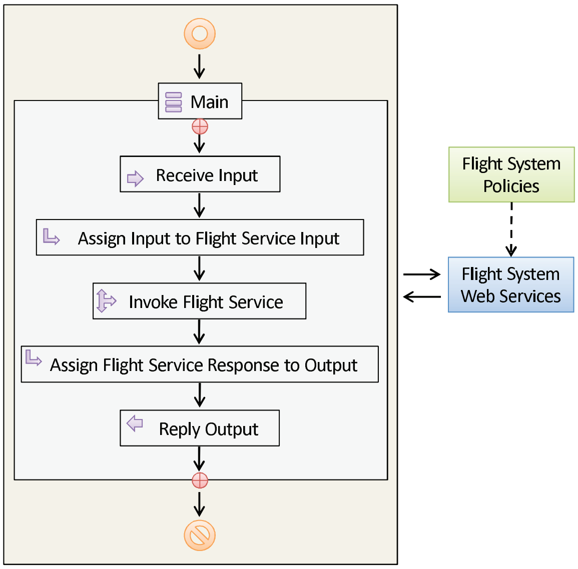

Figure 4: Flight system BPEL process

3.3.1. Specify AOP aspects

We leverage the ideas proposed by Charfi et. al in [8] about separating concerns (e.g. business, behavior, and security concerns in our case) and transform each context-based WSPL policy into a corresponding aspect. The separation of concerns allows better accommodation of changes in Web service policies. In fact, any change in a policy can be smoothly realized through the update of the related context element in the aspect file without any effect to the business logic of the Web service. There is no need to update the business logic of a Web service. This capability adds a level of flexibility in developing composite Web services.

Figure 4 outlines the Flight System BPEL process. This process can invoke any (atomic) Web service of the Flight System. All the business, behavior and security policies are related to that invoked Web service.

Figure 5 outlines the weaving of business, behavioral and security aspects into the the BPEL process of the Flight System. This process can invoke any of the four Flight System Web services. All three aspects are related to the invoked Web service. The idea is to translate all three policies into aspects first and then to perform a merging of the three aspects. Due to the limitation of the page numbers, the weaving codes of the three aspects are not presented.

3.3.2. Aspect oriented policy specification tool

In our research, we have successfully implemented an aspect oriented policy specification tool, which provides a framework for developing aspect oriented policies for Web services, including a menu option to load and compile a context-based WSPL policy as depicted in Figure 6. The tool verifies the syntactic correctness of a WSPL policy before translating it into an aspect as shown in Figure 7. The generated aspect can be automatically weaved into a BPEL file of a composite Web service by simply clicking the `Weave Aspect` menu option (see Figure 7).

Besides the weaving capability, there are other functionalities provided by this tool, such as detection of possible errors in policies or aspect codes. Under development is another capability of merging two context-based WSPL policies of atomic Web services into a common policy which is then translated into an aspect and weaved automatically into the BPEL code.

4. Discussion and conclusion

We propose in this paper a new approach to achieve highly adaptable Web services through superbly adaptable Web service policies which are context-adaptable at both policy and rule levels. In particular, we extend WSPL to allow context specification of policy rules. Furthermore, we establish a synergy between context and aspect in order

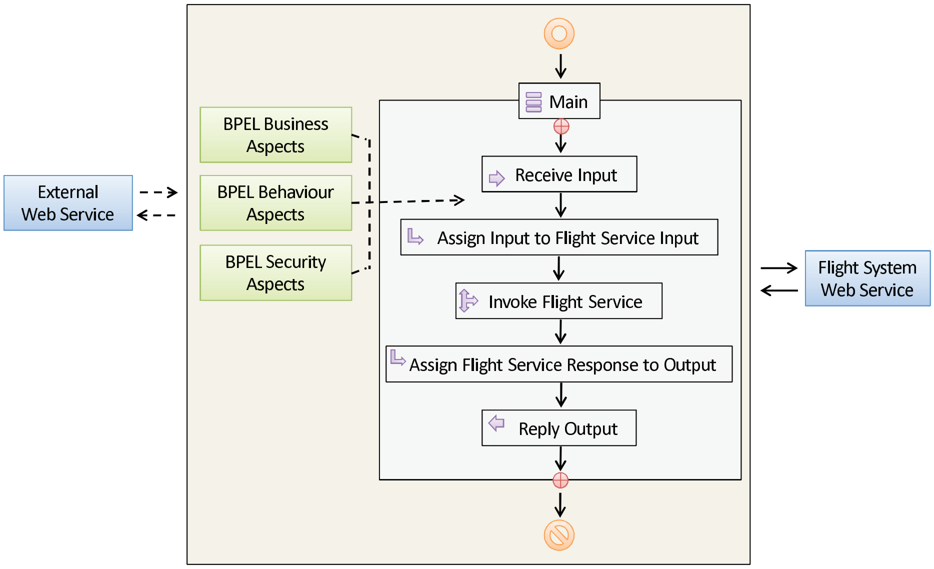

Figure 5: Weaving the three context-based aspects with the flight system BPEL process

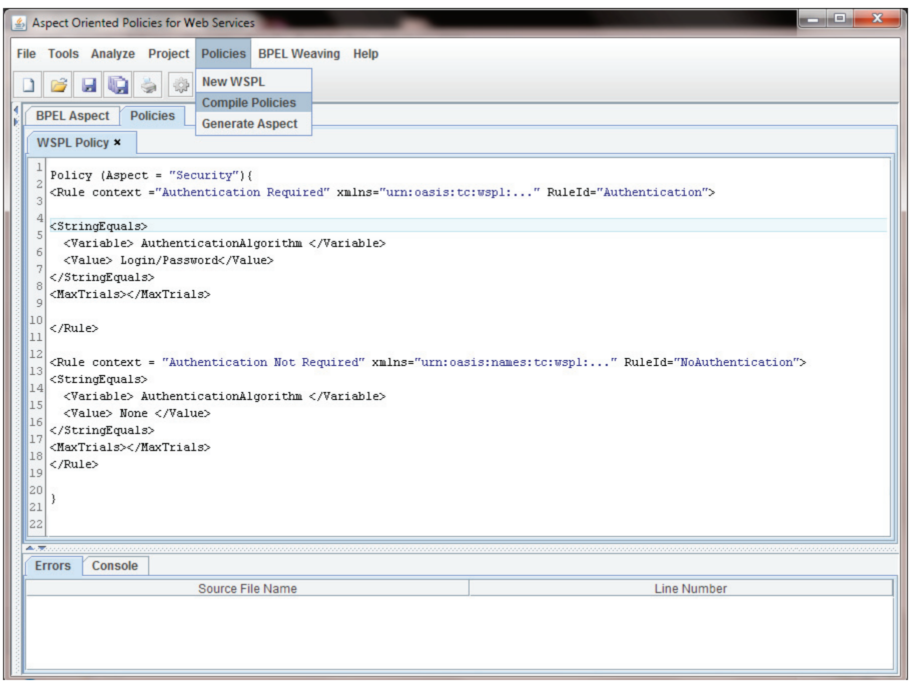

Figure 6: Context-based WSPL Security Policy

to apply context-adaptable WSPL policies to composite Web services. The proposed approach is embedded in an *Aspect Oriented Policy Specification Tool*, which allows to compile context-aware policies and weave them statically and automatically into the BPEL code.

The work presented in this paper is related to developing adaptable Web services through context-based Web service policies. Regarding Web service policies, most research focuses on security policies, such as Assertion Markup

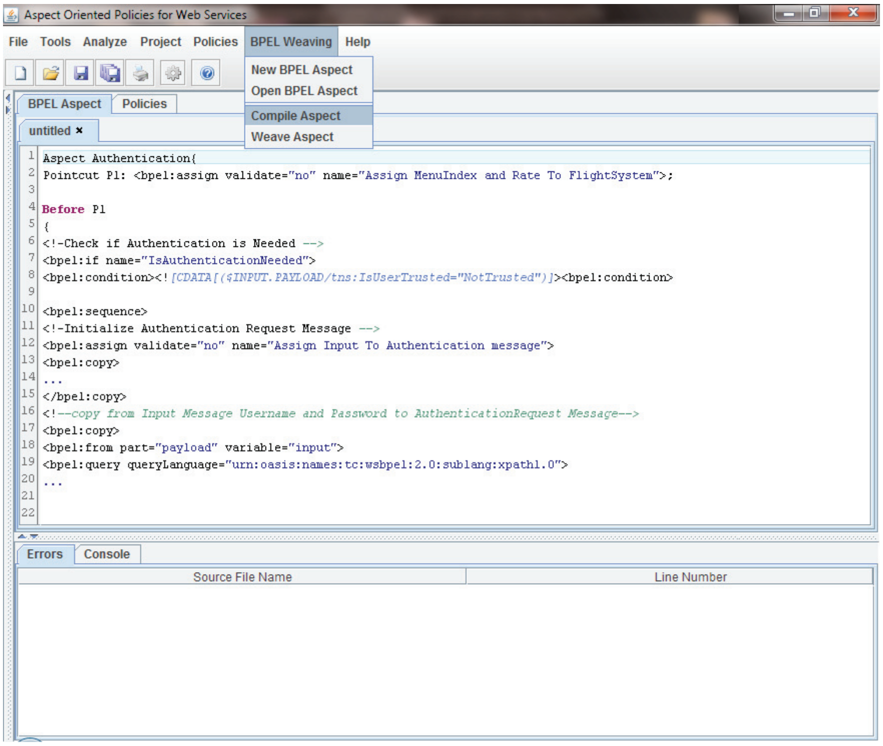

Figure 7: Authentication Aspect

language (SAML) [25], WS-Security [26], The Web Service eXtensible Access Control Markup Language (WS-XACML) [20], X-RBAC [27] etc. The main issue with these policies is their inflexibility, i.e., they do not take into account any change in the environment parameters. The deployed Web services have to be stopped and updated to take in any change, which is a cumbersome, tedious, error-prone and inefficient approach. There are some research initiatives in introducing the context concept in different levels and stages of Web services engineering [28, 29, 30, 31, 32, 10, 33, 11, 34]. However, the majority of these initiatives focused particularly on the definition and modeling of context information. Only a few, such as the OASIS initiative for specifying *WSContext* [35], did investigate the issue of devising possible languages for expressing context information. Yet the main objective of *WSContext* is to relate and manage interactions between Web services rather than specifying rules that govern the business logic of a Web service itself. In our approach, we focus on the specification and implementation of adaptable Web service policies based on the context at both policy and rule levels. Upon applying the superbly adaptable policies to Web services we achieve highly adaptable Web services which are most desirable but rarely available.

Several future research directions arise. For instance, the context can be used as a factor in enforcing the security of Web services and enhancing its performance. Another interesting issue is the use of context in matching user preferences and Web service capabilities. A context-based discovery protocol would allow a better matching and a fine-grained ranking of Web services.

References

[1] T. Andrews, F. Curbera, H. Dholakia, Y. Goland, J. Klein, F. L. et al, Business Process Execution Language for Web Services, Version 1.1, Standard proposed by BEA Systems, IBM Corporation, and Microsoft Corporation (2003).

[2] B. Di Martino, Semantic web services discovery based on structural ontology matching, International Journal of Web and Grid Services 5 (1) (2009) 46–65.

[3] M. Papazoglou, Web services and business transactions, World Wide Web 6 (1) (2003) 49–91.

[4] B. A. Schmit, S. Dustdar, Model-driven development of web service transactions, Enterprise Modelling and Information Systems Architectures 1 (1) (2005) 46–55.

[5] W. Den Heuvel, K. Leune, M. Papazoglou, EFSOC: A layered framework for developing secure interactions between web-services, Distributed and Parallel Databases 18 (2) (2005) 115–145.
[6] S. Agarwal, B. Sprick, Specification of Access Control and Certification Policies for Semantic Web Services, in: Proceedings of the 6th International Conference on Electronic Commerce and Web Technologies (EC-Web'2005), Copenhagen, Denmark, 2005.
[7] A. Anderson, Predicates for Boolean Web Service Policy Language (WSPL), in: Proceedings of The International Workshop on Policy Management for The Web (PM4W2005) held in conjunction with The Fourthen International World Wide Web Conference (WWW2005), Chiba, Japan, 2005, pp. 53–55.
[8] A. Charfi, M. Mezini, Using Aspects for Security Engineering of Web Service Compositions, in: Proceedings of the Internationa Conference on Web Services (ICWS'05), Orlando, USA, 2005, pp. 59–66.
[9] F. Clemente, G. Pérez, J. Blaya, A. Skarmeta, Representing Security Policies in Web Information Systems, in: Proceedings of the Fourthen International World Wide Web Conference (WWW'2005), Chiba, Japan, 2005.
[10] Z. Maamar, N. Narendra, D. Benslimane, S. Sattanathan, Policies for Context-driven Transactional Web Services, in: Proceedings of the 19th International Conference on Advanced Information Systems (CAiSE'2007), Trondheim, Norway, 2007.
[11] B. Medjahed, Y. Atif, Context-based matching for Web service composition, Distributed and Parallel Databases 21 (1) (2007) 5–37.
[12] H. Wang, S. Jha, M. Livny, P. McDaniel, Security Policy Reconciliation in Distributed Computing Environments, in: Proceedings of the 5th IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY'2004) in conjunction with The 9th ACM Symposium on Access Control Models and Technologies (SACMAT'2004), Yorktown Heights, NY, USA, 2004.
[13] Z. Maamar, D. Benslimane, P. Thiran, C. Ghedira, S. Dustdar, S. Sattanathan, Towards a context-based multi-type policy approach for Web services composition, Data & Knowledge Engineering 62 (2) (2007) 327–351.
[14] Q. Z. Sheng, B. Benatallah, ContextUML: a UML-based modeling language for model-driven development of context-aware web services, in: Mobile Business, 2005. ICMB 2005. International Conference on, IEEE, 2005, pp. 206–212.
[15] M. Keidl, A. Kemper, Towards context-aware adaptable web services, in: Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters, ACM, 2004, pp. 55–65.
[16] J. Yu, Q. Z. Sheng, J. Swee, Model-Driven Development of Adaptive Service-Based Systems with Aspects and Rules, Web Information Systems Engineering–WISE 2010 (2010) 548–563.
[17] P. Nolan, Understand WS-Policy processing, Tech. rep., IBM Corporation (2004).
[18] J. Schlimmer, Web Services Policy Framework (WS-Policy), http://www-128.ibm.com/developerworks/webservices/library/specification/ws-polfram/ (2004).
[19] A. Anderson, An Introduction to the Web Services Policy Language (WSPL), in: POLICY '04: Proceedings of the Fifth IEEE International Workshop on Policies for Distributed Systems and Networks, IEEE Computer Society, Washington, DC, USA, 2004, p. 189.
[20] T. Moses, OASIS eXtensible Access Control Markup Language (XACML), OASIS Standard 2.0, http://www.oasis-open.org/committees/xacml/.
[21] Z. Maamar, Q. Z. Sheng, H. Yahyaoui, D. Benslimane, F. Liu, On Checking the Compatibility of Web Services' Policies, in: Proceedings of the International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT'2007), Adelaide, Australia, 2007, pp. 125–130.
[22] Q. Z. Sheng, J. Yu, Z. Maamar, W. Jiang, X. Li, Compatibility Checking of Heterogeneous Web Service Policies Using VDM++, in: Proceedings of the IEEE Workshop on Software and Services Maintenance and Management (SSMM'2009) held in conjunction the 2009 IEEE Congress on Services, Part I (SERVICES I'2009), 2009.
[23] G. Abowd, A. Dey, P. Brown, N. Davis, M. Smith, P. Steggles, Towards a Better Understanding of Context and Context-Awareness, in: HUC '99: Proceedings of the 1st International Symposium on Handheld and Ubiquitous Computing, Springer-Verlag, Karlsruhe, Germany, 1999, pp. 304–307.
[24] G. Kiczales, E. Hilsdale, J. Hugunin, M. Kersten, J. Palm, W. G. Griswold, An Overview of AspectJ, in: Proceedings of the 15th European Conference on Object-Oriented Programming (ECOOP'01), Springer-Verlag, London, UK, 2001, pp. 327–353.
[25] B. Lockhart, al., OASIS Security Services TC (SAML), http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=securit
[26] B. Atkinson, al., Web services security (WS-Security), http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wss.
[27] R. Bhatti, J. Joshi, E. Bertino, A. Ghafoor, Access Control in Dynamic XML-Based Web-Services with X-RBAC, in: Proceedings of the International Conference on Web Services (ICWS'03), 2003, pp. 243–249.
[28] S. Bassil, S. Rinderle, R. Keller, P. Kropf, M. Reichert, Preserving the Context of Interrupted Business Process, in: Proceedings of the 7th International Conference on Enterprise Information Systems (ICEIS'2005), Miami, USA, 2005.
[29] K. Henricksen, J. Indulska, A Software Engineering Framework for Context-Aware Pervasive Computing, in: Proceedings of the Second IEEE International Conference on Pervasive Computing and Communications (PerCom'2004), Orlando, Florida, USA, 2004.
[30] M. Keidl, A. Kemper, A Framework for Context-Aware Adaptable Web Services, in: Proceedings of the 9th International Conference on Extending Database Technology (EDBT'2004), Heraklion, Crete, 2004.
[31] G. Kouadri Mostéfaoui, P. Brézillon, Modeling Context-Based Security Policies with Contextual Graphs, in: Proceedings of the Workshop on Context Modeling and Reasoning (CoMoRea'2004) held in conjunction with The 2nd IEEE International Conference on Pervasive Computing and Communication (PerCom'2004), Orlando, Florida, USA, 2004.
[32] Z. Maamar, S. K. Mostéfaoui, H. Yahyaoui, Toward an agent-based and context-oriented approach for Web services composition, IEEE transactions on knowledge and data engineering 17 (5) (2005) 686–697.
[33] O. Marjanovic, Managing The Normative Context of Composite E-services, in: Proceedings of the International Conference on Web Services (ICWS-Europe'2003), Erfurt, Germany, 2003.
[34] M. Mrissa, C. Ghedira, D. Benslimane, Z. Maamar, F. Rosenberg, S. Dustdar, A context-based mediation approach to compose semantic web services, ACM Transactions on Internet Technology (TOIT) 8 (1) (2007) 4–es.
[35] M. Little, E. Newcomer, G. Pavlik, Web Services Context Specification, http://docs.oasis-open.org/ws-caf/ws-context/v1.0/wsc