Model-Driven Development of a Web Service-Oriented Architecture and Security Policies

Juan P. Silva Gallino, Miguel A. de Miguel, Javier Fernández Briones, and Alejandro Alonso
DIT-Universidad Politécnica de Madrid (UPM),
Madrid, 28040 Spain
e-mail: {psilva, mmiguel, jfbriones, aalonso}@ dit.upm.es

Abstract— **Applying model-driven development methodologies provide inherent benefits such as increased productivity, greater reuse, and better maintainability, to name a few. Efforts on achieving model-driven development of web services already exist. However, there is currently no complete solution that addresses non-functional aspects of these services as well. This paper presents an ongoing work which seeks to integrate these non-functional aspects in the development of web services, with a clear emphasis on security.**
Keywords- MDA, WS, SOA, WS-SecurityPolicy.

I. INTRODUCTION

Service-Oriented Architectures (SOA), of great popularity nowadays, and Web Services (WS), the technology generally used to implement them, go hand in hand in such a way that they are even referred to as WSOA (Web Service Oriented Architecture). They achieve the integration of heterogeneous technologies, providing interoperability, and yielding to the reutilization of pre-existent systems.

At the same time, Model Driven Applications (MDA) [1] arises as a new paradigm that tries to shift models out of a mere documentational role, and turn them into a first class development artifact. This approach provides, among other benefits, a greater understanding of the system as a whole and a platform-independent development, improving the reusability of the design, and simplifying the evolution of the system, thus increasing productivity.

WS-* standards provide a strong foundation for the development of services. Ws-Policy and related specifications also offer the possibility of considering extra-functional concerns of those services. However, these specifications are XML-based, and its syntax is not thought to be read/written by humans. An abstraction of such languages is desirable.

Some approaches applying MDA to the development of Web Services already exist (e.g. [2], [3], [4]). However, these approaches do not offer thorough support for access control descriptors, code generation, Web Services Description Language (WSDL) [5], and WS-Policy [6]. This paper presents ongoing work towards achieving a solution that includes all these previously mentioned concepts, focusing on a Component-Based Software Development (CBSD).

II. RELATED WORK

Among the proposals that deal with the different aspects supported by WS-Policy, the approach by Ortiz and Hernández [2] is among the most representative ones. This work makes use of model-driven development for including extra-functional aspects in web services parting from a platform independent model. The focus of Ortiz's work is somehow more generic than ours in the sense that it does not have an emphasis in security. Their solution makes use of Service Component Architecture (SCA) [7] modeling and a UML (Unified Modeling Language) profile. Ours, in change, proposes the use of domain specific languages (DSLs), more adapted to each of the extra-functional aspects in hand, to later combine these DSLs into one integrated model. Ortiz's work also does not provide any means to generate the implementation of the security aspects previously mentioned.

Along the same line is the proposal by Jegadeesan and Balasubramaniam [3]. Their model driven development approach is very similar to Ortiz's, dividing aspects in three levels: service, business, and domain aspects. These different aspects are later expressed in the shape of WSDL 2.0 and WS-Policy documents. A service policy metamodel is defined for the later. Jegadeesan selects WS-Policy due to being a widely accepted standard, but does not maintain this criterion and uses WS-PolicyConstraints, a somehow stalled standardization process (the latest draft [8] was published in October 2005). Our approach, on the contrary, leaned towards the use of the much more accepted WS-SecurityPolicy [9] standard. Even though Jegadeesan and Ortiz present use cases in their publications to probe their concepts, no tool implementing their approaches have been provided so far.

Shifting to more access-control-oriented proposals, we found the SECTET framework [4], perhaps the most remarkable one. SECTET makes use of a model-driven approach to focus on confidentiality, integrity, and non-repudiation characteristics of SOAs, using WS as the implementation technology. SECTET's primary goal is the generation of eXtensible Access Control Markup Language (XACML) [10] policies derived from Role Based Access Control (RBAC) [11] models. The work behind this framework appears to be very thorough, making it very interesting to analyze. In relation to the work presented in this document, while SECTET is mainly focused in access

control and XACML policies, the goal of the presented work is the integration of this and other (principally, but not only) security aspects in one unique solution.

In this work, it has been decided to follow WSDL and WS-Policy (and WS-SecurityPolicy) standards, due to their deep penetration and its almost universal utilization. Notwithstanding, the same criterion (to follow standards preferably) has not been applied in the case of XACML.

XACML, although a defined standard, has still not been so widely adopted by the industry. A clear sign of this is that from most possible target platforms with aspect-oriented support (e.g., Spring, JBoss), few or none provide XACML support, and most implement a proprietary access control description mechanism, less verbose and complicated. The presented work has, therefore, not adopted XACML as the access control language of choice, but left it as one of the many possible implementation options.

There are also other interesting ongoing research studies on the subject, (e.g. [12] and [13]). However, these works are either taking their first steps, or they focus on a more limited scope than the one of the work being presented here.

III. IN SEARCH OF THE INTEGRATED SOLUTION

A. *Requirements and Objectives*

From within the different requirements and objectives placed on this research, the ones with a higher influence on the result follows:

- Where feasible and convenient, reuse pre-existent tools/solutions (a.k.a. "don't reinvent the wheel").
- Use a model-driven approach.
- Relating to target platforms, make use of the new aspect-oriented functionalities that are becoming available in the different platforms to satisfy extra-functional requirements.
- Adopt standards as much as possible, if practical.

B. *Defined Architecture*

Fig. 1 shows the overall structure of the designed solution. It consists of a chain of model transformations (T1, T2, T3, and T4) and compositions (C1 and C2). Every colored box in the figure indicates a different type of model. These models include:

- Functional system model (the main input model).
- Resource access control model.
- Non-functional aspects models (Policy model).
- Intermediate Meta Model (iMM) model.

The iMM metamodel was defined with the objective of achieving an abstraction both of input modeling techniques/metamodels, and output target platforms. It is also designed for holding in one unique model, all information necessary to generate the different output artifacts. The iMM models are automatically generated, and the developers need never interact with them.

The presented approach proposes the development of each individual concern as an independent model, in a similar fashion to that of Multi-Dimensional Separation of Concerns [14]. Those models are later composed into a complete model of the system by weaving them together. This approach allows for the use of different DSLs, each one appropriate to the particular concern in hand. This set of metamodels is presented in section IV.A.

Ideally, different access control and policy models, previously developed by security experts, would be contained in repositories. As a result, the developer of the system need not be security savvy, and its responsibility on this stage would only be to properly assign policies or permissions to the different system resources. Furthermore, this task could also be performed by a security expert, and the developer of the functional part of the system may fulfill his work independently.

The different resulting outputs to be obtained, indicated in Fig. 1, include:

- Access control descriptors. These platform dependant descriptors describe the permissions each role in the system has to access the different resources available.
- Code templates, derived from the system model. These templates are also platform dependant.
- WSDL descriptors and the WS-Policy documents referenced by those. These are common to all target platforms.

C. *Sequence of Operation*

The sequence of steps to perform in the proposed approach is the following:

1) The user feeds the tool with the input functional model (Platform Independent Model 1, PIM 1).

2) The input model is transformed (T1) into an instance of the intermediate metamodel (iMM). At this stage this model only contains functional information.

3) From within the available access control and security policy models, the user selects the most appropriate one. The developer could tune it to better fit his system or, alternatively, define his own. Please notice that, at this stage, the artifacts being referred to are access control and security policy models, not documents.

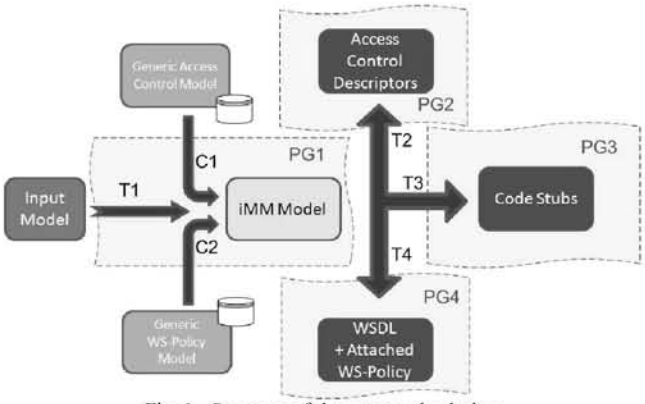

Fig. 1. Structure of the proposed solution

4) By means of a model composition tool, the user indicates which access control permissions should be associated to which resources. By the same token, the user also indicates the security policies to be applied to the operations offered by the service . (Marked as C1 and C2 correspondingly in Fig. 1)

5) All information (functional and extra-functional) is now contained in the platform-independent iMM model (PIM2).

6) At this stage, service and policy descriptors are derived from the PIM 2 model. Resulting WSDL documents hold references to the appropriate policies defined in the different generated WS-Policy documents (T3).

7) Next, access control descriptors and code templates are automatically created (T2 y T4).

IV. REALIZATION OF THE APPROACH

As a proof of concept, a prototypical implementation is under development. All input transformations and compositions (T1, C1 and C2) are complete, as well as the generation of WSDL and WS-Policy artifacts (T4). The implementation of the metamodels in Eclipse (some definitions have been reused, but no implementations were available at the time), the definition of the transformation and composition rules, and the implementation of the different assistants (wizards, cheat sheets, etc.) implied a considerable amount of work.

The current state of such prototype also includes a cheat sheet that guides the development process. Only the platform specific parts (code-stubs and access controls descriptors) remain to be completed. Implementation has taken place over the Eclipse platform, trying to make the most of the functionalities provided by it.

The design in Fig. 1 divides the implementation in different plugins (PG#), in order to allow for easier maintainability and evolution. Accordingly, the different artifacts to be generated (WSDL, WS-Policy, code templates), and the different metamodels to be composed (access control, policies) are to be delivered in different plugins.

The plugin in charge of the composition of the intermediate model, marked as PG1 in Fig. 1, is independent of latter generation plugins. The generation of the different output artifacts has also being made independent between themselves. This allows an autonomous evolution of each output with respect to new standard versions, for instance.

A. Metamodels

Referring to the metamodels employed in this solution, some pre-existent metamodels have been adopted, some other where merged together, whilst the rest had to be defined for the occasion. The different metamodels are described in the following sub-sections.

1) Input Metamodel: UML

UML, the modeling standard of choice in most cases nowadays, has been selected as the input's metamodel. A very simple UML profile is applied to these input models to guide the T1 transformation. The objective of this profile is to abstract the transformation of the different ways in which UML is being used (e.g., different modelers prefer to represent software components with UML component, class, or package model elements). This profile is basically used to identify components, interfaces, entities, etc.

2) Access Control Metamodel: SecureUML

Although the current state of the design incorporates a more generic metamodel [15] (indicated in Fig. 2 as the access control part), that support multiple access control techniques, the prototype currently uses SecureUML [16], a Role Based Access Control (RBAC, one of the currently most used access control techniques) metamodel.

3) Weaving Metamodels

Weaving metamodels guide the composition mechanisms of the different models. In this case, weaving access control models and weaving policy models require different weaving associations. For this reason, two different metaclases for weaving associations are defined: "AddPolicyReference" for policy-weaving, and "CombineResource" for access control.

The semantics of the associations are implemented as part of the composition rules. In the case of policy references, one new "PolicyReference" object is created on the model, with its URL value derived from the ID property of the policy, and the location property of the document containing it.

For access control compositions, a "resource" is associated with some particular actions (e.g. read, write, execute) described in the access control model. Correspondent actions are then created for the resource, and all associated permissions are introduced in the model (permissions connect roles with actions on resources, according to some attached constraints).

Currently, "resources" that are subject of access control rules include operations, entities, and interfaces. Permissions on interfaces propagate to all its operations in all its realizations. By the same token, all permissions on entities propagate to all operations accessing them. Conflict resolution between propagated permissions is not discussed in this paper.

4) iMM Metamodel

The intermediate metamodel developed for this approach is one of the fundamental parts of the system. It was designed to be as general as possible, in order to maximize access control techniques and policy standards support.

An excerpt of this metamodel can be seen in Fig. 2. It is composed of three differentiated parts: System Design, Access Control, and Policy.

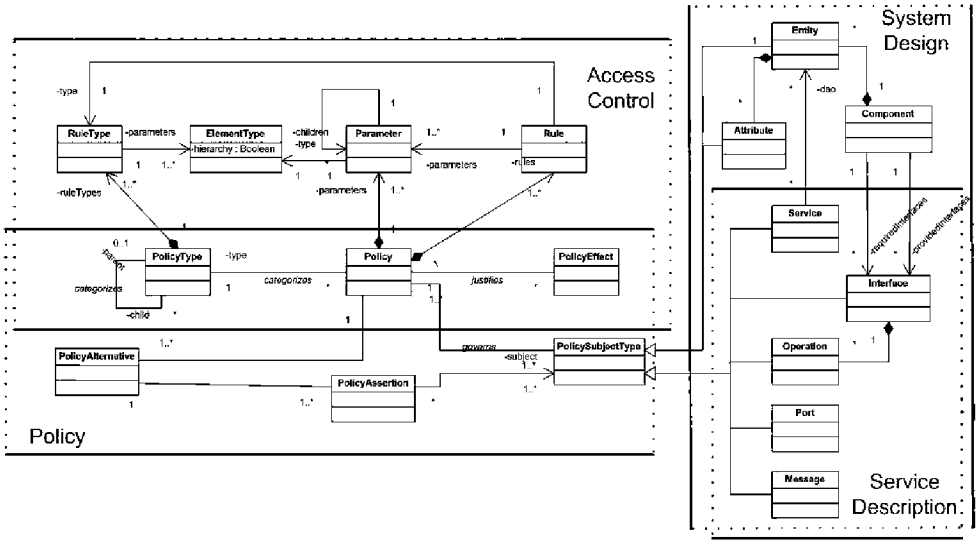

Fig. 2. iMM Metamodel (conceptual partial view)

The functional part of the metamodel follows a component-based approach, and is used to describe service components, entities and interfaces, among other metaclasses.

The generic metamodel [15] incorporated in the "Access Control" part of Fig. 2, allows for the use of different techniques (DAC [17], MAC[18], RBAC[11], OBAC[19]).

Finally, the "Policy" package of CBDI-SAE Meta Model for SOA [20] was selected for the service policies part due to its grade of maturity and flexibility. It was decided not to use the complete CBDI-SAE metamodel itself for two main reasons: it didn't include access control support, and its business-to-business orientation made it too heavy to implement for the objectives placed on this work.

The three selected metamodels where then studied to identify equivalent concepts that provided merging points.

5) WSDL Metamodel

A WSDL metamodel has been used for T4 transformation, and in C2 composition in order to visualize the future WSDL document as a model and facilitate the specification of policy application points. This metamodel was first derived from the WSDL 1.2 specification schema, and then modified to better fit its use in editors and transformations.

6) WS-Policy Metamodel

A WS-Policy metamodel, used both in T4 transformation and C2 composition, has been developed for the prototype. It was necessary to implement it from scratch, as the metamodel obtained from the specification schema was not usable in an editor, and that all other previously existent WS-Policy metamodels (e.g., [3]), do not allow for the use of both "compact" and "normal" WS-Policy mode, as intended.

The transformations and later generation of WS-Policy documents operate on this metamodel, including the case of WS-SecurityPolicy models.

7) WS-SecurityPolicy Metamodel

A WS-SecurityPolicy metamodel has been defined for this prototype as an extension to the WS-Policy metamodel. It leverages the policy editor, facilitating the expression of standard security policy assertions, and allowing the incorporation of validation rules (defined in the standard and already implemented in the prototype) for such assertions. It is, to the best knowledge of the authors, the first available implementation of a WS-SecurityPolicy metamodel.

B. Implementation and Validation of the Prototype

Although the proposed approach is not tied to a specific implementation technology, after an evaluation of the available tools [21], it was decided to base the implementation on the Eclipse platform.

The user interface of the tool is currently composed by a cheat sheet and a menu. The user can follow in the cheat sheet the steps described in section III.C or, alternatively, take the different steps independently as he sees fit by selecting them in the menu.

With respect to the validation of the prototype, Eclipse's validation tools are being used to validate resultant WSDL and WS-Policy documents respectively.

The obtained result as a whole is planned to be validated by implementing the WS-I [22] "Supply Chain Management" use case and companying validation tools, with security requirements included.

V. CONCLUSIONS AND FUTURE WORK

This paper has presented ongoing work towards the achievement of an integrated, model-driven solution for the development of service-oriented architectures with policy awareness and access control support. It is, to the knowledge

of the authors, the first model-driven approach to integrate all these functionalities together.

The different participant metamodels have been presented. Among them, two stand out: the intermediate metamodel, containing all necessary information for generating the resulting artifacts, and the WS-SecurityPolicy metamodel, leveraging validation security awareness.

The proposed solution is not tied to any implementation or target technology, allowing for great flexibility in its evolution regarding the appearance of new standards or technologies. This is an improvement compared to editor-based approaches, tightly coupled to the technologies they have been developed for. The prototype under implementation provides a set of functionalities not always found together in one only tool.

Moreover, the WS-Policy framework itself has been defined in a generic fashion, allowing for many standards to be formulated based on it. Any new standard that may be defined under its umbrella will automatically be supported by this solution, and its assertions can readily be applied.

Consequently, all already available or in process of being defined policy standards are supported by the tool (e.g., WS-Addressing [23], WS-AtomicTransaction [24]). This leverages its use for developing not only security-aware services, but also include timing constraints, secure exchange, transactions, etc.

With respect to future work, in the short term it will be focused on finishing the implementation of the prototype, mostly on the generation of code templates and access control descriptors for a determined platform. Some work has been done to integrate Eclipse's Connected Data Objects (CDO) repositories within it.

In a longer term, it is planned to consider other policy aspects under standardization process. As previously mentioned, the approach is flexible enough to support any WS-Policy-based standard, but it is not able to use the information intrinsic to the assertion for its benefit (shaping of code generation, configuration of target platform, generation/use of an extra-functional aspect, e.g.).

Additionally, researching on the semantics of timing constraints and requirements on services, its expression as policies, and its effects on the shaping of the generated code could provide interesting results.

On a different token, the shifting from a metamodel-based generation approach towards a weaving-based generation approach (in which any metamodel could be used to generate artifacts based on a set of weaving associations) could provide an alternative line of research.

ACKNOWLEDGMENT

This work was supported in part by the Centro Español de Desarrollo Tecnológico (CDTI, Ministry of Industry, Commerce and Turisms), by means of the ITECBAN (Infraestructura Tecnológica y Metodológica de Soporte para un Core Bancario) project, from the INGENIO 2010 program.

REFERENCES

[1] OMG Model Driven Architecture, http://www.omg.org/mda/

[2] G. Ortiz and J. Hernández, "Service-Oriented Model-Driven Development: Filling the Extra-Functional Property Gap," Service-Oriented Computing – ICSOC, 2006, pp. 471-476.

[3] Harshavardhan Jegadeesan, Sundar Balasubramaniam: "A Model-driven Approach to Service Policies", in Journal of Object Technology, vol. 8, no. 2, March-April 2009, pp. 163-186

[4] M. Hafner, R. Breu, B. Agreiter, and A. Nowak, "Sectet - an extensible framework for the realization of secure inter-organizational workflows," in WOSIS, E. F. Medina, Mariemma, E. F. Medina, and Mariemma, Eds. INSTICC Press, 2006, pp. 47-57.

[5] "Web Services Description Language (WSDL) 1.1, W3C Note", March 2001

[6] Web Services Policy 1.2 - Attachment (WS-PolicyAttachment), http://www.w3.org/Submission/WS-PolicyAttachment/

[7] Beisiegel, M., et al. "Service Component Architecture. Building Systems using a Service Oriented Architecture". http://download.boulder.ibm.com/ibmdl/pub/software/dw/specs/ws-sca/SCA_White_Paper1_09.pdf, , November 2005

[8] XACML-Based Web Services Policy Constraint Language (WS-PolicyConstraints), http://research.sun.com/projects/xacml/ws-policy-constraints-current.pdf

[9] "WS-SecurityPolicy 1.2, OASIS Standard", July 2007

[10] OASIS "eXtensible Access Control Markup Language (XACML) Version 2.0", IN Moses, T. & Godik, S. (Eds.), OASIS, February 2005.

[11] Sandhu, R., Coyne, E. J., Feinstein, H. L., and Youman, C. E. Role-Based Access Control Models. In IEEE Computer, 29(2):38–47, 1996.

[12] X. Larrucea and R. Alonso, "ISOAS: Through an independent SOA Security Specification," Composition-Based Software Systems, 2008. ICCBSS 2008. Seventh International Conference on, 2008, pp. 92-100.

[13] Fumiko Satoh, Nirmal K. Mukhi, Yuichi Nakamura, Shinichi Hirose, "Pattern-based Policy Configuration for SOA Applications," scc, vol. 1, pp.13-20, 2008 IEEE International Conference on Services Computing Vol. 1, 2008.

[14] P. Tarr et al., "N Degrees of Separation: Multi-Dimensional Separation of Concerns." Proceedings of the International Conference on Software Engineering (ICSE'99), May, 1999.

[15] T. Mouelhi, F. Fleurey, B. Baudry, and Y. Le Traon, "A Model-Based Framework for Security Policy Specification, Deployment and Testing," Model Driven Engineering Languages and Systems, 2008, pp. 537-552.

[16] T. Lodderstedt, D. Basin, and J. Doser, "SecureUML: A UML-Based Modeling Language for Model-Driven Security," «UML» 2002 — The Unified Modeling Language, 2002, pp. 426-441. [17] B. Lampson, \Protection," in 5th Princeton Symposium on Information Sciences and Systems, March 1971, pp. 437-443.

[18] D. E. Bell and L. J. LaPadula, "Secure computer systems: Unified exposition and multics interpretation," Tech. Rep. ESD-TR-73-306, The MITRE Corporation, March 1976.

[19] A.A.E. Kalam, S. Benferhat, A. Miège, R.E. Baida, F. Cuppens, C. Saurel, P. Balbiani, Y. Deswarte, y G. Trouessin, "Organization based access control," Proceedings of the 4th IEEE International Workshop on Policies for Distributed Systems and Networks, IEEE Computer Society, 2003, p. 120.

[20] "CBDI-SAE™ Meta Model for SOA Version 2", September 2007

[21] J.P. Silva, M. de Miguel, J.F. Briones, A. Alonso, "Composing Models with Six Different Tools: a Comparative Study." CEUR-WS MtATL, July 2009, pp. 103-118

[22] Web Services Interoperability Organization (WS-I), http://www.ws-i.org/

[23] W3C "Web Services Addressing 1.0 - Metadata", IN Gudgin, M., Hadley, M., Rogers, T., & Yalçinalp, Ü. (Eds.), W3C, July 2007.

[24] OASIS "Web Services Atomic Transaction (WS-AtomicTransaction) Version 1.1", IN Little, M., & Wilkinson, A. (Eds.), OASIS, February 2007.