Tool support for code generation from a UMLsec property

Conference or Workshop Item

How to cite:

For guidance on citations see FAQs.

© 2010 The Authors

Version: Accepted Manuscript

Link(s) to article on publisher’s website:
http://soft.vub.ac.be/ase2010/

Copyright and Moral Rights for the articles on this site are retained by the individual authors and/or other copyright owners. For more information on Open Research Online’s data policy on reuse of materials please consult the policies page.

oro.open.ac.uk
tool support for code generation from a umlsec
property

lionel montrieux†
the open university
milton keynes
united kingdom
l.m.c.montrieux@open.ac.uk

jan jürjens
tu dortmund & fraunhofer
isst
dortmund
germany
http://jan.jurjens.de

charles b. haley
the open university
milton keynes
united kingdom

yijun yu
the open university
milton keynes
united kingdom
y.yu@open.ac.uk

pierre-yves schobbens
university of namur
namur
belgium
pyschobbens@fundp.ac.be

hubert toussaint
university of namur
namur
belgium
hto@info.fundp.ac.be

abstract
this demo presents a tool to generate code from verified
role-based access control properties defined using umlsec.
it can either generate java code, or generate java code for
the uml model and aspectj code for enforcing said rbac
properties. both approaches use the java authentication
and authorization service (jaas) to enforce access control.

categories and subject descriptors: d.2.2 software
engineering: design tools and techniques [computer-aided
software engineering (case)]

general terms: security

1. introduction

security requirements can be made explicit on the design
level, such as annotations on a uml model. umlsec [4]
extends uml to allow one to express security properties on
a model, but it is still the developer’s responsibility to im-
plement the code that will actually enforce those properties.
this process can generate bugs and will not give any guar-
antee about how the implementation conforms to the model.

in this demo, we present a tool that generates java and
aspectj code from a uml with a verified umlsec property.
ith can either generate only java code, or, alternatively, im-
plement the security property using aspectj while still using
java for the functional code. the tool also has other fea-
tures for umlsec models verification that are not discussed
here.

the next sections are organised as follows: we first give
a short overview of umlsec in section 2, then in section 3

∗this work was partially supported by the eu project “se-
curity engineering for lifelong evolvable systems (secure
change)” (ict-fet-231101)
†part of this author’s work was done as a msc student at
the university of namur, belgium, under the supervision of
pierre-yves schobbens and hubert toussaint

we describe the tool, with a particular attention towards the
new features we are focusing on in this demo. in section 4
discusses related work, and we finally discuss future works
in section 5.

2. expressing access control as an
umlsec property

umlsec [4] is an uml profile allowing one to define secur-
ity properties, using standard uml extension mechanisms
like stereotypes and tagged values. one of those properties
that can be defined on a uml model is role-based access
control. a uml activity diagram can be annotated to as-
sign roles to users, grant permissions to roles, and protect
actions. each swimlane in the activity diagram represents
a user. it is therefore possible to check the defined rbac
property by making sure no protected action is in the swim-
lane of a user that is not allowed to perform it. currently,
only a subset of the rbac standard is supported by the
umlsec specification. for example, it is assumed that all
roles are granted to a user at the start of a session, and that
no roles can be dropped or delegated to another user.

3. enforcing access control prop-
erties through code generation

the umlsec tool [2] allows one to check whether or not
a model enforces a umlsec property [5]. it also allows one
to generate code conforming to the model.
This demo focuses on a new feature: code generation from a UML model with an UMLsec RBAC property, as described on Figure 1. Given an UML model with UMLsec annotations describing an RBAC property, we generate code that conforms to the RBAC property. Two different approaches have been implemented, both using the JAAS [1] framework: the first one produces only Java code, while the second one uses AspectJ to enforce the RBAC property.

Both code generation techniques have been implemented using Aspect-Oriented Programming [3]. While code generation from the UML model is done in Java, we use AspectJ to add generation of code enforcing the UMLsec property, which has several advantages over an Object-Oriented only implementation. First, it allows us to clearly separate the code generation of the UML model itself from the code generation of the associated UMLsec property. Second, it allows us to easily extend the tool to generate code from other UMLsec properties by simply writing a new aspect, and reusing the existing Java code for generating to code corresponding to the UML model. And finally, it allows us to study the potential conflicts between several UMLsec properties in terms of an aspect composition problem.

3.1 Object-Oriented code generation

The first approach produces only Java code, both for implementing the UML model and the UMLsec property. The aspect responsible for generating code enforcing the RBAC property simply monitors the code generation process, and adds the necessary lines of code in the places where they are needed. The access control code is therefore spread all over the code base. While it might make modifications like adding (resp. removing) access control protection to a non-protected (resp. from a protected) method or attribute, it has the advantage of not requiring the use of Aspect-Oriented Programming and its drawbacks, like potential conflicts between aspects or negative impact on performances.

Figure 2 shows an example of the lines of code added to a UML model to enforce an RBAC property. Figure 3 shows the same example as Figure 2, but using aspects.

4. RELATED WORK

There exist a lot of tools for generating Java (or other Object-Oriented language) code from UML models, like IBM’s Rational Rose or ArgoUML. None of them, however, allows one to verify that a model enforces an UMLsec RBAC property, and generate code that conforms to it.

SecureUML [6] is an alternative to UMLsec for developing RBAC properties on a UML model. The authors developed a prototype implementation of their approach to translate RBAC properties expressed using SecureUML into EJB code. However, their approach does not include model-level verification as UMLsec does prior to the code generation, and they do not offer the opportunity to chose between several paradigms for the target code.

5. CONCLUSION AND FUTURE WORK

The tool we presented allows one to generate code from a RBAC property expressed in UMLsec in two different ways: by generating only Java code, or by generating Java code for the functional model as well as AspectJ code for enforcing the RBAC property. The generation of the code that enforces access control properties is implemented in AspectJ.

In future work, we will add support for code generation using other security frameworks than JAAS, like EJB. We will also work on code generation from other UMLsec properties. This will raise new and interesting challenges, as we will need to generate code that enforces several different properties, without introducing conflicts.

6. REFERENCES

tutorials/index.html (Last accessed September 2009).