AO4AADL
Aspect Oriented ADL for Embedded Systems

NOTERE'2010
31 Mai - 2 Juin 2010, Tozeur - Tunisie

Sihem Loukil, Slim Kallel, Bechir Zalila
ReDCAD Laboratory
University of Sfax, Tunisia

Context & Objectives

Context

Managing embedded systems complexity and scalability:

- Supporting the crosscutting concerns in architecture level
- Modeling and implementing non functional properties

Problematics

Traditional ADLs do not provide appropriate formalisms to separate any kind of crosscutting concerns.

- Lack of appropriate formalisms providing modularity and scalability
- Poor description of the software architectures
- Tedious adaptation to constantly changing user requirements and specifications

The specification of non-functional concerns in architectural level is not well-modularized

- Tangled with the specification of each component's core functionality
- Scattered across the specification of different components.

Objectives

1. Propose an architecture description language that support the crosscutting concerns
2. Propose an approach based on the defined language for modeling and implementing non functional properties .
3. Develop a compiler of the defined language and a code generator to generate the appropriate code
4. Validate the new language and the whole approach by different case studies

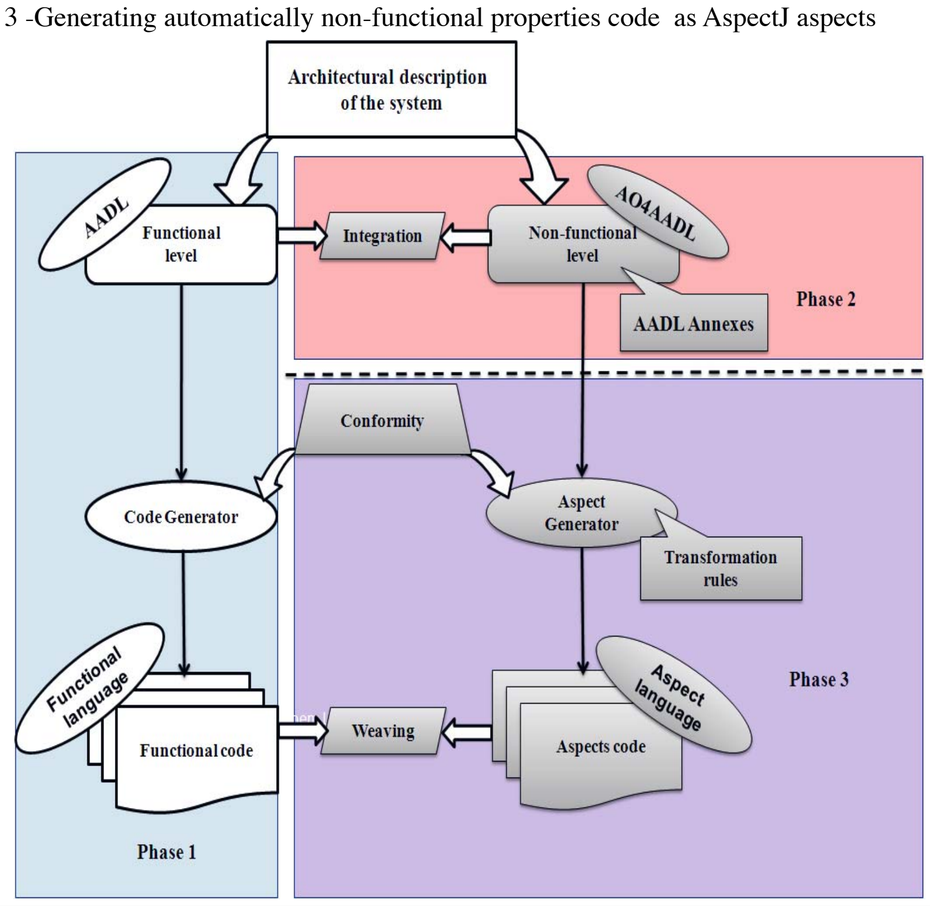
Proposed Approach

We propose an approach for implementing non-functional properties that resumes three principal steps:

1- Implementing of the functional code using AADL

2- Designing non-functional properties as an aspects using AO4AADL

3 -Generating automatically non-functional properties code as AspectJ aspects



AO4AADL language

AO4AADL is an aspect oriented ADL that expresses aspect concepts at the architectural level. Based on the annex extension mechanism, we propose to enrich AADL specifications with AOSD concepts.

We defined the **grammar** for our new language AO4AADL. The syntax is inspired from that of AspectJ.

An AO4AADL aspect consists of two parts:

- *Pointcut specificat*ion determines the conditions under which the aspect is invoked by the corresponding functional components. AO4AADL explicitly defines the architectural joinpoints as places where the effect of aspect annex can occurs.
- *Advice specification* encapsulates the behaviour of the aspect depending on its location. This specification is inspired from the Annex Behavior with some modifications to satisfy other requirements that are needed to be expressed.

We propose an aspect **code generator** which automatically translates AO4AADL aspect into AspectJ code in terms of pointcuts and advices.

The full grammar of AO4AADL and the transformation rules are available on: http://www.redcad.org/projects/AO4AADL/

AO4AADl example

As an example of embedded system, we apply our approach on Automated Teller machine. The following aspect specifies that the client has exactly three authentication attempts in order to use his credit card.

```
aspect CheckCode{
pointcut Verification ( ) : call outport RestoreCount_out ( . . ) ;
advice around ( ) : Verification ( ) {
Variables{
      counter : Base Types : : Integer ;
      result : Base Types : : Boolean ;
      message : Base Types : : String ;}
 initially {
      counter :=1 ;
      message : = ” Card Rejected ” ; }
 CardVerif ication ! (NumCard in , Code in , result ) ;
if ( ( counter = 3) and ( result = false ) ){
      RejectedCard out ! (message ) ;
      counter := 1;}
 else  proceed ( ) ;
counter := counter + 1;
} } }
```

Publications

Sihem loukil, Slim Kallel, Bechir Zalila, and Mohamed Jmaiel
AO4AADL: an Aspect Oriented ADL for Embedded Systems
Submitted to the 4th European Conference on Software Architecture (ECSA 2010)
LNCS, Springer, Copenhagen, Dennmark

This work is a part of the master thesis of Sihem Loukil.
NTSID master, National School of Engineers of Sfax, University of Sfax, Tunisia

Future Works

We plan to develop an eclipse plug-in for supporting the graphically modeling of AO4AADL aspects within AADDL specifications, as a part of OCARINA tool suite.

Grammar	Compiler & Code generator	Tool support
AO4AADL language		