Study on Weaving Process at Software Architectural level*

Youcong Ni1,2, Shi Ying1, Jing Wen1, Linlin Zhang1, Peng Ye1,3, Jiajing Zhang2
1 The State Key Lab of Software Engineering, Wuhan University, Wuhan 430072, China
2 Department of Mathematics and Physics, Anhui Institute of Architecture & Industry, Hefei 230601, China
3 Wuhan University of Science and Engineering, Wuhan, 430074, China
nyc@mail.whu.edu.cn

Abstract

The aspect-oriented software architectural weaving mechanism, to integrate aspectual components that encapsulate crosscutting behaviors and features into components and connectors constituting Software Architecture (SA), contributes to analyze and verify overall behaviors and quality attributes of SA. This paper proposes a new kind of weaving process and rules, which offer the measure to transform the unwoven SA model into the woven SA model. The unwoven SA model is described in a special aspect-oriented architecture description language, called AC2-ADL, and specifies the locations, times and constrains of injection. Then a woven SA model only containing components and connectors is acquired after weaving, and easier to analyze and verify.

1. Introduction

Nowadays, SA [1, 2] has become an area of intense research in the software engineering and academic communities. During the increasingly in-depth study and widely application of SA, it is well known that there always have a little crosscutting behaviors and features in conventional SA modeling approaches, and these crosscutting features and behaviors cut across the boundaries of other modules in different location and time, leading to high tangling and scattering between components and connectors. To solve this problem, researches have developed flourishingly in the area of aspect-oriented SA [3, 4]. For example, there is a common recognition that these crosscutting behaviors and features need to be encapsulated with independent design element by using the principle of separation of concerns [5]. However, It is a difficult issue that how to weave those design units that encapsulate crosscutting behaviors and features, trivial components, and trivial connectors together in order to form a whole, which embodies correct overall behavior and quality attribute.

At present, there have already many design approaches of aspect-oriented SA for resolving this problem by their weaving mechanism. For example, FAC[6], which exposes join point by interception interface of component, sets and unsets the binding between component and aspect by use of weaving interface of component. Composition between aspect and component is implemented by aspect binding in FAC. DAOP-ADL[7], in which an special evaluated interface is defined. This interface specifies how aspects affect interfaces of component, and connection between components and aspect are defined in the SA configuration. AspectualACME[8] is an aspect-oriented extension of ACME. AspectualACME introduces crosscutting role and interaction protocol in division of connector so as to define crosscutting influence on interactions among base roles. In AspectualACME, configuration describes the players of both base role and crosscutting role, and then connector along with configuration realizes the weaving between aspect and component. AO-ADL [9] is the extension of the semantic of traditional connectors in order to represent the crosscutting effect of “aspectual” components. Some information about weaving is localized in connectors by introducing aspectual role and aspectual binding.

In Aspect-Oriented SA design approaches mentioned above, aspect and other elements can be implicitly woven by increasing or extending existing elements of SA. Consequently, it is a little difficult to analyze and verify the overall behavior and quality attribute of SA because of more complicated specification of Aspect-Oriented SA model. To solve this problem, this paper proposes weaving process and rules based on unwoven model described in AC2-ADL. By use of this process and rules defined clearly, aspectual components designed independently before weaving is explicitly woven into components. A model of SA only containing components and connectors is acquired after weaving and is easier to analyze and verify.

The other parts of this paper are organized as follows: Section2 gives overview of AC2-ADL

*Research supported by the National Natural Science Foundation of China under Grant No.60773006 and the Doctoral Fund of Ministry of Education of China No. 20060486045
language; Section 3 depicts five steps of weaving process in detail. Section 4 presents some conclusions and ideas for future work.

2. Overview of AC2-ADL

In order to weave the aspects, components, and connectors designed separately together, and acquire the whole architecture that can work together harmoniously, it is very necessary to extend the traditional architecture description language (ADL) to describe the SA models, including unwoven and woven ones. So we introduce a new ADL named AC2-ADL [10] to solve this problem. There are many differences between unwoven SA and woven SA, such as elements and their specifications. For unwoven SA, we employ components and connectors to model the basic SA, in which way the concerns related to functional requirements (FRs) and relationships among them are satisfied. Meanwhile, we use aspectual component to encapsulate those behaviors and features that crosscut the basic SA elements, in which way the concerns associated with non-functional requirements (NFRs) and relationships among them are satisfied. Furthermore, we apply aspectual connector and software configuration to model the relation of injection by which aspectual components can be injected into basic SA, indicating the impact imposed on the FR concerns by NFR concerns. According to the relation of injection which is well-defined and contains four elements, such as locations, contents, times and constraints of injection, through weaving, the woven SA model which is only composed of component and connector is acquired, and crosscutting effect can be reflected within specification of a component. Modeling elements of SA in AC2-ADL relevant to weaving are introduced as follows:

1) Components

Aiming at describing the impact caused by crosscutting after weaving, it is necessary to extend the traditional components. A block named InternalProcess is introduced into specification of component, and can represent both operational behaviors and crosscutting behaviors.

2) Aspectual components

Aspectual components are employed to encapsulate all the behaviors and properties that scatter and tangle in modeling elements of SA. They are comprised of crosscutting interfaces and required interfaces. Crosscutting interfaces in an aspectual component provide the crosscutting services for other components, while required interfaces require the services which aspectual components need. Crosscutting interface can include a few crosscutting operations, which can actually affect basic SA elements.

3) Aspectual connectors

Aspectual connectors are used to specify the crosscutting interactive ways and interactive protocols between various kinds of modeling elements before weaving. They are composed of baseRole, crosscuttingRole, and crosscuttingProtocol. In the crosscutting relation, BaseRole defines set of elements in which anyone is to be crosscut, while crosscuttingRole represents set of elements in which anyone is to crosscut elements defined by baseRole. CrosscuttingProtocol depicts the interactive rules and constraints between baseRole and crosscuttingRole. There are “before”, “replace” and “after” three types of CrosscuttingProtocol. Both baseRole and crosscuttingRole should contain one or more behaviors.

4) SA configuration

SA configuration can be partitioned into instances of components, instances of connectors, instances of aspectual components, instances of aspectual connectors and description of configuration which indicates the connection relation between various instances. To indicate that an aspectual component may crosscut other model elements of SA at one or more places, we introduce pointcut designator (PCD) in SA configuration to explicitly define the joint point at architectural level. Here PCD and crosscutting interface of aspectual component respectively play baseRole and crosscuttingRole of aspectual connector instance. So the complicated relation of injection between aspectual components and basic SA elements can clearly be modeled. The definition of PCD is shown as Fig 1.

Here we employ logic operands, keywords such as PROINTF, REQINTF, PROORLE, REQROLE, and wildcard "*" to describe the same baseRole of aspectual connector instance conveniently and precisely. For example, PCD: and(comInstName.PROINTF, *. REQROLE) denotes the provided interfaces in component instance named comInstName and all the required roles in each connector instances are the join points at the architecture level. Those join points are waiting for being crosscut by aspect components.
3. The weaving process

In our work, we propose a weaving process which can change the unwoven SA model into the woven SA model under support of weaving rules. The woven SA model is only comprised of components and connectors. So we can simply verify the whole behaviors and quality attributes about the woven SA in uniform way. A weaving process is made up of initializing, searching for PCD, analyzing PCD, resolving conflict, and interweaving five steps.

1) Initializing

This step focuses on creating a blank weaving table, called weaveTbl, which is a two-dimensional table. Its row title represents the crosscutting operation name in crosscutting interface, while its column title expresses the operation name and type of interface. The content of each cell of the weaveTbl may be filled with the time and constrains of injection. For example, the crosscutting operation named AspectInstNamem.crintf.k.optr affects the operation named comInstNamen.intf.j.optk, the time of injection is "before", and the priority of injection is "10". So this information is filled in the appropriate cell shown in Table 1. At the beginning of weaving, weaver scans the original configuration of SA, finds out all the crosscutting operations in every crosscutting interface, as well as all operations in every interface and the types of interfaces for these operations. Row title and column title of weaveTbl are determined. Consequently, blank weaveTbl is created.

2) Searching for PCD

Weaver first seeks instance of aspectual connectors in the configuration of unwoven SA. Then, according to the description of configuration, weaver finds out the players that play the baseRole. These players are just PCD. Next, weaver scans and finds out those crosscutting operations that affect the PCD by means of player of crosscuttingRole and behavior attachment. As a result, the corresponding rows in weaveTbl are selected.

3) Analyzing PCD

There are two kinds of PCD: atomic PCD which does not include logic operands, and composite PCD which includes logic operands. For an atomic PCD, the weaver directly filled out the cells in weaveTbl according to appropriate matching rule. For a composite PCD, weaver iteratively parses it till an atomic one, according to appropriate matching rule. For example, if atomic PCD is comInstName.PROINTF, it will match weaveTbl's columns, whose titles denote operation from the provided interfaces in component instance named comInstName. And then According to crosscutting interactive protocol of aspectual connector, the time of injection and some relevant constraints such as priority are filled in the corresponding cell of weaveTbl.

4) Analyzing conflict

The filling of the whole weaveTbl are also finished after analyzing all of PCD in unwoven SA configuration. Now, Weaver constructs aspectual chains for columns of weaveTbl. Concretely, each column will be scanned from top to bottom, and then according to time of injection filled in cell, crosscutting operations will be set in aspectual chain, which is composed of "before segment", "replace segment" and "after segment". The "before segment" is used to store crosscutting operations whose corresponding type of crosscuttingProtocol is "before". The "replace segment" is used to store crosscutting operations whose corresponding type of crosscuttingProtocol is "replace". The "after segment" is used to store crosscutting operations whose corresponding type of crosscuttingProtocol is "after". Thus for one column, an aspectual chain is created. Following, this chain will be analyzed by the rules of analyzing conflict in order to find out conflicts. If each segment of this chain has more than two crosscutting operations, then these cells relevant to these operations will be examined. After examining, there is one of two results: One is that conflict occurs because the priority in the cells has not been defined; another is that the order of operation in

---

**Table 1. weaveTbl**

<table>
<thead>
<tr>
<th>operation</th>
<th>comInstNamex, intf.x Optx (provided)</th>
<th>...</th>
<th>comInstNamex, intf.x Optx (required)</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>AspectInstNamex, crintf.x Optx</td>
<td>...</td>
<td>...</td>
<td>time=before, priority=10</td>
</tr>
</tbody>
</table>

---

1576
same segment of chain is adjusted in accord with its priority.

5) Interweaving

After analyzing conflicts is successfully finished, all aspectual chains are formed and non-conflict. Crosscutting operations in an aspectual chain will be woven into component by interweaving rules, and the aspectual chain will also be woven into internalProcess block of component. The specification of component after weaving is as follow:

```plaintext
component aCom is
  parameters ...
  methods aMethod is ...; aAdvice1 is ...; aAdvice2 is ...
  providedInterface aProIntf is { Operation aMethod ... }
  requiredInterface aReqIntf { ... }
  InternalProcess is { aAdvice1 before aMethod; aAdvice2 after
                        aAdvice2 ... }
  constraints is { ... }
```

Fig 2. The partial specification of component after weaving

4. Conclusion and Future Work

The primary contribution of this paper is that the weaving process and relevant rules are proposed. The problem of weaving the aspects, components, and connectors together can be resolved by them. The process is composed of five steps which are initializing, searching for PCD, analyzing PCD, analyzing conflict, interweaving, and the rules is composed of three groups of rules which are matching rules, rules of analyzing conflict and interweaving rules. Through the process and the rules, aspectual components designed separately is woven into components, and an SA model only composed of components and connectors is formed, which make the whole behaviors and quality attribute easier to understand, analyze and verify.

The future research will pay more attention to modify, improve weaving rules. In addition, we will optimize the weaving process and formalize the weaving rules based on AC-ADL’s semantics.

5. References

            EarlyAspects/Papers/ Tekinerdogan.pdf.
            221.
            274