IEEE Trasactions on Software Engineering

Aspectual Feature Modules



Sven Apel
Thomas Leich
Gunter Saake

Cleice Souza

Reutilização de Software 16/10/2013

Roteiro

- Introdução
- Quais as diferenças e semelhanças?
- Integração de Feature com Aspecto
- Decomposição feature-basead e aspect-based
- Simbiose de AOP e FOP
- Aspectual Feature Modules
- Contribuições

Introdução

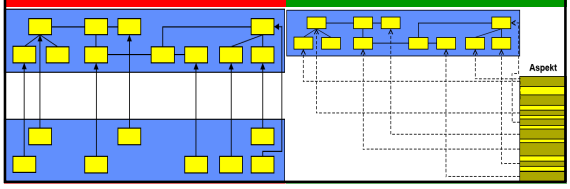
- SPL está ganhando cada vez mais atenção em engenharia de software e é tema de pesquisas futuras;
- FOP e AOP são paradigmas de programação usados no desenvolvimento em SPL;
- Por que em SPL?
- Porque os programas em SPL são implementados incrementalmente através da composição de código associado com características (incremento funcional).

Quais as diferenças e semelhanças?

Diferenças		
	FOP	AOP
Aspect		✔
Feature Modules	✖	
AspectJ linguagem mais difundida		✔
Jak extensão de Java	✖	
Classes, Pontos de corte, Pontos de Junção...		✔
Foca na separação, modularização de interesses transversais		✔
Foca na modularização de características	✖	
Classes, Refinamento, Mixin/Jampack	✖	
As diferenças são terminologias, métodos, mecanismos de transformações e ferramentas		

A diferença entre os paradigmas torna-se mais evidente quando consideramos implementação em larga-escala

FOP X AOP



Quais as diferenças e semelhanças?

- A situação tornou-se confusa, pois **features** são tratadas como **interesses transversais**;
- Mas, como resolver?
- **Proposta do artigo:** dissipar a confusão através da comparação de **FOP** e **AOP** em níveis de programação;
- **Foco:** linguagens que permitem comparar **FOP** e **AOP** como **Java**;
- **Ideia:** combiná-los.

Integração de Feature e Aspecto

Passo i:Foi estudado a decomposição de **POO** que resulta em uma hierarquia de classes;

Passo ii: Foi estudado a decomposição de **FOP** através de fragmentos de classe dentro da **POO**;

Passo iii: Foi estudado a decomposição de **aspectos** através dos seus construtores.

Integração de Feature e Aspecto

- **Decomposição feature-basead e aspect-based**

Lida: Programas de diferentes estruturas por causa de suas diferentes intenções; e

Lida: Diferentes implementações porque suas linguagens são diferentes.

- **Features** devem implementadas de forma modular e serem tratadas como **interesses transversais**;

Decomposição feature-basead e aspect-based

- **Aspectos** podem afetar códigos associados com muitas **features** e **features modules** podem conter códigos com muitos **aspectos**;
- A decomposição de **AOP** e **FOP** trata software em três dimensões: **classes, aspectos** e **features**;
- A ideia é fazer com que os programadores implementem **features** como unidades estruturais em **AOP** em vez de **POO**.

Simbiose de FOP e AOP

- Foi realizado uma simbiose entre **AOP** e **FOP** através de critérios de avaliação.
- **Propósito:** Saber como combinar os mecanismos de linguagens **AOP** e **FOP** para integrar **aspectos** como artefatos de software dentro de **FOP**

Critérios de Avaliação	FOP	AOP
Ponto de corte heterogêneo	Bom suporte	Suporte limitado
Ponto de corte homogêneo	Não apresenta suporte	Bom suporte
Ponto de corte estático	Bom suporte	Suporte limitado
Ponto de corte dinâmico	Suporte fraco	Bom suporte
Feature de coesão	Alto grau	Baixo grau

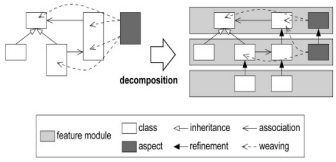
Aspectual Feature Modules

- É uma abordagem concreta da simbiose entre **AOP** e **FOP**;
- Estende a linguagem **Jak** de **FOP** para integrar **aspectos** nas classes que apresentam **refinamento**;
- Encapsula **classes** e **aspectos** que contribui com **feature**;
- **Feature** é implementada por uma coleção de artefatos, entre eles **classes**, **refinamentos** e **aspectos**

Aspectual Feature Modules

- AFMs refina um programa em duas formas:

1. Usando **Jak** como **refinamento**; ou
2. Usando os construtores de **AOP** mais especificamente **declarações inter-tipo**.



Contribuições

- Avaliação sistemática da comparação entre **FOP** e **AOP** através dos critérios de avaliação;
- Simbiose entre **FOP** e **AOP** chamada de **AFMs**;
- Avaliação da **AFMs** em um estudo de caso que indica uma previamente a integração de **aspectos** com **features modules**