Current state and future challenges in Optional Weaving

Constanze Michaelis
Student Conference on Software Engineering and Database Systems
June 27, 2009
Agenda

Motivation

Feature optionality problem

Optional Weaving

Conclusion
Motivation

- software product lines at JETI
  - firmware was built according to pcb layout (spl with preprocessor)
  - PC software was built “new” according to measurement device
  - copy and paste including errors
  - new errors emerging
  - software was full of features ⇒ one feature ↔ one customer
Motivation

- software product lines (spl) become more and more important in software development
- major design principle of spl: separation of concerns
- features
  - describe concerns of spl
  - selectable units within spl
  - mandatory or optional
FOP/AOP

• separation of concerns realized with
  • aspect-oriented programming (AOP)
    • aims on separating the crosscutting concerns (code scattered across multiple components)
    • implementation of crosscutting concerns as aspects
    • pointcuts and advice for additional features, traditional design concepts for core
  
• feature-oriented programming (FOP)
  • aims on feature traceability
  • idea: build program by composing features, where feature refines another feature incrementally
  • features composed by mixin approach within AHEAD toolsuite
FOP/AOP

• features in software product lines are often optional and interact with / depend on each other
• leads to the feature optionality problem when features interact and are optional
• often interacting optional features were implemented mandatory
Feature optionality problem
Feature optionality problem

- occurs when features that interact are optional
Feature optionality problem

- occurs when features that interact are optional
- first idea: encapsulation of interacting code as derivative feature
Optional Weaving

- implementation of optional interactions within feature
- i.e. the interaction code remains within feature but is optional
- optional interaction code is woven when both features are implemented
Optional Weaving

• FeatureC++
  • combination of AOP and FOP
  • Approach:
    • improvement of mixins to cope with optional features by introducing AOP concepts to mixins
    • refinements with the keywords before, after, around are optional
Optional Weaving

- AspectJ
  - not usable for optional weaving in the current state because of
    - the lack of referencing optional classes, methods or member variables in optional advice statements resulting in code replication
    - this approach only for advice statements and not for inter type member declarations
Optional Weaving

• AspectJ
  • not usable for optional weaving in the current state because of
    • the lack of referencing optional classes, methods or member variables in optional advice statements resulting in code replication
    • this approach only for advice statements and not for inter type member declarations
  • need to overcome these lacks because of
    • avoiding the need of creating derivative features
    • implementation of optional extension within the genuine feature ⇒ maintain locally
Conclusion

- optional weaving is a promising approach
  - optionality is important for software product lines
  - derivative approach will scale according to the derivatives
- within FeatureC++ optional weaving implemented for two features
- with AspectJ optional weaving leads to code replication and unnecessary code for runtime semantics ⇒ improvements to the language must be made
Thank you for your attention!