Identifying Semantically Cohesive Modules within the Palladio Meta-Model

Symposium on Software Performance: Joint Descartes/Kieker/Palladio Days 2014
Misha Strittmatter, Michael Langhammer
Context

- Palladio refactoring
  - Vision: provide holistic extension mechanism
- Focus now: Palladio Metamodell (PCM)
Problem

- PCM’s original focus: performance
  - later extended by further content
- Not designed for extensibility
  - Extensions implemented inconsistently
  - Package structure partly aligned with submodels
- Dependencies not constrained
  - Insufficient modularization
→ Hard to
  - understand
  - evolve
Idea

- Modularize the metamodel
- In a layered fashion
Benefit

- Reduce technical debt (wrt. future extensions)
- Separation of concerns
  - Reduction of complexity
  - Understandability & maintainability
- Extensibility
- Very specific content factored out
  - Usability
  - Quality independent ADL
  - Flexibility & more general
→ PCM as a platform/framework
Goal of this Presentation

- Present the results of a metamodel inspection
- Not to propose a definite modularization
- But concerns contained in the PCM
- To form a basis for further discussion
- A concern may be mapped to
  - Metamodel module
  - Package
Procedure

- Starting point: our top down understanding
  - Coarse grained concerns
  - Regarded every class
  - Mapped to concern
  - Dependencies

Identifying Semantically Cohesive Modules within the Palladio Meta-Model
Current Package Structure

General information:
- Packages: 20
- Classes: 147
- References: 284
- Containment: 89

 Dependencies not shown:
- Transitive
- To/from inner packages
Identifying Semantically Cohesive Modules within the Palladio Meta-Model
Identifying Semantically Cohesive Modules within the Palladio Meta-Model
Identifying Semantically Cohesive Modules within the Palladio Meta-Model
Identifying Semantically Cohesive Modules within the Palladio Meta-Model
Identifying Semantically Cohesive Modules within the Palladio Meta-Model
Identifying Semantically Cohesive Modules within the Palladio Meta-Model

Component Type Hierarchy → Repository → DataTypes

Composition

SEFF

Resource Interfaces

Resource Types

Resource Environment

RDSEFF

Performance

Infrastructure
Identifying Semantically Cohesive Modules within the Palladio Meta-Model
Identifying Semantically Cohesive Modules within the Palladio Meta-Model
Lessons Learned

- Bad modularity impedes understandability
  - Grouping of multiple concerns in one package
  - Distribution of a concern over multiple packages

- Packages do not constrain dependencies
  - Problematic, if core concerns are dependent on extending concerns
Conclusion

- Binding conventions for extensions needed
- Extensions should internally reflect which parts are extended
- Provide preferred metamodel extension techniques
- Clearly defined extension -> framework
BACKUP
Refactoring Challenges from last SoSP

- Feasibility of metamodel extension & editor fragments via plugin
- Compositional transformations
- Extensible simulators
- ADL forming & scale invariance
- How to slice the metamodel
- Bidirectional relations