Specifying and Analyzing Program Refactorings With AGG

Javier Pérez, Olga Runge, Gabriele Taentzer

Universidad de Valladolid, Spain
Technische Universität Berlin, Germany
Philipps-Universität Marburg, Germany
Program Graph

- complex abstract syntax structure
- GXL import/export
- type distinction by colors
- visualization options:
  - *zooming*
  - *hiding of attributes and types*
Program Graph - Simplified
Specifying Program Refactoring

- Refactoring Preparation
  - transitive closure of hierarchies (simulate path exp.)
Specifying Program Refactoring

- Refactoring „MoveMethod“
  - enabling
  - preparation
  - refactoring action
  - updating
  - clean up

- Metrics:
  - 17 rules
  - 2 - 12 object nodes per rule
  - no control flow elements
  - 5 rule layers

Pérez, Runge, Taentzer

AGG – http://tfs.cs.tu-berlin.de/agg
Specifying Program Refactoring

- Enabling „MoveMethod“

- no method with same name
- no call to super class
Specifying Program Refactoring

- Performing „MoveMethod“
Specifying Program Refactoring

- Update program structure
Specifying Program Refactoring

- Update program structure
Analyzing Program Refactoring

- Potential dependencies between rule applications
Analyzing Program Refactoring

- Potential conflicts between rule applications
Conclusions

- useful for refactoring designers
- simple definition language
  - rule-based, no code
  - simple control structures for rule applications
  - no path expressions
- analysis facilities
  - potential dependencies and conflicts
  - different forms of applicability checks
  - termination check