Towards a Refactoring Benchmark

Serge Demeyer
Lab on Reengineering (LORE)
University of Antwerp
Presentation for the ELISA - Workshop
(September 2003, Amsterdam - The Netherlands)
Story #1

Test provider type

Transform Client Type Checks

Test self type

Transform Self Type Checks

Test external attribute

Transform Conditionals into Registration

Test object state

Factor Out Strategy

Factor Out State

Test object state

Introduce Null Object

Test null values

© Serge Demeyer
Story #2

Boxes: Classes
Width:  # methods added
Height: # methods overridden
Color:  # method extended
Story #3

Split $B$ into $X$ and $B'$

/* Hierarchy nesting level increased */
(delta_HNL(B') > 0) and

/* Number of methods decreased */
((delta_NOM(B') < 0)

/* Number of attributes decreased */
or (delta_NOA(B') < 0))
Classification

Curative
(i.e. Which refactorings are good?
How do tools support refactoring?)

Predictive
(i.e. Where to apply
Which Refactoring?)

Retrospective
(i.e. Which Refactorings have been Applied?)
## Benchmark proposal

**Characteristics**
- Life Cycle
  - (analysis, design, ...)
- Evolution
  - (scale, #iterations, ...)
- Domain
  - (problem, solution, ...)

### Case studies
- Toy Example (LAN - Simulation)
- Industrial System (VisualWorks & Swing)
- Public Domain (HotDraw & ET++)
- Open-source (Mozilla)
Case Study: LAN Simulation

Add functionality

Refactor
Case Study: LAN Simulation

Curative?
- Version 0.x is "better" than version 0.x-1?
- Does tool P support 0.x to 0.x+1?

Predictive?
- Does technique Q predict 0.x to 0.x+1?

Retrospective?
- Does technique R discover 0.x to 0.x+1?
Discussion

- Does it makes sense to work out this LAN benchmark?
  - Would you use it?               o yes   o no