Detecting overly strong preconditions in refactoring engines

ABSTRACT

Refactoring is the process of changing a program to improve its internal structure while preserving its observable behavior. Refactoring can be applied manually, which may be time consuming and error prone, or automatically by using a refactoring engine, such as Eclipse, Net Beans and JastAdd Refactoring Tools (JRRT). These engines contain a number of refactoring implementations, such as Rename Class, Pull up Method, and Encapsulate Field. For correctly applying a refactoring, and thus, ensuring behavior preservation, the refactoring implementations might need to consider a number of preconditions, such as checking whether a method or field with the same name already exists in a type.

EXISTING SYSTEM

In Existing System, Refactoring engines may have overly strong preconditions preventing developers from applying useful transformations. We find that 32% of the Eclipse and JRRT test suites are concerned with detecting overly strong preconditions. In general, developers manually write test cases, which is costly and error prone. Our Existing technique detects overly strong preconditions using differential testing. However, it needs at least two refactoring engines

DIS ADVANTAGES

- It is time consuming and error prone.
- It rejects transformations.

PROPOSED SYSTEM

In Proposed System, we propose a technique to detect overly strong preconditions in refactoring engines without needing reference implementations. We automatically generate programs and attempt to refactor them. For each rejected transformation, we attempt to apply it again after disabling the preconditions that lead the refactoring engine to reject the transformation. If it applies a behavior preserving transformation, we consider the disabled

preconditions overly strong. We evaluate 10 refactoring of Eclipse and JRRT by generating 154,040 programs. We find 15 overly strong preconditions in Eclipse and 15 in JRRT. Our technique detects 11 bugs that our previous technique cannot detect while missing 5 bugs. We evaluate the technique by replacing the programs generated by JDOLLY with the input programs of Eclipse and JRRT test suites.

ADVANTAGES

- It detects overly strong preconditions in refactoring implementations.
- It prevents a warning or error message.

SYSTEM REQUIREMENTS

H/W System Configuration:-

Processor	-	Pentium –III
RAM	-	256 MB (min)
Hard Disk	-	20 GB
Key Board	-	Standard Windows Keyboard
Mouse	-	Two or Three Button Mouse
Monitor	-	SVGA

S/W System Configuration:-

Operating System	:	Windows95/98/2000/XP
Application Server	:	Tomcat5.0/6.X
Front End	:	HTML, Jsp
Scripts	:	JavaScript.
Server side Script	:	Java Server Pages.
Database	:	MySQL 5.0

Database Connectivity : JDBC