Research on Support tool for Software Development using a History of Refactoring Operations

Katsushi Takahashi (410072)

School of Information Science,
Japan Advanced Institute of Science and Technology

February 9, 2006

Keywords: refectoring, history of operations, metrics.

Progressive extention on specifiation and function of software makes softwares more and more complex, hard to read or maintain. Refactoring[Fowlor] is one of the technique for regaining its possilbility for maintenance. It is a general technique for changing structures of software without changing its semantics. However, it is not enough popular to developers in actual fields.

One of the reason is its low granularity. In order to regain the readability or maintenance, it requires an application of series of refactoring operations, especially large and complex software. Moreover, the result of refactoring depends on the order of application of each operations. Novice programmers cannot find the way to successful route to their goals. Though they will do so many times of trial and errors, few of traditional supporting tools can handle its record of operations. The goal of this research is to propose a new development method using a supporting tool which can record and manage continuous change of source codes by the form of refactoring operation histories.

The largest characteristics of this tools is it can keep the record of opeartions in tree. Each operations are stored as a node of the tree. We choosed Java as the target language, and extract 19 primitive operations from [Fowler] and give a formal definitions as mapping from a program structure to another. It is called primitive operations(PO), consists of

identifier for kind of operations, location and argument which depends on the operations. Location is a tuple of (package name, class name, method name, relative position). Argument is, for example, in the case of "extract method", the new name of the method.

Combination of PO is called Refactoring Plan(RP), it consits of series of PO and the goal which must be achieved just after the operation are applied. User can see the history and result at a glance of RP, and detect whether if an operation fails, or it seems to be failed or not. Fail means that is cause no effect or might lose an effort which gained by previous application of refactoring operations. RP allows us to compare the result of more than one application of refactoring, and mininize the effort for us to reach out indented goals. The reason why we keep not only the result but also a history is that has many opportunity to branch the history, and more number of candidates can be compared simultaneously.

In order to detect the location which an operation must be applied, metrics is useful. It gives many kinds of metrices, usable both detection and evaluation of each refactoring operations. Especially for novice programmes, metrics might be helpful to find the strategy of refactoring. Also it helps us to compare many candidates of refactoring strategies by some concrete evidences.

We proposed a new methodlogy for development with a tool for handling history of refactoring. It can show the result and history which reaches to in the same time. And also helps the user to compare many strategies. It seems to useful for makes refactoring to easy for actual developemnt field and reduce the cost of software developments.