Unifying Code Refactorings of Different Languages

Introducing CHAST, using Parser Generators as a Refactoring Meta Language

Graduate

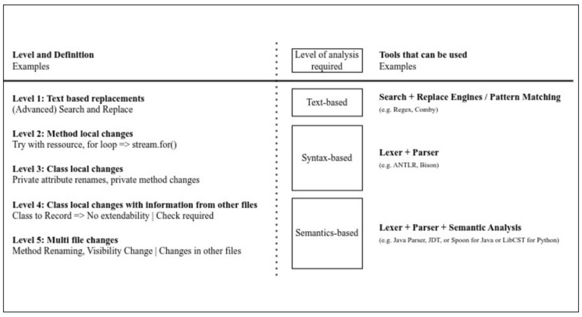
Raphael Jenni

Introduction: In every software project, the need for refactoring emerges sooner or later. Refactoring is the process of restructuring existing computer code without changing its observable behavior. It is intended to improve the design, structure, or implementation of the software to make it more maintainable, scalable, and efficient. Refactoring is a well-known technique in software engineering and is widely used in practice. However, much refactoring work is still done manually, which is time-consuming and error-prone. Many refactorings turn out to be repetitive and very similar to one another. Some refactorings may be automated and aided by IDEs. For others, no automated support is available. Some projects aim to provide automated support for refactoring.

Approach: CHAST, a tool that automates refactorings in a secure and reliable manner by using the concept of change isolation is proposed and prototyped. The tool allows users to preview and confirm changes before they are applied to the actual codebase, avoiding unintended side effects and enabling users to easily roll back changes if necessary. CHAST is designed to be language-agnostic and to support a wide range of tools and functionality. It also simplifies the process of creating and sharing refactorings by providing a command-line interface and recipe format. The project furthermore investigates the feasibility of implementing multi-language refactorings using parser generators as a refactoring meta language. It also evaluates CHAST in terms of change isolation and the effort required to create a refactoring.

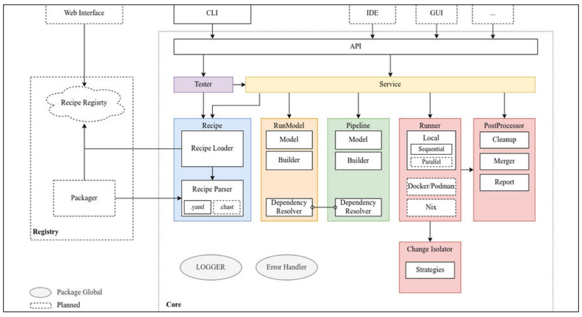
Conclusion: In conclusion, the project presents the feasibility of multi language refactoring and explains how it can be achieved through the use of parser generators as a refactoring meta language. This approach has the potential to significantly reduce the effort required to implement and maintain refactorings for multiple languages, as well as improve the user experience by providing a unified interface for accessing these refactorings. Thus, it highly contributes to the efficiency and effectiveness of developers in their strive to continuously improve their software.
As part of this research project, the CHAST tool was developed and introduced, which aims to address the challenges of finding and using refactoring tools by providing a unified interface for accessing refactorings and other code-related tools, as well as a packaging system for distributing these tools. CHAST is designed to be language-agnostic and to support a wide range of tools and functionality. While CHAST is still in its early stages of development, it has the potential to significantly improve the accessibility and usability of refactoring tools, as well as facilitating the development of new refactorings and code-related tools.

Classification of Refactoring Levels
Own presentment

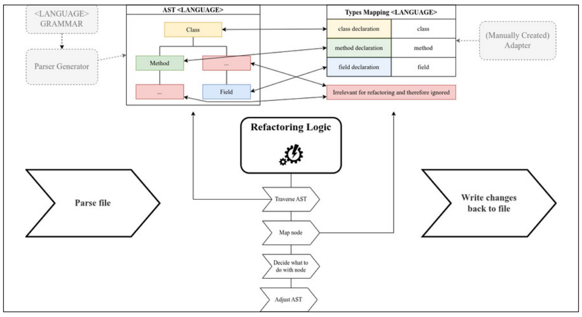


Level and Definition Examples	Level of analysis required	Tools that can be used Examples
Level 1: Text based replacements (Advanced) Search and Replace	Text-based	**Search + Replace Engines / Pattern Matching** (e.g. Regex, Comby)
Level 2: Method local changes Try with ressource, for loop => stream.for()	Syntax-based	**Lexer + Parser** (e.g. ANTLR, Bison)
Level 3: Class local changes Private attribute renames, private method changes		
Level 4: Class local changes with information from other files Class to Record => No extendability | Check required	Semantics-based	**Lexer + Parser + Semantic Analysis** (e.g. Java Parser, JDT, or Spoon for Java or LibCST for Python)
Level 5: Multi file changes Method Renaming, Visibility Change | Changes in other files		

Architecture of CHAST
Own presentment



Using Parser Generators and Mappings for Refactorings
Own presentment



Advisor
Prof. Dr. Farhad D. Mehta

Co-Examiner
Benno Baumgartner, Siemens Mobility AG, Wallisellen, ZH

Subject Area
Software and Systems, Computer Science