Abstract

SeisSol is a scientific software for the numerical simulation of seismic wave phenomena. However, there are three main problems in the SeisSol development project. First, the project documentation is incomplete. Second, the source code comprehensibility is low. Third, the dependencies between the modules in the system are complicated. To solve the problems and to enhance the software quality, we perform a reengineering process on SeisSol. The process contains four steps, reverse engineering, requirements reengineering, redesign and source code refactoring. In the requirements reengineering step, we employ a novel approach to elicit requirements efficiently for such a scientific computing project.

Through the reengineering process, the documentation of source code, requirements and the improved design is generated, the system is more modularized and easier to be extended, as well as the source code are more comprehensible. We also discuss the lessons learned during the reengineering process.

SeisSol Project

SeisSol Software
- Numerical simulation of seismic wave phenomena
- Supports seismological research to understand observed data from the field
- Supports the determination of the source characteristics and geological surface structure
- Applies discontinuous Galerkin finite element method
- Parallel computing is achieved by employing Message Passing Interface (MPI)
- Successfully run on SGI Altix 4700 and IBM Bluegene P supercomputers up to ~50,000 cores

SeisSol Development
- Started from 2006 and 5 to 10 developers have been working on the software
- Main programming language is Fortran 90
- Approximately 90,000 lines of executable code
- Automated overnight compiling and validation with basic test cases

Main Problems
- The project documentation is incomplete
- The source code comprehensibility is low
- The dependencies between the modules are complicated

In other words, the comprehensibility and maintainability of the SeisSol software shall be improved.

Solution
We perform a reengineering process on SeisSol to mitigate the above mentioned issues.

Reengineering

1. Reverse Engineering
Reverse engineering is the process of analyzing an existing system to create representations of a higher level of abstraction.
- Doxygen - a source code documentation generator tool, to generate a document that includes lists of functions and data, as well as call graphs
- Information retrieval techniques, to index and extract the information of the functions and subroutines in the Fortran source code

2. Requirements Reengineering
Requirements define what the software is expected to do, and are also used as important input into other development stages, such as design, coding and testing.
- Use the extracted information of the existing SeisSol functions
- Apply a novel requirements modeling approach

3. Redesign
Based on the SeisSol requirements model instance, we redesign the system. Especially, we want to clearly divide the functional components of the system and to develop hierarchies to deal with the complex parts.
- Group into packages

4. Source Code Refactoring
Refactoring is the process of changing a software system in such a way that it does not alter the external behavior of the code, yet improves its internal structure (Fowler et al., 1999).
- Modularization: The source code is grouped into packages to make dependencies among different components explicit and clear. Fortran modules are already well used in the SeisSol software. We then group external procedures into a set of modules and give them explicit interfaces.
- Move functions: We move the functions to the modules, where the functions can be used better or in a more logical way than in the modules they are defined, without increasing the computational complexity.
- Extract functions: A large function is broken into smaller pieces of functions.
- Clean untouched or duplicated code: Untouched and duplicated codes are detected. We analyze the rationale of having these codes and handle them accordingly. For example, duplicated code can be extracted to a new function and called from different places.
- Code optimization: To improve the performance and efficiency of the code, code optimization especially parallel optimization is also carried out.
- Testing and performance analysis: Testing and performance analysis are performed during the whole procedure of refactoring to avoid incorrect refactoring and negative impact on the performance.

Conclude

We presented an ongoing work of reengineering the SeisSol software. Up to this time, the source code is easier to comprehend, the system is more modularized and becomes more extensible. The recovered and refactored software deal with the complexity of the software and help the developers to communicate.

Further investigations on the quality of the reengineering and the improvement of comprehensibility and maintainability shall be conducted in the future.

References

Prerequisite - A Domain Specific Requirements Model

We employ this model to elicit and manage requirements for SeisSol (Li et al., 2011). The model provides domain specific abstractions and notations to bridge the gap between the scientific computing domain and the software engineering domain.

1. Scientific Knowledge Modeling
2. Software Requirements Modeling