《Design recovery and maintenance of build systems》Review

Paper Info

《Design recovery and maintenance of build systems》 Bram Adams, Herman Tromp, Kris De Schutter, Wolfgang De Meuter

Major Contribution

The paper introduces an approach to make the build's dependency graph available in a tangible way. The major contributions of their work include:

- Visualize the build's dependency graph
- Enable powerful querying of all build-related data, and various filtering techniques to define views on the build architecture.
- Validate rules after refactoring the build

The paper is a tool paper on build system visualization, query, refactoring and validation. This tool in this paper is useful in many later studies, and help researchers to have a more clear view of build system.

Main Work

The authors implement the tool named MAKAO to visualize the build systems, and design a DSL to support the build-related data querying, factoring and validation. The ideas are very interesting. More importantly, their tool is useful and this work has been referenced by many researchers.

The key insight of this work is simple and straightforward. As other similar work, they construct the CDG(Concrete Dependency Graph) at first, and CDG is the basic structure in the visualization phase. Some refined operations such as filtering and reducing make the visualization easier for researchers to find new structures in the build systems.

Other applications include refactoring and validation are also important parts of this work. However, in my personal view, the validation part should not be implemented by writing checkers in DSL code. This will be explained in the next section.

Overall, this work is a good tool demo paper, and MAKAO really helps researchers a lot to find some meaningful results in build systems.

Some Criticism

This work is the first work to propose a method of build system visualization and all build-related data querying. The tool MAKAO is practical in the research field, and helps researchers find plenty of other interesting results. However, it has several limitations and needs to be discussed and improved.

- The authors design a DSL to support all build-related data querying. In the section 6.5, they give an example to show how to use the DSL to find unused targets. The example is nice, but obviously, the

problem itself is limited, and many excessive perquisite issues can not be detected by their checker. The root cause is that the expressive power of the DSL is limited. Some other issues except excessive perquisite issue can not be detected.

- The tool can only analyze the build scripts in build systems. Although this is enough in build systems debug, some other analysis upon configuration files and products of systems should also be included. The problem which the author are concerned about is the maintenance of build systems, so it is obvious to consider these files besides build scripts.
- Because they have achieved the visualization of build systems, it is a natural way to find some dependency issues just based on some graph algorithms performed on the results of the visualization. However, they choose to write the checkers in DSL, and this really limits the power of checkers, for the DSL itself may have several restrictions and can not express the semantics of some issues.