Review 10

2/20/12

CS6V81.502

Seungtack Baek

**Using Dependency Models to Manage Complex Software Architectures**

In this paper, Sangal et al. used a tool named LDM (which was never explained what it stands for) to detect design rule violations. They first explained how dependency structure matrix (DSM) works and then, demonstrated how LDM utilizes DSM to find out software’s design rule. To show how LDM works, the researchers chose an ongoing project (or ended, yet never released) from MIT, Haystack, to evaluate how LDM analyzes architectural design of the project. As a result, they were able to find some modules which did not conform to the design rule.

As they said, there are two main drawbacks for this approach. The first drawback is in cognitive overhead that matrix inherently imposes on us. If you have too much rows/columns, it is hard to read each cell’s row/column position. Yet, they refuted this drawback with the fact that the researchers’ ability to read row and column from a cell became better. The second drawback is in that this tool, LDM, cannot recognize orthogonality of a project, which can lead to LDM’s inefficiency.

Speaking of efficiency, in their conclusion, Sangal et al. mentioned that LDM provides “lightweight enough” approach to analyze the design rule of a project “in practice”. Yet, what I wanted to see was something like “how fast it runs” or “how many classes (corresponds to number of modules) are there for given LOC”? Answering these questions might help us to get an idea about what to expect on how long reviewing a project’s architecture will take.

**Question:**

1. This paper looks extremely similar to the paper we just read. What is their “novelty”