

# Multi-View Architecture Trend Analysis for Medical Imaging

Tobias Röttschke  
Philips Research Laboratories  
Prof. Holstlaan 4  
5656 AA Eindhoven, The Netherlands  
tobias.roetschke@philips.com

René Krikhaar, Danny Havenith  
Philips Medical Systems  
Veenpluis 4-6  
5684 PC Best, The Netherlands  
{rene.krikhaar,danny.havenith}@philips.com

## Abstract

*We show how two web-based architecture trend analysis tools, considering different views and their changes over time, contribute to our daily effort to cope with the complexity of a large, software-intensive medical imaging system.*

## 1 Summary of the presentation

Philips Medical Systems develops large SW-intensive embedded imaging systems, comprising hundreds of hardware and software components. To be competitive in the market, the development process must meet several demanding and often conflicting requirements: high product quality, short time-to-market and low costs. Success depends, among other things, on an appropriately managed software architecture.

Software architecture can be described by four related views [2], i.e. conceptual, module interconnection, code and execution architecture. Software architects take the responsibility for specifying these views correctly and keeping them consistent.

To manage the software architecture, architects and designers continuously need up-to-date information about ongoing activities, problems that arise, and recent progress. This information can be related to any of the views mentioned above, and must be structured in such a way that navigation from general to more detailed information is possible and dependencies between different views can easily be traced. Considering the huge amount of available data, adequate tool support is needed to collect, process and present the required information.

We describe first experiences with two trend analysis tools, which are integrated in the daily software development process. One is the Code and Module Architecture Dashboard (CMAD) for monitoring changes of different code, module, and interconnection metrics organized according to the module hierarchy. The other, called Execu-

tion Architecture Dashboard (EXAD), helps to keep track of changes in system performance and to relate them to recent modifications in the code architecture.

Unsurprisingly, the CMAD tool has revealed a couple of architectural inconsistencies immediately after introduction: the code archive contains code that does not belong to a specified subsystem and, vice versa, concurrent versions of the same interface files have been found at different locations, and a subsystem supposed to be self-contained has some unintended dependencies on application subsystems, which could be traced to include directives in the code.

Fixing most problems was a matter of weeks and could be monitored and triggered by the software architect. Via the intraweb, unsolved problems are reported to all developers. This helps responsible persons to keep an overview of open tasks and to assign adequate priorities.

The EXAD tool is used to analyze the cause of unexpected performance decreases. Comparing performance information of actual and historical test runs makes it possible to quickly narrow the range of possible reasons, and by combining execution and code architecture information, it is possible to judge whether recent code changes caused the performance decrease.

Stimulated by the initial feedback from our developers, we are currently extending the amount of information and introducing additional modes of presentation. In particular, we plan to add statistics about violations of architectural rules [1] to monitor the consistency of the architecture.

## References

- [1] R. Krikhaar. *Software Architecture Reconstruction*. PhD thesis, University of Amsterdam, 1999.
- [2] D. Soni, R. Nord, and C. Hofmeister. Software architecture in industrial applications. In *Proc. 17th International Conference on Software Engineering*, pages 196–207. ACM Press, 1995.