

## DASHBOARDS: A BRIEF HISTORY OF DART

Lorensen, Bill<sup>1</sup>; Blezek, Dan<sup>1</sup>  
<sup>1</sup>GE Research, Niskayuna, NY

For any large software project to be successful in the long term, it is imperative to develop a testing infrastructure to monitor the quality of the system as it develops and matures. In general, developers of such systems are averse to this requirement. As a result, code begins to become stale, and developers no longer understand the ramifications of their changes, and are unable to refactor code from the past. So they press on in the hopes that their foundational code does not crumble beneath them. NAMIC seeks the higher ground. The following is an account of the history of our software quality system called Dart.

In 1997, General Electric added a new quality initiative, called Six Sigma. At GE Research, we focused a collection of our quality projects on the development of one of our software toolkits, the Visualization Toolkit or VTK (<http://www.vtk.org/>). We integrated our original 14 projects into an automated system that collected their outputs and integrated them into an online, daily VTK dashboard.

In 1999, the National Library of Medicine commissioned the development of an open source, cross platform project called the Insight Segmentation and Registration Toolkit, ITK (<http://www.itk.org/>). As part of GE Research's contribution to the ITK effort, GE Research developed the first version of Dart. Dart's goals were

- Remove the dependence on tcsh, awk, sed and cron scripts to perform a build and test sequence
- Allow testing machines from around the world to submit test results to a Dart server
- Separate the data from its presentation
- Apply the Dart testing system to a variety of project (ITK, VTK, VXL)
- Make the testing system itself open source

Dart met its original design goals and was successfully applied to many software projects. Dart clients were easy to use and allowed for testing machines to be distributed around the world. The Dart server allowed anyone to view the results of a test sequence and monitor the software development process. Dart allowed a cross-platform system to be tested in multiple configurations, though the server side of Dart was still difficult to maintain.

In 2004, NIH sponsored the National Alliance of Medical Image Computing, NA-MIC (<http://www.na-mic.org/>) as part of NIH Roadmap for Medical Research, Grant U54 EB005149. GE Research is developing the next generation of Dart as part of NA-MIC. The goals remain broadly the same, however, two new goals have been identified

- Simplify the Dart server setup and maintenance
- Allow for longitudinal or temporal analysis of test results

To this end, we introduce the new version of Dart. We affectionately refer to the previous version of Dart as Dart Classic. The new Dart still accepts build/test results in the Dart Classic format. Dart has been completely rewritten in Java. It uses an embedded web server and servlet engine (Jetty) and an embedded database (Derby). XML-RPC is used to transmit build/test results to the Dart server. Dart is distributed as two jar files. The first jar file, DartServer.jar, contains everything to create and run a Dart server managing several Dart projects. The second jar file, DartClient.jar, is a small utility to shutdown a server, refresh its resources, query its status, and can be used as an XML-RPC messenger.

Dart has recently passed the 1.0 release milestone and is moving into scale-up testing and debugging for broad adoption across the multitude of NAMIC software projects. More information may be found on the Dart homepage: <http://na-mic.org/Wiki/index.php/Dart2Summary>