

Distributed Computing With Triana, A Short Course

Ian Taylor, Matthew Shields and Ian Wang

What is Triana?

Triana (<http://www.trianacode.org>) is an open source distributed workflow environment for constructing applications from components. Originally designed to be a quick look analysis system for signal processing, Triana has evolved into a fully featured distributed, Grid-enabled, Web service aware, multi-domain Problem Solving Environment.

Why Should I Care?

Triana was designed to be simple enough for a child to use, but complex enough to be of real use to scientists. In its current incarnation it provides straight forward access to distributed computing technologies such as, Web services: including service discovery, service choreography and service publishing; Grid computing: including support for Globus (<http://www.globus.org/>) and the GridLab GAT (<http://www.gridlab.org/>) for job submission, file and data handling, and security; without the need writing code.

The ease of use, together with powerful workflow composition, Grid and Web service capabilities, make Triana a useful tool for anyone wanting access to distributed computing.

Distributed Components

Distributed components within Triana can be split into two categories, which we refer to as Grid-oriented components and service-oriented components:

Service-Oriented Service oriented components, such as Web services and P2PS services, are network accessible components that exist on remote computing

resources. These components can be discovered by Triana and then invoked from within a Triana workflow

Grid-Oriented Grid-oriented components represent jobs launched by Triana on a remote machine using a Grid resource manager, e.g. GRAM or GRMS. Unlike service components, these jobs do not typically have network accessible interfaces so communication is only available via input/output files.

Triana uses different interfaces to access these two categories of distributed component. For service-oriented components the Grid Application Prototype (GAP) Interface [1] is used, while for Grid-oriented components the GridLab Grid Application Toolkit (GAT) API [2] is used. Both these interfaces have multiple bindings which allow different service/Grid middleware to be employed without amending the Triana application code. For service-oriented components, Web services and P2PS services can currently be invoked, while for Grid-oriented components, we use the Java GAT interface which gives us access to a number of underlying bindings for the various capabilities. We use GridFTP for file copying and GRAM/GRMS etc for job submission.

Course Content

In this short course, we will provide a hands on demonstration for the use of Triana in both service-oriented and Grid-oriented environments. Help will be available to install Triana, requires Java 1.4 or later.

Service-oriented

This example is interactive (subject to technical constraints ...). Participants can install and set-up Triana in preparation following the guidelines in the user manual at www.trianacode.org. This part of the course will describe how remote services can be launched using the P2PS binding of the GAP interface. The users will be able to use the P2P discovery to discover the other participants attending the course and then launch a service remotely on this machine. For this, we will use the audio toolkit to launch a "Play" unit on the remote machine to dynamically play an audio file on another persons laptop. This somewhat fun example, however will demonstrate the ease in which any Triana unit (whether it be an audio component or complex iterative solvers) can be launched across the network.

In this section we will also cover the use of Web services within Triana. We will demonstrate how we can discover Web services using a UDDI server, compose the Web services within workflows together with Triana components to choreograph complex Web service interactions.

Grid-oriented

This example will involve a complex Grid workflow which will stage files onto the GridLab testbed (using GridFTP), then submit a job (Cactus) on a remote resource. Once running the output files from this job will be fed back into the the Triana workflow and connected to appropriate visualisers depending on the file type, for example image viewers, graph renderers etc. The remote Cactus simulation will also be dynamically steered in order to change the “orbit radius” parameter, which will result in a change in the realtime visualisations. This example is meant to demonstrate the concept of representing Grid primitives, such as *files* and *jobs*, graphically within complex workflows. The demonstration shows an integrated application involving a number of common Grid operations interacting together, including job submission, data management, monitoring, information services, security and adaptive components.

Who Should Attend?

This course is suitable for anyone interested in workflow based Grid solutions. There is no programming content and Grid concepts will be explained in context with Triana.

References

- [1] Ian Taylor, Matthew Shields, Ian Wang, and Omer Rana. Triana Applications within Grid Computing and Peer to Peer Environments. *Journal of Grid Computing*, 1(2):199–217, 2003. [\(document\)](#)
- [2] Gabrielle Allen, Kelly Davis, Konstantinos N. Dolkas, Nikolaos D. Doulamis, Tom Goodale, Thilo Kielmann, André Merzky, Jarek Nabrzyski, Juliusz Pukacki, Thomas Radke, Michael Russell, Ed Seidel, John Shalf, and Ian Taylor. Enabling Applications on the Grid: A GridLab Overview. *International Journal of High Performance Computing Applications: Special Issue on Grid Computing: Infrastructure and Applications*, 17(4):449–466, November 2003. [\(document\)](#)