

Working on the CSEGrid

CSE 487/587
Mar 21, 2005

Infrastructure

- The CSEGrid is a 4 node Linux-based grid.
- We call this network of machines, a "grid", since we've installed the Globus toolkit on it (ver 3.0.2)
- You will log in remotely to either one of the following nodes depending on the one that you have been already allocated (via email)
 - `cerf.cse.buffalo.edu`
 - `mills.cse.buffalo.edu`
 - `vixie.cse.buffalo.edu`
- Access to these machines is restricted from off-campus. If off-campus, connect through VPN.

Setup

- Follow instruction sheet to setup your `.cshrc` file with the appropriate environment variables, etc.
- You need to request for a grid certificate from the administrator (Ken Smith, who else?). Refer to the instruction sheet.
- A few sensible aliases would be helpful in navigating through directories.

Working with the Globus Architecture

- Globus can be thought of as a web server that can host multiple (grid) services.
- The architecture can be thought of as a form of RPC.
- "Services" contain methods that can be remotely called by "Clients"
- You will write up services, clients and deployment descriptors for your services.

Building your service: `build.sh`

- This script (part of Sotomayor's tutorial) invokes the ANT tool
- The ANT tool uses the `build.xml` file to carry out the following tasks:
 - Compiles service
 - Generate stubs
 - JARs the service-related files together, making it deployable
- This script takes as input:
 - The directory of the service implementation
 - The `.gwedl` file
- It outputs:
 - GAR

Service components

- GAR (Grid JAR), representing the deployable service, contains the following components:
 - A JAR containing the actual service implementation
 - A JAR containing service stubs (for RPC)
 - Deployment files
- GWSDL is a modified version of WSDL (as in, Web Services Definition Language) for the Grid

Deploying the service

- The service (GAR) needs to be deployed onto the "Grid" (Globus).
- This is where the deployment descriptor comes into the scene.
- You are not given access to deploy services explicitly.
- Deploy using an automated script. (As per instruction sheet).

Starting the container

- When you deploy a service onto one of the nodes, your service is accessible to everyone else who uses that node.
- Next step, is starting up the container. This is equivalent to starting up the "server".
- When starting up the "server", you need to specify the port number on which the container should listen for connections from clients.

Starting the container

- Note that even though the deployed services are common across all users of a node, the container is a process that you own. It can only be stopped by you.
- However, as all servers, it listens to remote connections. Hence, anyone's client can connect to it.

Compiling and executing the client

- Once you've started the container, you should `source` the script `$GLOBUS_LOCATION/setenv.csh` to add the newly added service files to your `CLASSPATH` environment variable.
- After this, you will be able to compile and execute your grid service client.

Executing a remote client

- Grid services are supposed to be executed remotely.
- To access your grid service (hosted on the CSEGrid) from outside, chiefly, you need the stub classes accessible to the client.
- Follow the instruction sheet to setup this type of remote access.
- You will need to copy the JAR file containing the stubs in the `$CLIENT_DIR/lib` directory.

Try it out!

- Please try this out within this week.
- Next week (or earlier), you will have more sample code to work with.