CI-BER Testbed

Design

The testbed is a combination of a web dashboard and one or more java processes that test Brown Dog servers with files from the CI-BER collection. Test batteries will run continuously using the current test profile. The results are logged, analyzed and presented, so that problems can be identified by the Brown Dog development team.

The Testbed samples the files in the CI-BER collection to find those meeting a pre-defined test profile. Tests are run in parallel to simulate future server load and each individual test operation result is recorded in the database, as well as the overall performance of the test battery. We should be able to diagnose individual service failures and the performance of services under a known load.
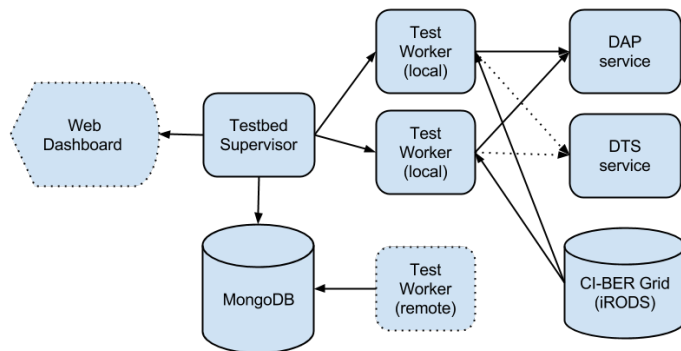
Technology

- Apache commons-daemon, Spring dependency injection
- Jargon iRODS client, Apache http client
- MongoDB used for data profile, test results and queuing
- JAX-RS controllers with UI in JQuery and D3

Test Profiles

The testbed runs a given set of operations, say many thousand, as fast as possible given the number of parallel processes allocated. The format and operations within a given set of tests, along with the number of parallel test workers are all part of a test profile logged in MongoDB. Currently the profile includes the number of files for each desired format, by extension.

The test profile seems like an important extension point:

- Specify both input and output formats for DAP.
- Specifying that test files met a minimum size criteria.
- What else??

Since the DAP conversions run through various backend software tools, it seems important that at least some test profiles fire up many converters in parallel to simulate real world conditions.

It also seems important to trace the performance of a given test profiles over time. So the web application will need to report out of that via the dashboard and/or JSON/REST. (in development)

Parallel Testing

In the current design there is a single daemon process, which will run on a VM at RENCI/UNC and perform tests in parallel threads. These tests consist mostly of network I/O, so we can probably run a lot of them in one java process on this VM.

The next step, when needed for performance evaluation, will be to add a test queue in the central MongoDB store. Then run test daemons on addition worker VMs.

We will need to pay some attention to the CI-BER IRODS host, to make sure it does not become the bottleneck, i.e. increase the number of server-side agent processes.

Code

On GitHub: https://github.com/CI-BER/BrownDog-Testbed