The Experiment Client

Michael Gohde

June 5, 2015

1 Overview

The Experiment Client provides an interface to the RSSE stack. While it would be best suited for integration into machine learning software stacks, this implementation of the Experiment Client provides a command line interface.

Since all of the protocols understood by the Experiment Client are detailed in their respective documents (CM and ESS), this document will contain an installation and usage guide for the Experiment Client as it is implemented here.

2 Installation

The Experiment Client should have been distributed as a .jar file. As such, all that it necessary is to either place it in a system directory (such as /usr/sbin) and build the provided wrapper, or to place it in the same directory as the current experiment. The latter method is preferable at the present time, as it allows the experiment client to write and dump data in a directory that is known and accessible to the current user.

If the first method is preferred based on the current setup and configuration of whichever machine is handy (the Experiment Client should be compatible with non-*NIX operating systems as it only hard-codes the directory "./" and can override this with directories specified by the user.

3 Command Line Arguments

As the Experiment Client as presently implemented is a command-line driven program, it has a multitude of possible arguments. All of the arguments will be listed in the table below, though it is recommended that the "Workflow" section be consulted before usage.

Argument	Description
-h, –help	Print a help message listing all command line arguments.
-i, –initial	Register with the specified experiment server.
–genconfig	Generates a configuration file using either defaults or the arguments provided.
-s, –server	Specifies the Experiment Server to connect to.
-p, –port	Specifies the port on the Experiment Server to connect to.
-l, –list	Lists all experiments that the Experiment server can serve.
-d, –usedir	Uses the specified directory instead of "./"
-n, –next	Fetch the next file from the server.
-a, –all	Fetch all files in the current experiment.
-cp, –cacheprt	Set the port to use on the Cache Module
-cs, –cachesrv	Set the Cache Module server to connect to.
-r, –respond	Send an XML response to the Experiment Server. *Not yet implemented*
-e, –expname	Experiment name to register with.
-g, –getinfo	Gets information about the selected experiment.

4 Usage and Workflow

From the command line options provided it is likely difficult to piece together a coherent idea of how to use the Experiment Client. This is so because several arguments need to be combined in order to produce a coherent command.

In general, based on this understanding, the following workflow is recommended:

1. Generate a configuration file with some base options.
2. List out all experiments on a server.
3. Get information on the 0 experiment.
4. Register the client for that experiment.
5. Fetch all URLs to catch up to the current state of the dataset.
6. Occasionally fetch next to stay updated/post results back.

4.1 Generating a Configuration File

Since it's excessively tedious and verbose to continuously specify all options for every single connection, the Experiment Client features the ability to generate a configuration file that will be read for every subsequent execution where no configuration arguments are passed.

Use the following template to complete this step:

```
java -jar ExperimentClient.jar -s servername -p port -cs cacheservername
-cp cacheport –genconfig
```

4.2 Listing all Experiments

This is one of the easiest steps. Be sure to copy and paste any selected experiment with no leading whitespace, as tab characters are printed leading into the name of each experiment.

java -jar ExperimentClient.jar -l

4.3 Getting Information

In order to get information about an experiment, all that should be done is as follows:

java -jar ExperimentClient.jar -e 'experiment name' -g

4.4 Registering For an Experiment

Now that you have selected an experiment, a modified version of the previous command is used to generate a registration file in "./registration.info". This file contains a randomly generated client ID number (used for the Experimental Server Service's own internal storage and retrieval) and the experiment registered for.

java -jar ExperimentClient.jar -e 'experiment name' -i

4.5 Fetching All URLs

If all has gone well so far, it's now time to fetch all of the data in the selected experiment. To do so, ensure that the Cache Module is running at the address specified (otherwise the Experiment Client may block for IO on the non-existent connection), then run the following command:

java -jar ExperimentClient.jar -a

Depending on internet traffic, the size of the dataset, and whether or not some of it is already 01, this operation may take a very long time. The Experiment Client will print out a message for every file fetched.

4.6 Periodically Checking for More URLs

Since datasets may change, it is worthwhile to occasionally check if more data has been added to a dataset. The Experimental Server Service automatically keeps track of which file your client is currently on, so this operation is very easy to perform.

java -jar ExperimentClient.jar -n

If there are no more files to fetch, the client will exit with return code 1. This allows scripts to be written to automatically update the current dataset.