

Building Scalable Distributed System

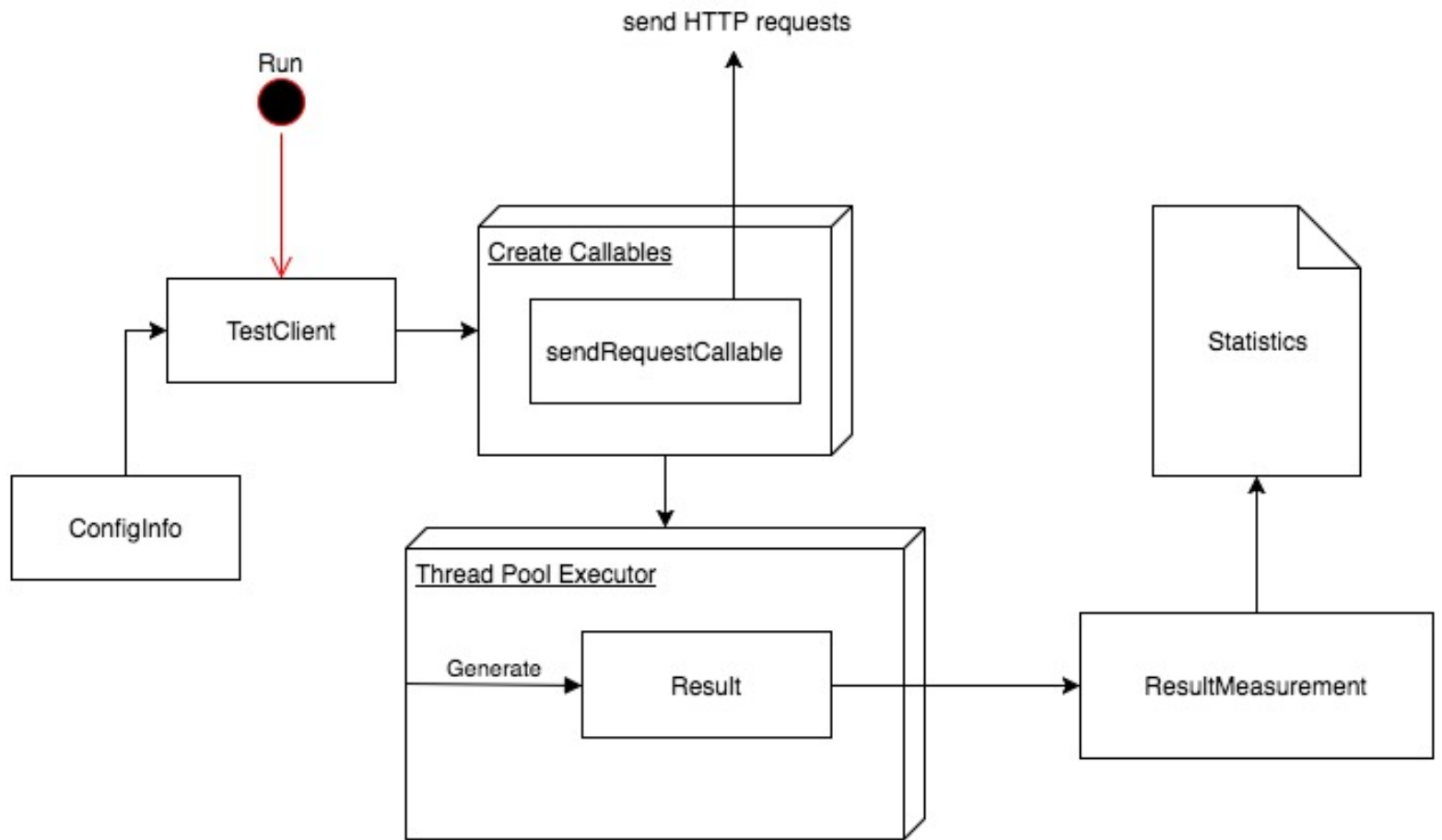
Assignment 1 Report

Xiyang He

he.xiy@husky.neu.edu

GitHubLink: https://github.com/hexiyang/BSDS_2017fall/tree/master/Assignment1

Design



URL to my git repo: https://github.com/hexiyang/BSDS_2017fall/tree/master/Assignment1

10 Threads, 100 Iterations

[illegible]

100 Threads, 100 Iterations

[illegible][illegible]

Step5:

10 Thread, 100 Iterations

100 Thread, 100 Iterations

```
*****
*           Statistic Results           *
*****
Wall Time: 5750 ms
```

Statistics for GET requests:

```
[   Total Requests]: 1000
[Successful Requests]: 1000
[   Failed Requests]: 0
[   Total Mean Latency]: 31 ms
[Successful Mean Latency]: 31 ms
[   Failed Mean Latency]: 0 ms
[   Total Median Latency]: 15 ms
[Successful Median Latency]: 15 ms
[   Failed Median Latency]: 0 ms
[   Total 99th Latency]: 15 ms
[Successful 99th Latency]: 15 ms
[   Failed 99th Latency]: 0 ms
[   Total 95th Latency]: 15 ms
[Successful 95th Latency]: 15 ms
[   Failed 95th Latency]: 0 ms
```

Statistics for POST requests:

```
[   Total Requests]: 1000
[Successful Requests]: 1000
[   Failed Requests]: 0
[   Total Mean Latency]: 25 ms
[Successful Mean Latency]: 25 ms
[   Failed Mean Latency]: 0 ms
[   Total Median Latency]: 15 ms
[Successful Median Latency]: 15 ms
[   Failed Median Latency]: 0 ms
[   Total 99th Latency]: 15 ms
[Successful 99th Latency]: 15 ms
[   Failed 99th Latency]: 0 ms
[   Total 95th Latency]: 15 ms
[Successful 95th Latency]: 15 ms
[   Failed 95th Latency]: 0 ms
```

```
*****
*           Statistic Results           *
*****
Wall Time: 16839 ms
```

Statistics for GET requests:

```
[   Total Requests]: 10000
[Successful Requests]: 10000
[   Failed Requests]: 0
[   Total Mean Latency]: 83 ms
[Successful Mean Latency]: 83 ms
[   Failed Mean Latency]: 0 ms
[   Total Median Latency]: 16 ms
[Successful Median Latency]: 16 ms
[   Failed Median Latency]: 0 ms
[   Total 99th Latency]: 16 ms
[Successful 99th Latency]: 16 ms
[   Failed 99th Latency]: 0 ms
[   Total 95th Latency]: 16 ms
[Successful 95th Latency]: 16 ms
[   Failed 95th Latency]: 0 ms
```

Statistics for POST requests:

```
[   Total Requests]: 10000
[Successful Requests]: 10000
[   Failed Requests]: 0
[   Total Mean Latency]: 74 ms
[Successful Mean Latency]: 74 ms
[   Failed Mean Latency]: 0 ms
[   Total Median Latency]: 15 ms
[Successful Median Latency]: 15 ms
[   Failed Median Latency]: 0 ms
[   Total 99th Latency]: 15 ms
[Successful 99th Latency]: 15 ms
[   Failed 99th Latency]: 0 ms
[   Total 95th Latency]: 15 ms
[Successful 95th Latency]: 15 ms
[   Failed 95th Latency]: 0 ms
```

Step 6: Charting

Using Python to plot the data file

For this part, I add a “timestamp” field for get and post operations to keep track of the start time of each request. After put them into a List, write them in the format of Timestamp::Latency into .dat file using writeData() method in ResultMeasurement class. (PS: I’ve commented out this method to let the client generate statistic outputs after I finished the charting, but it is still there.) And then use Jupyter Notebook to read the .dat file and translate into data frame. Then save the latency charts generated by pandas.dataframe.plot().

Here is the results for thread10,iteration100, other 2 charts are too long to put here. Please check my “step6” folder. There are codes and .dat files.

