COP5615

Distributed Operating Systems Principles

Fall 2014

Project IV – Part II

TWITTER SIMULATOR

(REST API)

Submitted by:
Yogesh Aggarwal
UFID: 1311-1361

<u>Code Submitted</u>

1. Project4 Part II
    a. Twitter Server
    b. Twitter Http Server
    c. Twitter Client

<u>Instructions for running the project</u>

1. Unzip the attached zipped file to temporary location on the machine.
2. Running Project4 server component
    a) Open command prompt and cd to the location of the "**Project4/Twitter_Server**" directory contained in the zipped file (make sure sbt is installed on the machine)
    b) Run as below

```
sbt run
```

3. Running Project4 client component
    a) Open command prompt and cd to the location of the "**Project4/Twitter_Http_Server**" directory contained in the zipped file (make sure sbt is installed on the machine)
    b) Run as below

```
sbt "run twitter_server_ip"
```

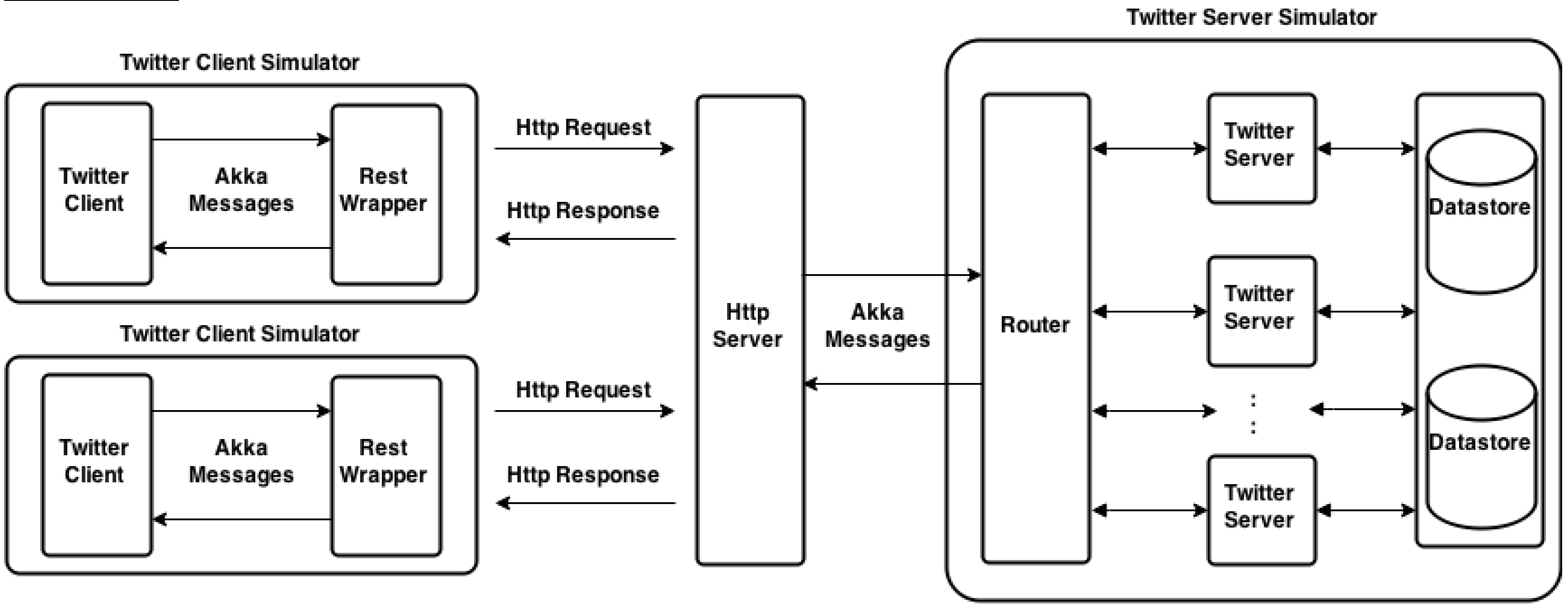
4. Running Project4 client component
    c) Open command prompt and cd to the location of the "**Project4/Twitter_Client**" directory contained in the zipped file (make sure sbt is installed on the machine)
    d) Run as below

```
sbt "run twitter_http_server_ip  load_factor rest"
```

Note:

- **twitter_server_ip** is the IP address of the remote machine on which the twitter server component is running
- **twitter_http_server_ip** is the IP address of the remote machine on which the twitter http server component is running
- **load_factor** should be greater than 0 and less than equal to 100
- While running Twitter_Client if the last argument is provided as "**rest**" then project will run as part II otherwise as part I

Architecture



Use Cases Implemented

1. User creation simulating
2. Followers/Following of user simulation
3. Tweeting simulation
4. Tweet Reply simulation
5. Retweet simulation
6. Request user stats simulation
7. Request user profile page tweets (user timeline) simulation
8. Request user home timeline simulation
9. Request server stats simulation
10. Request mentions simulation

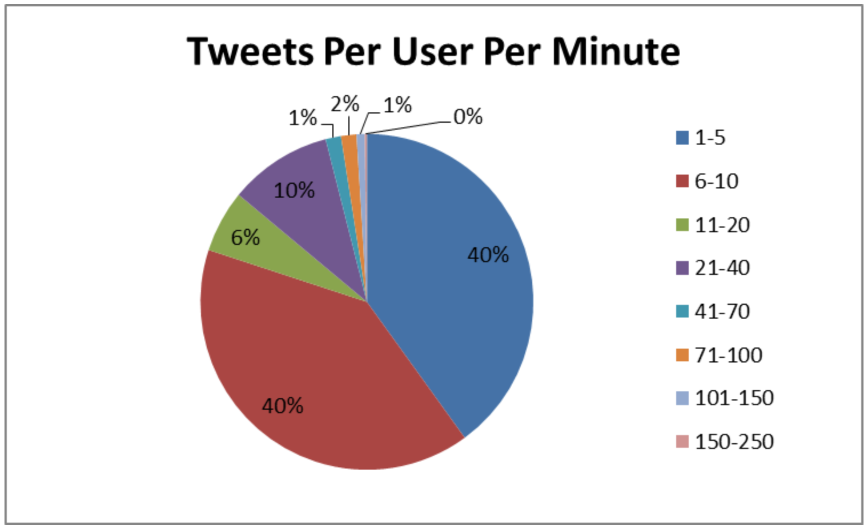
Rest API

Request Type	API Method	Request Parameters/Body	Description
Post	/add/client		Adds a new client/datastore
	/add/user	{“userId”:11}	Adds a new user
	/add/follower	{“userId”:11,”followerId”:12}	Adds a follower to a user and updates following list of follower
	/add/followers	{“userId”:11,”followers”:[12,13,14]}	Adds a list of followers to a user and updates the following list of all followers
	/add/tweet	{"userId":11,"tweet":"abcd"}	Adds a tweet
	/add/reply	{"userId":11,"replyToTweetId":"121","reply":"abcd"}	Adds a reply to a tweet
	/add/retweet	{"userId":11,"tweetId":"11"}	Retweets an existing tweet
Get	/user/stats	userId=11	Reports user statistics (total tweets, total followers, total following)
	/user/usertimeline	userId=11	Reports tweets from user’s profile page
	/user/hometimeline	userId=11	Reports tweets from user’s home timeline
	/user/mentions	userId=11	Reports tweets where user was mentioned (i.e. replies from other users to this user’s tweets)
	/server/stats		Reports server statistics (total users, total tweets, average tweets)

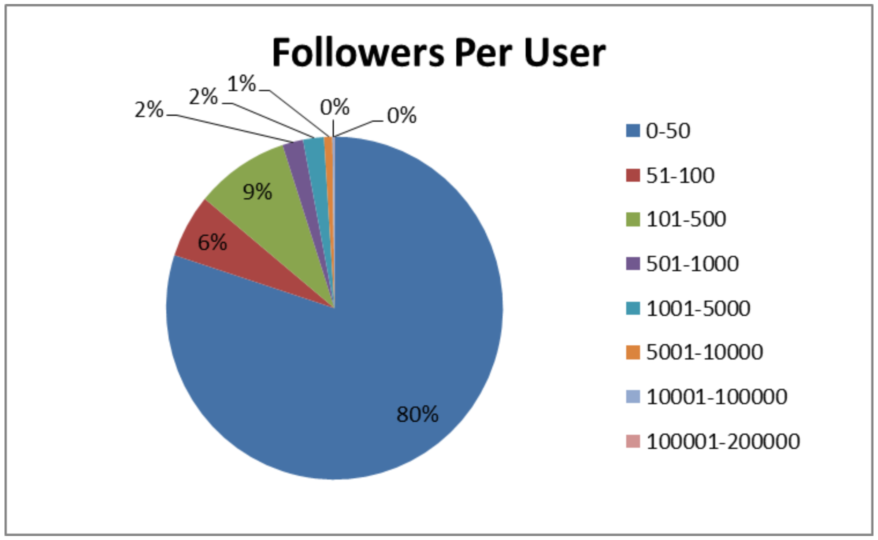
Note: content-type for all responses is application/json

Simulation Criteria

1. **Maximum Users**: 100 million
2. **Number of tweets per user per min:**
   - This is simulated as per the below chart. For eg. 40% of users will send 1-5 tweets every 60 seconds. Not all users tweet at the same time. Each user will tweet after every few seconds which is randomly decided after each tweet. This prevents all users loading the server with tweets at the same time.



3. **Number of followers per user:**
    - This is simulated as per the below chart. For eg. 80% of users will have 0-50 followers. A random number is chosen between 0-50 for each user in that class.



4. **Request user stats in second:**
    - If this is equal to 0 then this functionality is not simulated.
    - If this is equal to X (> 0) then after every X seconds a random user will request for its statistics from the server.

5. **Request user's profile page tweets (user timeline) in seconds:**
    - If this is equal to 0 then this functionality is not simulated.
    - If this is equal to X (> 0) then after every X seconds a random user will request for its user timeline from the server.

6. **Request user's home timeline tweets in seconds:**
    - If this is equal to 0 then this functionality is not simulated.
    - If this is equal to X (> 0) then after every X seconds a random user will request for its home timeline from the server.

7. **Reply to random tweet in seconds:**
    - If this is equal to 0 then this functionality is not simulated.
    - If this is equal to X (> 0) then after every X seconds a random user will select a random tweet from its home timeline and send reply to it.

8. **Retweet random tweet in seconds:**
    - If this is equal to 0 then this functionality is not simulated.
    - If this is equal to X (> 0) then after every X seconds a random user will retweet a random tweet from its home timeline.

Note:

- All simulation will happen on client side and no simulation on server side.
- User can decide the % simulation he/she wishes to run
- 100% simulation will create "Maximum Users" as defined above (not tested)

Results

No. of Clients	Users/Client	Total Users	Total Tweets	Tweets/sec	Max Tweets/sec	Runtime (sec)	No. of User Actors	Tweets/Actor/sec
1	1000	1000	209500	9435	9435	440	10	1000
1	5000	5000	11349816	30675	30675	360	50	1000
1	10000	10000	884820	9412	21284	100	100	1000
1	20000	20000	796370	13730	26554	60	200	1000
1	50000	50000	663806	11444	29317	60	500	1000
1	80000	80000	569914	12389	31029	50	800	1000
1	100000	100000	495700	10776	31740	50	1000	1000