Ashwin Chati A20354704

Design Document

Problem Statement:

To implement a distributed task execution framework on Amazon EC2 using the SQS. The framework should be divided into two components.
1. Client: Command line tool which submits tasks to SQS.
2. Worker: Extracts tasks from SQS and executes them.

Also Implement a local client-worker framework where,
1. Client: Command line tool which submits tasks to local queue.
2. Worker: Extracts tasks from queue and executes them.

Architecture:

a. Local Framework:

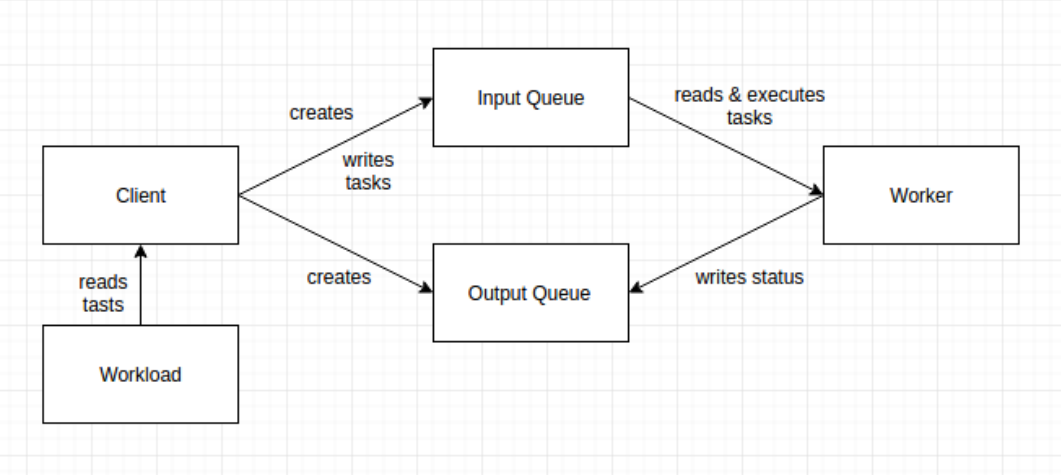

Fig: Local Framework

The description of components is as follows:

1. Workload: It is a file in which tasks to be executed are kept.

2. Client: It takes the tasks from the workload file and save them to the local queue.

3. Input Queue: It is the local queue which contains the tasks to be executed.

4. Worker: It fetches the tasks from the Input Queue and executes them. It then stores the status into the output queue.

5. Output Queue: It stores the status of the execute tasks.

Implementation:

1. CloudKon_Main: It is the main function where execution of the program starts. It calls client and worker functions and also calculates the execution time of the program.

2. Create_Client: It creates client.

3. Create_Worker: It creates the number of workers specified by the user.

4. Run_Client: It fetched the tasks from workload file and stores them in input queue.

5. Run_worker: It fetches the tasks from input queue, executes the tasks and stores the status of execution in output queue.

b. Remote Framework:

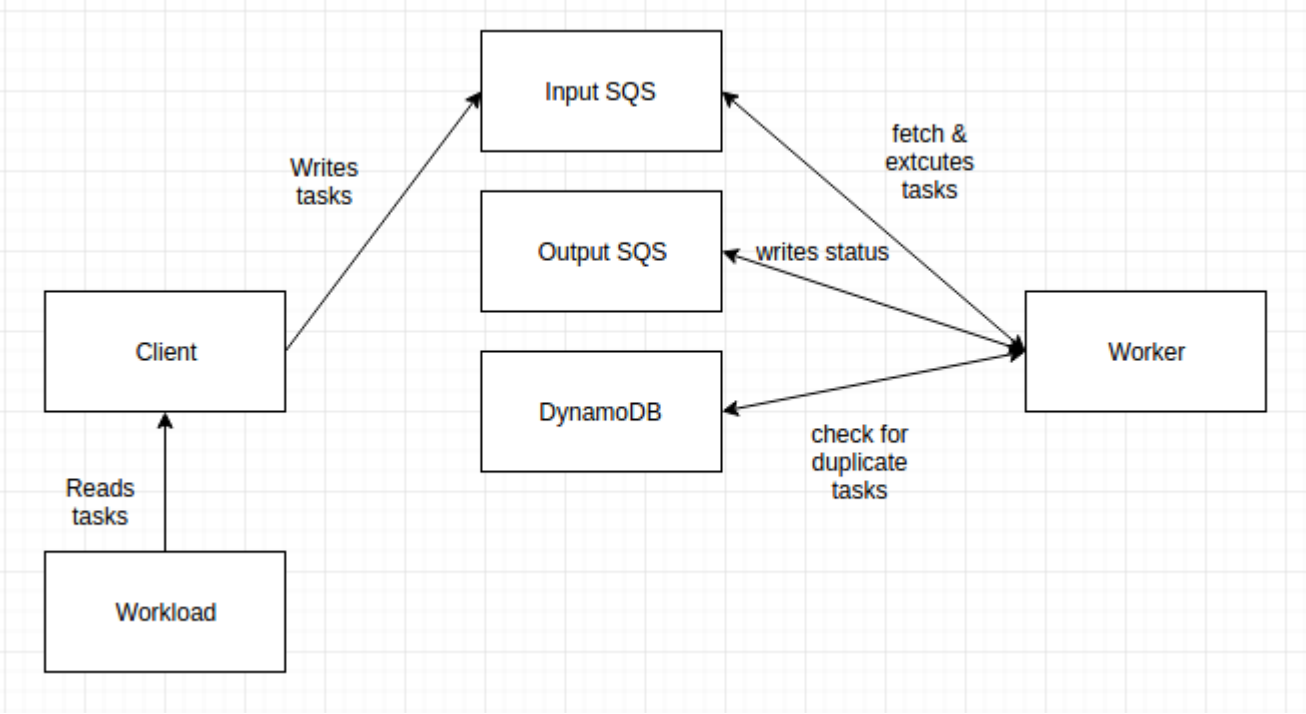

Fig: Remote Framework

The description of components is as follows:

1. Workload: It is a file in which tasks to be executed are kept.

2. Client: It takes the tasks from the workload file and save them to the input SQS.

3. Input SQS: It is the SQS which contains the tasks to be executed.

4. Worker: It creates the input and output queue and dynamoDB table, fetches the tasks from the Input Queue and and checks in the dynamoDB for duplicate tasks. If no duplicate task then it executes them and then stores the status into the output queue and insert the task_id into dynamoDB table. If duplicate tasks found then task is ignored.

5. Output SQS: It stores the status of the executed tasks.

6. DynamoDB: It is used for checking for duplicate task.

Implementation:

Client:

1. Client_Main: It is the main function where execution of client program starts. It also calculates the time required for execution.

2. Create_Client: It creates client.

3. Run_Client: It fetches the tasks from workload file and stores them in input SQS.

Worker:

1. Worker_Main: It is the main function where execution of worker program starts.

2. Create_Queue: It creates the Input and Output SQS.

3. Create_table: It creates the dynamoDB table.

4. Create_Worker: It calls Worker function.

5. Run_Worker: It fetches the tasks from input SQS, checks for duplicate task in input SQS and if no duplicate task found then executes the task and then store task in output table and make entry in dynamoDB table. If duplicate task found then task is discarded.