Group7 - Team1

Members - Chitra Kumari, Akshat Gupta, Aayushi Nigam

Components - Scheduler, Deployment Manager, Load Balancer, Platform initializer

May 8, 2021

Contents

1 Overview

Our team deals with 3 components - **Scheduler, Deployment Manager** and **Load Balancer** (which will be a sub-component of the deployment manager), and boostrapper service.
Deployer's request to deploy an application is sent to the application manager which passes the application instance id(or algorithm) to be run to the scheduler. At the scheduled time, it is sent to deployment manager. Deployment manager after balancing the load among the nodes, makes sure that the app(or algorithm) runs on the appropriate node (or virtual machine).
Platform initializer service is used to get the platform running on a set of client machines. It installs the prerequisites, configures the different distributed machines, and gets the platform services running on these machines

2 Block Diagram

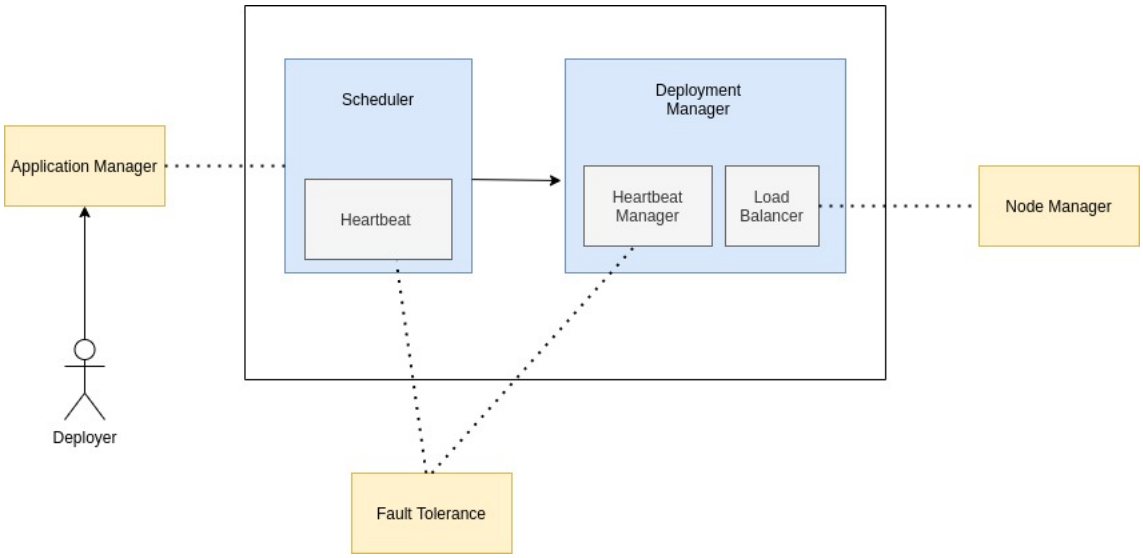


Figure 1: Block diagram for deployment and scheduling of jobs

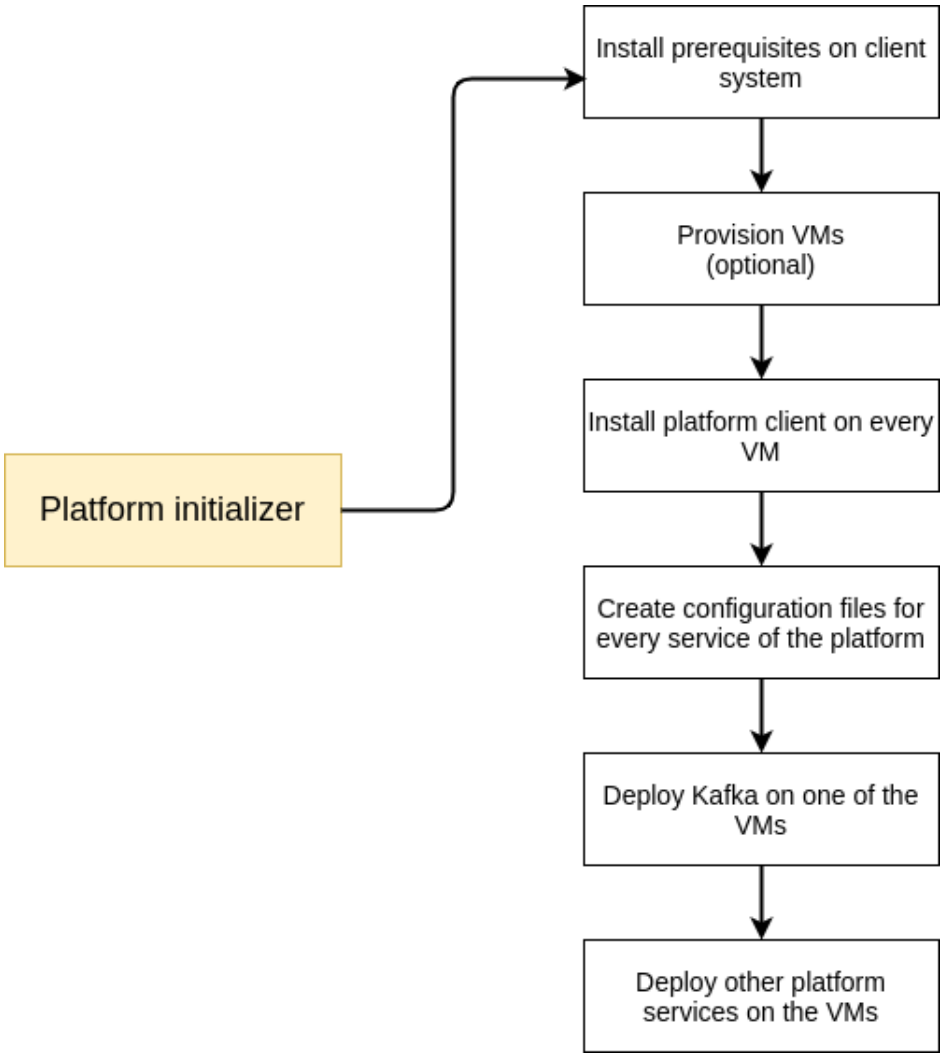


Figure 2: Block diagram for Platform initializer service

3 Technologies Used

- **Python framework** - overall development
  - Presence of Third Party Modules
  - Extensive Support Libraries
  - User Friendly data structures
- **Apache Kafka** - for communication
  - Scalable
  - Fault Tolerant
  - Publish-Subscribe Model
- **MySQL** - for database
  - Data Security
  - High Performance
  - On demand scalabilty
- **Docker** - for platform independence and portability
  - Consistent and isolated environment
  - Mobility
  - Easy collaboration

4 Environment Used

- OS - Linux 64-bits
- RAM - min 2 GB
- Storage - min 10 GB

5 List of subsystems

5.1 Scheduler

- Heartbeat Manager
- In memory heap
- Kafka Consumer
- Kafka Producer
- scheduler DB

5.2 Deployment Manager

- Load Balancer : Load balancer is used to find the node with the least node so that it can be used to run the application on.
- Heartbeat Manager
- Kafka Consumer
- Kafka Producer
- Database Handler
- Deployer DB

6 List of sub-modules

6.1 Platform initializer service

- Prerequisites installer
- VM provisioner and config updater
- Platform client installer
- Platform components bootstrapper

7 Actors Involved

7.1 Platform Deployer

Actor involved for the **Initializer service** is the **Platform Deployer**, who deploys the platform on the client system / machines. The platrofm configurer updates the addresses of the client machines, if they already exist, or provisions them using the service, if they don't exists. The bootstrap service then takes over to handle the client machines, and deploy various components of the platform on different machines, in a distributed manner.

7.1.1 Flow with respect to the platform deployer

- The platform deployer configures the information of the client machines on the bootstrap service.
- If the client machines do not exist, deployer can also configure the initializer service to create distributed machines for client.
- The platform initializer then configures the machines to run the platform components and initializes them on the machines in a distributed manner

7.2 Configurer

Actors involved in these components is the **Configurer**. Configurer will bind the physical instances of the sensor at configuration time. During this time, configurer will provide the scheduling information. This scheduling info for each instance, includes start time, end time, duration, interval of repetition if application is to be run repetitively.

7.2.1 Flow with respect to the configurer

- The app developer deploys the application on the platform.
- The configurer then selects the application to be run using the unique id provided to the application.
- The configurer provides information like the place where the application is to be deployed and the scheduling information (mentioned above).
  - This information can be provided by the configurer in two ways.
    - Through the UI dashboard of the scheduler
    - By uploading the configuration files.
- The information provided by the configurer is validated by the app manager and further stored in app database.

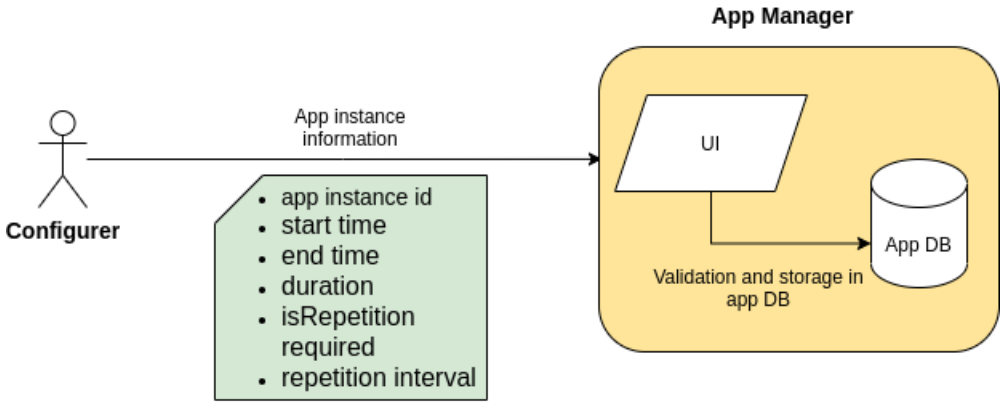


Figure 3: Configurer interaction with platform

8 Services

8.1 Platform Initializer

- The platform initializer initializes the client machines so that they are able to run the platform components. The platform components are independent of the number of machines the client has, and can run all on one machine, or even if they are distributed.
- After installing the platform clients, the initializer decides which service to run on which machine, and deploys the service on the machine, with the help of bootstrapper.

8.2 Scheduler

- The scheduler will receive application instance id from application manager. Based on this id, scheduler will fetch the scheduling info from the app repository like start time, end time, duration, isRepetition required and interval of repetition (if required).
- The application instances are placed in the priority queue as per their start times
- The applications are popped out of priority queue as per their scheduled time and a notification is sent to the deployer of deployment manager to start the application.
- Information sent to deployment manager are: app instance ID, a command to start/kill the app, whether the application is to be run on a standalone machine or shared machine

8.3 Deployment Manager

- The deployment manager receives the request from the scheduler to start/stop the app.
- It then contacts with the node manager to get active nodes.
- Load is balanced among the nodes and the given application gets deployed thereafter.

8.4 Load Balancer

- We use two metrics to calculate the load on the machine.
  - Load on the CPU
  - Percentage of RAM in use.
- These metrics are retreived from the nodes using a library called paramiko.
- We use the harmonic mean of the above two values to calculate the load on a particular machine, and select the node with the least load.

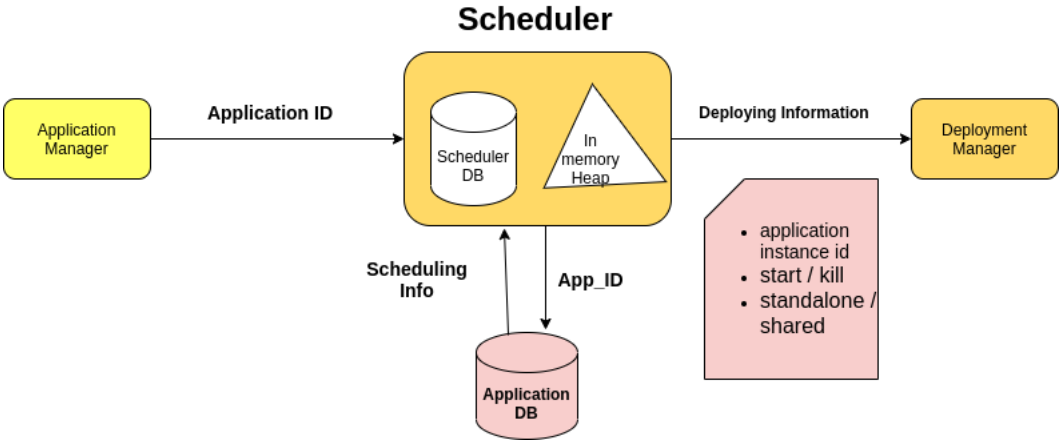


Figure 4: Scheduler Service

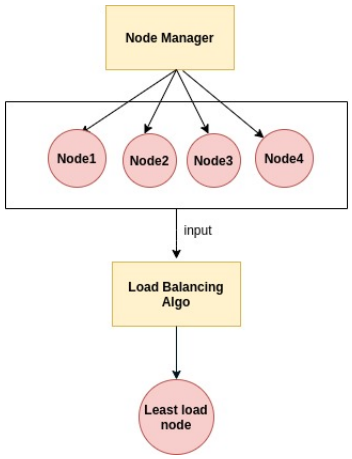


Figure 5: Load Balancing

9 Key Data structures

9.1 Platform initializer

- Map: Mapping of platform services with the client machine they are running on. The data is stored as files, and other configurational details, like database config, kafka config etc.

9.2 Scheduler

- Heap: In memory data structure for storing the scheduling info of jobs as per their start time and end time.
- Queue: For storing the currently running jobs, after they've been popped out of the heap.
- Scheduler DB: used to store scheduler state in order to recover scheduler after a failure

9.3 Deployer

- Map: For storing the mapping between job-ids and node-id

9.4 Load Balancer

- Heap: For storing the load on every node returned from the node manager, and to get the node with least load.

10 Database Interactions

10.1 Deployment manager

- For this project, we have used 1 Database for Deployment Manager. The database is used by the deployment manager to store the mapping between the application instance ID and the node address on which that instance is running.
- It also stores the corresponding job id for every instance running.
- When the application instance is initialized on a machine, it stores its mapping with the node in the database.
- When the application instance needs to be killed, it uses this database to identify the node on which the application instance is running, and sends the request to the particular node for killing the instance.

10.2 Scheduler

- Scheduler contacts app DB to retrieve scheduling data
- Scheduler also uses a scheduler DB in order to maintain persistence.

11 Initialization of the module

- Before starting any of the module, it is necessary to create the database that will be used by the modules. This is done by running the **schedulerInit.py** in scheduler and **migrations.py** file in deployment manager that initializes the database on the system, along with the necessary tables.
- All the components of this module will be started by platform initialiser.

12 Interaction with other components

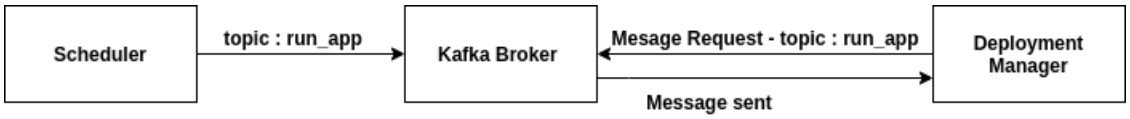
Figure 2 shows the interaction between various components.

12.1 Interaction between Application Manager and Scheduler

- Application Manager will provide Scheduler with the application instance id. Scheduler will then go to the repository, and based on the instance id, will retrieve the scheduling config file for the application.

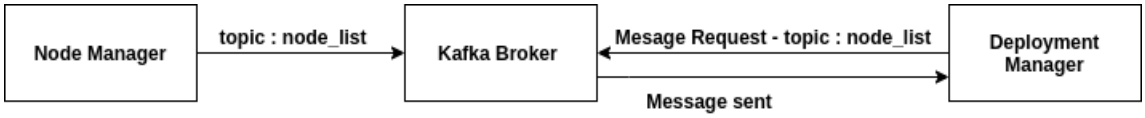
12.2 Interaction between Scheduler and Deployment Manager

- Scheduler on receiving the requests from application manager, will add the jobs to its priority queue, and will send the event to deployment manager through single **one to one topic**, which will contain the job info. The frequency will depend on the number of dependencies of the job and the number of instances requested by the user.Similarly, the scheduler will send the message to kill the job on the former topic.
- In order to start the application, the scheduler will contact the deployer in the Deployment Manager. A list of such jobs will be placed in the job queue by the scheduler and retrieved by the deployer.



12.3 Interaction between Deployment Manager and Node Manager

- The Load balancer part of Deployment Manager will contact the Node Manager in order to get a list of active nodes.
- After getting this list from the Node Manager, the Load Balancer will calculate the load of each node using Load Balancing Algorithm.
- A node with lowest load will be selected.
- The deployer will contact the Node Manager with the selected node on which the application with application id is to be run.



12.4 Interaction between Platform Initialiser and Scheduler/Deployer

- Platform initialiser will initiliase the scheduler and the deployer to get them up and running.

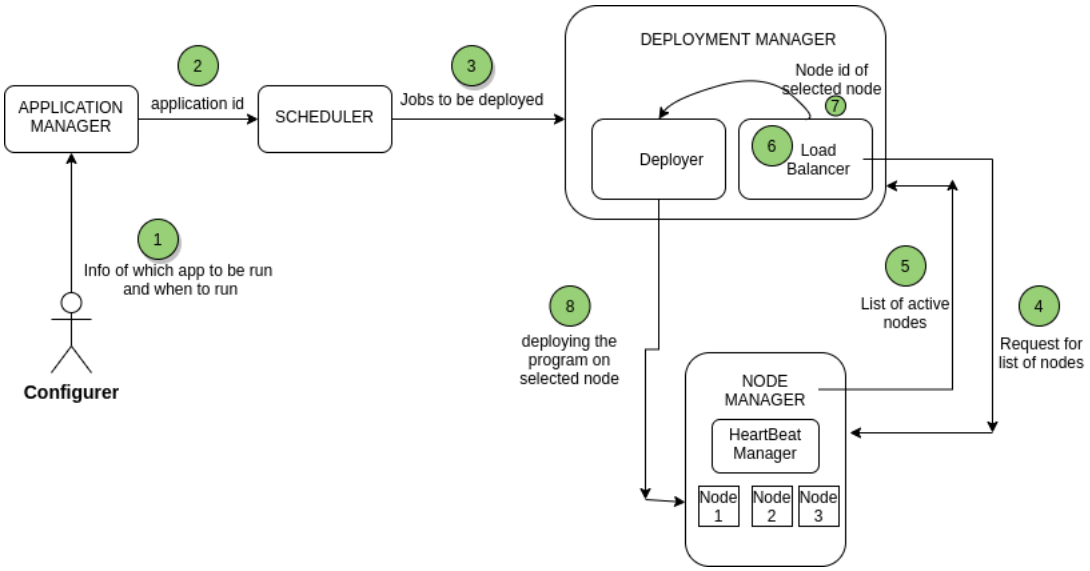


Figure 6: Diagram showing interactions among different components