AppMill

LOAD BALANCER GUIDE

Table of Contents

1. Overview

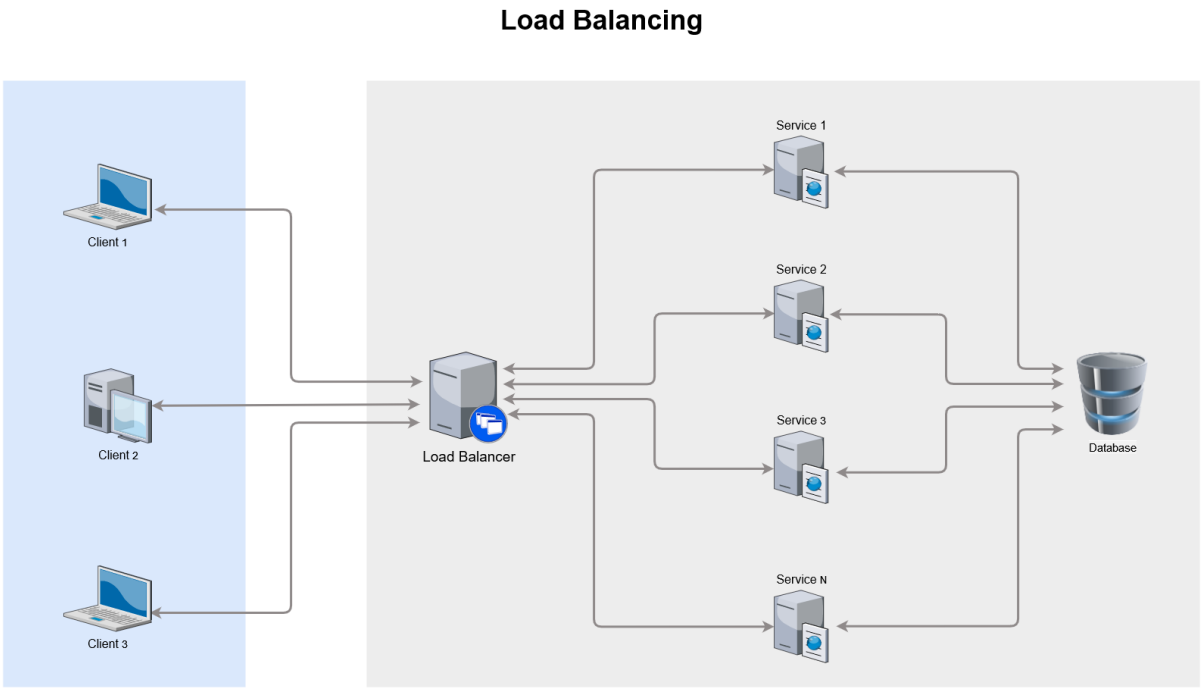
A **Load Balancer** is an AppMill service mode that was designed to distribute the load when processing client requests between a group of regular services.

In case of service overload, when clients send many requests to services, significant system slowdown and interruption of requests due to timeout may occur. To avoid such situations the service can be run as a **Load Balancer**.

The running service in the balancer mode acts as a proxy server.

The **Load Balancer** redirects the clients' requests to the end services based on the selected balancer mode. In this case the system will work faster and will greatly improve the productivity.

The following diagram shows the base structure of the cluster with the balancer:



Where:

- **Client 1…N** – is AppMill clients.
- **Load Balancer** – is the AppMill service that redirects the user's requests to the end services based on the selected balancer mode.
- **Service 1…N** – is AppMill services that handle the user's requests.
- **Database** – is a database supported in AppMill services.

2. Load Balancer Modes

Depending on the requests count, their resources utilization, time of the execution and other related parameters, the Load Balancer can be set in different modes. The service as a balancer supports the following balancer modes of operations:

- **Next in line**
- **CPU load**
- **Any server no sessions**

2.1 Next in line

The **Next in line** mode is set for the service in case all requests from a client session will be executed on one selected service. The balancer selects the service based on the order that they were added into the cluster.

For example: We have 3 clients, and Client 1 created a new session (start communication with the cluster). All requests from this client will be redirected to the Service 1, until the end of the session or invalidate it by the timeout. Next client's requests will be redirected to the Service 2 and so on. Client's session N+1 will be redirected again to the Service 1.

2.2 CPU load

The **CPU load** mode is set for the service in case all requests from a client session will be executed on one selected service based on the minimal CPU utilization at the time of the service selection.

For example: We have 3 services, and Service 2 is with the minimal CPU utilization. Then all requests from a new client will be redirected to Service 2 and will be executed there until the end of the session regardless of subsequent CPU loading.

2.3 Any server no sessions

The **Any server no sessions** mode means that the user's session does not depend on one service and requests could be executed on any service that has minimal count of active requests (requests from other clients that are in progress on this service) at the selection time.

For example: A client starts communication with the clusters and has 10 independent requests that should be executed at the same time. So, if we have more than 10 services and all of them have the same count of the active sessions, it will be executed on 10 separate services. In case we have 5 active services with the same count of the active requests, each service will execute 2 requests from this client.

3. Load Balancer Configuration

To configure the AppMill cluster with the balancer, it is necessary to add the needed settings to the *Codiac.HTTPService.config* file on the balancer side and on each service as well.

3.1 Balancer Settings

The following balancer settings should be added to the *Codiac.HTTPService.config* file:

- **is_balancer** – the indicator of the balancer activation. In case the value is set to:
  - ***true*** - the balancer is in active status,
    or
  - ***false*** – the balancer is disabled.
- **heartbeat_timeout** – the time interval in the seconds, when the balancer checks that all services from the list are in the available status. The value by default is 300 seconds.
- **server_choose** – algorithm that will be used at the request redirections to the end services. Possible values: **cpu_load, next_in_line, any_server_no_sessions**.

The services list, that is in the balancer cluster, should contain at least one **service** element with the following attributes to allow connection and communication with these services:

- **address** – IP or FQDN name of the services.
- **port** – port on which this service listens to incoming requests.
- **useHttps** – boolean value (true or false) that shows the type of the protocol for communication with this service. When the value is equal to:
  - ***true*** - HTTPS will be selected,
    or
  - ***false*** - HTTP will be selected.

The Configuration file stores data in the XML format and should have the following settings:

```
<config>
  <service name="appmill.service" useHttps="false"/>
  <logs store="false"/>
  <bind address="10.0.2.82" port="34567"/>
  <balancer is_balancer = "true" heartbeat_timeout = "300" server_choose =
"any_server_no_sessions">
    <service address="10.0.2.82" port="34571" useHttps = "false" />
    <service address="10.0.2.82" port="34572" useHttps = "false" />
    <service address="10.0.2.82" port="34573" useHttps = "false" />
  </balancer>
</config>
```

The **balancer** element, that contains the balancer settings, and the services list, that is included into the current cluster, should be presented in the config file.

3.2 Service Settings

Each service in the cluster branch should have the **service** element with the attribute **balancer_node** set to ***true***, which indicates that this node is in the balancer and excludes variants of using the nodes that were removed from the cluster. For example,

Service 1:

```
<config>
  <service name="Codiac.Service" useHttps="false" balancer_node="true" />
  <logs store="false" />
  <bind address="10.0.2.82" port="34571" />
</config>
```

Service 2:

```
<config>
  <service name="Codiac.Service" useHttps="false" balancer_node="true" />
  <logs store="false" />
  <bind address="10.0.2.82" port="34572" />
</config>
```

Service 3:

```
<config>
  <service name="Codiac.Service" useHttps="false" balancer_node="true" />
  <logs store="false" />
  <bind address="10.0.2.82" port="34573" />
</config>
```

3.3 Database

All services should be connected to one Database to use the same data and avoid data corruption.

The AppMill services support the following database types:

- PostgreSQL
- OracleSQL
- MicrosoftSQL
- MySQL
- SQLite

Please note that SQLite Database usage is not allowed. At the starting a service, it takes exclusive lock of the Database file. Therefore, in this case each service will have the database with its own version that is not synchronized with others.

3.4 Usage

The balancer service should be started when at least one end service in the cluster is running to avoid failed requests from clients. In case the balancer was started and there are no available end services, all

requests will be failed. But after starting a new one, after the heartbeat timeout interval it will be marked as allowed and will be included into the available services list.

In case the service becomes unavailable and the **Next in line** and **CPU load** modes are in the active sessions that were assigned to this service, the service will be invalidated and the user will have to login again and start new sessions on the available services.

In case the service is in the **Any server no sessions** balancer mode, all requests will be redirected to available services without interrupting current sessions.