Contents

[ORDER MANAGEMENT SYSTEM 2](#_Toc430448219)

[Introduction: 2](#_Toc430448220)

[Modules: 2](#_Toc430448221)

[RECEIVER: 4](#_Toc430448222)

[PROCESSOR: 4](#_Toc430448223)

[SCHEDULER: 4](#_Toc430448224)

[DATABASE: 4](#_Toc430448225)

# ORDER MANAGEMENT SYSTEM

## Introduction:

The order management system handles the requests received by its upstream and processes it to the downstream consumers based on the priority set by the upstream. It processes the requests efficiently, taking care of the SLA (Service License Agreement) period which is configurable. As per the SLA configured no request should be on order management system more than the SLA period. In addition to processing the request order management system handles the load balancing (i.e) when the threshold limit is reached it spawns process to handle the request load.

## Modules:

* RECEIVER
* PROCESSOR
* DUMPER
* SCHEDULER

The draft version of order management system process flow is pictured below in the figure.

**SCHEDULER**

DATABASE

RECEIVER WEB SERVICE

OVERWRITES THE DUPLICATE REQ

UPDATES THE REQ PRIORITY AS PER SLA

DUMPER n

DUMPER 2

DUMPER 1

PROCESSOR

### RECEIVER:

Receiver process is a web service which is deployed in a tomcat server, which can be accessed by the upstream and send the request. Once the request is received, the request is loaded into the oracle DB. When the request is inserted into the table a trigger is triggered in order to validate whether the previously sent low priority request to be overwritten. Along with the request data sent by the upstream, receiver updates the request status in table. The request status are as follows.

1. REQUEST RECEIVED(1)
2. REQUEST IN PROCESS(2)
3. IGNORE REQUEST(3)
4. ERROR RQUEST(4)
5. REQUEST PROCESSED(0)

### PROCESSOR:

Processor thread reads the received request from the database table based on the priority, request status and SLA period.IT reads the request from the table if the request status is REQUEST RECEIVED(1) and priority is high. The read request is sent to the corresponding downstream consumer based on the market place ID.A configurable market place ID and downstream consumer mapping file would be maintained in order to dynamically update the downstream consumers.

The processor thread handles the number of request and amount of payload to be sent to each target system.

### SCHEDULER:

Scheduler thread takes care of the load balancing based on the number of pending request to be processed. If the pending request threshold limit is reached, scheduler would spawn the additional processor in order to take care of the pending requests. Scheduler thread supports the order management system to be scale horizontally thus handling the high availability.

### DATABASE:

The database has table to load the request data into it which is sent by the upstream. Two triggers are running on the DB one while inserting the request data and the other while updating the processed request data. The first trigger is triggered when a data is inserted from the upstream in order to validate if the received request is already received in same time with a low priority. If found the previously sent request is to be ignored from sending to the downstream consumer by updating the request status column to IGNORE REQUEST(3), considering the upstream has updated the request data.

The second thread is triggered while updating the processed request back to the table by updating the request status column to REQUEST PROCESSED (0).