|  |
| --- |
| Distributed systems |
| Jms and Jmx |
| Master M1 MOSIG, Grenoble Universities |
| GRÄBNER Laurent |
| TCHOUGOURIAN Tigran |
|  |
| **10/04/2011** |



|  |
| --- |
| **Supervisor :**  BOUCHENAK Sara |

|  |
| --- |
| The assignment is about building an asynchronous-communication application with JMS. |

Table of contents

[1. Architecture of the application 2](#_Toc290196984)

[2. Overlay creator 3](#_Toc290196985)

[3. Node 3](#_Toc290196986)

[3.1. JMS 3](#_Toc290196987)

[3.2. JMX 3](#_Toc290196988)

[4. Handbook 3](#_Toc290196989)

# Architecture of the application

The architecture of the application is quite simple. The OverlayCreator generate the structure of the servers and run one virtual node by server. Each virtual node used to transfer information to the top layer, has a unique topic where its children publish. Every 15 seconds, a node access, thanks to JMX, some information about the server. Once the local data are recovered, the node collects data from its children, aggregates them and sends them to the upper layer thanks to JMS. Because the system is asynchronous, two threads are running every time. The collecting thread is a simple listener which receives JMS messages and updates data. In the other hand the publishing thread sends information about the server every 15 seconds



Figure 1‑1: Global architecture

The Figure 1-1 shows an instance of the overlay. Each node runs on a server and collect information every 15seconds. Send a message to the upper lever. The messages are always up to date in the mail box. Only the higher node (master node) saves the finale message in a log file.

|  |  |
| --- | --- |
| Legend | |
|  | A node running on a server |
|  | JMX collects information about the server state. |
|  | A message waiting to be collected in the topic of the upper node. |
|  | Shows the topic where a node publishes. |
|  | A message sends to a topic. |

# Overlay creator

# Node

The node is the virtual representation of a server. It runs on each server from the Overlaycreator. The nodes are communicated between them with JMS whereas all the monitoring is made with JMX.

## JMS

To be able to communicate between nodes:

* Each node with a parent starts publishing to its parent topic.
  + Every 30sec the node publishes data to the topic.
* Each node with children, create a unique topic
  + A listener (MemoryListener here) is set to receive asynchronously data from children.

## JMX

To be able to monitor the server status, JMX is used. Every 30 sec JMX provide information about the memory of the server. To avoid the node to waste CPU time, the thread managing publishing information sleeps when not needed.

# Handbook

Step 1: Execute openjms server, startup.bat (windows) or startup.sh (linux).

Step 2: Launch overlaycreator.jar

|  |
| --- |
| >java –jar Overlaycreator –N 7 –d 4 |