Imagen que contiene Icono

Descripción generada automáticamente

**MTIS PRÁCTICA 2**

**MOM Message Oriented Middleware**



**9 de marzo de 2025**

**METODOLOGÍAS Y TECNOLOGÍAS DE INTEGRACIÓN SOFTWARE**

**Escuela Politécnica Superior de la Universidad de Alicante**



Realizado por:

Cristian Andrés Córdoba Silvestre

Profesor:

Alejandro Sirvent Llamas

**ÍNDICE**

[**1 Descripción de la práctica** 2](#_Toc192443739)

[**2 Archivos entregados** 3](#_Toc192443740)

[**3 Puesta en marcha** 4](#_Toc192443741)

[**4 Ejemplo de ejecución** 5](#_Toc192443742)

[**5 Implementaciones para la máxima nota** 6](#_Toc192443743)

# **1 Descripción de la práctica**

El objetivo de esta práctica es hacer uso de la tecnología **MOM**, en particular de **ActiveMQ**, un intermediario que permite intercambiar mensajes entre distintos sistemas y/o aplicaciones.

Se ha implementado un sistema que permite manejar y controlar la temperatura y la iluminación dentro de un **edificio inteligente**.

Para ello, se dispone de dos aplicaciones servidoras (una implementada en **C#** y otra implementada en **Node.js**), así como de una aplicación cliente que actúa como una **“Consola Central”** (implementada en **Python**). Estas tres forman en conjunto un sistema en el que se intercambia información y se realiza una serie de acciones en consecuencia (aumentar/disminuir la temperatura, aumentar/disminuir la intensidad de la iluminación).

# **2 Archivos entregados**

Se han entregado los siguientes proyectos:

* **oficina1\_cs:** Proyecto en **C#** asociado a la **Oficina 1**.
* **oficina2\_js:** Proyecto de **Node.js** asociado a la **Oficina 2**.
* **consola\_central\_python:** Proyecto de **Python** asociado a la **Consola Central.**

# **3 Puesta en marcha**

1. **Iniciar ActiveMQ**

Acceder al directorio **/bin** desde donde se ubica **ActiveMQ** y ejecutar el siguiente comando:

/**activemq** **start**

1. **Ejecutar el proyecto de Python (consola\_central\_python)**

Acceder a la carpeta raíz del proyecto y ejecutar el siguiente comando:

**python Consola\_central.py**

1. **Ejecutar el proyecto de C# (oficina1\_cs)**

Acceder a la carpeta raíz del proyecto de **C#** y ejecutar el siguiente comando:

**dotnet build**

**dotnet run**

1. **Ejecutar el proyecto de Node.js (oficina2\_js)**

Acceder a la carpeta raíz del proyecto de **Node.js** y ejecutar los siguientes comandos:

**npm i**

**node Oficina2.js**

# **4 Ejemplo de ejecución**

En la siguiente imagen se muestra un ejemplo de ejecución (La consola de arriba a la izquierda se corresponde con el proyecto en **C#** de la **Oficina1**; la consola de abajo a la izquierda se corresponde con el proyecto en **Node.js** de la **Oficina 2**; la consola de la derecha se corresponde con el proyecto en **Python** de la **Consola Central**):

Texto

El contenido generado por IA puede ser incorrecto.

**Figura 1: Los tres programas (C#, Node.js y Python) comunicándose entre sí**

# **5 Implementaciones para la máxima nota**

Se han realizado las siguientes implementaciones para poder obtener la máxima calificación en esta práctica:

* Se ha implementado el sistema utilizando tres lenguajes de programación diferentes **(C#, Node.js y Python).**
* Se ha implementado seguridad sobre **ActiveMQ** haciendo uso del plugin **“Simple Authentication Plugin”**.
* Se ha utilizado **comunicación asíncrona** en todo el sistema (mediante el uso de **listeners** y **tasks**).