UNIVERSITAT POLITÈCNICA DE VALÈNCIA

etsinf

Escola Tècnica Superior d'Enginyeria Informàtica

Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

Entregable 3

Trabajo IAP

Grado en Ingeniería Informática

Autor: Sopeña Urbano, Enrique
Lambaraa Ben Razzouq, Anass
Lorente Núñez, Vicente Rafael
Raga Riera, Pablo

Curso 2024-2025

Resumen

En este proyecto se desarrolló un sistema de integración para la sincronización de expedientes académicos entre un Centro Docente y la Generalitat Valenciana (GVA), utilizando RabbitMQ como middleware de mensajería. La solución incluye la generación, transformación y consumo de datos en formatos CSV y JSON, permitiendo la automatización del intercambio de información académica. Además, se implementó un cálculo automático de la nota media de los alumnos como funcionalidad adicional.

El diseño del sistema se basa en una arquitectura modular y flexible, empleando *exchanges* configurados para el enrutamiento eficiente de mensajes. Los componentes principales incluyen generadores de expedientes, consumidores de datos y un visualizador (`Visualizer`) que facilita la supervisión del flujo de información. Este enfoque asegura la escalabilidad y la adaptabilidad del sistema frente a nuevas exigencias o cambios en los formatos de integración.

Los resultados obtenidos demuestran la efectividad de la solución para manejar flujos de datos complejos de manera confiable y estandarizada, cumpliendo con los requisitos planteados y aportando un modelo eficiente para resolver problemas similares en contextos académicos e institucionales.

Palabras clave: integración, RabbitMQ, expedientes académicos, automatización, mensajería.

Abstract

This project presents the development of an integration system for synchronizing academic records between an Educational Center and the Generalitat Valenciana (GVA) using RabbitMQ as a messaging middleware. The solution involves the generation, transformation, and consumption of data in CSV and JSON formats, enabling the automation of academic information exchange. An additional functionality for automatically calculating students' grade point average was also implemented.

The system's design is based on a modular and flexible architecture, leveraging exchanges configured for efficient message routing. The key components include record generators, data consumers, and a visualizer (`Visualizer`) that facilitates the supervision of data flow. This approach ensures the scalability and adaptability of the system to accommodate new requirements or changes in integration formats.

The results demonstrate the solution's effectiveness in reliably and standardizedly handling complex data flows, fulfilling the stated requirements, and providing an efficient model for addressing similar challenges in academic and institutional contexts.

Key words: integration, RabbitMQ, academic records, automation, messaging

Índice general

Índice de figuras

Índice de tablas

CAPÍTULO 1
Introducción

El presente trabajo corresponde al tercer entregable del proyecto de la asignatura de *Integración de Aplicaciones*. El objetivo principal es desarrollar una solución de integración que permita la sincronización de expedientes académicos entre un Centro Docente y la GVA (*Generalitat Valenciana*) a través de un sistema basado en colas de mensajería, utilizando RabbitMQ como middleware. Este entregable pone un énfasis especial en la automatización y estandarización del intercambio de información académica mediante el diseño de procesos eficientes y escalables.

El caso de estudio aborda la necesidad de extraer y transformar datos de notas académicas del Centro Docente, generándolos en diferentes formatos (`CSV` y `JSON`), y remitiéndolos a un *exchange* de RabbitMQ con un formato predefinido por la GVA. Asimismo, se integra una funcionalidad para calcular la nota media de los estudiantes y un visor que muestra en pantalla los datos sincronizados, incluyendo información clave como el nombre del estudiante, el número de asignaturas enviadas y la nota media calculada.

Este documento describe la implementación de generadores de expedientes, una solución basada en mensajería para la transferencia de datos y un visor de expedientes importados. La solución propuesta destaca por garantizar la coherencia de los datos, cumplir con las restricciones planteadas y facilitar la validación mediante procesos automatizados. Este entregable forma parte integral de un proyecto global de la asignatura, contribuyendo al desarrollo de competencias avanzadas en el diseño de sistemas de integración robustos dentro del campo de la ingeniería informática.

Figura 1.1: Esquema general de la solución de integración

CAPÍTULO 2
Extracción de expedientes

2.1 Expedientes CSV

El módulo `GeneradorExpedienteCSV` permite generar expedientes académicos en formato CSV y enviarlos a través de RabbitMQ. Su principal objetivo es extraer datos de los estudiantes desde la base de datos del Centro Docente, estructurarlos en un archivo CSV estandarizado y publicarlos en un *exchange* para integrarlos con otros sistemas.

2.1.1. Funcionamiento

El módulo opera siguiendo los pasos descritos a continuación:

1. **Interacción con el Usuario:** El programa solicita el DNI del estudiante y el año académico mediante la consola.
2. **Extracción de Datos:** Utilizando el patrón DAO, se conecta a la base de datos del Centro Docente y obtiene la información del estudiante y sus calificaciones del año solicitado.
3. **Generación del Archivo CSV:** Los datos extraídos se estructuran en un archivo CSV con el siguiente formato:

```
notas-alumno-anyo
format-01, version 1.0
<dni>,<nombre>,<apellidos>
<año>
<id_asignatura>,<nota>
```

4. **Publicación en RabbitMQ:** El archivo generado se convierte a un arreglo de bytes y se envía como mensaje al *exchange* `primer-exchange`, utilizando la clave de enrutamiento `generador.csv`.

2.1.2. Ejemplo de Uso

Para un estudiante con DNI `12345678W` en el año 2023, el archivo generado sería:

```
notas-alumno-anyo
format-01, version 1.0
```

```
12345678W,Juan,Pérez
2023
IAP,9.5
DCU,8.0
DSW,7.0
```

2.2 Expedientes JSON

El módulo `GeneradorExpedienteJSON` permite generar expedientes académicos en formato JSON y enviarlos a través de RabbitMQ. Este componente se encarga de estructurar los datos de estudiantes en un formato JSON estandarizado y publicarlos en un *exchange*, facilitando su integración con otros sistemas.

2.2.1. Funcionamiento

El módulo realiza las siguientes tareas principales:

1. **Interacción con el Usuario:** Solicita al usuario el DNI del estudiante y el año académico mediante la consola.
2. **Extracción de Datos:** Utiliza el patrón DAO para conectarse a la base de datos del Centro Docente y obtener la información del estudiante y sus calificaciones.
3. **Generación del Archivo JSON:** Los datos se estructuran en un archivo JSON con el siguiente formato:

```
{
    "documento": {
        "tipo": "notas-alumno-anyo",
        "formato": "format-02",
        "version": "1.0"
    },
    "alumno": {
        "dni": "<dni>",
        "nombre": "<nombre>",
        "apellidos": "<apellidos>"
    },
    "anyo": <año>,
    "asignaturas": [
        {"codigo": "<id_asignatura>", "nota": <nota>},
        {"codigo": "<id_asignatura>", "nota": <nota>}
    ]
}
```

4. **Publicación en RabbitMQ:** El archivo JSON generado se lee como un arreglo de bytes y se envía al *exchange* `primer_exchange` con la clave `generador.json`.

2.2.2. Ejemplo de Uso

Para un estudiante con DNI `12345678W` en el año 2023, el archivo JSON generado sería:

```
{
    "documento": {
        "tipo": "notas-alumno-anyo",
        "formato": "format-02",
        "version": "1.0"
    },
    "alumno": {
        "dni": "12345678W",
        "nombre": "Juan",
        "apellidos": "Pérez"
    },
    "anyo": 2023,
    "asignaturas": [
        {"codigo": "IAP", "nota": 9.5},
        {"codigo": "DCU", "nota": 8.0},
        {"codigo": "DSW", "nota": 7.0}
    ]
}
```

CAPÍTULO 3

Diseño de la Solución para la Entrega de Notas a la GVA

En este capítulo se describe de manera detallada la solución desarrollada para la sincronización de los expedientes académicos entre el Centro Docente y la GVA mediante el uso de colas de mensajería, específicamente RabbitMQ. El objetivo principal es transformar los datos generados en el Centro Docente al formato estándar requerido por la GVA y enviarlos de forma eficiente y estructurada.

La solución se basa en una arquitectura modular donde varios componentes colaboran para procesar y distribuir los datos académicos, garantizando el cumplimiento de los requisitos especificados.

3.1 Descripción General del Sistema

La solución utiliza un sistema de mensajería centralizado que se compone de:

- **Exchanges de RabbitMQ**: Encargados de enrutar y distribuir los mensajes entre los diferentes componentes.
- **Consumidores**: Dos programas principales denominados `CSVConsumer` y `JSONConsumer`, que procesan datos en formatos específicos.
- **Middleware de Control**: Un componente llamado `NotasMiddleware`, que administra la ejecución de los consumidores y garantiza la operación concurrente del sistema.

El sistema recibe mensajes generados por el Centro Docente, los transforma al formato requerido por la GVA y los publica en un canal de distribución que puede ser consumido por sistemas externos.

3.2 Exchanges Utilizados

En el diseño de la solución se utilizan dos exchanges principales, cada uno con un propósito específico:

3.2.1. Exchange de Origen: `primer-exchange`

- **Tipo:** `topic`.

- **Descripción:** Es el punto de entrada del sistema, donde se reciben los expedientes académicos generados por el Centro Docente.
- **Función:** Gestiona los mensajes en función de su formato (CSV o JSON) utilizando claves de enrutamiento.
- **Claves de Enrutamiento:**
  - `generador.csv`: Para mensajes en formato CSV.
  - `generador.json`: Para mensajes en formato JSON.
- **Comportamiento:** Los consumidores se suscriben a este exchange mediante las claves correspondientes para procesar solo los mensajes del tipo esperado.

3.2.2. Exchange de Destino: `notas.alumno.anyo`

- **Tipo:** `fanout`.
- **Descripción:** Distribuye los mensajes transformados al formato estándar requerido por la GVA.
- **Función:** Asegura que todos los sistemas conectados reciban los expedientes transformados, independientemente del origen del mensaje.
- **Comportamiento:** Los consumidores publican los mensajes procesados directamente en este exchange, el cual los entrega a todas las colas vinculadas, sin necesidad de una clave de enrutamiento.

3.3 Componentes del Sistema

El sistema está compuesto por los siguientes elementos principales:

3.3.1. `CSVConsumer`

- **Función:** Procesa mensajes en formato CSV recibidos desde `primer-exchange`.
- **Comportamiento:**
  - Escucha los mensajes con clave de enrutamiento `generador.csv`.
  - Transforma los datos al formato JSON estándar especificado por la GVA.
  - Calcula la nota media del alumno y añade este dato al mensaje.
  - Publica el mensaje transformado en el exchange de destino `notas.alumnos.anyo`.

3.3.2. `JSONConsumer`

- **Función:** Procesa mensajes en formato JSON recibidos desde `primer-exchange`.
- **Comportamiento:**
  - Escucha los mensajes con clave de enrutamiento `generador.json`.
  - Calcula la nota media del alumno y añade este dato al mensaje recibido.
  - Publica el mensaje actualizado en el exchange de destino `notas.alumnos.anyo`.

3.3.3. `NotasMiddleware`

- **Función:** Coordina y ejecuta los consumidores de forma concurrente, asegurando que ambos estén activos para procesar mensajes en tiempo real.
- **Comportamiento:** Inicia y supervisa las ejecuciones de `CSVConsumer` y `JSONConsumer`, permitiendo que trabajen en paralelo sin interrupciones.

3.4 Flujo de Procesamiento de Datos

El sistema sigue un flujo definido para procesar los expedientes académicos:

1. **Recepción de Mensajes:** Los mensajes son publicados en `primer-exchange` por los generadores del Centro Docente, indicando su formato mediante las claves de enrutamiento.
2. **Transformación de Datos:**
   - Los consumidores procesan los mensajes recibidos, reorganizando los datos para cumplir con el formato estándar de la GVA.
   - Calculan la nota media del alumno, agregando este dato al mensaje.
3. **Publicación en el Exchange de Destino:** Los mensajes transformados se publican en `notas.alumnos.anyo`, el cual los distribuye a todas las colas vinculadas.

3.5 Ventajas del Diseño

El diseño presenta las siguientes ventajas:

- **Escalabilidad:** Permite agregar nuevos consumidores o productores sin afectar a los componentes existentes.
- **Modularidad:** Cada componente funciona de manera independiente, facilitando su mantenimiento y evolución.
- **Flexibilidad:** La configuración de los exchanges permite adaptarse a nuevos formatos o destinos adicionales.
- **Eficiencia:** La ejecución concurrente asegura un procesamiento rápido y continuo.

CAPÍTULO 4
Visualización de Mensajes: Componente Visualizer

En este capítulo se describe la solución implementada para la visualización de los mensajes enviados al exchange `notas.alumno.anyo`. Este componente es fundamental para verificar que los datos transmitidos por el sistema se encuentran en el formato esperado y cumplen con los requisitos establecidos.

El componente, denominado `Visualizer`, permite conectar al exchange mencionado, escuchar los mensajes publicados y mostrarlos en pantalla para su inspección.

4.1 Descripción General

El componente `Visualizer` está diseñado para conectarse al sistema de colas de RabbitMQ y consumir mensajes desde el exchange `notas.alumno.anyo`, que opera bajo el tipo `fanout`. Su objetivo principal es recibir los mensajes transformados por los consumidores y mostrarlos en tiempo real, proporcionando una herramienta para supervisar la correcta sincronización de los datos.

4.2 Exchange Utilizado

4.2.1. Exchange: `notas.alumno.anyo`

- **Tipo:** `fanout`.
- **Descripción:** Este exchange distribuye los mensajes transformados a todas las colas que estén vinculadas a él.
- **Función en `Visualizer`:** El componente se conecta a este exchange, crea una cola temporal, y la vincula al exchange para recibir todos los mensajes publicados.
- **Configuración:**
  - Se utiliza una cola anónima y temporal, que se elimina automáticamente una vez que el componente deja de ejecutarse.
  - No se requiere clave de enrutamiento, dado que el tipo `fanout` envía los mensajes a todas las colas vinculadas.

4.3 Flujo de Operación

El funcionamiento del componente se puede describir en los siguientes pasos:

1. **Conexión al Servidor RabbitMQ:** Se establece una conexión con el servidor RabbitMQ utilizando las credenciales y configuración predefinidas (host, puerto, usuario y contraseña).
2. **Declaración del Exchange:** El componente declara el exchange `notas.alumno.anyo` con el tipo `fanout` para asegurarse de que exista antes de iniciar la recepción de mensajes.
3. **Creación de una Cola Temporal:** Se declara una cola anónima y temporal que se vincula al exchange `notas.alumno.anyo`. Esta cola se elimina automáticamente cuando el componente finaliza su ejecución.
4. **Recepción de Mensajes:** Se define un callback para procesar los mensajes recibidos desde la cola vinculada. Cada mensaje se imprime en la consola, proporcionando una representación directa de los datos.
5. **Ejecución Continua:** Para garantizar que el componente esté siempre escuchando los mensajes, se utiliza un mecanismo de bloqueo (`CountDownLatch`) que mantiene viva la ejecución del hilo principal hasta que el proceso es detenido manualmente.

4.4 Ventajas del Componente

El diseño del componente `Visualizer` presenta varias ventajas:

- **Simplicidad:** El componente es sencillo de implementar y utilizar, proporcionando una herramienta directa para supervisar el flujo de mensajes.
- **Flexibilidad:** Al usar una cola temporal, se evita la necesidad de configuraciones persistentes, facilitando su uso en diferentes entornos.
- **Fiabilidad:** El uso del exchange `fanout` garantiza que todos los mensajes sean entregados a la cola vinculada.

CAPÍTULO 5
Resultados

En este capítulo se presentan los resultados obtenidos tras la ejecución de los diferentes programas desarrollados en este proyecto. Para ello, se incluyen capturas de pantalla, registros y salidas relevantes que evidencian el correcto funcionamiento de los componentes implementados, así como su capacidad para cumplir con los objetivos planteados. Además, se analizan los resultados para evaluar el desempeño y la eficacia de las soluciones propuestas.

5.1 Script de inicio

La Figura 5.1 muestra el script encargado de inicializar todos los programas desarrollados en esta solución. Para garantizar su correcto funcionamiento, es fundamental que este script se encuentre en el mismo directorio que el resto de los programas. Esta disposición asegura la adecuada integración y coordinación de los diferentes módulos del sistema, facilitando la automatización de su ejecución.

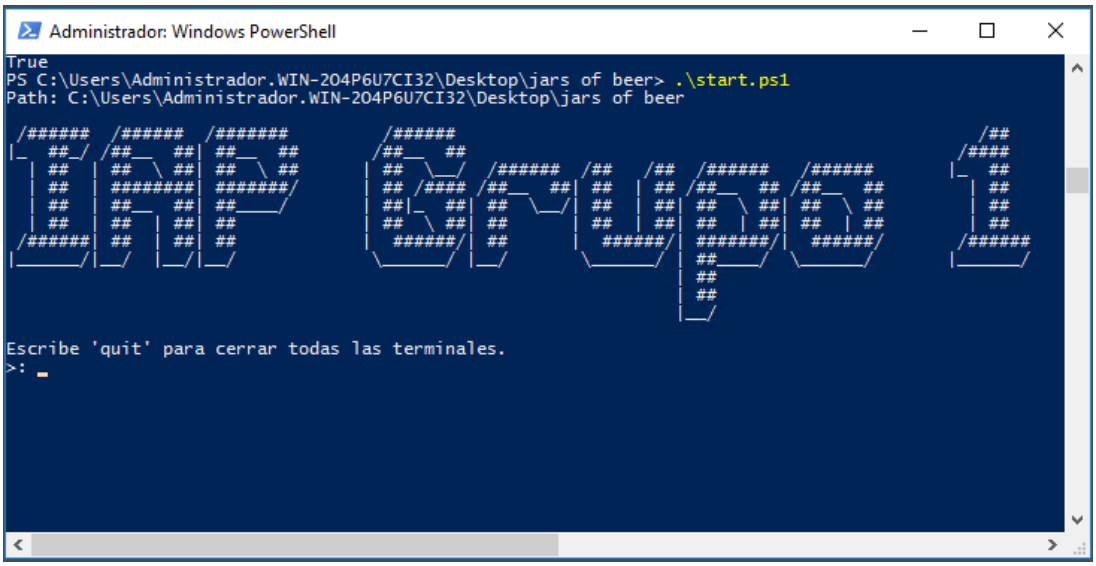

Figura 5.1: Script de inicio de los distintos programas

5.2 Resultado tras la ejecución

La Figura 5.2 muestra el resultado de la ejecución de todos los programas desarrollados en el sistema. Esta imagen permite verificar el correcto funcionamiento e integración de los distintos componentes, evidenciando el cumplimiento de los objetivos planteados en el proyecto.

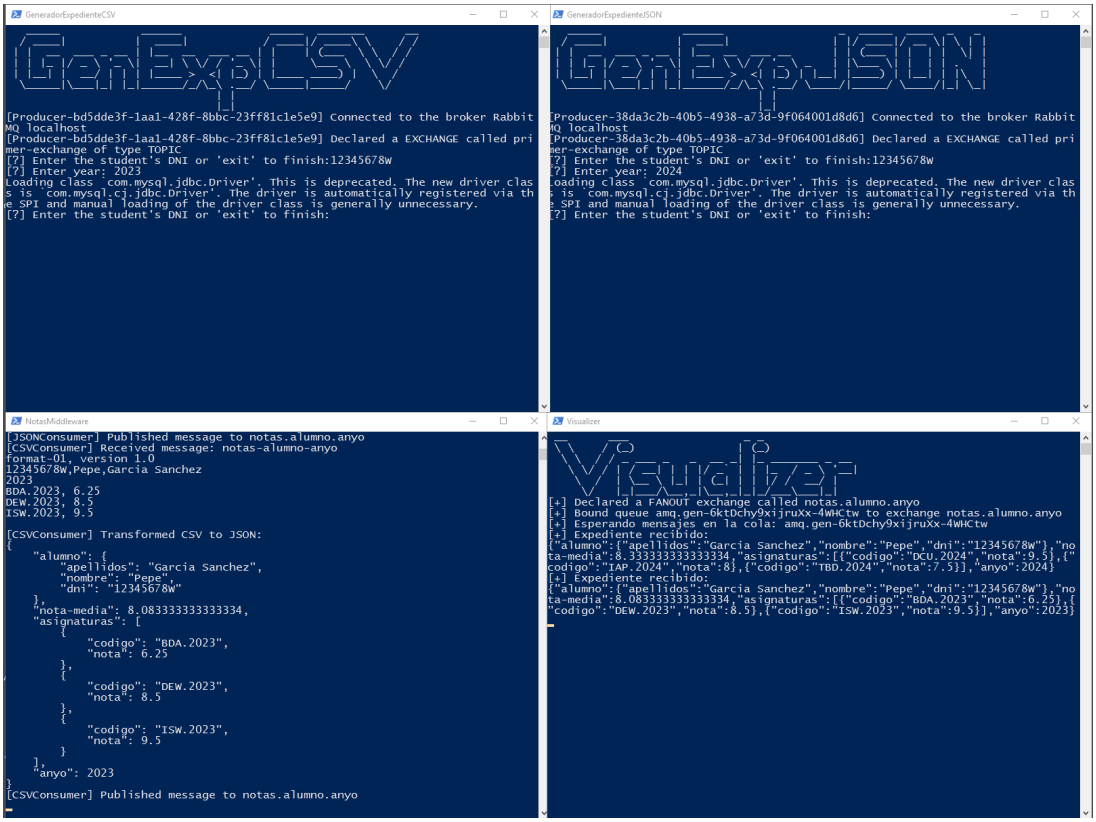

Figura 5.2: Resultados de la ejecución de los programas

CAPÍTULO 6
Conclusiones

El proyecto presentado demuestra la implementación de una solución robusta y eficiente para la sincronización de expedientes académicos entre un Centro Docente y la GVA, mediante el uso de RabbitMQ como middleware de mensajería. A través de un diseño modular y flexible, se logró automatizar y estandarizar el intercambio de información, permitiendo cumplir con los requisitos de integración planteados.

La implementación de los generadores de expedientes académicos en formatos CSV y JSON asegura la adaptabilidad a distintas necesidades, mientras que los consumidores desarrollados permiten transformar y procesar los datos según los estándares definidos por la GVA. Además, la inclusión del cálculo de la nota media como una funcionalidad adicional aporta un valor significativo al sistema al proporcionar información procesada y de fácil interpretación.

El uso de RabbitMQ como sistema de colas de mensajería destaca por garantizar un flujo eficiente y organizado de los mensajes entre los diferentes componentes. La configuración de *exchanges*, como `primer-exchange` y `notas.alumno.anyo`, asegura un enrutamiento efectivo, permitiendo la distribución de mensajes tanto en formato de origen como transformados.

La solución propuesta presenta varias ventajas clave: escalabilidad, modularidad, flexibilidad y eficiencia. Estas características no solo garantizan el correcto funcionamiento del sistema en su estado actual, sino que también permiten su futura ampliación y adaptación a nuevas exigencias o formatos. La utilización de un componente visualizador (`Visualizer`) permite validar la integración en tiempo real, brindando a los usuarios y desarrolladores una herramienta de supervisión intuitiva y útil.

Finalmente, los resultados obtenidos durante la ejecución del sistema evidencian el cumplimiento exitoso de los objetivos planteados, destacando la capacidad del sistema para manejar flujos de datos complejos de manera automatizada, estandarizada y confiable. Este proyecto refuerza la importancia de utilizar arquitecturas basadas en mensajería en el diseño de soluciones de integración y proporciona un modelo efectivo para resolver problemas similares en el ámbito de la ingeniería informática.