Arquitectura de graphPaname

Amin Kasrou Aouam, Alejandro Calle González-Valdés

UNIVERSIDAD
DE GRANADA

2020-04-11

Índice

Introducción

El patrón arquitectónico es una solución general a un problema de ingeniería de software. Es esencial elegir un diseño adecuado, dado que esto nos facilita la implementación, además de evitar antipatrones de diseño.

Descripción del sistema

El objetivo de este proyecto es crear un sistema de información que permite visualizar datos de una Smart City. Nos centraremos en la ciudad de París, dado que ofrece una gran cantidad de recursos públicos y actualizados.

Es un sistema de procesamiento de datos, que podría enmarcarse en la disciplina de ciencia de datos. El funcionamiento de éste es lineal, a partir de una entrada (fuentes de datos), pasamos por distintas etapas de procesamiento, hasta llegar a la salida deseada.

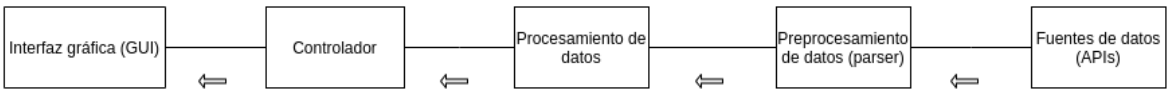

Figura 1: Diseño del sistema

Requisistos del sistema

Requisitos funcionales

1. **RF1**: Obtención de datos

   El sistema debe recolectar los datos necesarios, para su posterior procesamiento, desde fuentes externas de información.

2. **RF2**: Preprocesamiento de datos

   El sistema debe adaptar la información procedente de distintas fuentes a su representación interna de datos.

3. **RF3**: Procesamiento de datos

   El sistema debe clasificar los datos, según los parámetros que decida el usuario, para obtener información relevante.

4. **RF4**: Creación de gráficas

   El sistema debe facilitar la información de forma visual, con distintos tipos de gráficas.

Requisitos no funcionales

1. **RNF1**: Seguridad

   La página web de consulta será accesible únicamente mediante HTTPS

2. **RNF2**: Escalabilidad

   Se podrá aumentar el rendimiento de sistema, con escalbilidad horizontal y/o vertical

3. **RNF3**: Disponibilidad

   El sistema estará disponible 24/7, dado que usaremos un clúster de alta disponibilidad

4. **RNF4**: Tolerancia a fallos

   Se usará un clúster para permitir que siempre funcione el sistema, aunque falle una parte.

5. **RNF5**: Copias de seguridad

   Se harán copias de seguridad diarias del sistema, además de enviarlas a otro servidor en caso de que se pierdan los datos locales de backup

6. **RNF6**: Rotación de logs

   Se eliminarán los logs del sistema antiguos, cada semana

Elección del patrón arquitectónico

A partir de la descripción del sistema, y de los requisitos de éste, procedemos a elegir (especificando las razones que nos llevan a nuestra decisión) el patrón arquitectónico adecuado.

Como ya establecimos, nuestro sistema funciona de forma lineal, es decir que tenemos distintos módulos agrupados, a partir de una entrada pasa por distintas etapas hasta llegar a la salida deseada.

Nuestro primer instinto fue elegir un patrón de diseño por capas, dado que nos permite separar la presentación de la lógica interna. Esta abstracción permite desacoplar la parte de procesamiento con

la parte gráfica, un paradigma muy interesante que permite reemplazar un módulo sin reemplazar todo el sistema.

Tras analizar que cada salida de un módulo, era la entrada del módulo siguiente, nos dimos cuenta que estábamos ante una *pipeline*, y decidimos adoptar el patrón aquitectónico de **flujos de datos**.

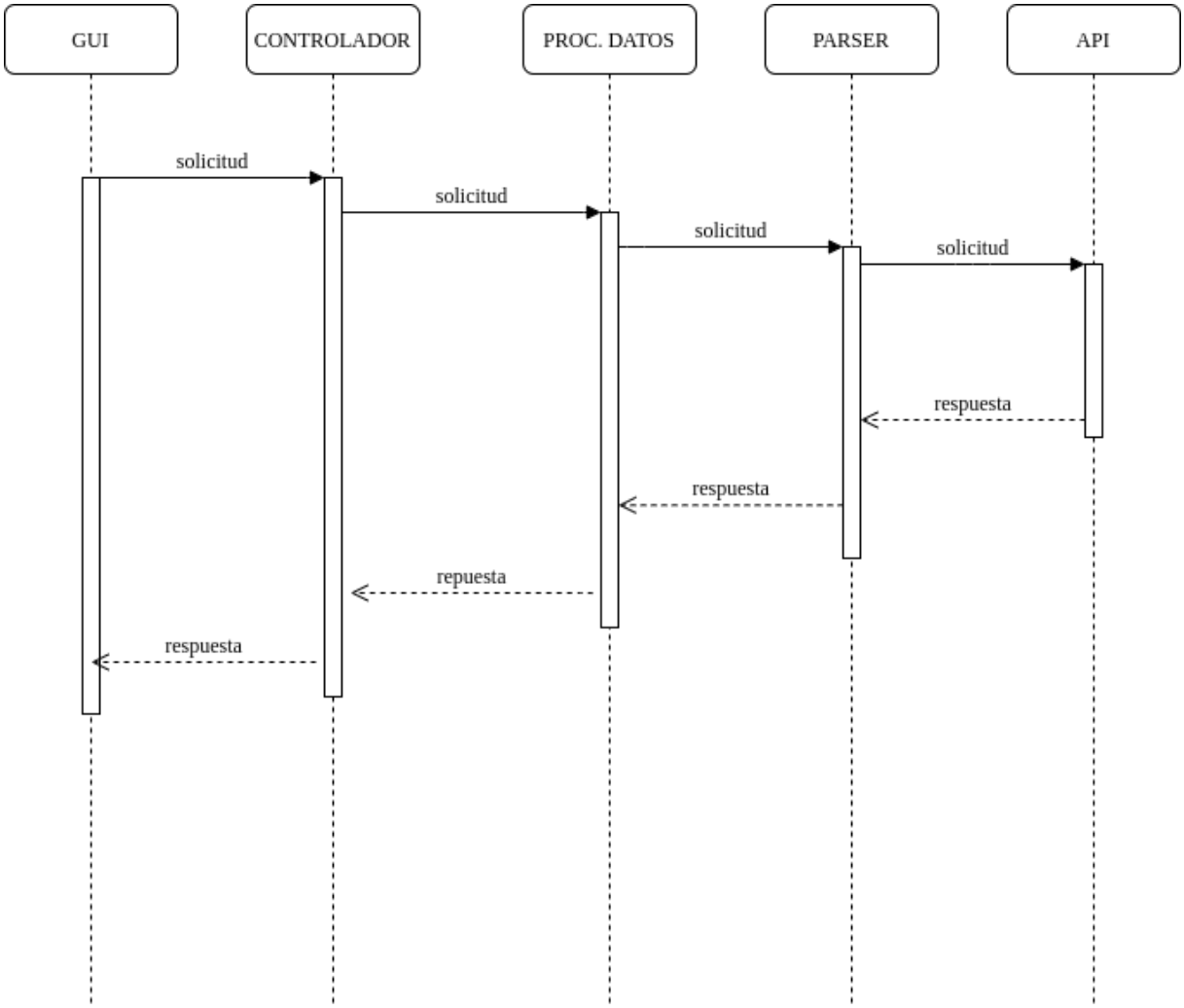

Figura 2: Pipeline del sistema

Descripción de los módulos

Nuestro objetivo es establecer un diseño modular, no monolítico, con el fin de poder reemplazar los componentes individuales, según las necesidades.

Para ello, dividimos nuestro sistema en 5 módulos:

1. Interfaz gráfica (GUI)
2. Controlador
3. Procesamiento de datos
4. Preprocesamiento
5. Fuentes de datos

Procedemos a desglosar las funcionalidades de cada uno de ellos en la siguiente figura.

Figura 3: Descripción de los componentes