UNIANDES LABS ORCHESTRATION SYSTEM

Esteban Gonzalez Ruales
Asesor: Camilo Escobar Velásquez
Universidad de los Andes
Ingeniería de Sistemas y Computación
Proyecto de grado

La Universidad de los Andes está en constante necesidad de modernizarse y mejorar sus capacidades para satisfacer la creciente curiosidad y capacidad de aprender de sus estudiantes. Es por esto por lo que la universidad está en constante búsqueda de proyectos que puedan aumentar las oportunidades educativas para los mismos. De aquí sale la motivación de hacer un sistema de orquestación para automatizar los laboratorios que usan los estudiantes de manera remota. Esto permitirá a estudiantes de diversas disciplinas en ingeniería hacer uso de recursos de laboratorios de la universidad de una manera más fácil y rápida. De esta manera, los procesos de laboratorios se verán acelerados y los estudiantes no tendrán que gastar tanto tiempo en los laboratorios.

OBJETIVOS

Objetivos generales

- Aumentar la capacidad de procesamiento de los laboratorios remotos de la universidad
- Aumentar el número de estudiantes concurrentes pueden usar los laboratorios remotos

Objetivos específicos

- Implementar un sistema que permita que las peticiones enviadas lleguen a sus respectivos ambientes de ejecución y los resultados de la misma sean guardados
- Poder acceder al resultado de la ejecución de los archivos enviados

¿Cómo?

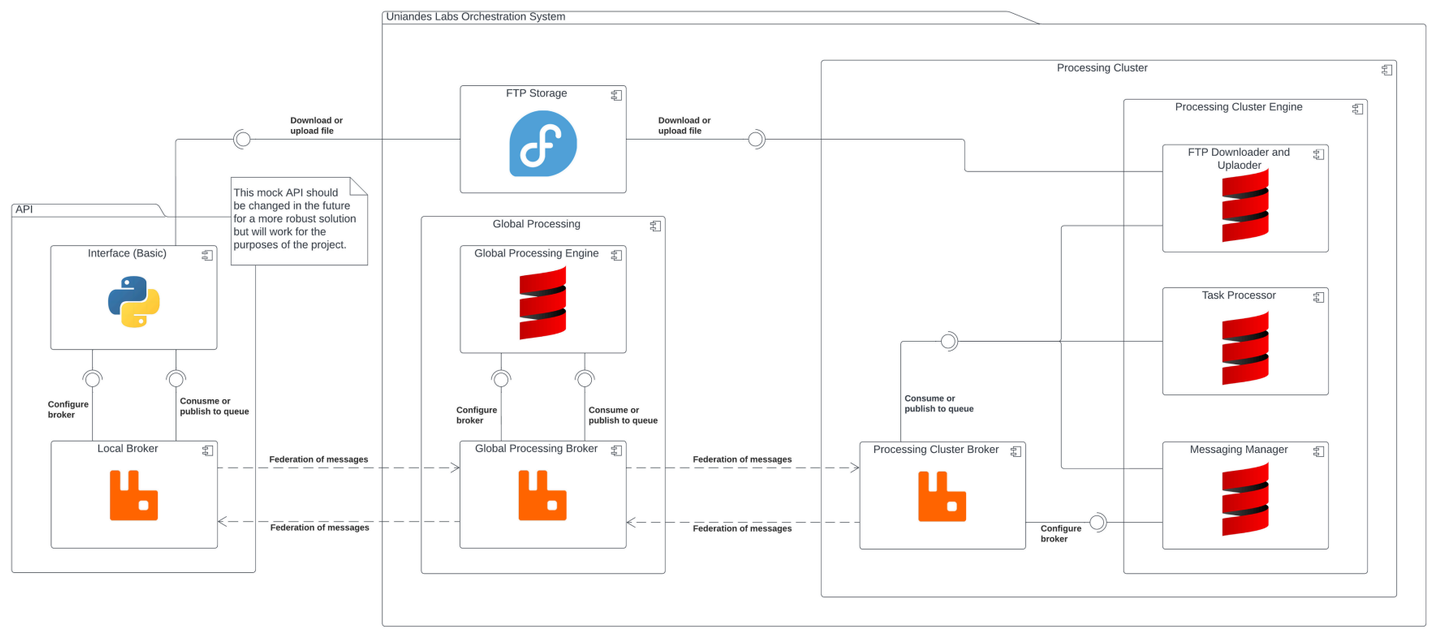
- Mediante un sistema autónomo que recibirá peticiones de los estudiantes para ser ejecutadas en el laboratorio
- Implementación de máquinas virtuales que realizan simulaciones sobre las peticiones que los estudiantes hacen al laboratorio

¿Por qué?

- La alta demanda de peticiones hace imposible que todas estas se ejecuten en el ambiente en el que se deben ejecutar en el tiempo en el que los estudiantes lo requieren

DISEÑO E IMPLEMENTACIÓN

Dada la naturaleza del proyecto varias decisiones de diseño e implementación fueron tomadas. En primer lugar, se evaluó que el sistema debería tener un **alto nivel de funcionalidad y un bajo nivel de fallas**, lo cual llevó a buscar un lenguaje de programación que fuera estáticamente tipado y compilado. Esto beneficia el proyecto debido a que un lenguaje estáticamente tipado reduce los errores en tiempo de ejecución debido al chequeo de tipos al momento de compilación mientras que un lenguaje compilado aumenta el desempeño del sistema. Por otro lado, se buscaba un lenguaje que tuviera un ecosistema de librerías suficientemente grande y robusto para poder implementar todas las funcionalidades necesarias del proyecto. Dadas estas consideraciones se decidió realizar el proyecto en **Scala**. Además de cumplir con los requisitos anteriormente mencionados, Scala se desempeña como lenguaje de programación funcional, lo cual permite un alto entendimiento del código. Por otro lado, dado que se busca **paralelizar y distribuir las peticiones** que llegan al sistema, se decidió implementar dos componentes separados que solucionaran este problema. El GPE (Global Processing Engine) es el encargado de distribuir las peticiones entre los PCs (Processing Clusters) y será el punto único de comunicación con el sistema que reciba las peticiones por parte de los estudiantes. Por otro lado, los PCs se encargan de consumir peticiones hechas al GPE y realizar todo el proceso de descarga, revisión y ejecución de las peticiones para finalmente devolver el resultado al GPE, el cual lo enviará al sistema que se comunica con los estudiantes.



Arquitectura general del proyecto

RESULTADOS

Los resultados del proyecto demostraron una **buena ejecución** del mismo dado que se cumplieron los objetivos específicos propuestos para contribuir a los objetivos generales. Se logró implementar un sistema que exitosamente procesara peticiones hechas al sistema. Para efectos de pruebas del sistema, únicamente se implementó el ambiente de ejecución de Cypress dado que la implementación de estos ambientes estaba fuera del alcance de este proyecto. Sin embargo, las peticiones de Cypress que se enviaban al sistema pasaban por todo el flujo necesario, eran ejecutadas y sus resultados eran devueltos por medio del mismo sistema. Esto demuestra que el sistema tiene un buen funcionamiento independientemente del ambiente de ejecución que se vaya a ejecutar. Con lo anterior **se deja un sistema funcional y robusto** como el motor central de la orquestación de los laboratorios de la universidad.

TRABAJO FUTURO

Aunque el sistema de orquestación implementado en este proyecto es crucial para el funcionamiento del sistema en general, aún hay varias cosas que son necesarias para la implementación y el buen funcionamiento del sistema total. En este momento, únicamente se tiene implementado el ambiente de ejecución de Cypress, lo cual significa que el sistema solo puede recibir peticiones que puedan correr en este ambiente de ejecución. Es necesario **implementar más ambientes de ejecución** además de complementar el ya existente. Por otro lado, es necesario que se realice el despliegue del sistema en contenedores para poderlos desplegar en cualquier lugar. El despliegue para componentes que solo usan Scala ya esta hecho, sin embargo, falta **agregarle dependencias a los archivos de docker** que usan otras tecnologías para que funcionen con ambientes de ejecución tales como Cypress y otros. Adicionalmente, es necesario implementar las conexiones con los respectivos equipos que también servirán como ambientes de ejecución tales como el brazo robótico, las impresoras 3D y el grid eléctrico de la universidad. Finalmente, es necesario **implementar el API y la interfaz gráfica** con la cual los usuarios se puedan comunicar con el sistema de orquestación. Con la implementación de todos estos elementos se tendrá un sistema de orquestación completado, funcional y utilizable por parte de los estudiantes y la universidad.