SD	Sistemas Distribuidos
24/25	Práctica no guiada: Seguridad y API Rest
	EasyCab

Preámbulo

El objetivo de esta práctica es que los estudiantes implementen, por un lado, dentro de los principios de arquitectura SOA (Service Oriented Architecture), la tecnología de comunicación de basada en servicios REST y por otro algunos de los principios de seguridad vistos en teoría.

Para ello, se consumirán una serie API Rest ya establecidos por un tercero y, por otro, se creará y expondrá otro API Rest desde un back que sea consumido por un front.

Adicionalmente, se implementarán tres aspectos relacionados con la seguridad: cifrado de canal, autenticación segura y principios de auditoría.

Se partirá de la misma práctica ya generada en la primera parte de la asignatura con la misma funcionalidad expuesta.

Especificación

Descripción funcional

La funcionalidad que deberá implementarse será idéntica a la implementada en la release 1 con las siguientes modificaciones:

1- EC_Central deberá comunicarse con un módulo externo EC_CTC (City Traffic Control) que proporcionarán la viabilidad o no de circular por la ciudad donde se presta el servicio.

2- Se añadirá un nuevo módulo llamado EC_CTC (City Traffic Control) el cual determinará el estado del tráfico. Para ello se comunicará vía API con sistemas externos (ChatGPT y Openweather) como se detalla en los siguientes apartados.

3- Se añadirá un nuevo módulo llamado EC_Registry que permitirá registrar un nuevo Taxi o darlo de baja. Este módulo se comunicará con los Taxis vía API_Rest.

4- Se creará un Front (mediante una simple página web) la cual mostrará el estado del mapa a cualquiera que la invoque. Es, por tanto, una web pública.

5- Se implementarán los siguientes mecanismos de seguridad:
   - Autenticación segura entre los Taxis y el Registry: cifrado del canal y protección segura de credenciales.
   - Auditoría de eventos en la Central.
   - Cifrado de los datos entre la Central y los Taxis.

Diseño técnico

Partiendo del diseño técnico ya implementado en la primera parte de la práctica, se modificará y ampliará el mismo para contemplar la siguiente arquitectura:

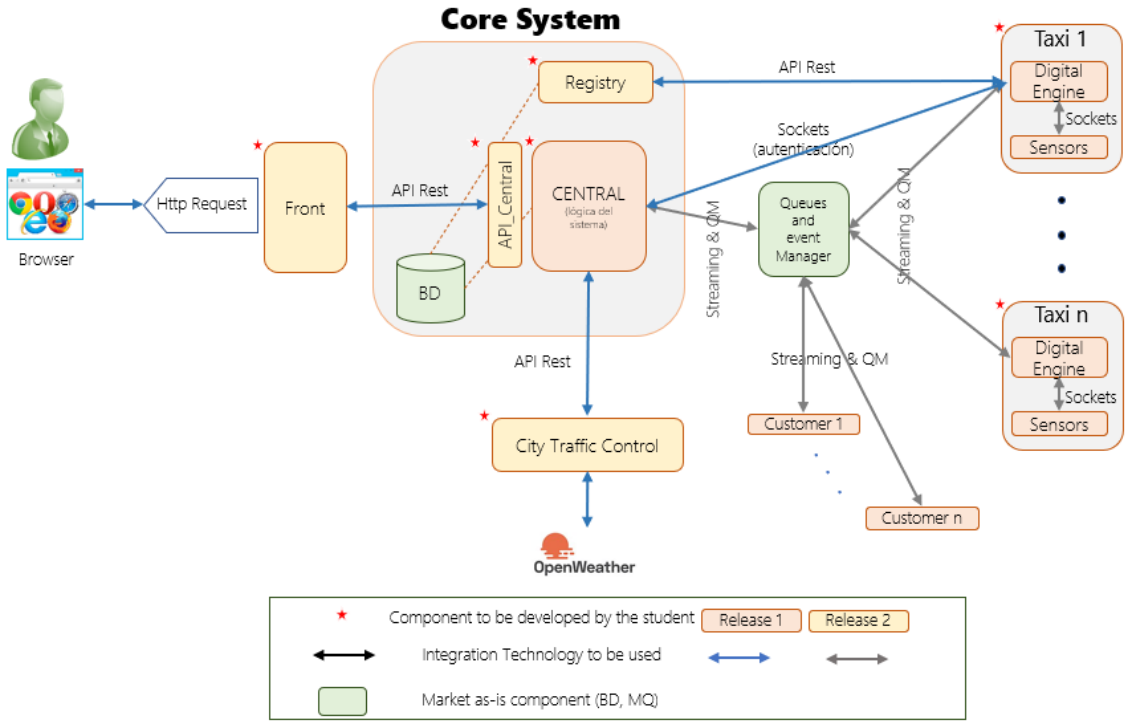


Figura 1. Esquema conceptual del Sistema software, interconexiones entre los componentes y tipos de interfaces.

Al igual que en la primera parte de la práctica, los componentes pueden ser desarrollados en el lenguaje de preferencia del estudiante: Java, C/C++, . NET, Python, etc. asumiendo el estudiante la responsabilidad de su conocimiento y forma de desplegarlo en el laboratorio.

Core System

Central:

Será idéntico al ya realizado en la primera parte con las siguientes modificaciones:

1- **Consumo de API_rest de un gestor de tráfico de la ciudad:** EC_Central se conectará a un nuevo módulo llamado EC_CTC (City Traffic Control) el cual le informará si el tráfico en la ciudad representada por el mapa es viable o no. Para ello, EC_Central, **cada 10 segundos** consumirá un API que ofrecerá EC_CTC el cual devolverá OK si la situación en la ciudad permite el tráfico o KO en caso de que la circulación no sea viable. Más adelante se indica el funcionamiento de EC_CTC.

2- **Implementación de la autenticación entre EC_Central y los Taxis**: En el momento de la autenticación, la central devolverá al taxi un token el cual deberá ser usado

por este en todos los mensajes que envíe a la Central. La central deberá comprobar que dicho token está en vigor antes de responder al taxi. Los tokens no tendrán caducidad temporal pero **sí expirarán cuando el taxi reciba una orden de la central de volver a la base (posición 1,1) y desconectarse**. Los tokens **NO podrán ser almacenados por el taxi ni tampoco de forma permanente en la BD**. Si se realiza almacenamiento en la BD será temporal y deberá quedar totalmente limpia de tokens una vez hayan expirado. Tal y como se refleja en el diagrama de arquitectura conceptual, la autenticación seguirá siendo por sockets pero el alumno que lo desee podrá implementar un API Rest entre Taxi y Central a tal propósito lo que, en la práctica profesional sería más correcto. En este caso Central expondría un API de Autenticación que el Taxi consumiría.

3- **Implementación de cifrado en el canal entre EC_Central y los Taxis**: Se establecerá un sistema de cifrado del canal que los Taxis depositan en los Topics de las colas para evitar ataques del tipo MITM (Man In The Middle). Se realizará un sencillo sistema de cifrado con intercambio de claves y certificados (simétrico o asimétrico) elegible por el estudiante con su justificación adecuada. *Nota: Kafka dispone de 3 componentes que permiten implementar mecanismos de seguridad a este propósito. Dichos componentes son: Cifrado de datos SSL/TLS, Autenticación SSL o SASL y Autorización mediante ACL. No se exige al estudiante la incorporación en la práctica de estos componentes si bien aquellos que lo deseen pueden hacerlo siendo valorado positivamente.*

4- Implementará un **registro de auditoría de todos los eventos que sucedan**, indicando, de forma estructurada, los datos de dicho evento como se indica a continuación:
a. Fecha y hora del evento.
b. Quién y desde dónde se produce el evento: IP de la máquina que genera el evento (ej. IP del taxi, IP del Usuario)
c. Qué acción se realiza: Autenticación o intentos de autenticación fallidos o no, Incidencias durante el servicio, cambio de la situación del tráfico, usuarios o taxis bloqueados, errores, etc.
d. Parámetros o descripción del evento.
*Se deberá incluir un API REST que exponga la información de la auditoría de seguridad. Se debe inluir un Front Web que acceda a dicha API a través de un navegador y la visualice de manera amigable.Se valorará la actualización en la parte del cliente (navegador Web) en tiempo real.
Nota informativa: Los sistemas profesionales deben incorporar estos conceptos de auditoría mediante herramientas específicas que integran un SIEM (Security Information and Event Management).

API_Central:

Este componente expondrá un API_Rest con los métodos oportunos (GET, PUT, DELETE,…) que permitirá desde cualquier componente externo consultar el estado de los Taxis, Usuarios y el mapa en curso. Al igual que en la anterior versión de la práctica, el interfaz front que consuma este API deberá contener todos los elementos necesarios para visualizar claramente el estado de situación de todos los elementos del sistema (taxis, usuarios, mensajes de error, estado del CTC, etc).

Registry (EC_Registry):

EC_Registry será el módulo mediante el cual los taxis pueden darse de alta o baja en el sistema. Solo dispondrá de las opciones de alta y baja de un taxi.

Los taxis, previo a su autenticación y posterior prestación de servicios, deberán realizar una petición de alta para registrar su ID en el sistema o darlo de baja. Sin este proceso los taxis no podrán autenticarse ni prestar servicios. Ante un intento de autenticación sin previo registro recibirán una denegación de esta.

El mecanismo de integración con los taxis se realizará mediante un API_Rest que implementará los métodos oportunos (GET, PUT, DELETE, …) para el registro de los Taxis.

La comunicación entre EC_Registry y EC_DE (Taxis) deberá realizarse mediante el establecimiento de un canal seguro (HTTPS, SSL, RSA, …).

La identificación de los Taxis podrá realizarse en base a un certificado que contenga las credenciales de este.

Base de Datos:

Podrá ser accesible tanto EC_Central como EC_Registry para compartir y actualizar la información de los taxis.

Digital Engine (EC_DE)

Adicionalmente a las funcionalidades ya disponibles, este módulo implementará una nueva opción (disponible en un menú) para poder conectarse al EC_Registry y registrarse en el sistema. Este proceso se implementará mediante el consumo de un API.

La conexión y autenticación deberá realizarse de forma segura evitando la exposición en claro de los datos de identificación del taxi.

City Traffic Control (EC_CTC)

Como se ha indicado en el apartado de descripción funcional de este documento, se añadirá un nuevo módulo al sistema llamado EC_CTC (City Traffic Control) el cual determinará la viabilidad de circulación en la ciudad donde se presta el servicio devolviendo un OK en caso que haya total normalidad o un KO en caso que la circulación no sea viable y los Taxis deban volver a la base.

La viabilidad de la circulación vendrá determinada por la temperatura de la ciudad.

Para ello, el módulo EC_CTC dispondrá, a efectos de simulación, mediante una opción de menú, la posibilidad de indicar una ciudad donde se presta el servicio.

EC_CTC se conectará al servidor de clima externo OPENWEATHER para preguntarle el tiempo que hace en la ciudad.

Si la temperatura de la ciudad está por debajo de los 0º C, EC_CTC devolverá un mensaje de KO al módulo Central. En cualquier otro caso enviará un OK.

Nota: A efectos de la corrección, **la ciudad podrá ser cambiada a voluntad del profesor en cualquier momento** mediante la opción de menú mencionada, sin necesidad de reiniciar ninguna parte del sistema.

Front

Este módulo consistirá en una simple página web que, haciendo peticiones al API_Central muestre el mapa en curso y el estado de los Taxis y los Usuarios.

Como se ha comentado anteriormente, este interfaz front deberá contener todos los elementos necesarios para visualizar claramente el estado de situación de todos los elementos del sistema: taxis (incluyendo su estado de registro, activación y token), usuarios, mensajes de error de cualquier parte del sistema, mensajes de estado del CTC, alertas ante cualquier fallo de cualquier módulo del sistema, etc).

Nota: Aunque al igual que en el resto de componentes el estudiante podrá elegir la tecnología de su preferencia se recomienda, por su sencillez, el uso de Node js con arreglo a las indicaciones que se han ofrecido en la práctica guiada entregada.

Sensores y Usuarios

No cambian su funcionalidad respecto de la entrega anterior.

Servidor de clima (OPEN WEATHER)

Mediante la funcionalidad aportada por la plataforma OPEN WEATHER (https://openweathermap.org/) es posible obtener múltiples datos relacionados con el clima de cualquier parte del mundo.

A través de su API el EC_Central podrá acceder a dichos datos.

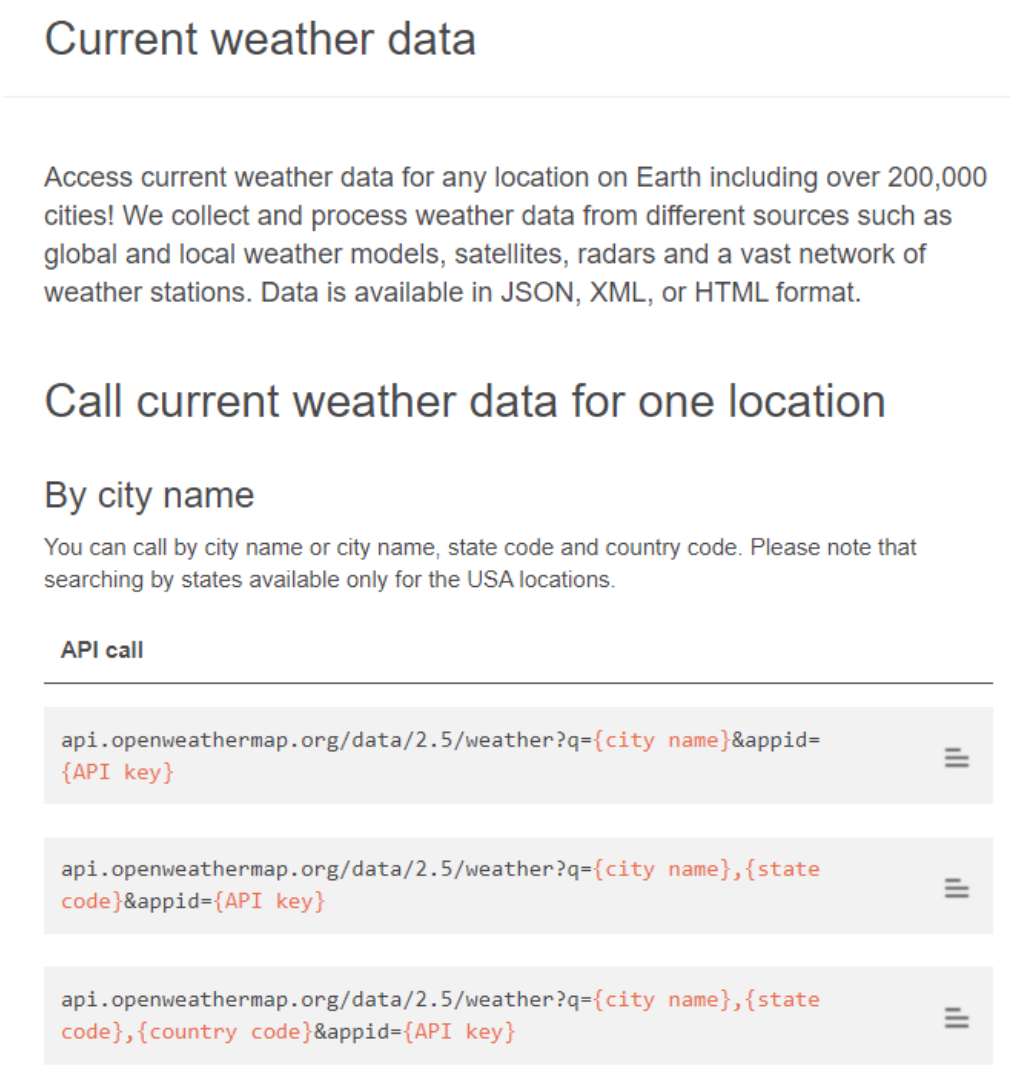
Para ello bastará con que el estudiante se registre en la versión FREE, solicite el API key (Get API Key) y realizar una petición para cada ciudad que quiera consultar.

Current weather and forecasts collection

Free	Startup	Developer	Professional	Enterprise
Get API key	40 USD / month Subscribe	180 USD / month Subscribe	470 USD / month Subscribe	2,000 USD / month Subscribe
60 calls/minute 1,000,000 calls/month	600 calls/minute 10,000,000 calls/month	3,000 calls/minute 100,000,000 calls/month	30,000 calls/minute 1,000,000,000 calls/month	200,000 calls/minute 5,000,000,000 calls/month
Current Weather	Current Weather	Current Weather	Current Weather	Current Weather
Minute Forecast 1 hour*	Minute Forecast 1 hour**	Minute Forecast 1 hour	Minute Forecast 1 hour	Minute forecast 1 hour
Hourly Forecast 2 days*	Hourly Forecast 2 days**	Hourly Forecast 4 days	Hourly Forecast 4 days	Hourly Forecast 4 days
Daily Forecast 7 days*	Daily Forecast 16 days	Daily Forecast 16 days	Daily Forecast 16 days	Daily Forecast 16 days
National Weather Alerts*	National Weather Alerts**	National Weather Alerts	National Weather Alerts	National Weather Alerts
Historical weather 5 days*	Historical weather 5 days**	Historical weather 5 days	Historical weather 5 days	Historical weather 5 days
Climatic Forecast 30 days	Climatic Forecast 30 days	Climatic Forecast 30 days	Climatic Forecast 30 days	Climatic Forecast 30 days
Bulk Download	Bulk Download	Bulk Download	Bulk Download	Bulk Download
Basic weather maps	Basic weather maps	Advanced weather maps	Advanced weather maps	Advanced weather maps
Historical maps	Historical maps	Historical maps	Historical maps	Historical maps
Global Precipitation Map	Global Precipitation Map	Global Precipitation Map	Global Precipitation Map	Global Precipitation Map
Road Risk API	Road Risk API	Road Risk API	Road Risk API	Road Risk API
Air Pollution API	Air Pollution API	Air Pollution API	Air Pollution API	Air Pollution API
Geocoding API	Geocoding API	Geocoding API	Geocoding API	Geocoding API
Weather widgets	Weather widgets	Weather widgets	Weather widgets	Weather widgets
Uptime 95%	Uptime 95%	Uptime 99.5%	Uptime 99.5%	Uptime 99.9%

Modelos de suscripción de Open Weather

De todas las posibilidades que ofrece el API es suficiente con que se acceda a las funcionalidades que se encuentran en los métodos de “Current weather data” como se aprecia en la imagen siguiente. Toda la información al respecto se encuentra disponible en la página web del proveedor (https://openweathermap.org/current):



API Current Weather Data

El JSON de respuesta de las peticiones que se muestran en la anterior imagen es el siguiente del cual solo necesitaremos la temperatura.

JSON

Example of API response


```
{
  "coord": {
    "lon": -122.08,
    "lat": 37.39
  },
  "weather": [
    {
      "id": 800,
      "main": "Clear",
      "description": "clear sky",
      "icon": "01d"
    }
  ],
  "base": "stations",
  "main": {
    "temp": 282.55,
    "feels_like": 281.86,
    "temp_min": 280.37,
    "temp_max": 284.26,
    "pressure": 1023,
    "humidity": 100
  },
  "visibility": 16093,
  "wind": {
    "speed": 1.5,
    "deg": 350
  },
  "clouds": {
    "all": 1
  },
  "dt": 1560350645,
  "sys": {
    "type": 1,
    "id": 5122,
    "message": 0.0139,
    "country": "US",
    "sunrise": 1560343627,
    "sunset": 1560396563
  },
  "timezone": -25200,
  "id": 420006353,
  "name": "Mountain View",
  "cod": 200
}
```

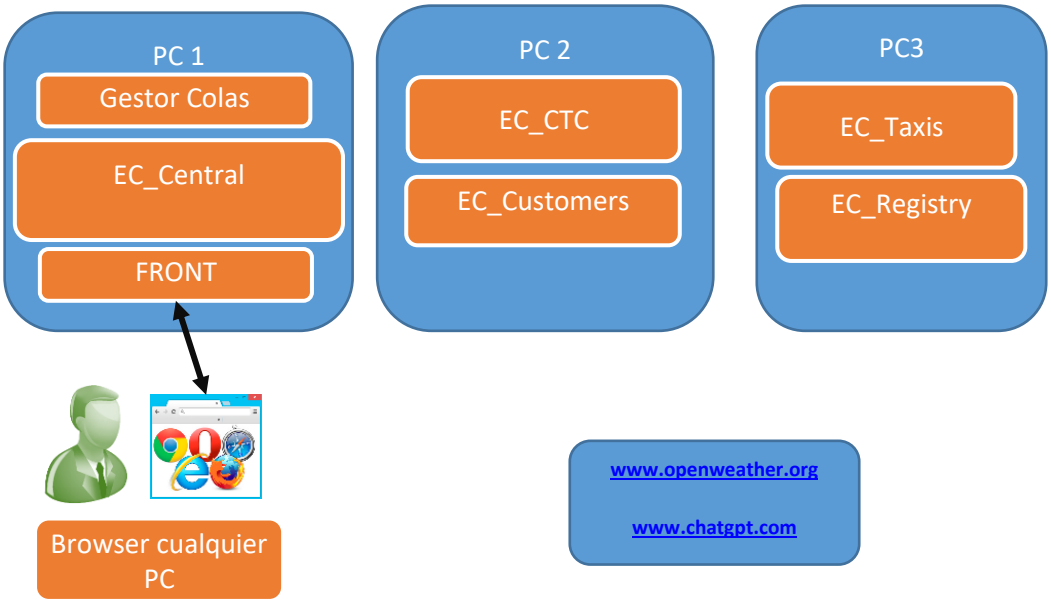

Temperatura en ºK

Ciudad

JSON de respuesta del request al API Current Weather Data

Guía mínima de despliegue

Para la correcta evaluación de la práctica es necesario comprobar que la aplicación distribuida solicitada es desplegada en un entorno verdaderamente distribuido. Es por ello que para su prueba es necesario al menos 3 PCs distintos en los que se desplegarán los componentes solicitados proporcionando el siguiente escenario:



Escenario físico para el despliegue de la práctica.

Entregables y evaluación

La evaluación de la práctica se realizará en los laboratorios. **Se podrá realizar en grupos de hasta 2 personas sin perjuicio de que, durante el momento de la corrección, el profesor pueda preguntar a cualquier estudiante del grupo por cualquier aspecto de cualquiera de los módulos**. Los estudiantes deben desplegar por ellos mismos la práctica que resuelve el enunciado anterior. Deben desplegar un sistema completo con todos los módulos interconectados entre sí. **Este requisito es indispensable para poder realizar la corrección**. Además, deben poderse evaluar positiva o negativamente todos los apartados que aparecerán en la Guía de corrección que se entregará a tal propósito. Cada uno de los apartados puntúa de forma variable, por tanto, cada apartado no implementado o que no pueda comprobarse su correcto funcionamiento no podrá ser tenido en cuenta y por tanto no puntuara. Los estudiantes deberán presentar para la evaluación el documento **"Guía de corrección"** cumplimentado para que el profesor pueda validar los apartados implementados.

Los estudiantes deberán entregar, además, mediante la funcionalidad de evaluación del UACloud antes de la fecha establecida a su profesor de prácticas una **memoria de prácticas**, con el código fuente y compilados generados, así como un documento donde se detalle la siguiente información. El formato es libre, pero debe ser un documento ordenado y debidamente formateado, cuidando la redacción y ortografía.

- Portada con los nombres, apellidos y DNI de los estudiantes, año académico y el título de la práctica.
- Un informe donde se indique el nombre de los componentes software desarrollados y una descripción de cada uno de ellos, explicando y enviando además el código fuente de todos ellos.
- El detalle, paso a paso, de una guía de despliegue de la aplicación, que deberá ser la misma que utilice cuando haga la corrección de la práctica.
- Capturas de pantalla que muestren el funcionamiento de las distintas aplicaciones conectadas.

Cada profesor de prácticas podrá solicitar a los estudiantes cualquier otra evidencia que el profesor considere adecuada para poder formalizar la evaluación.

La fecha de entrega será en la semana del 16/12/2024.