MAXCT

ENTREGABLE R9

"ANÁLISIS DE LAS NECESIDADES DE SOFTWARE"

19 de Octubre, 2015

F.E.D.E.R.

JUNTA DE ANDALUCIA
CONSEJERÍA DE FOMENTO Y VIVIENDA

UNIVERSIDAD DE MÁLAGA

NEO

Movilidad Inteligente: Wifi, Rutas y Contaminación
Proyecto I+D+i Ene-Oct, 2015. N° GGI3003IDII. OTRI-UMA # 8.06/5.47.4356.

Contenido

1. Introducción

En este entregable se describirán las necesidades **software** requeridas tanto durante el desarrollo como el despliegue de las dos herramientas (HITUL y CTPATH). Asimismo, incluimos detalles del software finalmente utilizado (y requerido para el uso de las herramientas) en el proyecto en general.

2. Infraestructura software para el desarrollo

En esta sección se describe la infraestructura utilizada para el desarrollo de HITUL y CTPATH. Esta infraestructura se compone de cuatro herramientas principales: Git, Maven, Jenkins y Sonar. Estas herramientas son la columna vertebral de nuestra infraestructura de desarrollo de software y en esta sección se describen cómo las utilizamos para ayudar en el **ciclo** de vida de **desarrollo** de software (ver Figura 1). Estas no son las únicas herramientas que asisten a nuestro equipo de desarrollo, también se usaron otras herramientas convenientes (como IDEs, componentes de seguimiento de incidencias, etc.).

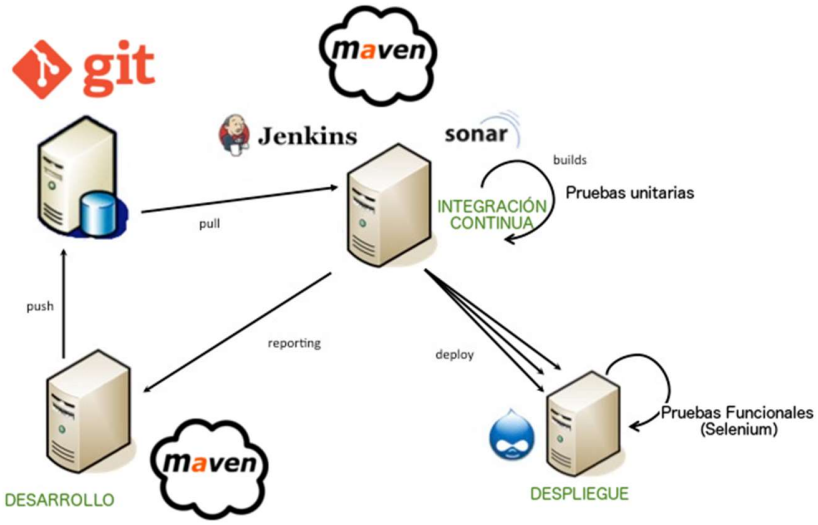

Figura 1: *La interacción entre las cuatro herramientas principales de nuestra infraestructura de desarrollo de software.*

Git es una herramienta de gestión de código fuente (SCM) que realiza un seguimiento de todas las versiones del código fuente generado durante el desarrollo software. Ha sido una

herramienta esencial para el trabajo en grupo y que cada uno pueda ir desarrollando parte del software e ir integrándose de forma apropiada con el resto.

Maven es una herramienta de software para ayudar a los desarrolladores en la construcción de software. Un software tan complejo como el desarrollado en este proyecto tiene con muchas dependencias de bibliotecas externas y requiere también de un proceso de construcción complejo. El uso de Maven facilitó todos estos aspectos permitiendo un diseño más ágil del software.

Mientras Git y Maven fueron utilizadas en las máquinas de los desarrolladores para realizar un seguimiento de la evolución del software y construcción del software, también dispusimos de un lugar centralizado donde todos los desarrolladores y los jefes de proyecto pueden echar un vistazo a la situación del software. **Jenkins,** una aplicación web de automatización de tareas relacionadas con el desarrollo de software, nos permitió disponer de este sitio centralizado y realizar tareas periódicas para construir el software de la rama principal de nuestro repositorio Git central, calcular estadísticas y métricas sobre el código fuente, todo esto sin intervención humana.

Entre las tareas de Jenkins, es detacable la automatización de las tareas relacionadas con la gestión de calidad realizada por SonarQube. **SonarQube** nos proporciona un lugar central para los desarrolladores y administradores de proyectos que permite evaluar la calidad de los proyectos y tomar medidas para mejorarlo. Durante el proyecto se utilizaron esos informes para mejorar la calidad del software (como se observa en la **calificación A**, la máxima posible, obtenida al finalizar los desarrollos).

Además de los mencionados previamente también se utilizó como IDE el entorno **Eclipse** para todos los desarrollos en Java. Este entorno soporta Maven y se integra perfectamente con Git. Como se comentó previamente esta fue de vital importancia en nuestro proyecto, ya que todo el software desarrollado se almacena en repositorios Git.

3. Tecnologías específicas para HITUL

Para el desarrollo del sistema HITUL se emplearon distintas herramientas y tecnologías. Estas se listan a continuación separadas en dos partes: (1) herramientas utilizadas para el desarrollo del sistema y (2) tecnologías empleadas para implementar las funcionalidades del sistema. Estas últimas, son requeridas para la instalación o despliegue del sistema en el entorno de ejecución del cliente.

3.1. Herramientas específicas para el desarrollo

El desarrollo de HITUL requirió un conjunto de herramientas específicas, que complementan las ya mencionadas anteriormente relativas a la infraestructura general. La siguiente lista describe la selección de herramientas realizada y la utilidad que se le dio:

- **Notepad++**, para la creación de la aplicación web (html, css y js).
- **JSOM 15.05**, para la edición de los mapas con información real de la ciudad.
- **NETCONVERT**, para la importación y conversión de mapas con las redes de carreteras reales de la ciudad dispuesta en otros formatos OSM.
- **DUAROUTER**, para la generación de rutas y flujos vehiculares.

3.2. Tecnologías utilizadas

El sistema HITUL está implementado como una aplicación de página única (SPA, por sus siglas en inglés). Según esta arquitectura, la mayoría de las funcionalidades se ejecutan en el navegador de internet utilizado (configuración de parámetros de optimización, visualización de planes semafóricos en un mapa, comparación entre planes, exporta/cargar planes seleccionados, etc.). Para ello se usa extensivamente HTML5, CSS3 y javascript. Específicamente se requieren las siguientes librerías:

- **Bootstrap, Bootstrap-filestyle v3.3.5**, para adaptar la disposición y el estilo de los contenidos de la interfaz a distintos tipos de pantallas y dispositivos.
- **Bootbox v1.0**, para la emisión de mensajes de dialogo desde la aplicación, por su fácil integración con Bootstrap.
- **JQuery v1.11.3+**, para simplificar la implementación de manejadores de eventos e interacción asíncrona con la capa de servicios.
- **CanvasJS Charts v1.6.2**, para dibujar gráficas con la estadística de los planes semafóricos bases y óptimos propuestos por el sistema.
- **Google Maps API v3**, para dibujar mapas especializados con información de los semáforos y marcas representativas de la diferencia entre distintos planes.

Todas las librerías antes mencionadas, forman parte del paquete de instalación o se enlazan en tiempo de ejecución desde los servidores o repositorios en internet y no requieren configuración adicional.

Para el cálculo de los planes semafóricos óptimos, la aplicación HITUL interactúa con una capa de servicio del lado del servidor, que ejecuta los algoritmos de optimización/simulación, y recupera la información depositada en la base de datos. Las plataformas y librerías requeridas se listan a continuación:

- **Apache/2.4.7 (Ubuntu)**, como servidor web para la publicación de los servicios implementados para soportar las funcionalidades de la aplicación.

- **PHP 5.5.9**, para la implementación de la capa de servicios que permiten ejecutar los algoritmos y obtener los planes óptimos encontrados. No se requiere ninguna extensión adicional a los estándares.
- **Java(TM) SE Runtime Environment (build 1.8.0_60-b27)**, entorno para la ejecución de programas de optimización codificados en el lenguaje JAVA.
- **jMetal (build 5.0 Beta 34)**, librería que facilita el diseño de los algoritmos de optimización basados en técnicas bioinspiradas.
- **SUMO v0.19+**, para las simulaciones aceleradas que permiten evaluar la calidad de un plan construido con los algoritmos de optimización
- **Text/XML**, para la codificación de datos reales de ciudad utilizados por el simulador y el almacenamiento flexible de planes semafóricos base provisto por expertos.
- **MySql v5.5.44**, para la gestión de planes propuestos por el sistema y otras estadísticas resultado de las simulaciones.

En general, el sistema no requiere la instalación de módulos o extensiones adicionales a las provistas en la instalación típica de las herramientas utilizadas.

3.3. Software requerido para el despliegue

El despliegue del sistema HITUL en el entorno del cliente es soportado por una imagen de máquina virtual (MV), que provee todas las herramientas requeridas previamente instaladas. Para que el sistema funcione correctamente, solo es necesario que la computadora donde se instale la MV esté conectada a internet. Todas las configuraciones necesarias también se han realizado como parte de la construcción de la mencionada imagen.

4. Tecnologías específicas para CTPATH

Para el desarrollo del sistema CTPATH se emplearon distintas herramientas y tecnologías. Estas se listan a continuación separadas en dos partes: (1) herramientas utilizadas para el desarrollo del sistema y (2) tecnologías empleadas para implementar las funcionalidades del sistema. Estas últimas, son requeridas para la instalación o despliegue del sistema en el entorno de ejecución del cliente.

4.1. Herramientas específicas para el desarrollo

El desarrollo de CTPATH requirió un conjunto de herramientas específicas, que complementan las ya mencionadas anteriormente relativas a la infraestructura general. La siguiente lista describe la selección de herramientas realizada y la utilidad que se le dio:

- **Android Studio 1.4**, para el desarrollo de las aplicaciones móviles (tanto del operario como del usuario de CTPATH).
- **SourceTree 2.0.5.2**, para la gestión de los repositorios GIT.
- **Arduino Software 1.6.3**, para el desarrollo del software para el sensor Arduino.

- **Spyder 2.3.7** como editor para desarrollar el código en **Python** (software) para el sensor Raspberry.
- **PuTTY 0.65** para conectar la máquina de desarrollo con el sensor Raspberry mediante una conexión segura (SSH) y probar el software del sensor.
- **Win32diskimager-0.9.5**, para copiar/clonar la imagen del software en la Raspberry.

4.2. Tecnologías utilizadas

El sistema CTPATH integra un servidor web Grizzly, por lo que no hay necesidad de incluirlo en ningún servidor de aplicaciones. Esto permite reducir la dependencia de otros paquetes software. No obstante, CTPATH está basado en OpenTripPlanner, que incluye una gran cantidad de bibliotecas de terceros. A continuación mostramos una lista del software que hemos usado durante el desarrollo de CTPATH (excluimos las que usa internamente OpenTripPlanner):

- **jQuery 1.9.1**, para el desarrollo del cliente Web.
- **AngularJS 1.2.20**, para el desarrollo del cliente Web.
- **Hibernate 5.0.1**, como motor de persistencia.
- **JPA 2.1**, como interfaz para el motor de persistencia
- **MySQL connector 5.1.6**, para el acceso a la Base de Datos MySQL.
- **OpenTripPlanner 0.13.0**, para la gestión de las peticiones de cálculo de rutas. Este módulo incluye otras 43 bibliotecas no listadas aquí.
- **Python 3**, para el desarrollo del software empleado por los sensores Raspberry.
- Paquetes **airdump-ng** y **airmon-ng** para el análisis del tráfico WiFI para la detección de vehículos usando el sensor Raspberry.
- Paquete **hcitool** que facilita la detección de dispositivos Bluetooth usando el sensor Raspberry.

Todas las librerías antes mencionadas, forman parte del paquete de instalación o se enlazan en tiempo de ejecución desde los servidores o repositorios en internet y no requieren configuración adicional.

4.3. Software requerido para el despliegue

Para el cálculo de las rutas, la aplicación CTPATH interactúa con un servicio Web en el lado del servidor, que ejecuta los algoritmos de optimización, y recupera la información depositada en la base de datos. Las plataformas y software requerido para poner en marcha este servicio se listan a continuación:

- **Java(TM) SE Runtime Environment (build 1.8.0_60-b27)**, entorno para la ejecución de programas de optimización codificados en el lenguaje JAVA.
- **MySQL v5.5.44**, para el almacenamiento de los datos de usuario, de los sensores y el estado del tráfico.

- **Apache/2.4.7 (opcional)**, como servidor web que actúe como **front-end**. En nuestro caso lo hemos usado para añadir una capa de seguridad a la aplicación durante el desarrollo, pero no sería necesario cuando el software esté en producción.

El sistema no requiere la instalación de módulos o extensiones adicionales a las provistas en la instalación típica de las herramientas utilizadas.

Para las aplicaciones móviles y los sensores, hemos usado el siguiente software durante el despliegue:

- **Android SDK Tools, 24.4**, para instalar el fichero **apk** de las aplicaciones en los dispositivos móviles con sistema operativo Android.
- **Arduino Software 1.6.3**, para el despliegue del software en el sensor Arduino.
- **Win32diskimager-0.9.5,** para desplegar/clonar del software en el sensor Raspberry.