UNIVERSITAT POLITÈCNICA DE VALÈNCIA

etsinf

Escola Tècnica Superior d'Enginyeria Informàtica

Escola Tècnica Superior d'Enginyeria Informàtica

Universitat Politècnica de València

SMARTROAD

Trabajo Fin de Grado

Grado en Ingeniería Informática

Autor: García Ferrando, Germán Adrián

Tutor: Joan Josep Fons i Cors

2015 / 2016

Resumen

Resum

Abstract

TABLA DE CONTENIDO

1 CONTEXTO TECONLÓGICO

Antes de poder introducir al lector en el tema de este proyecto, he de situarlo en el contexto tecnológico en el que se sitúa el proyecto. En este punto se explicarán brevemente las tecnologías utilizadas a la vez que se justificará porqué se ha utilizado cada una de ellas.

Este proyecto se sitúa dentro de un entorno donde el Internet de las Cosas (IoT) nos brinda la oportunidad de conectar miles de dispositivos para obtener cantidades de información inimaginables hasta el momento. Con el problema de recopilar información solucionado, el siguiente paso es diseñar sistemas capaces de armonizar dicha información y, en consecuencia, actuar con la mayor precisión posible.

Actualmente, uno de los escenarios de estudio más conocido es el de las SmartCity, ciudades capaces de auto gestionarse por sí solas. La cantidad de eventos que suceden en una gran ciudad diariamente es enorme, y en consecuencia la casuística del sistema que ha de gestionar dicha ciudad, también lo es. Por ello son necesarias tecnologías que proporcionen una gran escalabilidad e interoperabilidad, lo cual ha sido un punto clave a la hora de seleccionar las tecnologías que se explican a continuación.

1.1 INTERNET OF THINGS (IOT)

Internet of Things nace de la idea de conectar a internet los objetos cotidianos que utilizamos día a día, tostadoras, vehículos, casas… y conectarnos a internet para poder analizar todos los datos que son capaces de recopilar. El concepto IoT se propuso en 1999 por Kevin Ashton, el cual apunta a que a finales de los 90, la mayoría de la información almacenada en internet había sido creada por seres humanos, a base de teclear. Él propone un giro, en el cual se pueda dotar a los computadores de las herramientas necesarias para que generen ellos la información de forma independiente (Ashton, 2009).

El concepto IoT ya se ha aplicado a varios campos con éxito. En el campo de la medicina, ya se están utilizando pequeños sensores para monitorizar las constantes vitales de un paciente esté donde esté, o simplemente para recordar al paciente cuando es hora de tomarse la medicación. También se han creado “SmartCar”, coches capaces de monitorizar su estado (motor, neumáticos, autonomía…) lo cual ayuda al usuario ya que tiene más información a sobre su vehículo y también ayuda a la industria, ya que puede recopilar la información generada por todos sus vehículos para hacer estudios sobre estos.

La tecnología IoT es algo que no se puede ignorar, está presente en nuestro día a día y cada vez lo va a estar más. En este proyecto IoT representa el punto de unión entre todos los elementos que componen la "SmartCity". Cada uno de los elementos que interviene, está conectado con el resto de dispositivos siguiendo una arquitectura IoT, lo cual nos permite extrapolar esta simulación virtual a una posible simulación real utilizando dispositivos embebidos como Arduino o Raspberry Pi.

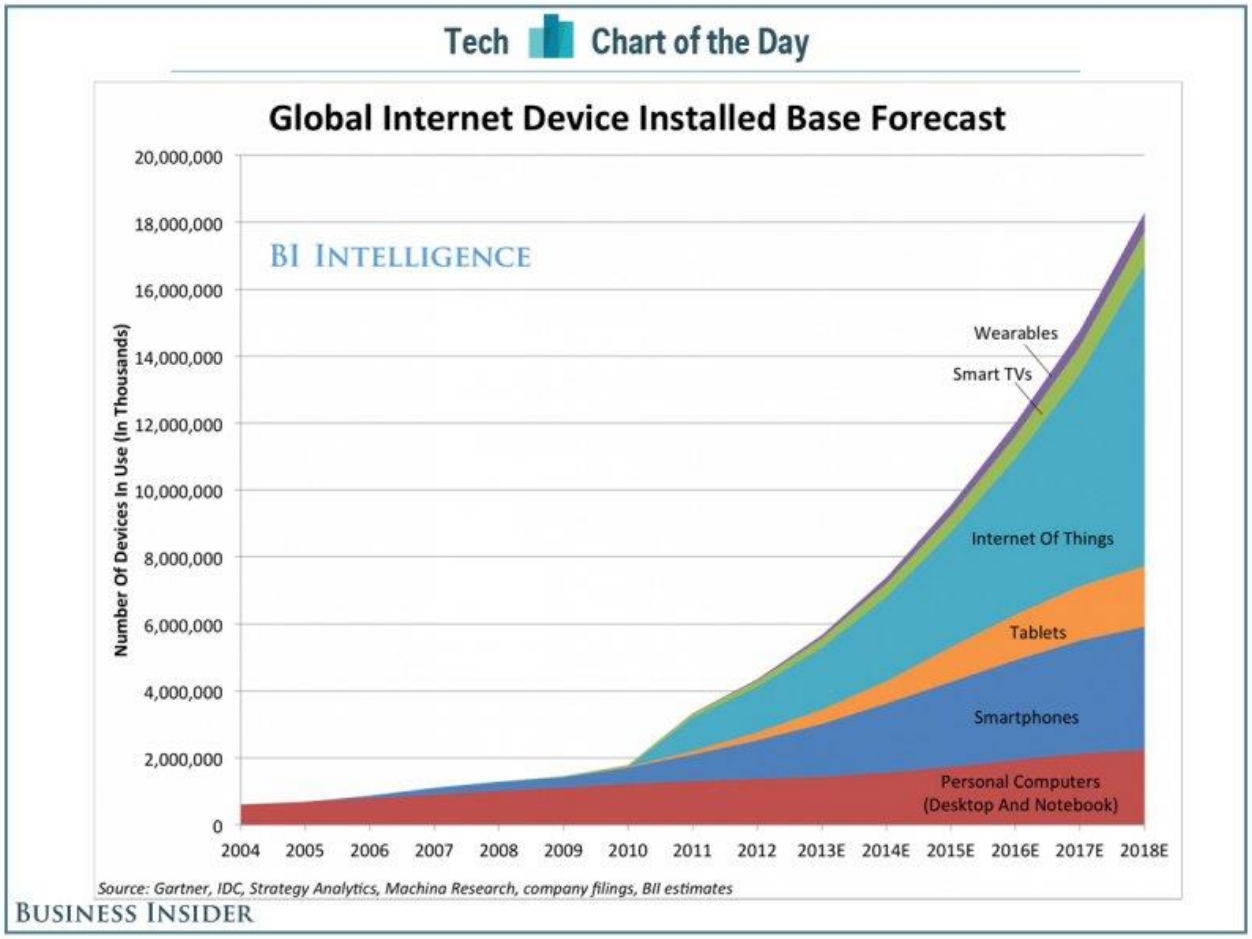


Ilustración 1: Predicción de la tecnología IoT por Business Insider

1.2 SMARTCITY

Nota: Aunque la ciudad inteligente (SmartCity) no es una tecnología en sí; sí es un concepto fundamental dentro de este proyecto. Por lo que, con la intención de ahorrar al lector posibles confusiones con este concepto, se ha decidido explicar la SmartCity dentro del contexto tecnológico.

El concepto de SmartCity aún no está definido con claridad, pero sí que podemos encontrar un patrón entre todos los proyectos que en la actualidad trabajan en este campo. Todos persiguen, a través de la tecnología: **aumentar la calidad de vida de los ciudadanos** mientras se crea una ciudad autosuficiente y sostenible. Por ejemplo, en el proyecto BCN Smart City (BCN, 2016) se habla de utilizar las nuevas tecnologías para hacer una gestión eficiente de los recursos y servicios de la ciudad; en el proyecto *eurupeansmartcities 4.0* dirigido por Vienna University of Technology (WIEN, 2015) se habla de combinar la competitividad de una ciudad con un desarrollo urbano sostenible centrado den las ciudades de tamaño medio – bajo; también podría nombrarse el PLEEC (Planning for Energy Effitient Cities) que apunto a utilizar la tecnología para crear SmartCities autosuficientes energéticamente, y así un largo etcétera.

El resumen que podemos extraer de este conjunto de proyectos, es que el modo en el que las ciudades han funcionado hasta ahora está cambiando, existe la oportunidad de mejorar el modelo de ciudad que conocemos actualmente. En un futuro, las ciudades serán mucho más interactivas con los ciudadanos, serán ciudades autosuficientes, capaces de auto gestionarse sin la constante supervisión de los seres humanos etc. En definitiva, un lugar mejor donde vivir.

1.3 SMARTROAD

Nota: La explicación de la SmartRoad se realiza dentro del contexto tecnológico por el mismo argumento relatado en el punto anterior (SmartCity).

Si el concepto de las SmartCity aún no está claro, el de la SmartRoad no es una excepción. A día de hoy hay pocos proyectos que se dediquen exclusivamente al campo de las SmartRoad. Aunque sí que podemos destacar el trabajo que se está realizando en el estado de Virginia, USA. Allí se está construyendo la SmartRoad conocida como *"Virginia Smart Road"*, se trata de una carretera de 3.5km de longitud en la cual se pretende testear varias de las nuevas tecnologías que se están introduciendo en el escenario de las SmartCity. Un ejemplo de estas pruebas fue el cuándo los estudiantes de la Universidad de Whasington instalaron en la Virginia Smart Road un pavimento el cual también actuaba como panel solar (UW, 2016).

De todas formas, ya sea aumentando la interacción con los SmartCar que circulen por la carretera, monitorizando el estado de la carretera, haciendo que genere su propia energía… Podemos

concluir en que el objetivo básico de las SmartRoad sigue siendo el mismo que en las SmartCities, y es **aumentar la calidad de vida los usuarios**.

1.4 JAVA & JADE

Java es uno de los lenguajes de programación más populares en la actualidad, nació en 1995 aunque su primer JDK se lanzó en 1996. Java es un lenguaje orientado a objetos, fue diseñado con la intención de tener pocas dependencias de implementación, lo cual le permite una gran interoperabilidad. El punto fuerte de Java es gestionar aplicaciones cliente-servidor, donde cuenta con más de 10 millones de usuarios reportados.

Además de la interoperabilidad que ofrece, y la gran escalabilidad demostrada hasta el momento con los 10 millones de usuarios, Java también cuenta con una característica de peso para este proyecto: JADE.

JADE es un *framework* implementado en Java el cual nos ofrece muchas facilidades a la hora de trabajar con agentes inteligentes, los cuales son una herramienta excelente a la hora de reproducir simulaciones a gran escala. Aunque JADE finalmente no ha sido utilizado en este proyecto, sí que entraba dentro del planteamiento inicial, lo cual influenció en la selección del lenguaje Java para desarrollar la simulación.

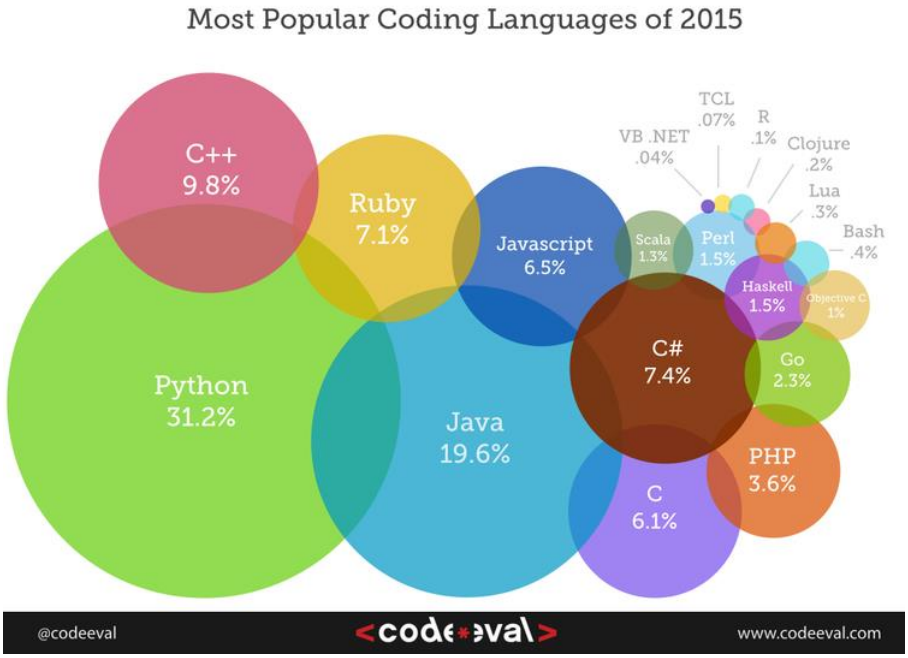


Ilustración 2: Lenguajes de programación más populares (2015)

1.5 MQTT

MQTT es un protocolo de mensajería pub-sub diseñado para funcionar en conexiones con poca banda ancha, alta latencia o en redes poco fiables. La principal característica de MQTT es su sencillez, lo cual le permite minimizar la cantidad de recursos necesarios para hacerlo funcionar.

Fue creado en 1999 por el Dr Andy Stanford-Clark (IBM) y Arlen Nipper (Arcom, actualmente Eurotech). En 2011 IBM y Eurotech hicieron una donación al proyecto "Eclipse Paho", el cual se utiliza en este proyecto para controlar los clientes MQTT utilizando Java. Además, desde marzo de 2013 MQTT se está sometiendo al proceso de normalización de OASIS, con el fin de convertirse en un estándar.

Los pilares de MQTT le convierten en un protocolo ideal para IoT (Internet of Things) o M2M (Machine-to-Machine), donde los dispositivos no disponen de una gran cantidad de recursos. Dentro de nuestro proyecto, se propone utilizar RaspBerry Pi junto a Arduino a modo de prototipo para empezar a articular una SmartCity, donde el protocolo MQTT encaja perfectamente.

Otro punto a favor para MQTT es que su integración con Java es muy sencilla, gracias a "Paho", el proyecto mencionado anteriormente, el cual nos permite gestionar los clientes MQTT con gran sencillez, ofrece una buena API y además cuenta con bastante documentación.

1.6 MOSQUITTO

Para utilizar MQTT es necesario un *broker,* en este proyecto se ha optado por utilizar Mosquitto. Una de las principales razones fue que Mosquitto es un proyecto incubado por Eclipse, el cual también está dirigiendo "Paho", por lo que es de esperar que la sinergia entre estos dos proyectos tienda a ser mejor que utilizando otros *brokers* también viables (RabbitMQ, VerneMQ, ActiveMQ…).

Por último, ya se había trabajado antes en un proyecto el cual gestionaba varias comunicaciones entre dispositivos Arduino via MQTT utilizando Mosquitto como *broker*; y como se suele decir, la experiencia es un grado.

1.7 JSON

JSON es un formato de mensajería especialmente ligero e independiente del lenguaje de programación. Es fácilmente legible por los humanos, y además fácil de *parsear* por las computadoras. Está basado en un subconjunto de JavaScript, y fue popularizado por Douglas

Crockford. Actualmente es el formato más utilizado en comunicaciones asíncronas buscador/servidor, remplazando a XML el cual es utilizado por AJAX.

JSON se ajusta perfectamente a las condiciones de este proyecto, donde los dispositivos no cuentan con una gran capacidad de cómputo, por lo que la sencillez del formato es un gran punto a favor. También se tuvo en cuenta que en el entorno de IoT se intenta armonizar una gran diversidad de dispositivos, y conectarlos con la web, JSON nos permite ajustarnos a lo que actualmente se encuentra "de moda" y nos brinda mayor compatibilidad con futuras aplicaciones que necesiten establecer comunicación con el sistema encargado de gestionar la SmartCity.

Por último, para este proyecto se ha tomado como referencia la iniciativa VLCi, y dentro de su documentación se puede observar que han optado por el formato JSON como estándar de mensajería para sus comunicaciones; por lo que esto es otro punto a favor en vistas a una posible implementación real en el proyecto VLCi.

1.8 ARDUINO

Arduino es un proyecto de hardware libre el cual diseñó una plataforma de hardware y software compuesta por placas de desarrollo que integran un microcontrolador y un entorno de desarrollo (ArduinoIDE). La placa en sí consiste en un circuito impreso con un microprocesador, además de esto cuenta con puertos analógicos y digitales de entrada/salida, los cuales pueden conectares a otros dispositivos para aumentar la funcionalidad del Arduino (sensores, Raspberry Pi…).

Actualmente existe una comunidad alrededor de Arduino a nivel internacional, lo cual ha generado una amplia cantidad de información disponible en la red. Esta sencilla placa destaca por ser una de las mejores formas para implementar prototipos, ya que es relativamente económica y cuenta con una gran compatibilidad con otros dispositivos.

(TODO: Relacionar con el proyecto de Germán?)

1.9 RASPBERRY PI

Raspberry Pi es un ordenador de placa reducida con un coste de fabricación atractivamente bajo. El objetivo por el cual fue diseñado fue estimular las ciencias de la computación en las escuelas, aunque ha terminado por abarcar una gran cantidad de campos, gracias a su sencillez y su precio.

Normalmente la placa cuenta con un procesador, una memoria RAM, conexiones USB, ranura para tarjetas SD y puerto Ethernet; aunque dependiendo del modelo de placa estas especificaciones pueden variar.

1. INTRODUCCIÓN

Dentro de este proyecto la placa Raspberry Pi ocupa el rol de intermediario entre la información captada por la placa Arduino (conectada a sus correspondientes sensores) y el software que armoniza la información generada por los distintos dispositivos. Por lo que lo que más nos interesa de este modelo es su capacidad para comunicarse tanto via Ethernet como a través de un puerto serie con un dispositivo Arduino.

2 INTRODUCCIÓN

En este documento se relata el diseño de un sistema capaz de, gracias a los avances tecnológicos, proponer mejoras considerables en el ámbito de las ciudades inteligente, concretamente en el campo de la asistencia sanitaria en carretera.

Pero antes de lanzarnos de cabeza al desarrollo del sistema en cuestión, se debe dar respuesta a una pregunta fundamental: ¿Existe la necesidad de invertir tiempo y dinero en el desarrollo de un sistema para mejorar la asistencia en carretera?

Con el fin de responder a dicha pregunta, y demostrar que **sí existe la necesidad de mejorar el funcionamiento de las ciudades actuales,** se ha realizado un pequeño estudio en el cual se analiza el estado actual de los principales servicios de emergencia del estado español.

2.1 ANÁLISIS DEL ESTADO ACTUAL DE LOS SERVICIOS DE EMERGENCIA

Nota: Toda la información aquí mostrada ha sido extraída de las páginas oficiales del Ministerio del Interior, en caso contrario se detallará la fuente.

Ante de nada, me gustaría remarcar el buen trabajo que se ha ido haciendo desde un tiempo atrás por los servicios de emergencia del estado español, los fallecidos en accidentes de tráfico han ido descendiendo desde 1989 a un ritmo considerable; aunque como se muestra en el gráfico "Accidentes de tráfico serie 1993 – 2014" el número de accidentes de tráfico que requieren la hospitalización del accidentado es elevada.

La principal demostración de que invertir en mejorar la seguridad vial no es en balde la tenemos en la Ilustración 3, donde podemos apreciar un gran descenso de los fallecidos en accidentes de tráfico una vez alcanzado el máximo de 1989; también se puede observar un gran descenso de los fallecidos desde 2004, lo cual coincide con el endurecimiento de las medidas de seguridad vial, como por ejemplo el carnet por puntos implantado en 2006.

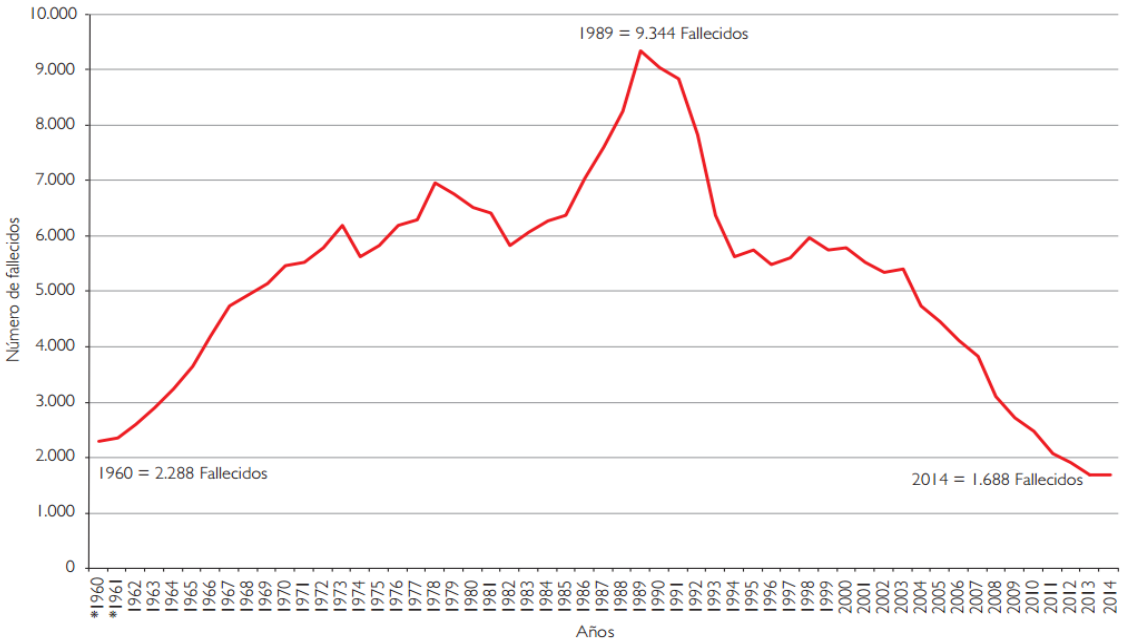


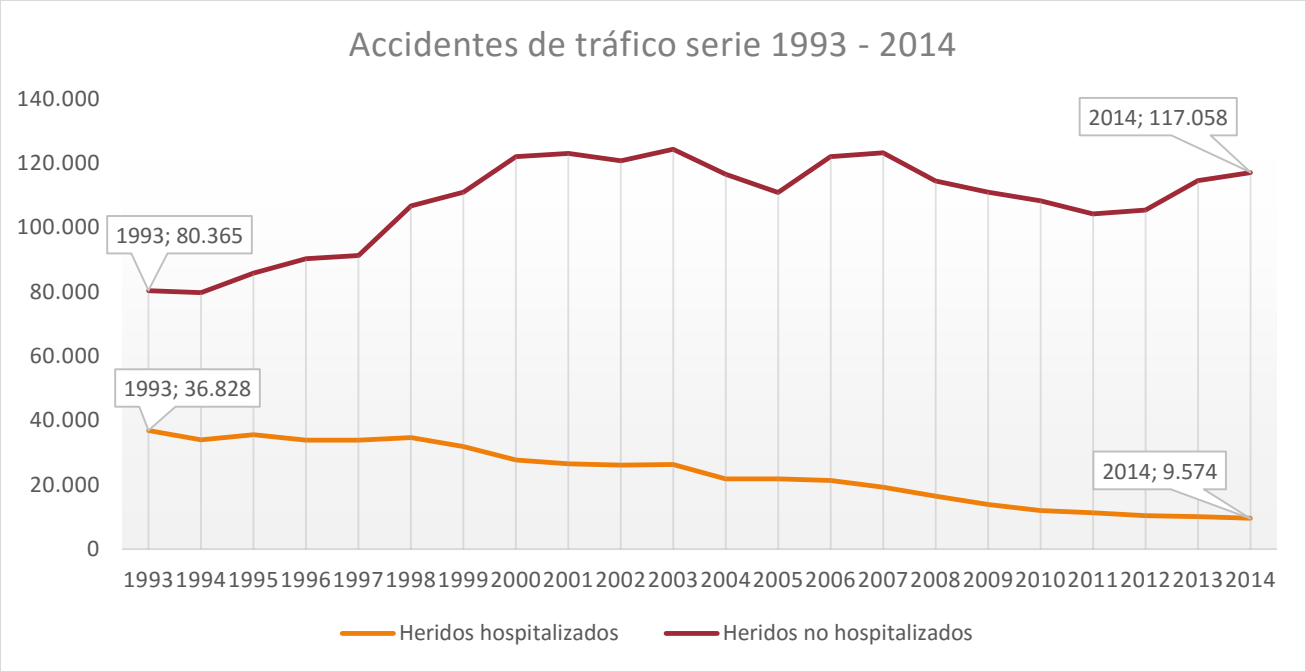
Ilustración 3: Fallecidos por accidente de tráfico 1960 – 2014

Un dato a destacar es que desde el año 2011 el número de accidentes ha aumentado, esto podría deberse a un aumento en la cantidad de vehículos en las carreteras, pero como se puede apreciar en el Censo de conductores, la cantidad de conductores se ha estabilizado desde el año 2009.

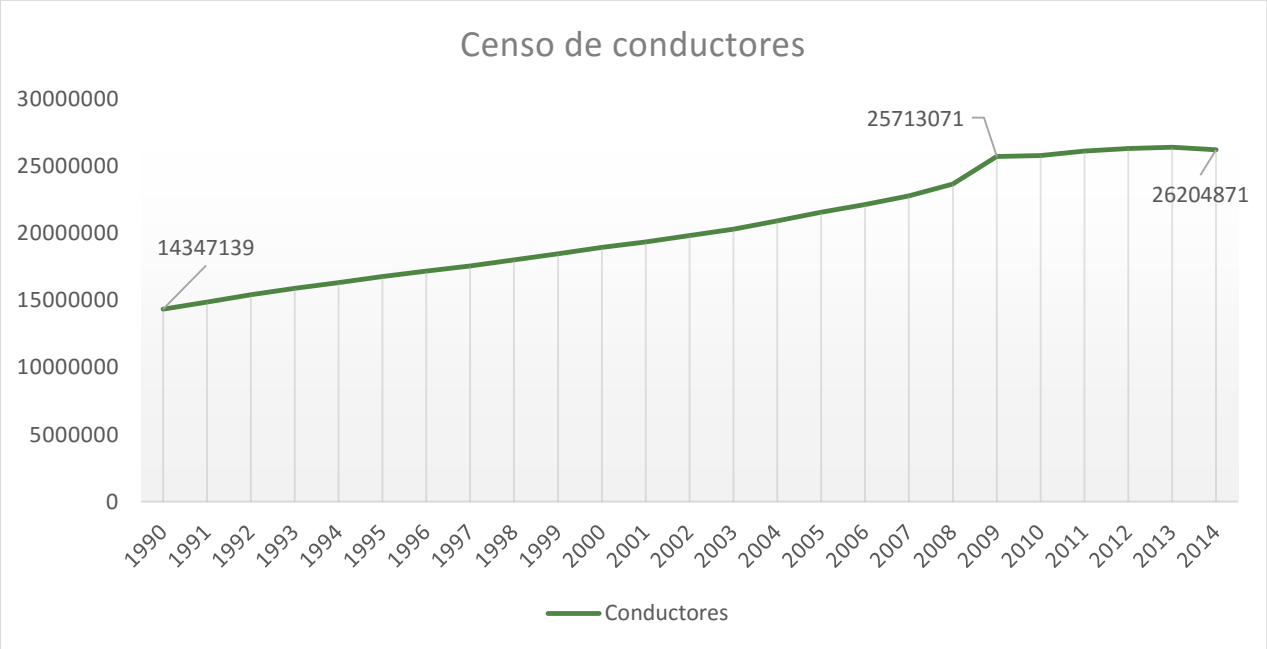
Aunque este análisis queda fuera del rango de este proyecto, se ha investigado acerca de una posible relación entre estancamiento en el número de conductores (2009) y el aumento de heridos en accidentes de tráfico (2011). En base a las declaraciones de expertos en la materia, el estancamiento de nuevos conductores está totalmente relacionado con la crisis económica de 2008; esto ha provocado que la ciudadanía opte más por mantener sus vehículos antiguos en vez de comprar nuevos.

"Una de las causas a tener en cuenta para entender la siniestralidad de 2015 es un mayor envejecimiento del parque. La edad media de los vehículos implicados en accidentes mortales continúa aumentando. En 2015 se situó en 11,3 para los turismos y en 9,6 para las motociclistas, casi dos puntos más que en 2014." - (DGT, Dirección General de Tráfico España, 2016)

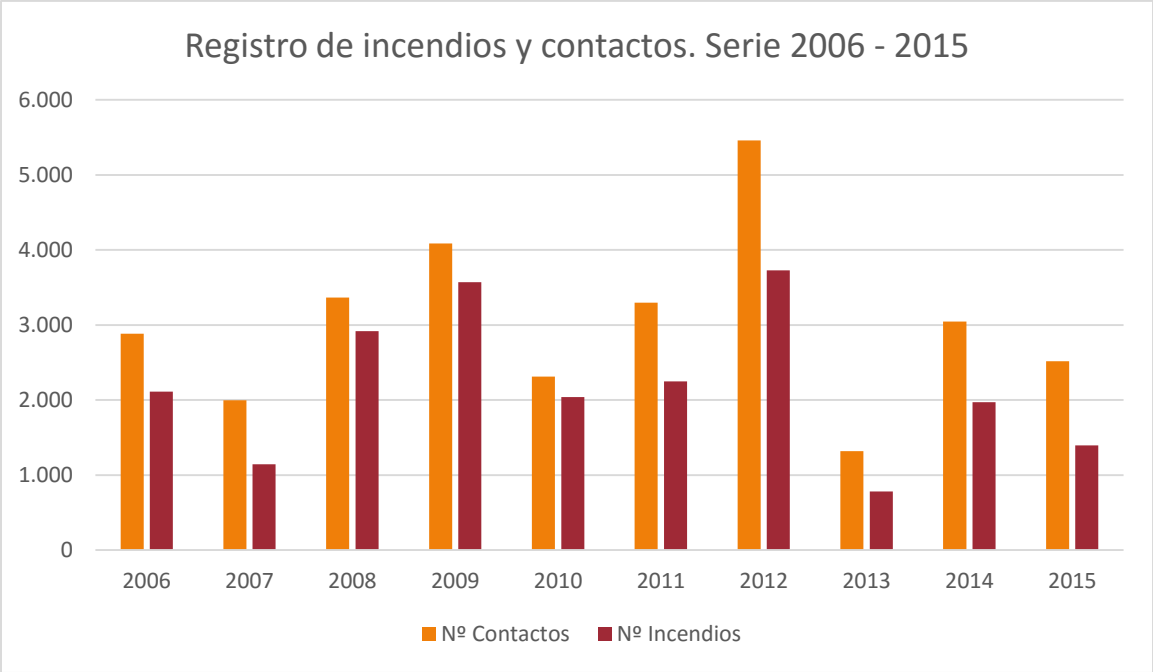
2. CASO DE ESTUDIO



Una de las conclusiones que se podrían extraer de estos datos, es que cada vez hay más usuarios utilizando las carreteras españolas, por lo que invertir tiempo en mejorar los servicios encargados de mejorar la seguridad de los ciudadanos es una causa más que justificada.



Las ambulancias no son los únicos servicios de los cuales dispone una ciudad inteligente para atender las distintas emergencias; a continuación, se hará un breve análisis del estado actual de los bomberos en el estado español, servicio al cual también se podría aplicar la idea general de este proyecto.



A diferencia de los datos mostrados en la sección de tráfico, el número de incendios no atiende a una función creciente o decreciente, sino que más bien es algo aleatorio. Lo que sí que se puede destacar de estos datos, es que la relación Número de contactos/Número de incendios es bastante elevada, es decir, **cuando alguien contacta con el servicio de emergencias para alertar de un posible incendio la probabilidad de que el incendio exista es del 72.32%.**

El servicio de bomberos lleva tiempo trabajando en la instalación de sensores de varios tipos para ayudar a la detección de incendios, como se aprecia en la Ilustración 4 hay varios sensores distribuidos por los distintos parques nacionales, cada uno con un distinto tiempo de reacción.

Por desgracia, la detección automática de incendios no es sencilla, por lo que recibir la información a tiempo se convierte en uno de los principales problemas a la hora de tratar estas emergencias. Al igual que en las ciudades inteligentes, la tecnología en el contexto de IoT trae muchos avances aplicables a este escenario, aunque esto ya no forma parte del alcance del proyecto.

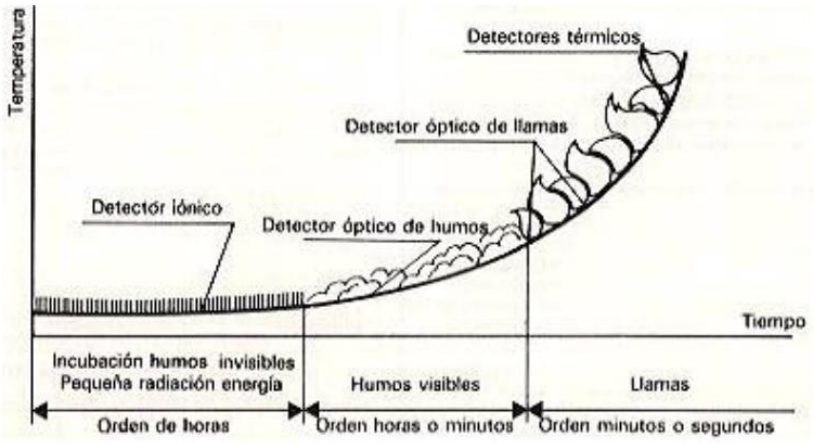


Ilustración 4: Fase de actuación de los sensores

TODO: AMPLIAR ESTUDIO O HACER UN PÁRRAFO QUE CIERRE EL TOPIC

Patron común en todos

Información = mejores resultados

Mayor rapidez con una respuesta automática…

2.2 MOTIVACIÓN

Ambulancias en mitad de una emergencia inmóviles en la carretera por atascos; accidentes que no se detectan hasta que un tercero llama al 112; vidas que se podrían salvar, pero no fuimos capaces de hacerlo… Cada día mueren en el mundo sobre 3.400 personas en accidentes de tráfico (OMS, 2016), en España en el año 2014 murieron 340 personas (DGT, 2014), gracias al trabajo de miles de personas el número es cada vez menor, pero, ¿podríamos haberlo hecho mejor?

Desde mi humilde ignorancia creo que algunas situaciones se podrían haber gestionado mejor si se hubiese contado antes con la información necesaria, por lo que mi granito de arena es diseñar una propuesta la cual nos permitiría obtener la información con mayor rapidez y ser capaces de actuar de una forma automática. Esta propuesta se basa en una tecnología ampliamente conocida, y que ya se ha usado con éxito en otros contextos, hablo de IoT junto a un software capaz de gestionar la información generada por los distintos dispositivos. (Redundante con el párrafo siguiente).

Creo que la propuesta realizada en este TFG es un boceto más, entre muchos otros, que demuestra la existencia de un camino donde la tecnología nos brinda un mundo mejor; una tecnología que ya se está utilizando en varios proyectos, una tecnología que es viable económicamente y que podríamos empezar a utilizar hoy mismo.

La motivación principal de este proyecto no es superar una última prueba para obtener un título, sino ser capaz de utilizar los conocimientos adquiridos durante los cuatro años de estudio en el Grado de Ingeniería Informática en pro de la causa más noble que conozco, salvar vidas.

2.3 OBJETIVO

El objetivo principal de este proyecto es **reducir el tiempo de respuesta de la asistencia sanitaria en carretera** dentro del contexto de una ciudad inteligente.

¿Cómo se va a reducir el tiempo? Bueno, si lo único que necesitamos es llegar antes a un sitio, todos conocemos la solución, solo necesitamos "salir antes" e "ir más rápido". En base a estos dos conceptos, el proyecto propone cambios en tres puntos clave:

- **Obtener información con mayor rapidez.**
- **Ser capaces de responder automáticamente.**
- **Gestionar las carreteras para reducir el tiempo de viaje y aumentar la seguridad.**

Como objetivo final, también **se diseñará una simulación** en la cual se comprobará la viabilidad de las soluciones propuestas en este proyecto.

2.4 METODOLOGÍA Y PLAN DE TRABAJO

TODO: No tengo ni idea de que poner aquí

2. CASO DE ESTUDIO

3 CASO DE ESTUDIO (¿ANÁLISIS?)

3.1 ELEMENTOS FUNDAMENTALES

Es hora de empezar a trabajar en los objetivos planteados anteriormente. El primer paso es definir qué elementos necesitamos tener en cuenta para trabajar en un entorno lo más realista posible. Para ello vamos a narrar el escenario en cuestión y extraer los elementos importantes.

El proyecto se basa en que un vehículo, el cual circula por una de las carreteras de la ciudad inteligente, sufre un accidente. Dicho esto, ya se pueden identificar dos elementos fundamentales dentro de nuestro escenario: la ciudad inteligente (**SmartCity**) y el vehículo (**SmartCar**).

Una vez se ha producido el siniestro, de alguna forma el vehículo ha de comunicar a la ciudad de su estado de emergencia; esto se realiza a través de las carreteras inteligentes (**SmartRoad**), las cuales realizan un control de todos los vehículos que circulan a través de ellas.

Por último, una vez la emergencia ha llegado a la ciudad, esta debe ser atendida. Para ello la ciudad hace uso de los servicios de asistencia sanitaria, los cuales son representados en este proyecto por una **Ambulancia.**

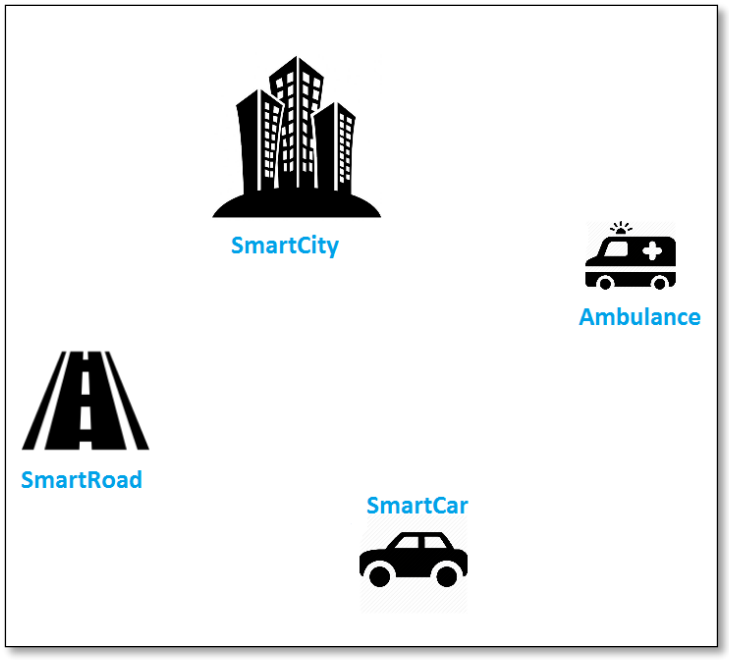


Ilustración 5: Elementos básicos del caso de estudio

3.2 IDEA BÁSICA DE LA SOLUCIÓN PROPUESTA

Una vez se han definido los elementos que toman parte dentro del entorno, podemos pasar a explicar cómo, a grandes rasgos, se espera mejorar la asistencia en carretera. La solución propuesta se puede dividir en tres partes, cada una de estas partes hace referencia a los puntos comentados en el apartado de Objetivo.

La información del siniestro llega al sistema. Una vez el vehículo ha detectado que ha sufrido un siniestro, o el conductor detecte que está en una situación de emergencia, se mandará una señal S.O.S a la ciudad.

Respuesta automática. Una vez la señal S.O.S ha llegado a la infraestructura del sistema, la ciudad inicia un proceso automático el cual dará una respuesta a dicha señal. Esta respuesta deberá seleccionar la ambulancia más apropiada para tratar dicho siniestro, la ruta óptima desde la ubicación de la ambulancia hasta la ubicación del siniestro…

Por último, y con el objetivo de que la ambulancia encuentre el **mínimo número de obstáculos en su ruta**, la ciudad tendrá la capacidad de abrir o cerrar carriles de las carreteras. En caso de ser necesario, se pondrá en contacto con la carretera responsable de cada carril para realizar dicha acción.

Una vez la ambulancia haya llegado a la ubicación del siniestro y comunique que es capaz de atender la emergencia con los recursos disponibles, la emergencia se dará por concluida.

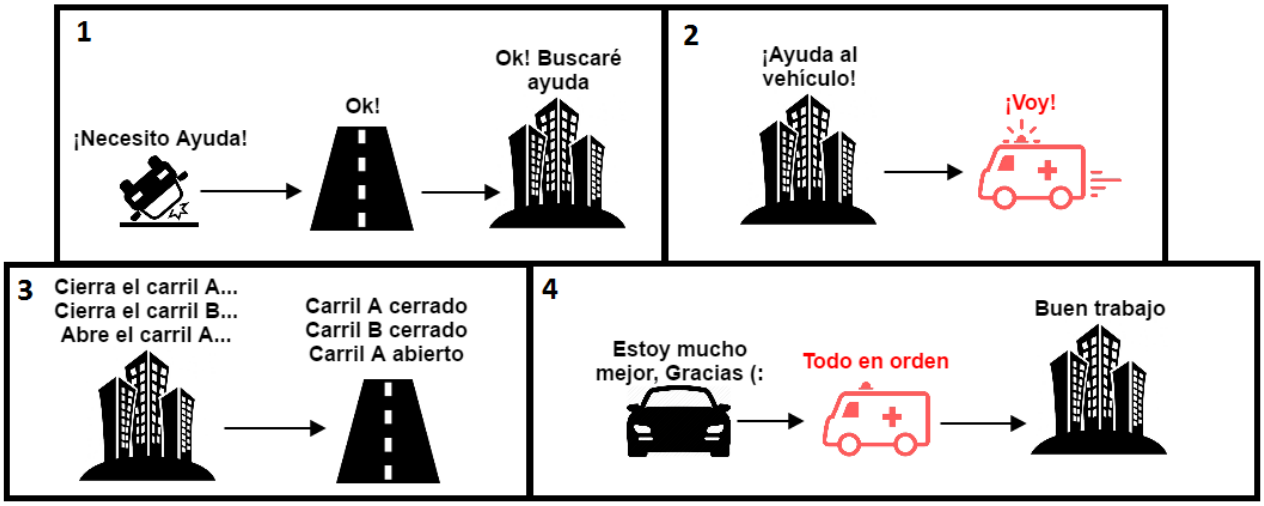


Ilustración 6: Representación gráfica de la idea básica

3.3 REQUERIMIENTOS

Como era de esperar para que se pueda llevar a cabo la idea principal, los elementos que participan en ella deben cumplir unos requerimientos básicos. El primero es que **todos los elementos deben poder conectarse a internet**; además de esta conexión **tanto los vehículos como las carreteras han de ser capaces de percibir su entorno** (vía sensores, por ejemplo).

Para cumplir estos requerimientos, se utiliza la tecnología de Arduino junto a Raspberry Pi. En este proyecto no se va a entrar en profundidad, ya que parte de la base del documento presentado por Germán Caselles Fonts (Fonts, 2014/2015), dónde se analiza más en profundidad cómo es posible conectar varios dispositivos utilizando una arquitectura IoT.

La **consistencia de datos** también es un requerimiento básico, por lo que los distintos elementos han de integrar un sistema capaz de mantener información en un dispositivo de memoria no volátil; o ser capaz de almacenar toda su información en la nube y acceder a ella cada vez que se inicia el dispositivo. En este proyecto se obviado este requerimiento, ya que no era necesario mantener la información de una sesión a otra para demostrar la viabilidad de la idea aquí propuesta.

Por último pero no menos importante, es necesaria una **supervisión humana en todo momento**, ya que aunque seamos capaces de diseñar un sistema capaz de tener en cuenta una gran cantidad de variables, estamos dando respuesta a escenarios muy sensibles donde se trata con vidas humanas. Este último requerimiento no se tendrá en cuenta ya que queda fuera del alcance de este proyecto, pero sería fundamental si se llega a aplicar en una situación real.

3. INTRODUCCIÓN A LA SIMULACIÓN

4 DISEÑO

Una vez captada la idea general, el siguiente paso es empezar a transformar los elementos que se han descrito en el tema anterior en componentes de un sistema informático. En este punto se explica paso a paso como se va diseñando el sistema que finalmente se implementará y se pondrá a prueba en la simulación.

4.1 DIAGRAMA DE CLASES

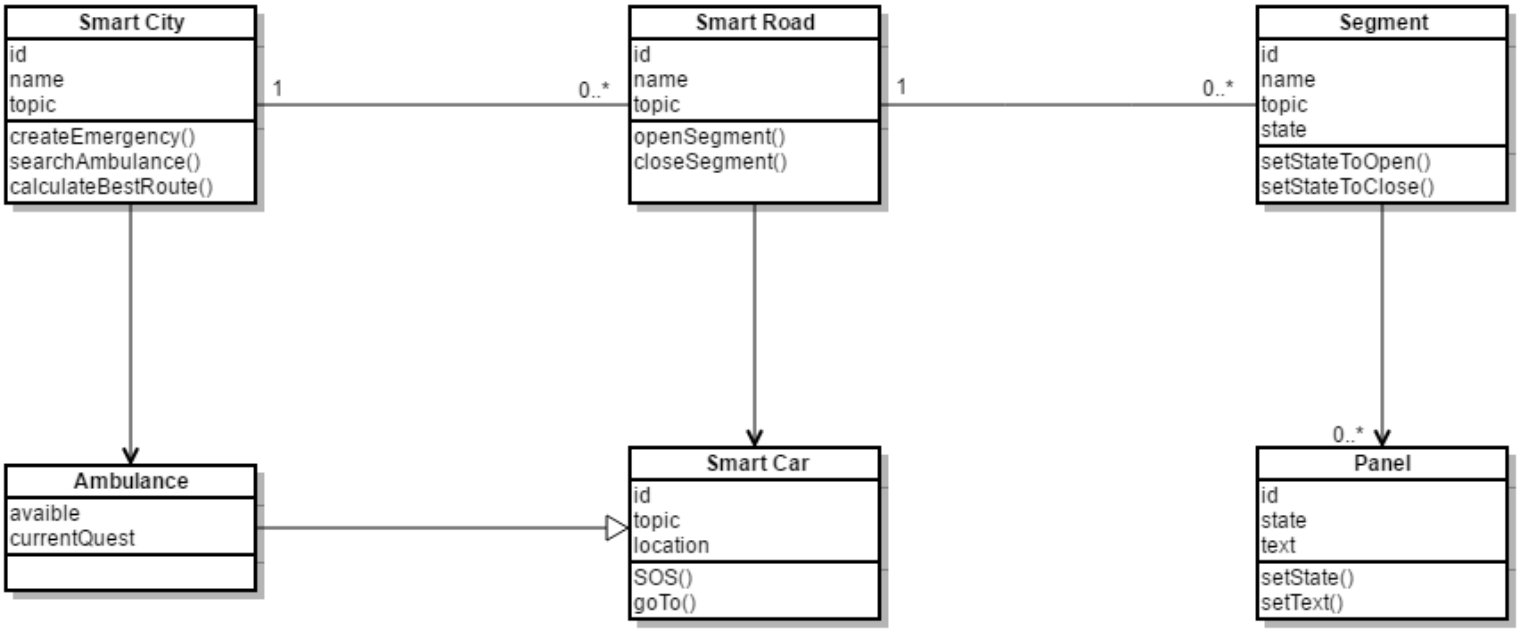


Ilustración 7: Diagrama de clases Diseño

Cada elemento presentado en la idea general se transformará en un objeto Java, el cual almacena la información que debería conocer dicho objeto mediante sus atributos de clase, y los métodos representan las acciones que debería ser capaz de realizar. En el diagrama presentado arriba, se nombran dos nuevos elementos *Panel* y *Segmento,* los cuales se tratarán con más profundidad próximamente.

Como se puede observar cada elemento contiene un **Identificador**, cuya función es distinguir explícitamente a los distintos elementos que conforman el sistema. El sistema utilizado para asignar los identificadores se explicará en el punto TODO: RELLENAR AQUI

Nota: En este proyecto solo se tendrá en cuenta una ciudad con pocos elementos, pero en un sistema globalizado interactuarían miles de ciudades con millones de carreteras, coches…

Tanto la ciudad, como la carretera y los segmentos cuentan con un atributo **Nombre**, cuya función es simplemente representar el nombre de dichos elementos, tal y como son conocidos por los seres humanos (Valencia, C/Murcia, V-30…).

El siguiente atributo común entre todos los elementos (excepto *Panel*) es el **Topic**, representa a qué cola de mensajerías deberá subscribirse cada elemento. El apartado de comunicación se detallará el punto TODO: RELLENAR AQUÍ, por lo que no entraremos en detalle en esta sección.

Por último, todos objetos mostrados en el diagrama cuentan con los métodos necesarios para realizar todas las acciones citadas en el apartado 3.2 Idea Básica. En el capítulo 6 Implementación, se explicará en profundidad como se les ha dado funcionalidad a dichos métodos.

4.2 FUNCIÓN DE CADA ELEMENTO DENTRO DEL SISTEMA Y CONCEPTOS CLAVE

Ahora que ya se conocen todos los elementos necesarios, se procede a explicar cuál es la función de cada uno. En este capítulo se explicará qué relación tiene cada elemento con la realidad, cuales son las simplificaciones que se han realizado (en caso de que existan) para adaptarlo a este proyecto, y qué funciones se espera que realice cada uno.

4.2.1 Smart City

Como se ha mencionado en la introducción de este documento, la definición de una Smart City aún no está del todo clara, pero podríamos definirla como el **"núcleo de unión entre todos los elementos del sistema"**, representa la máxima autoridad dentro de la jerarquía de elementos, y es el encargado de armonizar a todos los participantes para asegurar un correcto funcionamiento del sistema.

Dentro de este proyecto el rol de la Smart City es el de articular la respuesta que se da a la señal SOS. Para ello, deberá ser capaz de comunicarse con las Smart Road y las ambulancias de las que disponga. Además de esto, debe ser capaz de realizar los cálculos correspondientes para elegir la mejor ambulancia y calcular la ruta óptima, como de realizar un seguimiento de todas las emergencias que se den dentro de dicha ciudad (por ejemplo, que se lance más de una señal S.O.S, la ciudad ha de ser capaz de atenderlas todas al mismo tiempo).

No existe ningún sistema catalogado como "Smart City" con el cual podamos comparar funcionalidades de la ciudad inteligente diseñada, pero es de esperar que un sistema realista sería inmensamente mayor que el propuesto en este documento; ya que la casuística de la ciudad aquí diseñada solo considera el escenario – simplificado – de la asistencia sanitaria en carretera.

4.2.2 Smart Road

La Smart Road es el **principal responsable del funcionamiento y estado de las carreteras**. Es un sistema el cual cuenta con cierta independencia para auto gestionarse, pero que se sitúa en un eslabón inferior dentro de la jerarquía de la ciudad inteligente. Adaptar los límites de velocidad dependiendo de las condiciones meteorológicas, cerrar o abrir segmentos de la propia carretera en base al flujo de tráfico, mantener a los usuarios informados de su estado… todo son acciones capaces de realizarse con la información que posee el sistema encargado de las carreteras.

La Smart Road trabaja junto a la Smart City para dar respuesta a los eventos que se escapan de su radio de acción, los cuales requieren una coordinación entre varios elementos del sistema (por

ejemplo, accidentes). En concreto, este es el escenario que se contempla dentro de este proyecto. El rol de la Smart Road será el de acatar las órdenes de la Smart City cuando se de una emergencia, al igual que comunicar a la Smart City todos los mensajes de importancia que sean retransmitidos por los vehículos que circulen por la carretera y viceversa.

Nota: En una primera versión del proyecto la SmartRoad informaba cada x tiempo a los vehículos del estado de la carretera, pero se ha decidido prescindir de esta funcionalidad ya que no aportaba nada al escenario de la asistencia sanitaria.

Por último, también se podría pensar en las Smart Road como **un filtro entre los usuarios de las carreteras y el núcleo del sistema**; ya que están pensadas como un sistema capaz de gestionar todos los eventos que sucedan dentro de las carreteras, dividiendo así el peso de la autogestión de la ciudad.

4.2.3 Segmento

El segmento representa la unidad en la que se dividen las Smart Road, exactamente igual que en las carreteras convencionales. Además de actuar como piezas de una carretera, también representan una capa intermediaria entre el software encargado de gestionar la autogestión de las Smart Road y el hardware que está desplegado en la carretera.

La función principal de los segmentos es la de traducir las órdenes de la Smart Road a los distintos componentes que se hayan instalado en dicho sector (Paneles, sensores, barreras…); mediante dichas órdenes se realiza la auto gestión de la carretera.

En este proyecto la única función de los segmentos es la de representar el estado de un determinado tramo de la carretera: cerrado o abierto. Para aproximar un poco el segmento aquí utilizado a un segmento realista, todos los segmentos cuentan con *Panels,* también con la capacidad de mandar órdenes a estos *Panels* en base a las órdenes recibidas por las diferentes Smart Road.

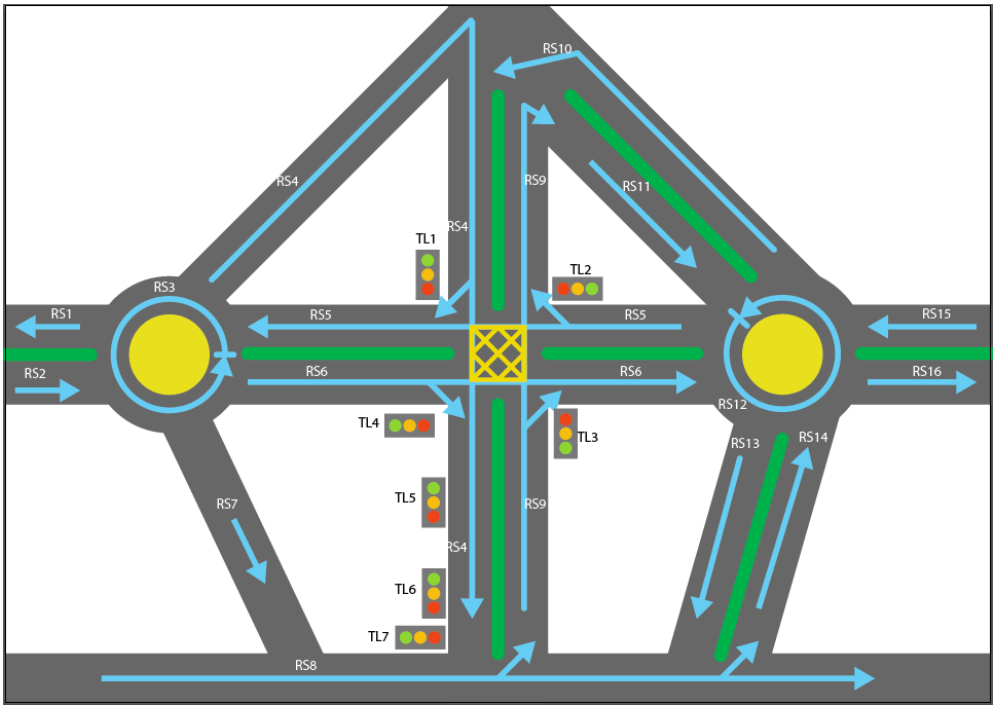


Ilustración 8: Segmento realista

En la TODO: RELLENAR ILUSTRACION AQUI se muestra como sería el diseño de un segmento realista, el cual estaría compuesto por varios carriles y distintos elementos como semáforos, indicadores de velocidad, paneles de texto… En una solución más realista la utilización de estos segmentos nos brindaría más estados que el "segmento abierto/cerrado" utilizado en este proyecto; se podría jugar con los semáforos, reservar carriles específicos, redirigir el flujo de tráfico… con el fin de crear un camino más ligero para los servicios de emergencia.

Como se puede vislumbrar con la especificación de este segmento, diseñar una solución totalmente realista necesita un nivel de detalles muy superior al aquí presentado.

4.2.4 Panel

Dentro de la categoría de "Panel" entraría todo lo que conocemos como señales viales. Todo lo que muestre una determinada información a los conductores y forme parte de la infraestructura de la carretera es considerado por este proyecto como un Panel.

La simulación supone que todos los paneles son digitales, por lo que su texto se podría modificar vía software. Esta suposición es inviable hoy en día, por lo que se aconseja distinguir entre

paneles con información “dinámica” y paneles con información “estática”; lo cual se mantendría durante todo el proceso de transición entre las carreteras convencionales y las Smart Road.

En un entorno realista los paneles serian la principal vía de comunicación entre el sistema y los usuarios de las carreteras, además de la comunicación directa entre las carreteras y los vehículos.

El rol del Panel dentro de este proyecto es únicamente el de cambiar el texto cuando el segmento al que pertenece ha sido cerrado o abierto, por lo que su función también es puramente simbólica.

Ilustración 9: Panel digital en una carretera convencional

4.2.5 SmartCar

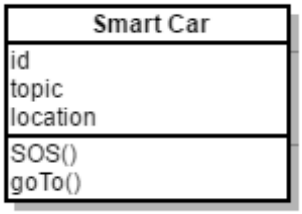


Ilustración 10: Clase Smart Car

Los vehículos son los principales usuarios de las Smart Road, y gran parte de los usuarios de la Smart City. Cuando se habla de SmartCar hablamos de un coche capaz de interactuar con el entorno que le rodea, además de ser capaz de monitorizar su estado actual, un ejemplo de SmartCar sería el modelo que utiliza IBM en sus demos (IBM, 2016).

La función básica de un Smart Car estándar es hacer uso de los servicios que brinda la ciudad inteligente. Para el caso que nos ocupa solo es necesario que los vehículos realicen dos acciones básicas, moverse dentro del mapa y lanzar una señal S.O.S. Cabe destacar que para que el

vehículo pueda realizar estas acciones se ha añadido el atributo ubicación, el cual contiene la ubicación del vehículo.

4.2.6 Ambulancia

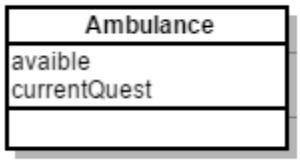


Ilustración 11: Clase Ambulancia

Las ambulancias están consideradas como Smart Car, ya que comparten la funcionalidad básica de moverse en el mapa o mandar una señal S.O.S, además de los atributos que contienen información fundamental como identificador, *topic* y ubicación.

El argumento principal por el cual las ambulancias ocupan una clase diferente a la de los vehículos estándar, es que las ambulancias son capaces de comunicarse directamente con la Smart City, ya que representan uno de los recursos que tiene el sistema para atender las distintas emergencias que se den dentro del rango de acción de la ciudad.

Por ende, las ambulancias amplían las funcionalidades de un Smart Car. Son capaces de completar misiones y atender emergencias, lo cual es la funcionalidad principal de las ambulancias dentro de este proyecto.

En el proceso de completar las misiones, las ambulancias mantendrán una comunicación con la ciudad en la cual informarán constantemente de su ubicación; el objetivo de esto es trasladar a la ciudad la información necesaria para que puede abrir o cerrar segmentos de las carreteras, y así evitar posibles escenarios en los cuales la ambulancia no podría avanzar con suficiente rapidez para atender el siniestro.

4.2.7 Emergencia & Misión

En los apartados anteriores se han mencionado los conceptos *Emergencia* y *Misión*, los cuales tienen un significado en particular dentro de este documento. Ambos conceptos representan clases dentro del sistema, cuyo objetivo es encapsular la información necesaria para atender una emergencia, y completar una misión.

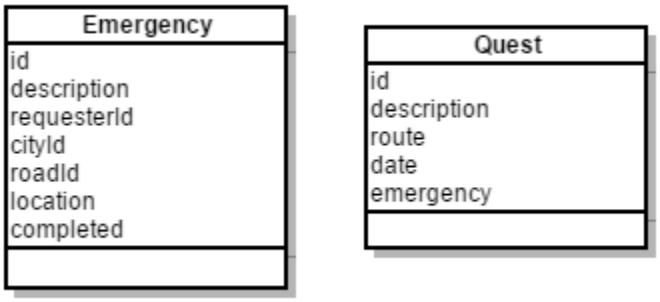


Ilustración 12: Clases Emergency y Quest

Como se puede observar la Ilustración TODO: NUMERO dichas clases no cuentan con métodos relevantes (únicamente getters and setters). El objetivo de cada clase se podría resumir de la siguiente manera:

- **Emergencia.** Representa el modo con el cual la ciudad almacena toda la información relevante de un evento el cual requiere atención urgente.

- **Misión.** Representa el modo en el que la ciudad gestionan sus recursos para atender las distintas emergencias.

4.3 ESCENARIO

Para poder visualizar la solución se creará una simulación en la cual transcurren los eventos, como en toda simulación, el entorno representa una simplificación de la realidad, con el fin de no caer en los detalles y conseguir el objetivo principal: demostrar la viabilidad de la solución propuesta.

La ciudad ficticia donde se desarrollan los hechos es Atlantis, y su mapa es siguiente:

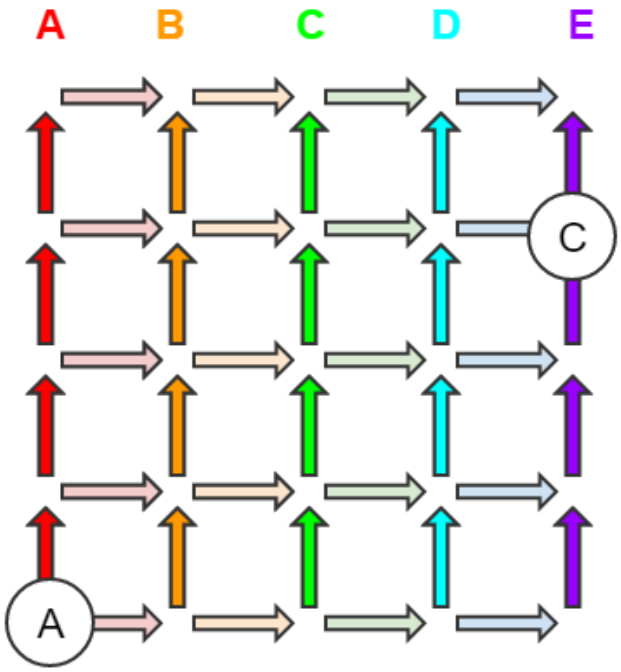


Ilustración 13: Escenario de la simulación

Cada flecha representa un segmento y el color representa a que carretera pertenece dicho segmento (que algunos segmentos tengan un color más claro solo es una ayuda visual).

Se cuenta pues con cinco SmartRoad: A, B, C, D y E. Cada SmartRoad contiene diez segmentos. La forma correcta de interpretar este mapa es como si fuese una matriz de cinco filas y cinco columnas.

Además de los segmentos y las carreteras, también se puede encontrar dos Smart Car. El círculo con la letra A en la posición (1,1) representa una ambulancia; mientras que el círculo con la letra C en la posición (4,5) representa un vehículo estándar. El vehículo C será el encargado de mandar la señal S.O.S y la ambulancia la encargada de ir a socorrerlo. Los vehículos solo pueden

desplazarse de nodo a nodo dentro de la matriz, y pueden utilizar un segmento siempre que esté abierto.

Se ha decidido utilizar una ciudad ficticia para poder focalizarse en la idea general. Me atrevería a decir que este mapa representa bastante bien lo que podría ser el esqueleto de las carreteras de una ciudad inteligente; para trabajar en un entorno más realista solo haría falta trabajar con Latitud y Longitud en vez de con los nodos de una matriz, y ampliar la cantidad de carreteras, segmentos y vehículos que circulan por esta (utilizando obviamente una definición más detallada de cada uno de estos elementos).

4.4 DISEÑO DE LA COMUNICACIÓN

Una vez se ha presentado el diseño de los distintos elementos, el siguiente paso es diseñar como va a funcionar la comunicación dentro del escenario propuesto. TODO: Añadir cosas aquí

4.4.1 Tipo de comunicación

Para diseñar la comunicación de este sistema hemos de responder a una pregunta clave **¿la información que se retransmite es de interés para el colectivo en general, o es innecesaria?**

Esta pregunta divide la comunicación en dos escenarios: la comunicación es de interés para más usuarios además de los que intervienen en la conversación, y, la información de interés solo es relevante para los usuarios que intervienen en la comunicación.

En base a estos dos escenarios se propone que las comunicaciones se realicen por dos medios distintos, en el caso de que la información sí sea relevante, se utilizarán colas de mensajería con un esquema *pub-sub*. Esto permite transmitir la información por *topics* en los cuales puede haber varios usuarios escuchando, y por lo tanto mantener informados de una forma sencilla a todos los interesados en dichos *topics.*

Por otro lado, cuando la información solo es de interés para los participantes de la conversación, se propone utilizar comunicación basada en el protocolo HTTP como podría ser REST. Con esto se consigue que cuando un usuario necesite una determinada información, pueda utilizar la API que presentan los distintos elementos de la ciudad inteligente para hacer las peticiones que se consideren oportunas.

Dicho esto, el esquema de comunicaciones quedaría como se muestra en la Ilustración TODO: RELLENAR AQUÍ. Toda la comunicación entre los elementos pertenecientes a la infraestructura de la Smart City se retransmitiría por colas de mensajería, ya que es información de interés público, además, de que puede ser utilizada por terceros.

Ejemplo: una Smart Road escucha que la ciudad ha cerrado X segmento, puede prepararse para un aumento de tráfico.

Por otro lado, la comunicación que tiene un vehículo con la carretera en la que se encuentra no es de interés público, por lo que se realiza vía REST.

Ejemplo: un vehículo acaba de incorporarse a una carretera y desea conocer el estado de esta. Para realizar dicha comunicación el vehículo haría una petición a la API de la carretera.

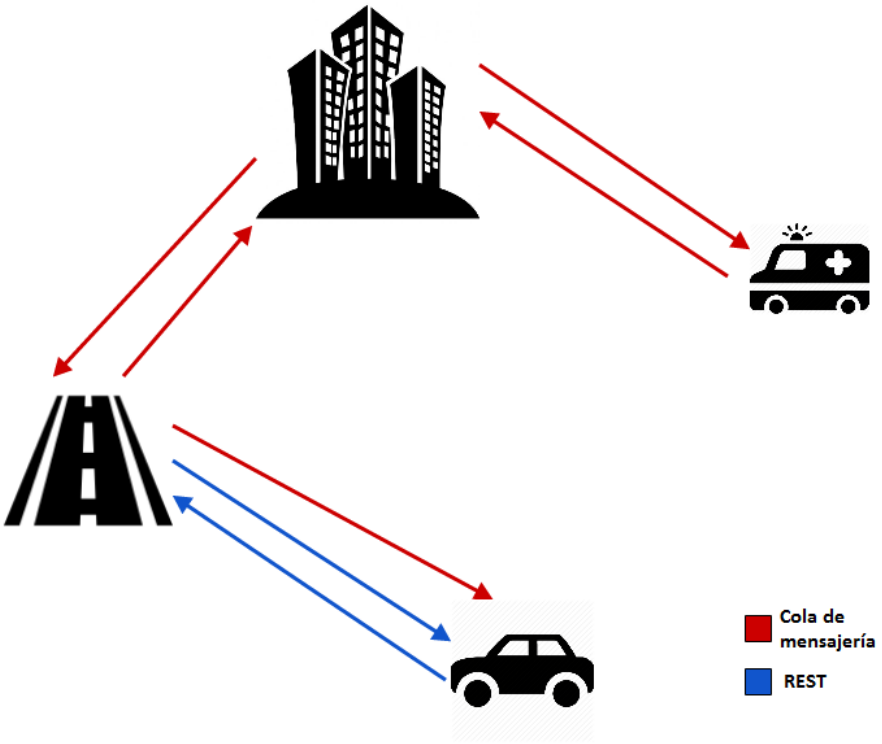


Ilustración 14: Esquema básico de la comunicación dentro del sistema

Con el diseño que acabamos de realizar, toda la comunicación necesaria para responder una solicitud de S.O.S es posible. En la ilustración TODO: NUM se muestra un diagrama de secuencia el cual representa todas las comunicaciones que se han de establecer desde la señal de SOS hasta que la ambulancia confirma que se ha atendido la emergencia correctamente. Destacar que como toda la comunicación es entre elementos de la ciudad inteligente se haría vía colas de mensajería; la comunicación vía REST no tiene un papel relevante dentro de la respuesta al S.O.S.

Todos los mensajes construirán utilizando el formato JSON, tal y como se ha indicado en el contexto tecnológico se considera que este formato es el más adecuado para este proyecto. El formato en concreto que tendrán los mensajes se verá en profundidad en el apartado de Implementación.

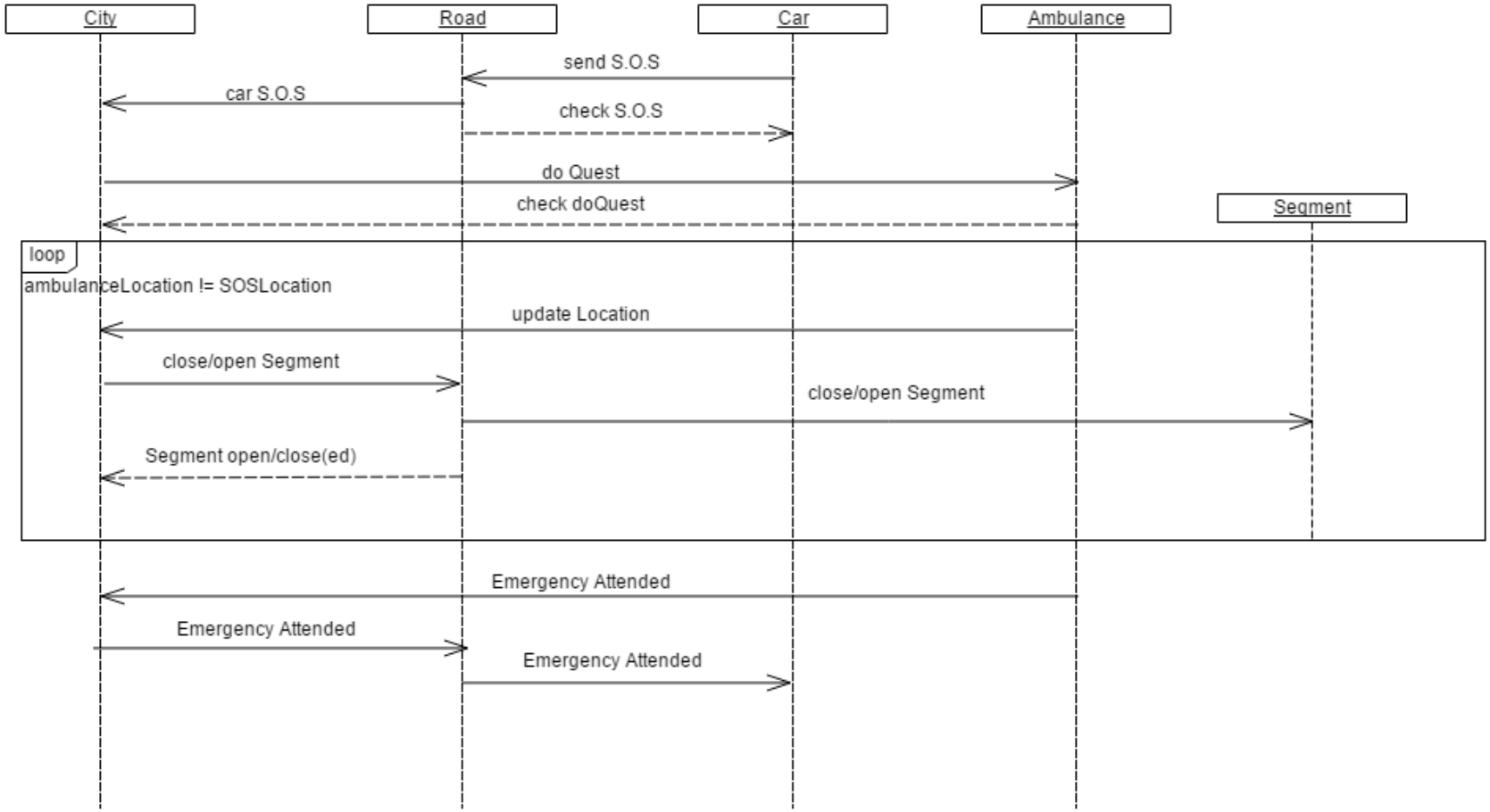


Ilustración 15: Secuencia UML diseño

4.4.2 Códigos

En esta implementación los códigos son la forma en la que el software identifica el objetivo que tiene cada mensaje. Se componen de cuatro dígitos (TCCC) donde el primer dígito T representa el “Tema” y los tres últimos representan el “Caso”.

El **Tema** recoge a todos los mensajes que están relacionados con un mismo campo. Ejemplo: el tema número 1 corresponde a “Información Request”, por lo que todos los mensajes que tengan como objetivo solicitar información utilizarán un código que empezará con el número 1.

Cabe destacar que todos los códigos del tema 1 al tema 4 representan “solicitudes”, mientras que los códigos que se sitúan entre el tema 5 hasta el tema 8 representan las “respuestas” a dichas solicitudes.

El **Caso** representa una determinada acción dentro de un tema. Ejemplo: dentro del tema 3 “Misiones” existen varios casos; 000 (Send Quest), 001 (Open Segment), 002 (Close Segment) …

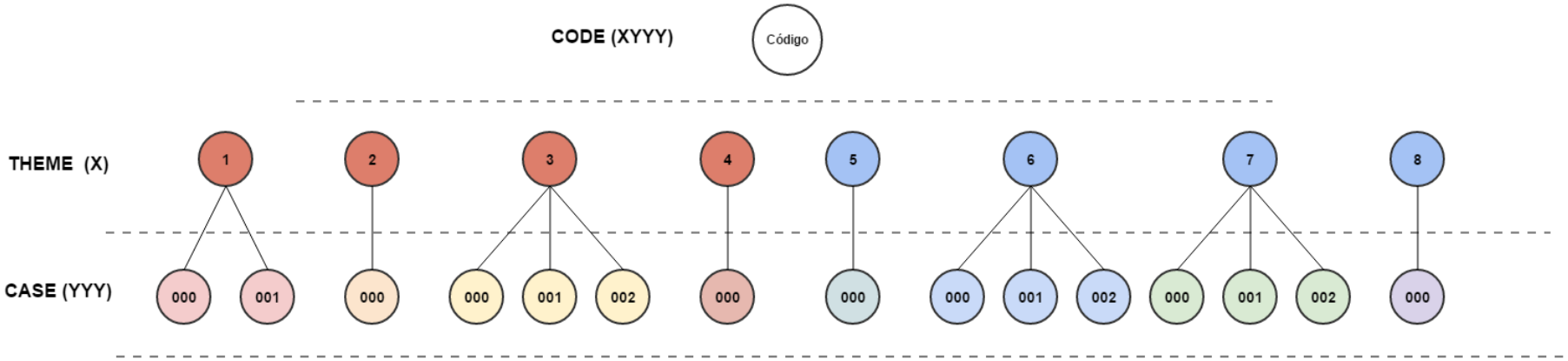


Ilustración 16: Esquema de los códigos utilizados en la comunicación

5 IMPLEMENTACIÓN

5.1.1 Formato de los mensajes

Todos los mensajes dentro de la simulación se estructuran en base al formato JSON. Además de seguir dicho formato, deben contener los siguientes atributos obligatoriamente:

- **Código:** Un código de cuatro dígitos que representa una determinada acción dentro del sistema. Cada elemento del sistema puede responder / mandar unos determinados códigos. La función principal es crear una coherencia y orden entre los mensajes que circulan por el sistema.

- **SenderId:** Identificador del elemento que ha enviado el mensaje.

- **ReceiverId:** Identificador del elemento que debe recibir dicho mensaje. Este campo puede ser *null* en determinadas situaciones.

- **Message:** El mensaje a menudo coincide con la descripción del código. Suele contener una breve descripción en "lenguaje humano" del objetivo del mensaje. El objetivo fundamental es que un profesional pueda leer este campo para poder entender el objetivo del mensaje sin tener que recurrir a la tabla de códigos.

- **Atributos opcionales:** En determinados casos la información básica no será suficiente para gestionar el mensaje, por lo que es posible que se deban añadir nuevos atributos con la información necesaria. Este campo no tiene límite, ya que cada código define los atributos necesarios para tratar una determinada operación.

La idea que subyace detrás de la obligatoriedad de estos atributos es poseer siempre una información básica para que no exista mensaje que no pueda recibir respuesta, ya sea porque el *sender* no esté autorizado, que el *receiver* no pueda atender la solicitud o simplemente que el código del mensaje no exista. En cualquier caso, con estos campos el sistema será capaz de responder en cualquier escenario.

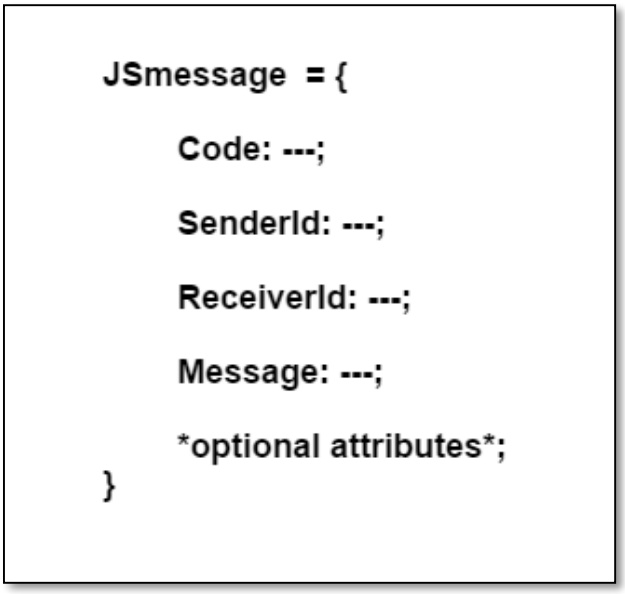


Ilustración 17: Formato Mensajes

5.1.2 Identificadores

Para identificar a los distintos usuarios de la comunicación dentro de la simulación se ha diseñado un formato de identificadores muy sencillo. Como se verá en el apartado de **Mqtt & Rest** solo hay cuatro elementos que se comuniquen: SmartCity, SmartRoad, SmartCar y Ambulance.

A cada uno de estos elementos se les asigna una letra:

- **C**ity (SmartCity)
- **R**oad (SmartRoad)
- **V**ehicle (SmartCar)
- **A**mbulance (Ambulance)

Esta letra representará el primer símbolo del identificador, seguido de un guion y un número de tres dígitos, los cuales representan un contador del número de elementos que se hayan creado. Dicho esto, en la Ilustración TODO se muestran algunos ejemplos.

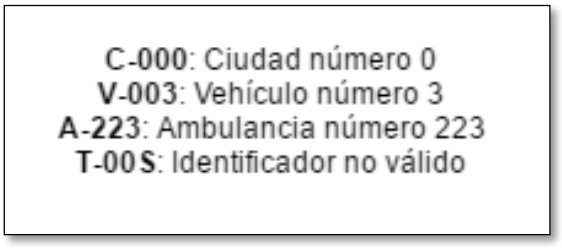


Ilustración 18: Ejemplos de Identificadores

Para finalizar, clarificar que no pueden existir dos identificadores iguales, y que en la simulación no se reciclan identificadores. En una solución más realística los elementos que formasen parte de la infraestructura del sistema contarían con identificadores estáticos, mientras que, en el caso de los vehículos seguirían un esquema como el de las IP dinámicas: el identificador sería solicitado cada vez que el usuario accede al sistema, y cuando deje de utilizarlo dicho identificador se le asignaría a otro vehículo.

5.1.3 Descripción de cada código (Implementación)

En la tabla a continuación se muestran todos los códigos que se utilizan en este proyecto.

CÓDIGO	TEMA	CASO	DESCRIPCION
1000	1	000	¿Dónde estoy? (Broadcast)
1001	1	001	Manda la ubicación del SmartCar.
2000	2	000	Llamada de auxilio (S.O.S)
3000	3	000	Envío de una nueva misión
3001	3	001	Orden de abrir un Segmento
3002	3	002	Orden de cerrar un Segmento
4000	4	000	Notificar a la SmartCity que hay un nuevo SpecialVehicle
5000	5	000	Respuesta a la solicitud ¿Dónde estoy?

6000	6	000	Confirmar que la llamada S.O.S ha sido recibida correctamente
6001	6	001	Tiempo hasta estimado llegue a la ubicación de la llamada S.O.S (Respuesta a una llamada de auxilio S.O.S)
6002	6	002	La emergencia ha sido atendida (Respuesta a una llamada de auxilio S.O.S)
7000	7	000	Check Quest (Confirmar que se ha recibido la misión correctamente)
7001	7	001	El segmento ha sido abierto correctamente
7002	7	002	El segmento ha sido cerrado correctamente
8000	8	000	Un nuevo SpecialVehicle ha sido añadido a los servicios de la ciudad correctamente.

5.1.4 Atributos de los códigos (Implementación)

A continuación, y en caso de que sea necesario, se explicarán los atributos opcionales (en caso de que existan) todos los códigos.

CÓDIGO	SenderId	ReceiverId	Opcionales
1000	Vehicle	*Null*	Location Type
1001	Ambulance	City	LastLocation CurrentLocation NextLocation
2000	Vehicle	*Null*	Location

3000	City	SpecialVehicle	Quest AmbulanceId
3001	City	Road	RoadName Segment Ini Segment End
3002	City	Road	RoadName Segment Ini Segment End
4000	SpecialVehicle	City	Location Type
5000	City	Vehicle	Topic
6000	Road	Vehicle	-
6001	Road	-	Tiempo
6002	Road	Vehicle	-
7000	Ambulance	City	-
7001	Road	City	-
7002	Road	City	-
8000	City	Special Vehicle	-

5.2 MQTT & REST

Para este proyecto se ha elegido MQTT y REST como bases de la comunicación entre dispositivos, el uso de uno u otro depende del escenario:

- Se utiliza MQTT cuando se quiere dar a la información un carácter público, es decir, cuando además de los interlocutores haya más interesados en dicha información.

Ejemplo. Una carretera ha sido cerrada por un accidente, esta información puede ser utilizada por otras carreteras gestionar su flujo de tráfico, por los GPS de los vehículos para rediseñar su ruta…

- Se utiliza REST para establecer comunicaciones privadas cuando se quiere transmitir información confidencial, o para evitar propagar información innecesaria para el resto de usuarios.

Ejemplo. Un vehículo acaba de incorporarse a una carretera y desea conocer el estado de esta. Al resto de usuarios no les aporta nada saber que X vehículo solicita saber el estado de la carretera.

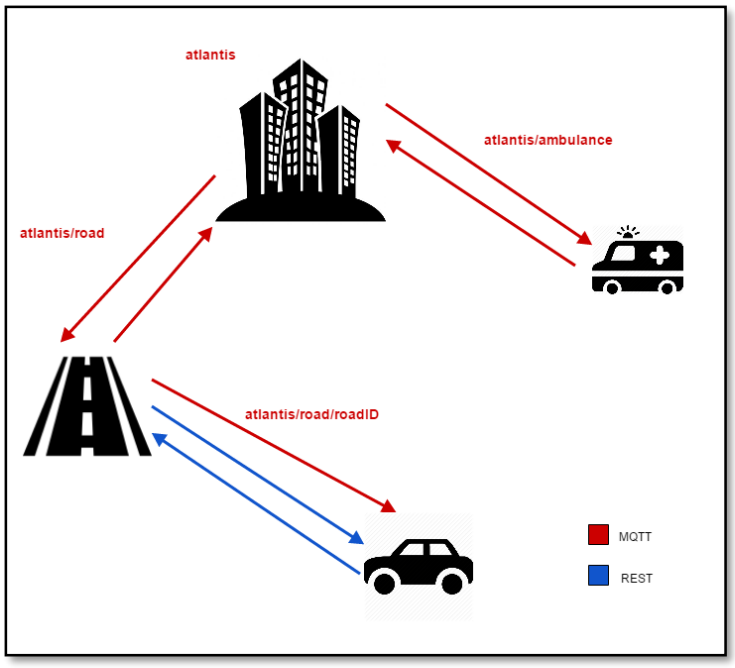


Ilustración 19: Esquema de comunicación

5.3 COMUNICACIÓN MQTT

Existen varios *topics* dentro de la simulación, y cada uno tiene una función definida. Quién puede o no puede suscribirse a cada uno de estos topics es algo que, por ahora, no está definido.

(TODO: información de las city totalmente publica, o habrá restricciones?)

- **atlantis** Este *topic* está reservada para eventos que solo puedan ser atendidos por las SmartCity (emergencias, gestión de misiones…), por lo que, los únicos que pueden hacer uso de el son los elementos que forman parte de la infraestructura de las SmartCity (SmartRoad, vehículos autorizados…)

- **atlantis/ambulance** Como bien indica el nombre, este *topic* representa el puente comunicativo entre las ambulancias que trabajan para la ciudad, y la misma ciudad. Gestionar misiones, seguimiento de la ubicación de las ambulancias, averías… son algunos de los temas tratados por este *topic.*

- **atlantis/road** *Topic* reservado para la comunicación entre las carreteras y la ciudad. Toda la información de las carreteras es retransmitida por aquí; también se utiliza para comunicar emergencias, cerrar o abrir segmentos…

- **atlantis/road/roadID** Cada carretera tiene su propio *topic* por el cual la carretera se comunica con los vehículos que circulan por esta. Cabe destacar que esta comunicación no es bidireccional, la carretera informa a los suscriptores de información válida para todos, pero si un vehículo necesita comunicarse con la ciudad, lo debe hacer vía REST y no utilizando este canal. La funcionalidad principal de este canal es mantener a los vehículos que circulan por la carretera informados del estado de esta.

5.4 COMUNICACIÓN REST

Como se ha mencionado anteriormente, el objetivo de la comunicación REST en esta simulación es mantener comunicaciones 'privadas' o evitar propagar información 'inútil' para el resto de usuarios.

Dentro del escenario de la ambulancia, la única comunicación REST que se puede encontrar es la que establece el SmartCar con la carretera, para hacer la llamada S.O.S.

(TODO: Explicar detalles de la comunicación REST cuando esté diseñada/implementada)

6 SIMULACIÓN

En el siguiente tema se explicará paso a paso como la solución propuesta se desarrolla dentro de la simulación, esto se hará desde una perspectiva más técnica con el fin de demostrar la viabilidad de la solución planteada anteriormente.

6.1 CARGA DEL ESCENARIO

El primer paso es realizar la carga del escenario en Java. Esto se hace a través de la clase *EnvironmentOne.java,* todos los elementos de la simulación son objetos Java inicializados en dicha clase. Los constructores de cada objeto Java almacenan toda la información requerida para realizar la simulación, los parámetros que necesitan representan la Input que necesitará el sistema para ubicar dicho elemento dentro de la infraestructura de la SmartCity.

Después de crear la SmartCity y todas las SmartRoad, el siguiente paso es añadir los segments a las SmartRoad con sus respectivos Panels. Una vez hecho esto solo faltará crear los dos SmartCar, la ambulancia y el vehículo estándar que será el encargado de dar la señal S.O.S. Se ha de destacar que una vez se crea la ambulancia, esta se registra como un "SpecialVehicle" dentro de la ciudad a la que pertenezca, lo que le permitirá a la ciudad mantener un seguimiento de los recursos disponibles para atender las emergencias.

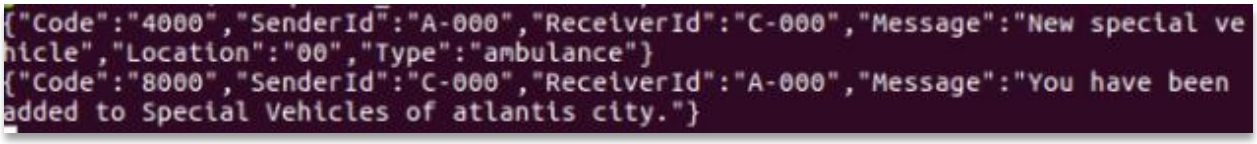


Ilustración 20: Ambulance avisa a la ciudad de que hay un nuevo SpecialVehicle. La ciudad lo añade y le manda un Check.

Una vez hecho esto, todo el escenario está preparado para empezar la simulación.

6.2 SEÑAL S.O.S

(Aclarar como manda la señal de SOS el coche, via Mqtt o via REST…)

El coche manda una señal S.O.S a través de (completar aquí) la cual es recibida por la SmartRoad a la que se ha suscrito. Esta señal se mandará cada “x” tiempo hasta que se reciba un mensaje de confirmación como señal de que la emergencia está siendo atendida.

Ilustración 21: Señal S.O.S (JSON)

Nota: La simulación supone que el SmartCar es capaz de reconocer en que carretera se encuentra en base a su ubicación GPS.

Una vez la señal ha sido recibida por la SmarRoad a través del hardware instalado en los segmentos, se activa el protocolo de actuación ante una señal S.O.S. La primera acción que realiza la carretera es mandar un mensaje a la ciudad usando el *topic* atlantis. Hecho esto, mandará un mensaje a modo de “check” al vehículo que hay mandado la señal S.O.S.

Una vez la ciudad haya mandado un mensaje corroborando que ha recibido la información del accidente, la señal habrá llegado al núcleo del sistema, donde dará comienzo una respuesta automática por parte de la ciudad.

6.3 PLANIFICACIÓN DEL PLAN PARA ATENDER LA EMERGENCIA

Una vez la señal ha llegado a la ciudad, el primer paso es crear una Emergencia. Dentro de la simulación las Emergencias son objetos java que representan toda la información relacionada con el evento en cuestión, hora de la llamada, ubicación, estado de la emergencia… Una vez se ha creado el objeto, se inserta en una cola de emergencias pendientes, que es el principal mecanismo de la SmartCity para llevar control sobre todas las emergencias que se están gestionando en un determinado momento.

Ilustración 22: SmartRoad notifica a la SmartCity de una nueva emergencia

Hecho esto, se procede a comenzar la búsqueda de la mejor ambulancia para atender el siniestro, la búsqueda se realiza en una lista que contienen todas las ambulancias disponibles a servicio de la ciudad en cuestión. Actualmente el único factor que se tienen en cuenta a la hora de buscar una ambulancia es la distancia a la que se encuentra del siniestro, aunque perfectamente podría hacerse en base a otros factores.

Una vez se ha seleccionado la ambulancia óptima para atender dicha emergencia se procede a calcular la ruta óptima desde la ubicación de la ambulancia hasta la ubicación del siniestro. El cálculo de una ruta óptima es complicado y difícil de definir, existen muchos factores: fluidez del tráfico, disponibilidad de carriles especiales, capacidad para abrir o cerrar segmentos, zonas urbanas por las que hay peligro de accidente… por lo que para evitar emplear tiempo en un método el cual necesitaría del apoyo de varios expertos, dentro de la simulación se ha simplificado a “la distancia mínima entre dos nodos de la matriz”.

Con la información de la emergencia, la ambulancia óptima seleccionada y una ruta calculada la SmartCity posee todos los elementos necesarios para crear una Misión. Dentro de la simulación las misiones son objetos java, al igual que las emergencias, que contienen toda la información necesaria para que un recurso de la ciudad realice las tareas que se le han encomendado de una forma eficiente. Una vez creado el objeto Misión, se mandará a la ambulancia a través del *topic* atlantis/ambulance. Una vez hecho esto, la SmartCity queda a la espera de una confirmación por parte de la ambulancia.

Una vez la ambulancia haya confirmado que puede realizar la misión encomendada, la ciudad marcará la Emergencia como “en progreso” y cambiará el estado de la ambulancia de disponible a “en misión”. Finalmente, la ciudad mantendrá un seguimiento de la emergencia hasta que un profesional confirme que la emergencia ha sido atendida.

6.4 GESTIÓN DE SEGMENTOS CON AMBULANCIA EN RUTA

Con el objetivo de ahorrar el máximo tiempo en el camino hacia el siniestro, la SmartCity puede abrir o cerrar segmentos al tráfico estándar. Para ser lo más preciso posible con estas acciones, una vez la ambulancia se pone en ruta, empieza a mandar un mensaje con su ubicación cada que va a abandonar el segmento en el que se encuentra, el mensaje se manda a través del *topic* atlantis/ambulance.

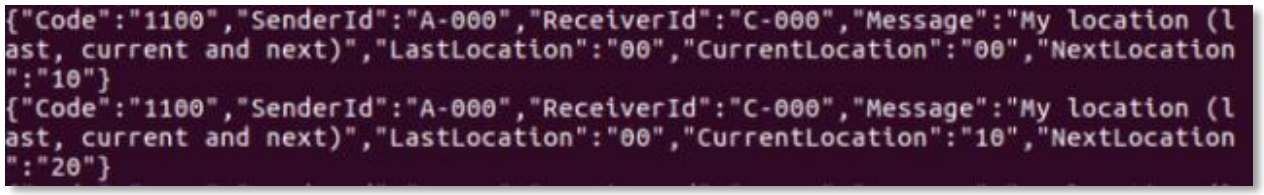


Ilustración 23: Ambulancia informando de su ubicación actual.

El mensaje es leído por la ciudad, y se considera si es necesario cerrar algún segmento en la ruta de la ambulancia. En la simulación que se ha implementado se utiliza un algoritmo simplista para abrir y cerrar segmentos. Cada vez que la ambulancia manda su ubicación, la smartCity manda cerrar el próximo segmento en la ruta de la ambulancia, y en caso de que sea necesario, manda abrir el segmento por el que acaba de circular la ambulancia.

Ilustración 24: Comunicación entre la ciudad y las Road mientras se abren y cierran segmentos.

Este proceso se realiza utilizando el *topic* atlantis/road con el objetivo de que todas las carreteras sean conscientes de los segmentos que se van a cerrar o abrir, y puedan prepararse para un posible flujo de tráfico superior al que esperaban.

Como es habitual, cada vez que se manda una orden se ha de esperar a recibir un mensaje de confirmación, en este caso por parte de las SmartRoad. Cuando la orden llega a las carreteras, el cierre o apertura de los segmentos se hace mediante métodos brindados por los objetos Java. En este caso, las carreteras cuentan con una lista en la cual almacenan todos los segmentos que forman parte de ellas, por lo que se busca el objeto con un identificador idéntico al que viene en el mensaje mandado por la SmartCity, y se procede a utilizar sus métodos para cambiar su estado. El cambio de estado de cualquier Segmento también provoca un cambio en el texto mostrado por los paneles que formen parte de dicho segmento.

Este bucle de abrir/cerrar segmentos se prolonga hasta que la ambulancia haya llegado a su destino. Esto se detectará porque la ambulancia mandará un mensaje donde confirmará que ha llegado a su destino.

6.5 LA AMBULANCIA LLEGA A SU DESTINO

Una vez la ambulancia ha llegado a la ubicación del siniestro pueden darse dos situaciones:

- El profesional considera que hacen falta más recursos para atender la emergencia. Esto se comunicará a la ciudad a través del *topic* atlantis, en el mensaje se deberá especificar la situación en la que se encuentra el siniestro, y cuantas son las unidades que el profesional piensa que son necesarias para atender el siniestro. El proceso encargado de gestionar la emergencia inicial deberá redefinir la emergencia con la información recibida, y cambiar su estado a "pendiente". *Nota: Este escenario no ha sido implementado dentro de la simulación.*

- El profesional se ve capacitado para atender la emergencia y procede a ello. Una vez la emergencia haya sido atendida se le comunicará a la ciudad vía el *topic* atlantis/ambulance. Hecho esto la ciudad borrará la emergencia de la lista de emergencias pendientes y cambiará el estado de la ambulancia de "en misión" a "disponible".

Ilustración 25: La SmartRoad comunica a el vehículo de que su emergencia ha sido atendida correctamente.

En este punto la simulación habría finalizado. La ambulancia se encontraría en la misma ubicación que el coche que envió la señal S.O.S y su estado volvería a ser "disponible". Todos los segmentos estarían abiertos, y la ciudad no tendría ninguna emergencia pendiente por tratar.

7 CONCLUSIONES

8 REFERENCIAS

Ashton, K. (22 de 06 de 2009). *RFID Journal*. Obtenido de http://www.rfidjournal.com/articles/view?4986

BCN. (15 de 06 de 2016). *BCN Smart City*. Obtenido de http://smartcity.bcn.cat/

DGT. (2014). *Dirección General de Tráfico - Ministerio del Interior - Gobierno de España*. Obtenido de http://www.dgt.es/es/seguridad-vial/estadisticas-e-indicadores/accidentes-30dias/tablas-estadisticas/2014/

DGT. (2016). *Dirección General de Tráfico España*. Obtenido de http://www.dgt.es/es/prensa/notas-de-prensa/2016/20160104-nuevo-minimo-historico-numero-victimas-mortales-accidente-desde-1960.shtml

Fonts, G. C. (2014/2015). *Diseño y Prototipación de servicios en una ciudad inteligente aplicando soluciones en la Internet de las Cosas. Control de Semáforos y de Carreteras.* Valencia: UPV - Escola Tècnica Superior d'Enginyeria Informàtica.

IBM. (2016). *Connected Car* . Obtenido de http://m2m.demos.ibm.com/connectedCar.html

OMS. (14 de 06 de 2016). *World Health Organisation* . Obtenido de http://www.who.int/violence_injury_prevention/road_traffic/en/

UW. (10 de 06 de 2016). *Inhabitat - Students Install the World's First Solar Pavement Panels in Virginia*. Obtenido de http://inhabitat.com/students-install-the-worlds-first-solar-pavement-panels-in-virginia/

WIEN. (2015). *Technische Universitat WIEN*. Obtenido de http://www.smart-cities.eu

¿Por qué Mqtt?

http://mqtt.org/

http://mosquitto.org/

http://www.eclipse.org/paho/

¿Por qué JSON?

http://www.json.org/

http://www.valencia.es/ayuntamiento/DatosAbiertos.nsf/0/2113BD9D1693D7EAC1257C6600449981/$FILE/API%20APPCIUDAD%20v3.pdf?OpenElement&lang=1

NOTAS

Nota: Para que esto sea posible hace falta una infraestructura básica, por lo que en el proyecto se da por hecho que todos los elementos cuentan con la tecnología necesaria para llevar a cabo las comunicaciones y cálculos que la solución presentada necesita.

TODO: PONER ESTO EN EL CASO DE ESTUDIO

¿Cómo se va a reducir el tiempo? Bueno, si lo único que necesitamos es llegar antes a un sitio, todos conocemos la solución, solo necesitamos "salir antes" e "ir más rápido". En base a estos dos conceptos, el proyecto propone cambios en tres puntos clave:

- **Obtener información con mayor rapidez.** No se puede atender un siniestro del cual se desconoce su existencia. La primera propuesta del TFG es dotar a los SmartCar la capacidad de mandar una señal S.O.S cuando detecten que el coche ha sufrido un accidente, o cuando el conductor considere que está en una situación de emergencia.

- **Ser capaces de responder automáticamente.** Una máquina siempre procesará la información con mayor rapidez que un ser humano. Esta solución propone crear un software capaz de actuar en caso de emergencia, con cierta independencia de un supervisor humano.

- **Si no existe un camino adecuado, hay que crearlo.** No se puede tolerar que un vehículo de asistencia sanitaria quede inmovilizado por el tráfico en mitad de una emergencia. Para solucionar esto, se propone que el software tenga la capacidad de gestionar la circulación del tráfico con el fin de evitar el mayor número de obstáculos en ruta.

TODO: PONER ESTO EN CASO DE ESTUDIO, PERO MÁS RESUMIDO

Visión general de la propuesta

6. REFERENCIAS

La solución propuesta se puede dividir en tres partes, cada una de estas partes hace referencia a los puntos comentados en el apartado de Objetivo.

La información del siniestro llega al sistema. Una vez el SmartCar ha detectado que el vehículo ha sufrido un siniestro, o el conductor detecte que está en una situación de emergencia, se mandará una señal S.O.S la cual será recibida por la infraestructura de la carretera.

Respuesta automática. Una vez la señal S.O.S ha llegado a la infraestructura del sistema, se inicia un proceso automático el cual dará una respuesta a dicha señal. Primero la carretera debe comunicar a la SmartCity la existencia de una nueva emergencia, una vez hecho esto mandará una señal de "Check S.O.S" al vehículo siniestrado, para que, si este puede recibirla sea consciente de que su emergencia está siendo atendida.

A continuación, la SmartCity debe decidir dos cosas, primero: **¿qué ambulancia es la más indicada para asistir la emergencia?** El criterio de evaluación, a priori, define como mejor ambulancia aquella que se encuentre más próxima a la ubicación del siniestro; y segundo: **¿cuál es la ruta óptima desde la ubicación de la ambulancia hasta el siniestro?** Este es un cálculo complejo, ya que además de la distancia, se han de contemplar factores como el tráfico, estado de la carretera, opciones para abrir o cerrar carriles etc…Una vez decidida qué ambulancia asistirá a la emergencia y qué ruta seguirá, la SmartCity mandará una misión a la ambulancia elegida, la cual debe corroborar que ha recibido la misión y que está en proceso.

Por último, y con el objetivo de que la ambulancia encuentre el **mínimo número de obstáculos en su ruta**, se entra en una etapa en la cual la ambulancia enviará constantemente su ubicación a la SmartCity; con esta información, la SmartCity decidirá si es necesario cerrar o abrir algunos carriles en la ruta de la ambulancia. En caso de ser necesario, se pondrá en contacto con las carreteras, las cuales tienen la autoridad de cerrar carriles al tráfico estándar.

Una vez la ambulancia haya llegado a la ubicación pueden darse dos escenarios:

- El profesional en cuestión considera que necesita más recursos para atender el siniestro, por lo que mandará una nueva petición a la SmartCity con un informe de la situación.

- La emergencia puede ser atendida por el profesional, el cual manda manda un "check" a la city como que la emergencia ha sido atendida.

Antes de cerrar la emergencia por completo, la SmartCity manda un mensaje a la carretera, la cual comunicará al coche que mandó la señal S.O.S que su emergencia ha sido atendida.