Sistema de interacción con la ECU de un automóvil empleando procesamiento de lenguaje natural

Martín Agüero, Maximiliano Colman, Fabrice Dubois,
Marcelo Rabadán, Santiago Amarilla

Universidad de Palermo, Facultad de Ingeniería, Buenos Aires, Argentina
aguero.martin@gmail.com, maximilianocolman@gmail.com, kerygme@riseup.net,
mrabadan@dilmi.com.ar, amarilla.santiago@gmail.com

Resumen. En este trabajo se realiza una breve revisión general acerca de las oportunidades y desafíos que presenta la intención de la industria de sumar a Internet objetos de la vida cotidiana. Esta tendencia denominada Internet de las Cosas, implica cambios en el modo como las personas interactuarían con esos objetos-dispositivos-cosas. En base a la disponibilidad de datos a través de la interfaz OBD2 (On Board Diagnostics 2) presente en todos los automóviles y la masividad de las redes sociales, se propone un sistema que permita a las personas interactuar con su auto empleando lenguaje coloquial y a través de Twitter. El prototipo convierte las consultas en comandos que son dirigidos a la ECU (Electronic Control Unit) y la respuesta es traducida a lenguaje humano no experto. Empleando técnicas de procesamiento de lenguaje natural (NLP) y un sistema experto, se confirma la viabilidad de estas herramientas para crear interfaces más humanas para los dispositivos. Para una siguiente etapa, se planea incorporar análisis por distancia semántica y comparación de estructura sintáctica con patrones locales.

Palabras clave: osgi, java, ecu, obd2, nlp, iot, siot, expert system.

1 Introducción

En las redes sociales, las personas comparten opiniones, emociones y todo tipo de contenidos en ámbitos mayormente de acceso público. En la vida cotidiana las personas no solo interactúan con otras personas sino que también lo hacen con los objetos que las rodean, como por ejemplo, libros, autos, películas y todo tipo de dispositivos en general. En la emergente Internet de las Cosas (IoT - Internet of Things), esos dispositivos comenzarán a comunicar tanto su estado como funcionalidad. A través de esos mensajes, un electrodoméstico podrá avisar cuando completó su trabajo o el estéreo del auto comunicará a los seguidores del conductor qué tema está escuchando [1]. El paradigma IoT agrega nuevas dimensiones a la conectividad: objeto-a-objeto, humano-a-objeto y negocio-a-objeto. Estos dispositivos, que poseen capacidad de sensar, actuar y procesar datos, definirán un entretejido omnipresente que modificará el modo como la sociedad interactúa con ellos [2]. Esta realidad está dando lugar a la creación de una IoT Social (SIoT) donde esas *cosas* sociabilizan [3]. El trabajo de Mendes [4] introduce la idea de objetos con

capacidad de participar en conversaciones donde anteriormente sólo los humanos formaban parte. Está claro que ese diálogo entre humanos y objetos deberá ser a través de mensajes en lenguaje natural y coloquial.

El proceso de comprender el significado de esos mensajes, no sólo estará basado en modelos determinísticos o sintácticos. Ese análisis también requerirá de técnicas semánticas para evaluar en contexto cada comunicación de modo tal que esos dispositivos sean capaces de interpretar y responder a otros mensajes emitidos por humanos y otros dispositivos.

Tomando como referencia la arquitectura propuesta para servidor de SIoT [5], este proyecto tiene como objetivo diseñar un prototipo de middleware cuya función principal será la de interpretar consultas en lenguaje humano coloquial y traducirlas en solicitudes a un dispositivo de uso cotidiano. A su vez, las respuestas emitidas por el objeto, serán traducidas a lenguaje humano de modo tal que puedan ser comprendidas por un usuario no experto.

Dado que todos los automóviles actuales están equipados con al menos una unidad de control electrónico (ECU – Electronic Control Unit) que gestiona las principales funciones del motor, está conectada a un gran número de sensores y ofrece como interfaz el estándar OBD2 (On Board Diagnostics 2), se seleccionó al automóvil como dispositivo-cosa-objeto sobre la cual realizar la prueba de concepto. Es por ello que actualmente los vehículos, equipados con tecnología wireless, ya pueden ser considerados como dispositivos IoT móviles [6].

El prototipo se probó con éxito en una SBC (Single-Board Computer) Raspberry Pi conectada a la ECU de un automóvil mediante un scanner de interfaz OBD2 y acceso a Internet por red celular. El canal de comunicación es la red social Twitter.

A continuación, la sección 2 describe brevemente las principales características de la tecnología seleccionada. Luego, la sección 3 describe el prototipo desarrollado. En la sección 4 se presentan posibles casos de uso cotidianos y en la sección 5 se mencionan las actividades de transferencia a la industria efectuadas a partir de este proyecto. Finalmente, la sección 6 presenta las conclusiones y trabajos futuros propuestos.

2 Tecnologías

A continuación se describen brevemente las tecnologías que forman parte del proyecto:

2.1 Plataforma de software: OSGi

A finales de los años 90 la industria detectó la necesidad de contar con un framework que permitiera interconectar pequeños dispositivos para domótica y creó la OSGi Alliance [7]. Este consorcio de empresas surgió como iniciativa marco para el desarrollo de especificaciones que promovieran la estandarización del software a partir de la modularización de sus componentes, facilitando así la reutilización y buscando un equilibrio en el acoplamiento entre sus elementos. OSGi también define un ciclo de vida para sus elementos con el principal propósito de permitir cambios en

sus enlaces, sin la necesidad de detener el sistema completo. El alcance inicial de esta iniciativa se amplió a punto tal que hoy numerosos sistemas de aplicación general han adoptado la especificación. Un software muy difundido y que implementa OSGi como tecnología troncal, es el entorno de desarrollo integrado (IDE) Eclipse, donde cada plugin es un módulo OSGi.

Con el advenimiento de la IoT y por la natural adaptabilidad a dispositivos de limitada capacidad de procesamiento, OSGi resultó ser ideal para el ecosistema IoT. Esta situación derivó en una serie de estándares y frameworks orientados de manera específica a esta nueva tecnología como la comunidad open source IoT dentro de la Fundación Eclipse [8].

2.2 Conectividad por red celular

En la actualidad las redes celulares poseen la capacidad de ofrecer diferentes grados de calidad de servicio dependiendo del nivel de conectividad en servicio de la zona. Las redes celulares y tecnologías disponibles actualmente son: GPRS (80 kbps), EDGE (236Kbps), 3G (2Mbps), HDSPA (14Mbps) y 4G LTE (24Mbps). El módulo GSM de los celulares tiene la capacidad de hacer soft-handover (traspaso de celdas en caliente). Este algoritmo de traspaso se basa en medir la intensidad de señal en la estación base actual y en las cercanas, al pasar un umbral de baja señal, la estación base pasa la comunicación a la nueva y le informa al módulo GSM a través del canal de señalización de la nueva conexión, de esta forma es totalmente transparente para la aplicación que esté utilizando ese canal de datos [9].

2.3 Plataforma de hardware: Raspberry Pi

La Fundación Raspberry es una organización de caridad. Su objetivo es promover el estudio de la ciencia computacional especialmente en las escuelas. El objetivo del Pi fue de crear una pequeña computadora que fuera accesible para cada usuario final. Importante esfuerzo e investigación se puso en el Raspberry Pi, manteniendo los costos a un mínimo removiendo componentes que no eran críticos. El microprocesador para el Pi 2 es el Broadcom BCM2836. Es un system-on-a-chip, tiene CPU y gráficos con 1080p nativo dentro del chip [10]. Uno de los sistemas operativos más empleados en esta computadora es el Raspbian que está basado en Debian y es una distribución de Linux.

2.4 Unidad de Control Electrónico (ECU) a través de OBD2

La Unidad de Control Electrónico o ECU (por sus siglas en inglés: Electonic Control Unit), son unidades de procesamiento que se encuentran presentes en equipos de inyección que gestionan el funcionamiento del motor. Son, por lo general, parte de una red de unidades electrónicas presentes en un vehículo, entre las cuales pueden o no compartir información a partir, de una o varias redes de comunicación. Controlan, en base a lectura de sensores, el funcionamiento del motor variando la apertura de

válvulas, accionamiento de bobinas o apertura de inyectores. Esta gestión electrónica trata mantener la estequiometria de un motor y de esa forma se obtienen emisiones de gases reducidos, potencias elevadas incluso con motores de pequeña cilindrada y considerables aumentos en autonomía.

Con el constante avance la tecnología, los desafíos para diagnóstico de fallas fue creciendo a la par con el avance de las ECU; por lo que las herramientas para diagnostico fueron siendo cada vez más complejas. A modo de estandarizar el acceso a esa información, la industria automotriz acordó desarrollar el protocolo OBD (On Board Diagnostic, por sus siglas en ingles), el cual permite, a través de una interfaz de hardware estándar, verificar el estado de los sensores involucrados en controlar emisiones, así como también emitir alertas de fallas [11].

2.5 Procesamiento de Lenguaje Natural y Sistema Experto

El procesamiento de lenguaje natural (NLP – Natural Language Processing) es un área de la inteligencia artificial que se especializa en la generación automática e interpretación del lenguaje humano [12]. Para realizar este procesamiento se debe contar con herramientas que permitan hacer análisis morfológico, lematización, etiquetado Part-of-Speech (PoS) y desambiguación en texto. El proyecto open source FreeLing [12] ofrece estas capacidades con muy aceptable precisión para el idioma español.

Los Sistemas Expertos habitualmente son empleados como herramientas para la toma de decisiones y también son considerados parte de la inteligencia artificial [14]. Estos sistemas imitan el proceso de razonamiento que los expertos utilizan para resolver problemas específicos. Una implementación muy difundida entre la comunidad open source es JBoss Drools que emplea y extiende el algoritmo Rete [15].

2.6 Medio de comunicación Twitter

Una de las opciones para acceder en a la API de la red social Twitter y a través de Java es con la librería[1] open source Twitter4J [16]. Este software actúa de interfaz simplificada a los servicios de Twitter desde un programa Java. Permite realizar siguientes funciones: autenticación, envío y recepción de tweets, gestión de usuarios, listas, mensajes directos, suscripciones y otros. Ofrece acceso transparente a tres APIs de Twitter: Streaming API, REST API y Search API.

3 Prototipo

A partir de la información disponible en la ECU (Electronic Control Unit) de los automóviles y a través del estándar OBD2 (On Board Diagnostics 2), se integra un sistema de reconocimiento de lenguaje humano que pueda interpretar las solicitudes y

[1] Es una mala traducción de la palabra ‘libraries’ la correcta sería ‘bibliotecas’

responder de forma amigable y fácil de comprender por cualquier persona. El sistema se ejecuta en una mini computadora Raspberry Pi (RPI) ubicada dentro del automóvil y se conecta a Internet a través de la red celular (3G). El canal de comunicación es la red social Twitter. En la Fig. 1 se muestra un diagrama general de los componentes principales del prototipo, donde se representa en contexto el despliegue de la solución.

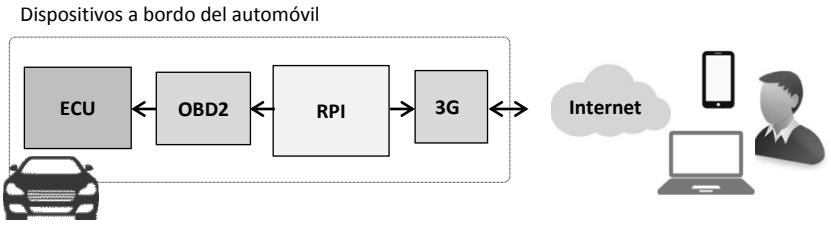

Fig. 1 - Diagrama general

La computadora RPI es la que recibe a través de Twitter las consultas y las traduce en comandos dirigidos a la ECU del automóvil. Prácticamente todo el software se ejecuta en la RPI, no obstante, el procesamiento de lenguaje natural es delegado a un Web Service publicado en la nube. Debido a que FreeLing es software que requiere de prestaciones que superan las capacidades de la RPI, se decidió desarrollar un Web Service (WS) que encapsule la funcionalidad de procesamiento de lenguaje natural (ver Fig. 2).

El software es un conjunto de 5 módulos OSGi donde un módulo Principal atiende y coordina solicitudes recibidas vía Twitter.

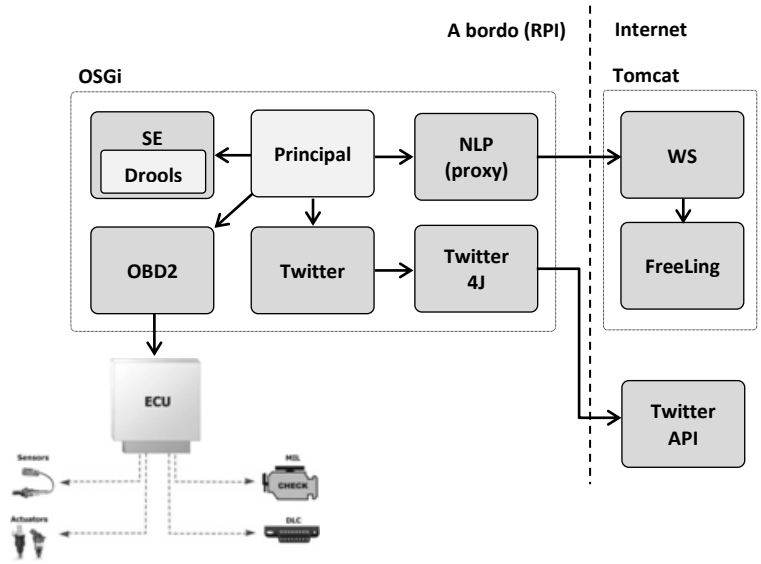

Fig. 2 - Componentes de software del prototipo

Principal: Coordina la ejecución de cada consulta.
Twitter: Accede a través del módulo Twitter4J a la API de Twitter.

Twitter4J: Consulta si existen nuevos mensajes donde se mencione la dirección asignada al automóvil (@irc_up_2015_iot[2]).
NLP: Envía a FreeLing todo el texto del mensaje y recibe los tokens, el etiquetado de cada token (PoS) y la lemmatización de cada uno.
SE: Reglas de Drools Expert analizan el resultado de NLP con el fin de determinar qué dato deberá ser leído a través de OBD2.
OBD2: Consulta a la ECU y devuelve el resultado.

3.1 Secuencia de ejecución

1 Una vez detectado un nuevo mensaje, el módulo principal obtiene el texto completo desde el módulo Twitter.
2 La cadena se envía al módulo NLP que este a su vez la reenvía al WS de FreeLing.
3 Con el resultado de FreeLing, el módulo principal lo envía al módulo SE donde en función de la estructura sintáctica y la presencia de palabras clave se construye dinámicamente la consulta al módulo OBD2.
4 El módulo OBD2 accede a través de la librería Java Simple Serial Connector [17] al puerto USB correspondiente al Scanner OBD2 y transmite la consulta.
5 La respuesta obtenida de la ECU es traducida a lenguaje no técnico por el módulo SE.
6 El módulo Principal envía la cadena resultante al módulo Twitter que este a su vez publica la respuesta anteponiendo la dirección del usuario (@usuario) que realizó la consulta.

Cabe destacar que todo el software de terceros es open source.

3.2 Entornos de desarrollo y pruebas

Una vez que se contó con una versión ejecutable de los módulos, se montó un entorno de pruebas (ver Fig. 3).

[2] https://twitter.com/irc_up_2015_iot/with_replies

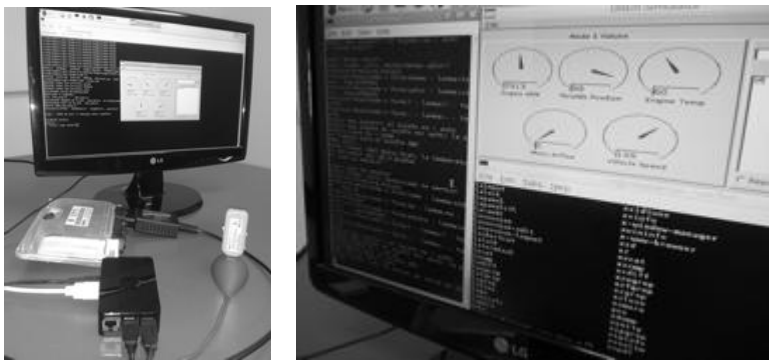

Fig. 3 - Entorno de pruebas

Desde allí se probó el software en la computadora RPI y ejecutando el simulador OBDsim [18]. Luego, como se observa en la Fig. 4, se probó conectado los componentes (Scanner OBD2, RPI y Módem 3G) a dos automóviles.

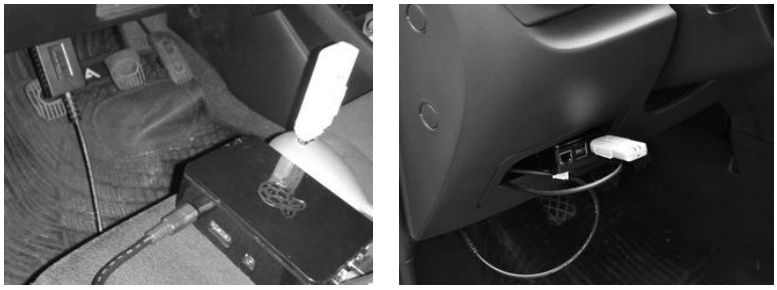

Fig. 4 – Pruebas en Nissan Tiida y VW Gol Trend

7 Casos de uso

La versión inicial del prototipo permite realizar un set limitado de consultas (ver Tabla 1), no obstante es suficiente para validar el concepto.

Tabla 1 - Consultas implementadas

Dato	Consulta
Si hay algún código de error activo (DTC[3])	"Cómo te sentís?", "Cómo estás de salud?"
Motor encendido (RPM)	"Cómo tenés el motor?", "Está encendido el motor?"
Temperatura motor (Coolant temperature)	"Tenés calor?"
Posición acelerador (Throttle position)	"Cómo te tratan?", "Cómo la pasás?"
Posición geográfica (GPS)	"Por dónde andás?", "Cuál es tu ubicación?"

[3] Diagnostic Trouble Codes

Para el caso de la consulta acerca de la ubicación actual, el software solicita a Google Maps una imagen con la posición geográfica y esta es incrustada en el mensaje de respuesta (ver Fig. 5).

Fig. 5 – Respuesta a ubicación con mapa incrustado

8 Transferencia a la industria

Conseguir el apoyo de la industria es el siguiente objetivo del proyecto, por eso se ha presentado en una charla organizada por la CESSI (Cámara de Empresas de Software y Servicios)[4] donde se estableció contacto con una empresa vinculada a la industria automotriz. Actualmente se está en conversaciones para firmar un convenio de colaboración. Asimismo también se postuló el proyecto para el concurso Innovar 2015 organizado por el Ministerio de Ciencia, Tecnología e Innovación Productiva[5]. Se está aguardando novedades respecto al proceso de evaluación.

9 Conclusiones y trabajos futuros

Este trabajo tiene como objetivo crear un prototipo que permita definir bases para avanzar en la creación de una plataforma de integración de dispositivos en Internet. Asimismo el proyecto también propone una interfaz de usuario de lenguaje natural en lugar de botones o comandos. Para ello se seleccionó un conjunto de tecnologías de

[4] http://www.cessi.org.ar/ver-noticias-charla-gratuita-sobre-internet-de-las-cosas-y-oportunidades-de-negocios-1852

[5] ID 18007 Sistema de interacción con la ECU de un automóvil empleando lenguaje natural

soporte a la solución que son OSGi, FreeLing (NLP), Drools (Sistema Experto) y OBD2 como interfaz al dispositivo objetivo (ECU del automóvil). También se emplearon la red celular y Twitter como medios de transmisión y comunicación.

Los resultados del proyecto permiten confirmar la viabilidad de un sistema experto y software NLP para convertir el lenguaje natural en comandos interpretables por un dispositivo y vice versa. Asimismo también se puso a prueba la compatibilidad de OSGi en un entorno de prestaciones reducidas, como es la computadora Raspberry Pi, y el desafío que implica consumir web services (FreeLing y Twitter) a través de una red celular de datos en movimiento.

Como siguiente fase del proyecto se planea ampliar y mejorar la capacidad de interpretar consultas mediante el análisis de la distancia semántica entre palabras y la comparación de la estructura sintáctica con patrones locales.

10 Referencias

1. Kranz, M., Roalter, L., Michahelles, F.: Things That Twitter: Social Networks and the Internet of Things. ETH Zurich (2010)
2. Amir, M., Hu, F., Pillai, P., Cheng, Y., Bibiks, K., Interaction Models for Profiling Assets in an Extensible and Semantic WoT Framework. 10th International Symposium on Wireless Communication Systems. VDE Verlag (2013)
3. Alam, K., Saini, M., El Saddik, A.: Towards Social Internet of Vehicles: Concept, Architecture, and Applications. IEEE (2015)
4. Mendes, P., Social-driven Internet of Connected Objects. Interconnecting Smart Objects with the Internet Workshop (2011)
5. Atzori, L., Iera, A., Morabito, G., Nitti, M., The Social Internet of Things (SIoT), when social networks meet Internet of Things: Concept, architecture and network characterization. Elsevier (2012)
6. Oka, D., Furue, T., Langenhop, L., Nishimura, T., Survey of Vehicle IoT Bluetooth Devices. 7th International Conference on Service-Oriented Computing and Applications. IEEE (2014)
7. OSGi Alliance, http://www.osgi.org/
8. Eclipse IoT, http://iot.eclipse.org/
9. Brunner, C., Caravaglia, A., Mittal, M., Narang, M., Bautista, J.: Inter-System Handover Parameter Optimization. Vehicular Technology Conference. IEEE (2006)
10. RasPi Magazine, Issue 11 (2015)
11. Diseño y Prueba de ECU usando Productos de National Instruments, http://www.ni.com/white-paper/3312/es/
12. Hirschberg, J., Ballard, B., Hindle, D., Natural Languaje Processing. AT&T Technical Journal (1988)
13. Padró, L., Collado, M., Reese, S., Lloberes, M., Castellón, I., FreeLing 2.1: Five years of open-source language processing tools. 7th Language Resources and Evaluation Conference (2010)
14. Rossini, P., Using Expert Systems and Artificial Intelligence for Real Estate Forecasting. 6th Annual Pacific-Rim Real Estate Socienty Conference (2000)
15. JBoss Drools, http://www.drools.org/
16. Twitter4J, http://twitter4j.org/
17. Java Simple Serial Connector, https://code.google.com/p/java-simple-serial-connector/
18. OBDSim, http://icculus.org/obdgpslogger/obdsim.html