I+D Revista de Investigaciones
ISSN 22561676
Volumen X Número X Año XX mes-mes 2014 p.p Artículo de Investigación

Uso de SDL y MSC para modelado de sistemas IoT[1]

IoT system modeling using SDL and MSC.

A.S. Rudqvist Valencia[2], E. Tamura[3]

CEA-IoT Nodo Occidente

Artículo recibido en mes agosto de 2016; artículo aceptado en mes XX de año

Citación del artículo: Rudqvist, A.S. & Tamura, E. (año). Uso de SDL y MSC para modelado de sistemas IoT. *I+D Revista de Investigaciones*, 1(2), pp-pp.

Resumen

La visión de un mundo interconectado mediante sistemas basados en Internet de las Cosas (*IoT*, por la sigla en inglés de *Internet of Things*) se espera genere mejoras en productividad y en la calidad de vida para toda la humanidad. Para llegar a este futuro deseado, sin embargo, persisten retos técnicos importantes por resolver. Uno de los retos claves es la interconexión masiva de los dispositivos. Existe un número de diferentes plataformas abiertas como Arduino, Raspberry Pi, entre otras, que han mostrado que el desarrollo de dispositivos inteligentes puede ser una tarea al alcance incluso de niños; sin embargo, la interconexión masiva de estos dispositivos es un reto complejo de ingeniería. Algunos autores han

---

[1] Artículo de original, resultado de un proyecto de investigación en curso.

[2] Ingeniero Electrónico e Ingeniero de Sistemas, Pontificia Universidad Javeriana Cali. Magister en Finanzas., Universidad Icesi. Investigador Grupo de Automática y Robótica. Pontificia Universidad Javeriana - Cali: Dirección Calle 18 No 118-250. Cali, Colombia. PBX: (+57 2) 3218200. Correo electrónico institucional: asrudqvist@javerianacali.edu.co

[3] Ingeniero en Electrónica, Universidad del Cauca. Doctor en Arquitectura y Tecnología de los Sistemas Informáticos, Universidad Politécnica de Valencia. Investigador Grupo de Automática y Robótica. Pontificia Universidad Javeriana - Cali: Dirección: Calle 18 No 118-250. Cali, Colombia. PBX: (+57 2)3218200. Correo electrónico institucional: tek@javerianacali.edu.co

propuesto el uso de lenguajes como el de Diagrama de Secuencias de Mensajes (*MSC*[4] por la sigla en inglés de *Message Sequence Charts*) y el Lenguaje de Especificación y Descripción (*SDL*[5] por la sigla en inglés de *Specification and Description Language*) para enfrentar esta tarea. En el presente artículo se presenta una descripción de estas notaciones para especificar sistemas IoT. Se incluye como caso de uso un sistema urbano de bicicletas compartidas.

Palabras clave: IoT, SDL, MSC, modelado, transporte multimodal

Resumen en inglés.

An interconnected world vision using Internet of Things (*IoT*) systems has generated expectations about improvements in the productivity and life quality for all the mankind. However, in order to achieve this foreseen future, there are some key technical challenges that need to be addressed. Open platforms such as Arduino, and Raspberry Pi, among others, have shown that smart device development could be done, even by a child. However, the massive interconnection of these devices poses a complex engineering challenge. Message Sequence Charts (*MSC*[4]) and Specification and Description Language (*SDL*[5]) have been proposed as appropriate languages for this task by some authors. This article illustrates a description of those languages for an IoT system specification. A shared-bicycle system is included as a study case.

Introducción

IoT es definida por la Unión Internacional de Telecomunicaciones (ITU por la sigla en inglés de *International Telecommunication Union*) como "la infraestructura global para la sociedad de la información, habilitando el avance de los servicios mediante la interconexión (física o virtual) de cosas basadas en las tecnologías de información interoperables existentes

[4] International Telecommunication Union. Recommendation ITU-T Z.120: Message Sequence Chart (MSC). https://www.itu.int/rec/T-REC-Z.120

[5] International Telecommunication Union. Recommendation ITU-T Z.100: Specification and Description Language. https://www.itu.int/rec/T-REC-Z.100

y futuras"[6]. Esta definición, que se enfoca en una de las visiones de IoT, la visión orientada a Internet de acuerdo a la clasificación de Singh (2014), es solo una de las muchas que se han dado por diferentes organizaciones y autores. En cualquier caso, un tema común en la literatura es el reconocimiento de los retos que subyacen hacia futuro en las soluciones de IoT. Autores como Stankovic (2014) y Borgia (2014) hacen referencia de su perspectiva frente a dichos retos en sus escritos.

En estas listas de retos de los sistemas IoT se encuentran el soporte al escalado masivo, la gestión de la arquitectura (de sistema) y dependencias, robustez, seguridad y privacidad de la solución planteada. Apuntando a la solución de estos retos, se puede pensar en emplear algunas de las técnicas que se han aplicado exitosamente en la solución del problema técnico más cercano a IoT, Internet. En esta dirección, SDL ha mostrado su potencial para contribuir en la solución de los retos anteriormente nombrados Sherratt (2015).

El uso de SDL como lenguaje para especificar y diseñar sistemas IoT ofrece ventajas tales como: legibilidad, al ser un lenguaje grafico es fácil discutir el comportamiento de la solución especificada, aún con personas que no tengan experiencia usando lenguajes de modelado. Es un lenguaje ejecutable, por lo que los modelos desarrollados pueden ser simulados para evaluar su comportamiento frente a la especificación del sistema. Debido a que el enfoque del lenguaje se centra en las interacciones, y estas a su vez son la raíz de uno de los principales retos técnicos en los sistemas IoT, el uso de este lenguaje se considera una decisión coherente en busca de enfrentar los retos que se tienen al diseñar soluciones IoT.

Como caso de estudio de la metodología propuesta se modela un sistema inteligente de bicicletas públicas. Un sistema de este tipo tiene el potencial de mejorar la calidad de vida de las personas en las ciudades donde es implementado, mejorando la movilidad y haciéndola más amigable con el medio ambiente. Los sistemas de bicicletas públicos han mostrado ser

[6] International Telecommunication Union. Recommendation ITU-T Y.2060: Overview of the Internet of Things. http://www.itu.int/ITU-T/recommendations/rec.aspx?rec=y.2060

una alternativa efectiva; entre los sistemas más populares de este estilo, se encuentran sistemas en Barcelona, Gotemburgo, Lyon, Montreal, Paris y Washington D.C, como se detalla en Midgley (2009).

Se usa PragmaDev Studio 5.0[7], como herramienta de diseño asistido por computador para la evaluación de la metodología de estudio propuesta y para el desarrollo del caso de estudio. Esta es una herramienta líder en la especificación y diseño de sistemas embebidos.

Metodología propuesta

La metodología propuesta en el presente artículo podría ser utilizada para seguir un flujo estructurado de gestión de proyectos, tipo modelo en V[8] o cascada. Sin embargo, dada la flexibilidad de la herramienta, otra opción es utilizar metodologías de gestión de proyectos ágiles basadas en pruebas (*Test-Driven Development - TDD*). En este sentido la posibilidad de desarrollar casos de uso utilizando MSC y luego poder validarlos utilizando el modelo especificado en SDL, permite la aplicación directa de metodologías ágiles. En esta línea están el trabajo realizado por Zhang,Y (2004) y por Boris (2014); trabajos en los cuales se busca tener metodologías ágiles basadas en modelos para tener lo mejor de ambos mundos.

Definición de sistema y escenarios

El primer paso, sin importar cuál de las alternativas de gestión de proyectos se seleccione (tradicional o ágil), es definir el dominio en el cual se trabajará, definir el problema, los actores y los requerimientos iniciales.

Para el caso de estudio sobre un sistema urbano de bicicletas compartidas, el problema se definió como se ilustra en la siguiente sección.

Definición de sistema y escenarios – Caso de estudio

[7] PragmaDev modeling and testing tools. http://www.pragmadev.com/
[8] Ryen(2008) y el Departamento de transporte de Estados Unidos (2007).

Se propone la implementación de un sistema de bicicletas público de cuarta generación, de acuerdo con la clasificación de DeMaio (2009). El sistema lo componen: bicicletas, parqueaderos ("puntos de estacionamiento"), un sistema centralizado de información, taller de reparación y camiones de transporte, como se muestra en la Figura 1. El sistema de información centralizado contará con un sistema de predicción de demanda y un algoritmo de redistribución de bicicletas. A partir de la información en el sistema central se tendrá una App donde los usuarios podrán consultar la disponibilidad de las bicicletas en los parqueaderos.

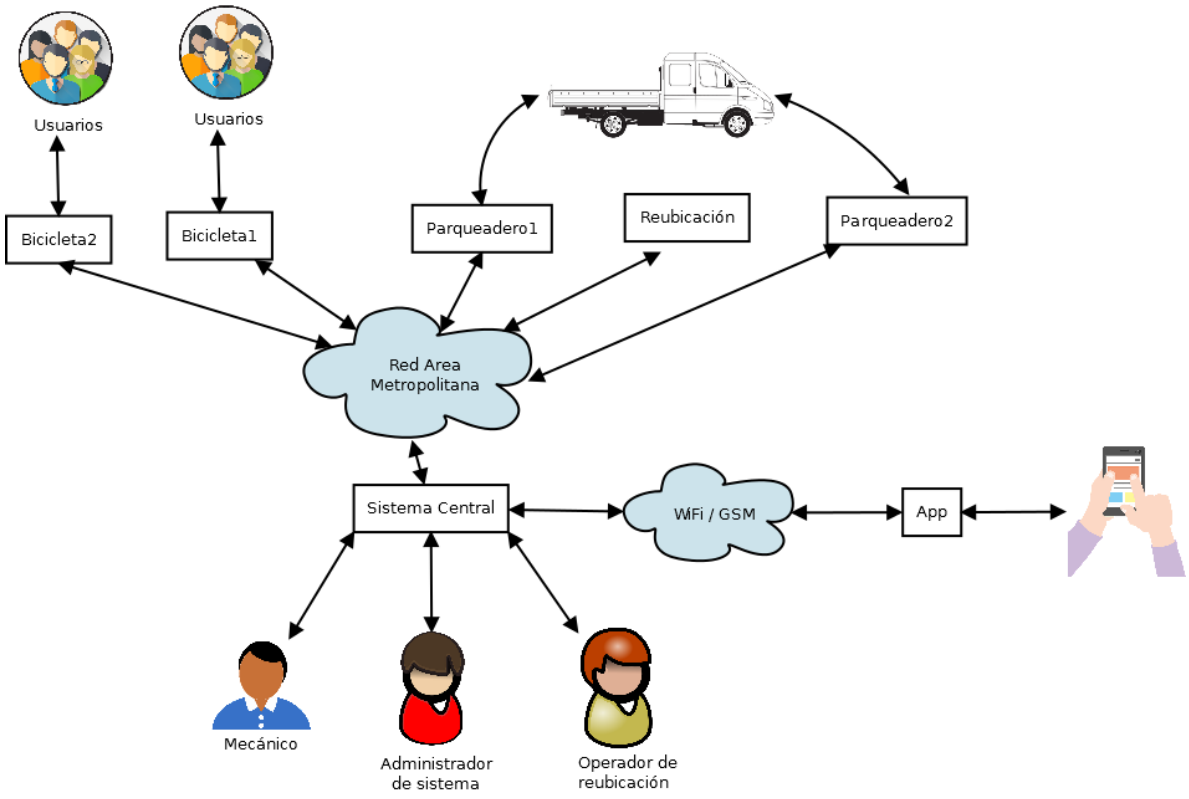

Figura 1. Estructura del sistema propuesto como estudio de caso – Sistema de transporte urbano de bicicletas compartidas.

Para completar la descripción inicial se definen los casos de uso de manera textual:

- **CU00** Se inicializa un parqueadero.

- **CU00A** Se inicializa una bicicleta.
- **CU01** Un usuario con tarjeta retira una bicicleta de un parqueadero.
- **CU02** Un usuario parquea su bicicleta en un parqueadero.
- **CU03** Un camión desplaza bicicletas de un parqueadero a otro, para balancear el sistema.
- **CU04** Una bicicleta es reportada como dañada y se lleva al taller de mantenimiento Usa CU01.
- **CU05** Una bicicleta es revisada en el taller y se pone a disposición del sistema. Usa CU02.
- **CU06** Una bicicleta deja de enviar señal (posible incidente: A: Accidente, B: Violación de barrera geográfica, C: Vandalismo)
- **CU07** Una bicicleta que dejó de reportar posición es encontrada.

Definición de trazas de ejecución

La definición de trazas de ejecución permite realizar una especificación semi-formal del sistema, de cómo los agentes que lo conforman deben comportarse ante una serie de estímulos. Para esta tarea se emplea MSC como lenguaje hermano de SDL (ETSI 2016). MSC es estandarizado por la ITU-T en la recomendación Z.120. Desarrollar las trazas de ejecución utilizando MSC para un sistema, que luego se modelará utilizando SDL, permite la posterior validación de dichas trazas contra el modelo. Los diagramas en MSC son similares, conceptualmente, a los diagramas de secuencia del Lenguaje Unificado de Modelado (UML[9] por la sigla en inglés de *Unified Modeling Language*).

Los diagramas en MSC permiten la representación gráfica de las trazas, sirviendo como un punto de partida para el diseño del sistema y para propósitos de validación y

[9] Object Management Group, OMG. UML. http://www.uml.org/index.htm

verificación. Una vez se tiene especificado el comportamiento del sistema, este puede ser simulado y validado contra la traza especificada originalmente. Las trazas de ejecución permiten describir casos normales, casos con alta concurrencia y casos excepcionales (con fallas en alguna parte del sistema).

Definición de trazas de ejecución – Caso de estudio

Para ilustrar el uso de MSC en la especificación de trazas de ejecución, se incluye la ejecución normal del caso de uso CU01. La Figura 2 representa el escenario en el que un usuario con su tarjeta, retira una bicicleta de un espacio del parqueadero. El proceso inicia cuando el usuario pasa su tarjeta por un controlador con una bicicleta disponible; el sistema valida que el usuario esté registrado y habilitado en el sistema centralizado. A continuación, la bicicleta es asociada al usuario y liberada por el control del parqueadero. Finalmente, el usuario valida el estado de la bicicleta y si esta está en condiciones aceptables, la retira para iniciar su recorrido. La bicicleta pasa a un estado en el que reporta su ubicación al sistema centralizado.

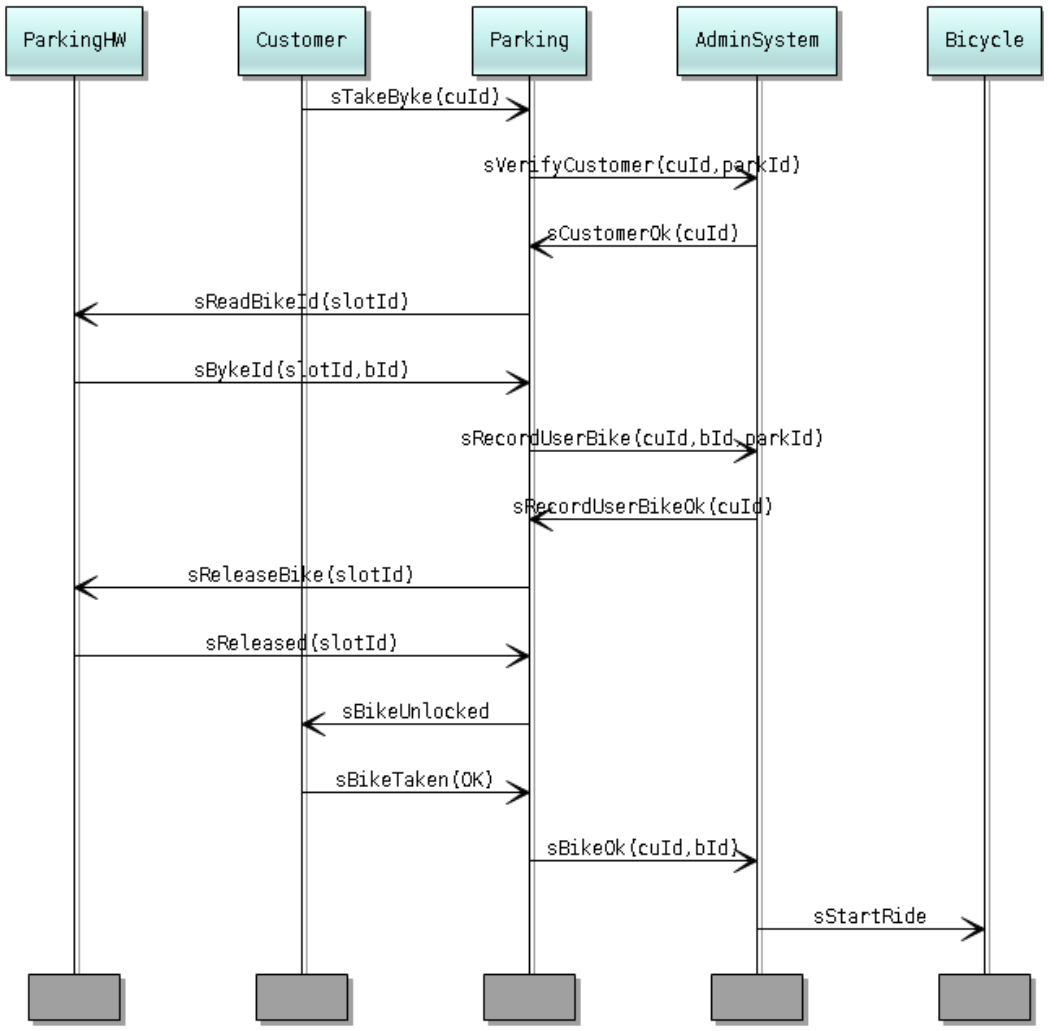

Figura 2. Traza de ejecución, caso de uso CU01 - Sistema de transporte urbano de bicicletas compartidas.

Definición arquitectura de sistema

Para describir el diseño de los agentes que conforman el sistema se usó SDL. Este lenguaje es definido por ITU-T en la Recomendación Z.100. SDL se desarrolló para la "especificación de sistemas complejos, orientados a eventos, de tiempo real e interactivos"

(según definición de IEC[10]). Esta definición refuerza la hipótesis de que SDL es un lenguaje adecuado para representar sistemas IoT.

SDL permite modelar la estructura, la comunicación, la representación de datos y el comportamiento de los sistemas especificados (además, soporta herencia). Estructuralmente el sistema se divide en bloques, y estos a su vez en sub-bloques o en procesos. La comunicación se especifica definiendo canales que pueden transportar determinados mensajes (los mismos definidos en los diagramas MSC). Asociado a la parte estructural también está la posibilidad de definir las estructuras de los datos usadas en el sistema.

En la vista de arquitectura en SDL se define: estructura, comunicación y datos. Todo ello se deriva a partir de lo especificado en la definición de sistema y las trazas de ejecución en MSC.

Definición arquitectura de sistema – Caso de estudio

La arquitectura del sistema se especifica como un diagrama de componentes en SDL. Como se puede apreciar en la Figura 3, se definen los agentes principales del sistema, la relación entre ellos y con el ambiente.

[10] IEC, International Engineering Consortium. Specification and Description Language (SDL) - http://www.sdl-forum.org/SDL/Overview_of_SDL.pdf

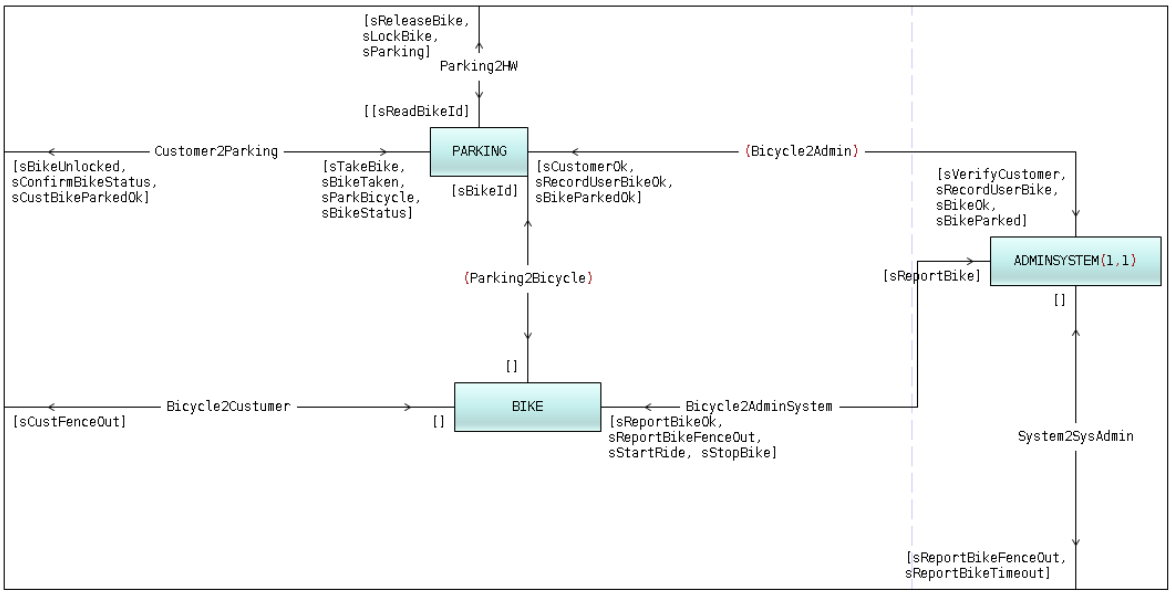

Figura 3. Arquitectura del sistema en SDL - Sistema de transporte urbano de bicicletas compartidas.

El ambiente está compuesto por los usuarios del sistema y las interfaces hardware. Los agentes incluidos en el sistema, son:

- ADMINSYSTEM: Representa el sistema central, encargado de coordinar todo el sistema. Tiene canales de comunicación con los bloques: PARKING, BIKE y comunicación con el ambiente para comunicarse con el administrador del sistema.
- BIKE: Representa el agente embebido en cada una de las bicicletas. Tiene canales de comunicación con los bloques: PARKING Y ADMINSYSTEM; además, con el ambiente para comunicarse con el usuario (indicador de fuera de zona permitida).
- PARKING: Representa el agente embebido en cada uno de los parqueaderos. Tiene comunicación con los bloques BIKE y ADMINSYSTEM; además, con el ambiente para comunicarse con el hardware de los espacios de aparcamiento y con el usuario del sistema.

Definición de comportamiento del sistema

En SDL el comportamiento asignado a los procesos al interior de los bloques se expresa en forma de máquinas extendidas de estados finitos. El comportamiento de los agentes planteados es reactivo: esperan en un estado una señal que los despierte; una vez recibida la señal, esta es eventualmente procesada emitiendo señales a otros agentes y se regresa a un estado de nuevo. Este esquema de operación permite que en el nivel de la implementación, cuando el sistema está en un estado, el hilo de ejecución del agente se suspenda y se conceda el uso del procesador a otro hilo; de hecho, si todos los agentes están un estado, es decir, no hay hilo alguno por ejecutar, lo que se puede hacer es poner el procesador en un estado de bajo consumo, lo cual es supremamente beneficioso si consideramos que en sistemas IoT el consumo energético es un factor muy importante, dado que la operación de las plataformas de adquisición, procesamiento y actuación podrían estar basadas en sistemas de alimentación que no están interconectados a la red eléctrica.

Definición comportamiento del sistema – Caso de estudio

Como ejemplo de la definición del comportamiento de los agentes especificados en el caso de estudio se muestra la máquina extendida de estados finitos del agente embebido en la bicicleta (ver Figura 4). Esta máquina lleva a cabo el proceso de inicialización de la bicicleta, proceso en el cual solicita un identificador al sistema central y se duerme. Cuando llega el mensaje con el identificador lo almacena y se queda esperando a que la bicicleta sea tomada por un usuario. Después de que un usuario solicita exitosamente dicha bicicleta y la retira, ésta activa el rastreo hasta que es parqueada de nuevo. Durante la conducción, se envía la posición y se emplea un temporizador para enviar la ubicación de manera periódica. Si durante la conducción la bicicleta sale de un área geográfica pre-determinada (*geofence*[11]) se informa al usuario usando un indicador de alerta auditiva.

[11] *Geofence* o geoperimetraje se refiere al uso de barreras geográficas virtuales.

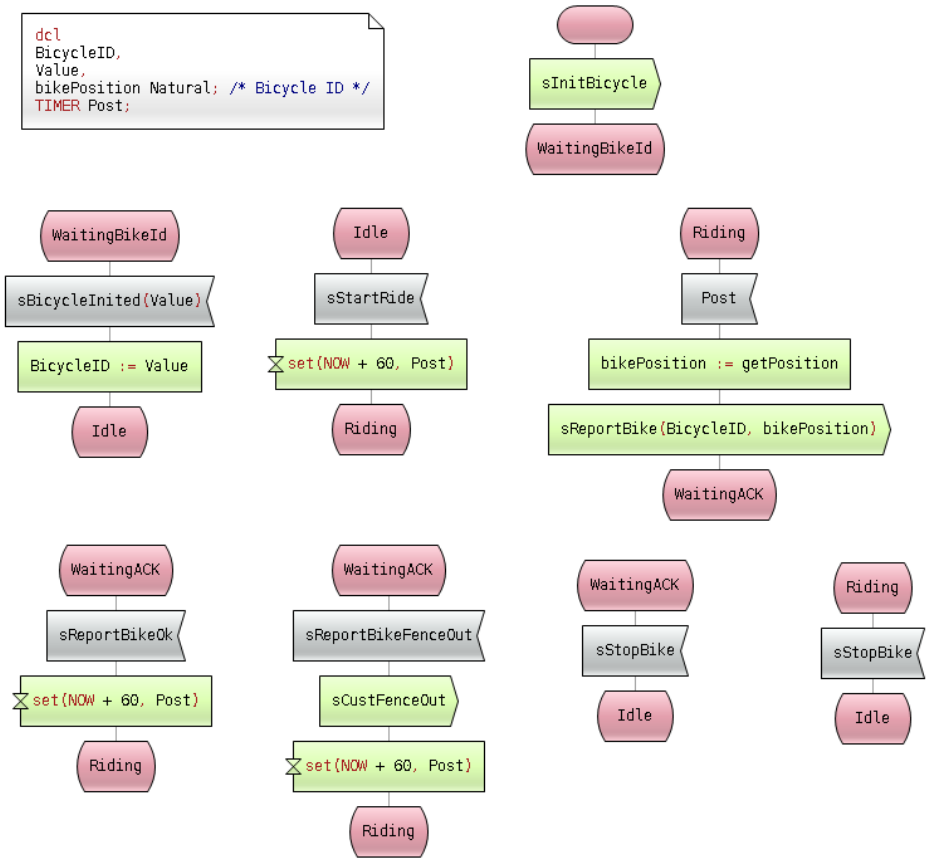

Figura 4. Comportamiento del agente bicicleta representado como una máquina de estado extendida, utilizando la notación SDL - Caso Sistema de transporte urbano de bicicletas compartidas.

Resultados

Se presenta una metodología para el desarrollo de sistemas complejos IoT basada en herramientas heredadas del desarrollo de estándares de telecomunicaciones. La aplicación de dicha metodología se ilustra con el caso de uso de un sistema de bicicletas compartido; el cual permite realizar una validación de la misma. El uso de una herramienta de diseño asistido por computador para este caso se muestra como una ayuda invaluable debido a las validaciones que permite realizar sobre el modelo desarrollado.

Los retos técnicos inicialmente mencionados para los sistemas IoT pueden ser tratados al utilizar notaciones que hacen explícito el proceso de comunicación entre agentes. En este sentido los diagramas MSC combinados con la especificación ejecutable del sistema en SDL, proporcionan una herramienta invaluable para enfrentar dichos retos.

Un aspecto adicional en términos de complejidad que se encontró en el diseño de sistemas IoT, es el hecho que estos típicamente están compuestos por varios tipos de agentes, cada uno de los cuales puede contar con múltiples instancias, con las consecuentes implicaciones en términos de concurrencia. A esto se suma el factor que usualmente en la fase de diseño se desea minimizar las restricciones impuestas sobre el número posible de dispositivos, de uno u otro tipo. Se pudo validar que el hecho de tener la posibilidad de especificar el sistema a partir de trazas de ejecución, permitió hacer visibles los puntos en los cuales se podrían tener errores de este estilo y corregirlos de manera directa, antes de que el sistema se lleve a un despliegue masivo.

Una de las dificultades principales que se enfrentó en el diseño del sistema del caso de estudio, fue el tema de la identificación específica de las diferentes instancias de los agentes, debido a que el número de parqueaderos y bicicletas desplegados en el sistema no es conocido a priori. Adicionalmente, se espera que todos ejecuten una copia idéntica del software desarrollado. Eventualmente esto podría causar que no se pudieran distinguir los agentes. Para solucionar esto, se incluyeron trazas explícitas de inicialización de los agentes. En el modelo del sistema se puso como requerimiento el acceso a una dirección física (similar, por ejemplo, a una dirección MAC) y a nivel de sistema incluyendo un sistema que pudiera asignar nombre y resolverlos (similar a DHCP y DNS).

Un tercer punto de interés es la flexibilidad que se tiene con la metodología propuesta para ser adaptada tanto a la gestión tradicional como a la ágil de proyectos. Se puede definir todo detalladamente al inicio del proyecto o se puede realizar una definición iterativa. Un aspecto que soporta el modelo ágil es la posibilidad de realizar generación de código a partir

de las máquinas extendidas de estados finitos, con lo cual todo el ciclo de vida del proyecto, aunque semi-formal, podría enmarcarse en un proceso no tradicional.

Comentarios

Se ha realizado un avance en la cobertura del proceso de modelado de sistemas IoT usando MSC y SDL. En el plan del proyecto en ejecución está la realización de pruebas automatizadas usando la Notación de Pruebas y Control de Pruebas Versión 3 (TTCN-3[12], por la sigla en inglés de *Testing and Test Control Notation version 3*) para luego pasar a una fase de construcción de un prototipo del sistema. Esto permitirá afinar la metodología propuesta en el presente artículo y continuar con su enriquecimiento de cara a futuros proyectos. Más aún, si se desea realizar verificaciones formales, se puede hacer uso de IFx[13] o Diagramas de Secuencia de Propiedades (PSC por la sigla en inglés de *Property Sequence Chart*)[14].

Parte de la utilidad de SDL como lenguaje radica en que es completamente independiente de la tecnología final donde se despliegue la solución. Por ello, constituye una solución ideal para el diseño y especificación de agentes a desplegar en diferentes tipos de plataformas; de hecho, la implementación se podría realizar en un sistema hardware haciendo uso de un lenguaje de descripción de hardware como VHDL[15] o Verilog[16]. Si a esto se le suma la posibilidad de generar código de manera automática (a partir de reglas de traducción)

[12] European Telecommunications Standards Institute, ETSI. Testing and Test Control Notation Version 3 (TTCN-3). http://www.ttcn-3.org/
[13] Institut de Recherche en Informatique de Toulouse – Verimag. IFx. https://www.irit.fr/ifx/
[14] Dipartimento di Ingegneria e Scienze dell'Informazione e Matematica, Università dell'Aquila. Property Sequence Chart. http://www.di.univaq.it/psc2ba/index.html
[15] Institute of Electrical and Electronics Engineers. IEEE Std 1076 - IEEE Standard VHDL Language Reference Manual
[16] Institute of Electrical and Electronics Engineers. IEEE Std 1364 - IEEE Standard Verilog Language Reference Manual

se tiene una metodología que permite el desarrollo de soluciones de IoT robustas utilizando plataformas heterogéneas.

Finalmente, vale la pena comentar que el caso de estudio presentado representa un problema de interés para las ciudades modernas en que tanto la congestión como la búsqueda de alternativas más verdes de movilidad tienen espacio. Contar con un sistema de bicicletas compartidas que incorpore elementos como los modelados en el presente artículo, tienen el potencial de hacer el sistema más eficiente y amigable. Además, el tener bicicletas geo-referenciadas permite tener un mayor control sobre las bicicletas del sistema.

Agradecimientos

Los autores agradecen la cooperación de todos los miembros del *Centro de Excelencia y Apropiación en Internet de las Cosas (CEA-IoT)*. Agradecen también a las instituciones que apoyan este trabajo: El *Ministerio de Tecnologías de la Información y las Comunicaciones de Colombia – MinTIC* y el *Departamento Administrativo de Ciencia, Tecnología e Innovación – Colciencias* a través del *Fondo Nacional de Financiamiento para la Ciencia, la Tecnología y la Innovación Francisco José de Caldas* (Proyecto FP44842-502-2015).

Agradecimientos especiales al Ingeniero Fernando Barraza por su colaboración en la conceptualización del caso de estudio.

Agradecimientos también a PragmaDev SARL.

Referencias

- ITU. Internet of Things Global Standards Initiative. Recuperado de: http://www.itu.int/en/ITU-T/gsi/iot/Pages/default.aspx
- DeMaio, Paul. 2009. Bike-sharing: History, Impacts, Models of Provision, and Future. Journal of Public Transportation, 12 (4): 41-56. DOI: http://dx.doi.org/10.5038/2375-0901.12.4.3 Available at: http://scholarcommons.usf.edu/jpt/vol12/iss4/3

- Singh, D., Tripathi, G., & Jara, A. J. (2014, March). A survey of Internet-of-Things: Future vision, architecture, challenges and services. In Internet of things (WF-IoT), 2014 IEEE world forum on (pp. 287-292). IEEE.
- Stankovic, J. A. (2014). Research directions for the internet of things. IEEE Internet of Things Journal, 1(1), 3-9.
- Sherratt, E., Ober, I., Gaudin, E., i Casas, P. F., & Kristoffersen, F. (2015, October). SDL-The IoT Language. In International SDL Forum (pp. 27-41). Springer International Publishing.
- Borgia, E. (2014). The Internet of Things vision: Key features, applications and open issues. Computer Communications, 54, 1-31.
- Midgley, P. (2009). The role of smart bike-sharing systems in urban mobility. Journeys, 2, 23-31.
- Ryen, Ed. North Dakota Department of Transportation (2008). Overview of the system Engineering process. Recuperado del portal del departamento de transporte de North Dakota, USA. https://www.dot.nd.gov/divisions/maintenance/docs/OverviewOfSEA.pdf
- Department of Transportation, Office of Operations (2007). Systems Engineering for Intelligent Transportation Systems. An Introduction for Transportation Professionals. Recuperado del departamento de transporte de Estados Unidos. http://www.iteris.com/itsarch/documents/seguide/seguide.pdf
- Boris, T., Alexey, V., & Vsevolod, K. (2014). Keyword-Driven Testing with Message Sequence Charts. In Proceedings of the Spring/Summer Young Researchers' Colloquium on Software Engineering (No. 8). Disponible en: http://syrcose.ispras.ru/2014/files/submissions/17_syrcose2014.pdf
- Zhang, Y. (2004). Test-driven modeling for model-driven development. IEEE Software, 21(5), 80-86.

- ETSI, European Telecommunications Standards Institute. (31 de julio de 2014). Message Sequence Charts (MSC). Recuperado del sitio de internet ETSI: http://www.etsi.org/technologies-clusters/technologies/protocol-specification/msc
- IEC, International Engineering Consortium. Specification and Description Language (SDL). Recuperado del sitio de Internet SDL Forum: http://www.sdl-forum.org/SDL/Overview_of_SDL.pdf