EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO
INSTITUTO TECNOLÓGICO DE ORIZABA

"2021, Año de la Independencia"

DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

OPCION I. - TESIS

TRABAJO PROFESIONAL

"Mecanismo de Composición de Servicios bajo el enfoque del Internet de las Cosas (IoT)".

QUE PARA OBTENER EL GRADO DE:
DOCTOR EN CIENCIAS DE LA INGENIERÍA

PRESENTA:
M.C. Isaac Machorro Cano

DIRECTOR DE TESIS :
Dr. Giner Alor Hernández

CO-DIRECTOR DE TESIS :
Dr. José Luis Sánchez Cervantes

ORIZABA, VERACRUZ, MÉXICO. FEBRERO 2021

Orizaba, Veracruz, 18/02/2021
Dependencia: **División de Estudios de Posgrado e Investigación**
Asunto: **Autorización de Impresión**
OPCION: **I**

C. ISAAC MACHORRO CANO
Candidato a Grado de Doctor en:
CIENCIAS DE LA INGENIERÍA
P R E S E N T E.-

De acuerdo con el Reglamento de Titulación vigente de los Centros de Enseñanza Técnica Superior, dependiente de la Dirección General de Institutos Tecnológicos de la Secretaría de Educación Pública y habiendo cumplido con todas las indicaciones que la Comisión Revisora le hizo respecto a su Trabajo Profesional titulado:

"Mecanismo de Composición de Servicios bajo el enfoque del Internet de las Cosas (IoT)"

Comunico a Usted que este Departamento concede su autorización para que proceda a la impresión del mismo.

A T E N T A M E N T E
Excelencia en Educación Tecnológica®
CIENCIA – TÉCNICA - CULTURA®

EDUCACIÓN | TECNOLÓGICO NACIONAL DE MÉXICO
INSTITUTO TECNOLÓGICO DE ORIZABA
DIV. DE EST. DE POSGRADO E INVEST.

DR. MARIO LEONCIO ARRIOJA RODRÍGUEZ
JEFE DE LA DIVISIÓN DE ESTUDIOS
DE POSGRADO E INVESTIGACIÓN

Agradecimientos

A Dios, por el regalo de la vida, por siempre estar a mi lado, por guiarme por el buen camino y por permitirme alcanzar une meta más en la vida.

A mi esposa Mónica Guadalupe Segura Ozuna, por su amor incondicional y comprensión, por siempre apoyarme en todas mis iniciativas, por alentarme en los momentos difíciles y por disfrutar juntos las pequeñas y grandes cosas de este hermoso juego que se llama vida, ¡Lo logramos!.

A mi mamá Irene Cano Martínez, por inculcarme que siempre debemos seguir adelante a pesar de las adversidades, por su apoyo y motivación, por siempre tener fe en mí y sobre todo por su cariño incondicional.

A mis padrinos Rosa Rodríguez May (†) y Enrique Antonio García Aguirre, por acogerme como un hijo, por sus sabios consejos, por su cariño y apoyo incondicional. Siempre los llevo en mi corazón.

A toda mi familia (Cano Martínez, García Rodríguez, Segura Ozuna y Machorro Paz), por confiar siempre en mí y apoyarme en todo momento, por todos los bellos momentos que pasamos juntos y por ser una inspiración para seguir adelante.

Al M.C. Héctor López Arjona, a mis amigos, conocidos y compañeros que de una u otra forma contribuyeron y me apoyaron para lograr esta importante meta.

A mi director de tesis Dr. Giner Alor Hernández, por su dedicación y profesionalismo para el desarrollo y culminación de esta tesis doctoral, por sus enseñanzas, por sus consejos, por compartir su gran experiencia y sobre todo por su amistad.

Al Dr. José Oscar Olmedo Aguirre por su apoyo y enseñanzas durante la estancia de investigación y en la tesis, pero sobre todo por sus consejos y amistad.

A mis sinodales, Dra. Lisbeth Rodríguez Mazahua, Dr. José Luis Sánchez Cervantes, Dr. Cuauhtémoc Sánchez Ramírez y Dr. Galo Rafael Urrea García, por su tiempo y por los comentarios para el mejoramiento de esta tesis.

Al Dr. Mario Leoncio Arrioja Rodríguez y al Dr. Mauricio Romero Montoya, por el apoyo incondicional en los procesos de ingreso, seguimiento y titulación dentro del programa de Doctorado en Ciencias de la Ingeniería.

A CONACYT, por el apoyo para el desarrollo del proyecto de investigación, el cual fue fundamental para la finalización del mismo en tiempo y forma.

Índice

Índice de figuras

Índice de tablas

Abstract

In the Internet of Things (IoT) paradigm, multiple smart devices communicate among themselves to fulfill a common objective. Similarly, they are primarily characterized by remarkable detection and processing capabilities. On the other hand, a service composition (SC) task involves performing the orchestration or choreography of services. Service orchestration is a centralized process for organizing interactions among the services of an activity or business process; however, orchestrators involved in a same service orchestration task rarely know each other. The choreography of services is collaborative, allowing each involved party to describe their participation in the interaction. The choreography follows the sequences of the messages between the multiple parties and typically obtains the exchanges in the public messages that occur between the services, instead of a specific business process that a single party executes. SC is frequently studied in the context of Web services (WS), where a series of standards have been developed and used in real-world implementations to support SC. Unfortunately, these standards are inadequate in the IoT paradigm, due to IoT devices are based on data/events, the resources are restricted, the heterogeneity of devices, the difficulty of integration and collaboration of devices, to representation of workflows and limited service coordination.

In this context, this thesis proposes the development of a service composition mechanism in the IoT approach that allows the orchestration and choreography of services offered and invoked in real time by smart devices, enabling Machine to Machine (M2M) communication and Person to Machine (P2M) through the sending and receiving of data or events in real-world scenarios.

For this reason, a service composition language was developed in the IoT that performs the services integration and coordination, offered by various smart devices such as *wearables*, sensors and actuators in healthcare and home automation scenarios. The language integrates the expressiveness of the compositional languages Business Process Execution Language (BPEL) and Web Services Choreography Description Language (WSCDL) and three new extensions were incorporated in BPEL (selection, update and notification), which allows to have a language with greater expressiveness. In addition, three case studies were developed and the CPN (Colored Petri Nets) were used for modeling, formalization and validation the service composition mechanism in the IoT context.

Resumen

En el Internet de las cosas (IoT), diversos dispositivos inteligentes se comunican entre sí para lograr un objetivo común. Adicionalmente, se caracterizan principalmente por una notable capacidad de detección y procesamiento. Por otro lado, una tarea de composición de servicios (SC) implica realizar la orquestación o coreografía de los servicios. La orquestación de servicios es un proceso centralizado para organizar interacciones entre los servicios de una actividad o proceso comercial; sin embargo, los participantes involucrados en una misma tarea de orquestación de servicios rara vez se conocen entre sí. La coreografía de servicios regularmente es colaborativa, por ello cada participante describir su participación en la interacción. La coreografía sigue las secuencias de los mensajes entre los participantes y generalmente obtiene los intercambios en los mensajes públicos que ocurren entre los servicios, en lugar de un proceso comercial específico que ejecuta una sola parte. La SC se estudia con frecuencia en el contexto de los servicios Web (WS), donde se desarrollan y utilizan estándares en implementaciones del mundo real para soportar la SC. Desafortunadamente, estos estándares son inadecuados en el paradigma del IoT, debido a que los dispositivos se basan en datos/eventos, a que los recursos son limitados, a la heterogeneidad de dispositivos, a la dificultad de integración y colaboración de dispositivos, a la representación de flujos de trabajo y a la limitada coordinación de servicios.

Ante este contexto, esta tesis propone el desarrollo de un mecanismo de composición de servicios basado en el enfoque del IoT que permite la orquestación y coreografía de los servicios ofrecidos e invocados en tiempo real por dispositivos inteligentes, permitiendo la comunicación M2M (máquina a máquina) y P2M (persona a máquina) a través del envío y recepción de datos o eventos en escenarios del mundo real.

Por tal motivo, se desarrolló un lenguaje de composición de servicios en el IoT que realiza la integración y coordinación de servicios que ofrecen diversos dispositivos inteligentes tales como *wearables*, sensores y actuadores en escenarios del cuidado de la salud y la domótica. El lenguaje integra la expresividad de los lenguajes composicionales BPEL (Lenguaje de ejecución de procesos de negocio) y WSCDL (Lenguaje para la descripción de coreografías de servicios Web) y se incorporaron tres nuevas extensiones en BPEL (de selección, actualización y notificación) lo que permite tener un lenguaje con mayor expresividad. Además, se desarrollaron tres casos de estudio y se utilizaron las CPN (Redes de Petri Coloreadas) para la modelación, formalización y sobre todo para la validación del mecanismo de composición de servicios en el contexto del IoT.

Introducción

El IoT es la importante evolución de Internet donde dispositivos y máquinas heterogéneos están interconectados entre ellos y con las personas. Recientemente, los micro-controladores utilizados para comunicarse a través de Internet presentan mayor popularidad, lo que ha dado lugar a una gran diversidad de dispositivos inteligentes y conectados en red, como objetos digitalmente mejorados, detectores de movimiento, dispositivos de vigilancia de la salud, medidores eléctricos, entre otros. Todos estos dispositivos se caracterizan principalmente por sus capacidades de detección, procesamiento y conexión en red.

Con el fin de permitir la comunicación en tiempo real de objetos inteligentes a través de Internet, algunos protocolos Web se están implementando en tiempo real. Estos protocolos son compatibles con objetos inteligentes, estándares Web de código abierto, el perfil de dispositivos para servicios Web (WS) y el protocolo de aplicación restringida (CoAP por sus siglas en inglés *Constrained Application Protocol*). En paralelo, los servicios que proporcionan los objetos inteligentes se acceden directamente en la Web e interactúan con una amplia variedad de WS convencionales para formar una nueva generación de aplicaciones ubicuas; sin embargo, existe un problema con respecto a la SC de los objetos inteligentes.

SC es un principio básico de la Computación Orientada a Servicios (SOC), donde los diferentes servicios se combinan para atender las peticiones de los usuarios. Dos aspectos importantes de la SC son la orquestación y la coreografía de servicios. La SC se estudia ampliamente en el contexto de los servicios Web y procesos de negocios, donde se desarrollan y utilizan una serie de estándares para implementaciones en el mundo real. Desafortunadamente, estos estándares son inadecuados en el paradigma del IoT, debido a que los dispositivos se basan en datos/eventos, a que los recursos son limitados, a la heterogeneidad de dispositivos, a la dificultad de integración y colaboración de dispositivos, a la representación de flujos de trabajo y a la limitada coordinación de servicios.

Por otra parte, el soporte en tiempo real para los protocolos basados en la Web facilita la llegada de nuevas aplicaciones en el IoT. Además, la SC busca reutilizar varios servicios de componentes existentes al unirse a ellos de una manera creativa; la idea es que cuando se aplica al contexto del IoT, optimiza el desarrollo de las aplicaciones en el IoT. Del mismo modo, la SC aplicada en el IoT permite combinar servicios de múltiples objetos inteligentes para satisfacer las necesidades complejas de los usuarios en una diversa variedad de domi-

nios de aplicación y se utiliza para crear aplicaciones innovadoras de una manera más eficiente.

Ante este contexto, esta tesis plantea el desarrollo de un mecanismo de composición de servicios bajo el enfoque del IoT que permita la orquestación y coreografía de servicios ofrecidos e invocados en tiempo real por diversos objetos, cosas o dispositivos inteligentes, permitiendo la comunicación M2M y P2M a través del envío y recepción de datos o eventos en escenarios del mundo real. Este documento está estructurado en cinco capítulos, los cuales describen los aspectos fundamentales de la tesis doctoral, los hallazgos y los principales resultados.

El capítulo 1 presenta la revisión de los fundamentos teóricos que sirven como base conceptual de esta tesis doctoral, y que permiten comprender su contenido. También se presenta el objetivo general y los objetivos específicos de la tesis, así como la hipótesis, el planteamiento del problema, la justificación y la contribución al conocimiento.

En el capítulo 2 se presenta una revisión del estado del arte, en la cual se discuten brevemente algunos de los trabajos más relevantes y que se encuentran relacionados con el presente trabajo de investigación. El estado del arte se encuentra dividido en cuatro secciones: 1) Composición de servicios en el IoT, 2) Orquestación de servicios en el IoT, 3) Coreografía de servicios en el IoT, y 4) Coordinación de servicios en el IoT. Adicionalmente, se presenta una tabla comparativa de los trabajos relacionados con la tesis doctoral.

El capítulo 3 describe la metodología de investigación utilizada para el desarrollo de este trabajo de investigación, así como los resultados de la aplicación de esta metodología de investigación. Cuatro fases son las etapas que conforman la metodología de investigación: 1) Etapa de análisis. Esta etapa presenta el análisis de diversos aspectos como las tendencias y desafíos del IoT, lenguajes de programación y plataformas para el IoT; herramientas de modelado, elementos de la orquestación y coreografía de servicios, la expresividad del mecanismo y las técnicas de formalización; 2) Etapa de diseño. Esta etapa presenta el diseño de la orquestación y la coreografía, el diseño de la expresividad, el diseño de los casos de estudio, su correspondiente modelado y el diseño de la formalización del mecanismo de composición de servicios; 3) Etapa de desarrollo. Esta etapa presenta el desarrollo de escenarios en el contexto del cuidado de la salud, la domótica y la integración y coordinación de servicios en el IoT, y 4) Etapa de pruebas. Esta etapa presenta las pruebas y validación del mecanismo de composición de servicios en el IoT.

El capítulo 4 describe los principales resultados obtenidos, los cuales incluyen: 1) Composición de servicios de salud para el control del sobrepeso u obesidad, 2) Composición de

servicios en domótica para el confort, seguridad y ahorro de energía de una casa inteligente, y 3) Integración y coordinación de servicios en el IoT para contribuir en el control de peso, confort, ahorro de energía y seguridad de un paciente con discapacidad física.

El capítulo 5 presenta las conclusiones al finalizar el desarrollo de la tesis y el trabajo a futuro del proyecto de investigación.

Finalmente, se presentan los anexos, productos académicos y las referencias utilizadas.

Capítulo 1
Antecedentes

En este capítulo se presentan los conceptos básicos relacionados con el tema de tesis. Se incluye el planteamiento del problema, objetivos, hipótesis, justificación y la contribución al conocimiento de este trabajo de tesis.

1.1 Marco teórico

1.1.1 Internet de las cosas

El IoT es un nuevo paradigma que está ganando terreno rápidamente en el escenario de la tecnología inalámbrica moderna. La idea básica de este concepto es la presencia en el entorno de diversas cosas u objetos interconectados, tales como la identificación por radiofrecuencia (RFID del inglés *Radio Frequency Identification*), teléfonos inteligentes, actuadores, sensores, entre otras cosas, los cuales a través de esquemas de direccionamiento únicos, son capaces de interactuar entre sí, colaborando y cooperando con otros objetos vecinos para lograr objetivos en común (Atzori et al., 2010). El término del IoT se utilizó por primera vez por Kevin Ashton en 1999 y posteriormente por David L. Brock en 2001, ambos fundadores del original centro MIT Auto-ID del Instituto de Tecnología de Massachusetts (MIT - *Massachusetts Institute of Technology*), el cual tuvo su auge en al año 2003 cuando el simposio ejecutivo del código del producto electrónico (EPC) marcó el lanzamiento oficial de la red EPC, una infraestructura de tecnología abierta que permite a las computadoras identificar automáticamente cosas u objetos y realizar un seguimiento de ellos desde su fabricación hasta la llegada a los centros de distribución en las tiendas; ante este contexto el RFID se convirtió en la tecnología clave para el crecimiento económico en los siguientes años. El término “Auto-ID” se refiere a cualquier clase amplia de tecnologías de identificación utilizadas en la industria para automatizar, reducir errores y aumentar la eficiencia. Estas tecnologías incluyen sensores, datos biométricos y reconocimiento de voz, tarjetas inteligentes y lectores de códigos de barras, además desde el año 2003 la tecnología Auto-ID se identificó como el RFID (Sundmaeker et al., 2010).

Por otra parte las tecnologías sensoriales de forma inalámbrica ampliaron significativamente las capacidades sensoriales de los dispositivos (cosas u objetos) y por lo tanto el concepto original del IoT. Actualmente, una serie de tecnologías están involucradas en el IoT, tales como las redes de sensores inalámbricos (WSN por sus siglas en inglés *Wireless sensor networks*), comunicación de campo cercano (NFC por sus siglas en inglés *Near Field Communication*), RFID, comunicaciones inalámbricas de poca energía, computación en la nube, entre otras (Li et al., 2015) (Kawsar et al., 2010) (Welbourne et al., 2009). Además, cada día el IoT tiene mayor atención en el mundo académico e industrial, dada a la integración de receptores móviles de corto alcance en una amplia gama de dispositivos adicionales y cosas de uso diario, que permiten nuevas formas de comunicación como son: persona a persona (P2P por sus siglas en inglés *Person to Person*), P2M y M2M; esto marca una nueva dimensión relacionada con el mundo de la comunicación e información (Bandyopadhyay y Sen, 2011). Por ello, el IoT se caracteriza por las cosas del mundo real, ampliamente distribuidas con capacidades limitadas de almacenamiento y procesamiento, que implican preocupaciones con respecto a la fiabilidad, rendimiento, seguridad y privacidad (Botta et al., 2015).

1.1.2 Dominios de aplicación del IoT

La principal fuerza de la idea del IoT es el alto impacto que tiene sobre varios aspectos de la vida cotidiana y el comportamiento de los usuarios. Desde el punto de vista de un usuario, los efectos más evidentes de la introducción o inserción en el mundo del IoT son visibles en los diversos dominios de aplicación del IoT.

En este contexto, son diversos los escenarios en los diferentes dominios de aplicación del IoT, haciendo una revisión en la literatura se identificaron hasta diez y ocho dominios de aplicación del IoT, por tal motivo, como se observa en la Figura 1.1, se clasificaron en siete dominios de aplicación: 1) Industrial; 2) Transportación y Logística; 3) Negocios Inteligentes/Manejo de Productos e Inventarios; 4) Medio Ambiente, Agricultura y Ganadería; 5) Personal y Social; 6) Seguridad y Vigilancia; 7) Cuidado de la Salud (Atzori et al., 2010) (Sundmaeker et al., 2010) (Bandyopadhyay y Sen, 2011) (Miorandi et al., 2012) (Gubbi et al., 2013) (Perera et al., 2013) (Aggarwal et al., 2013) (Said y Masud, 2013) (Xu et al., 2014) (Gluhak et al., 2014) (Whitmore et al., 2015) (Li et al., 2015) (Botta et al., 2015).

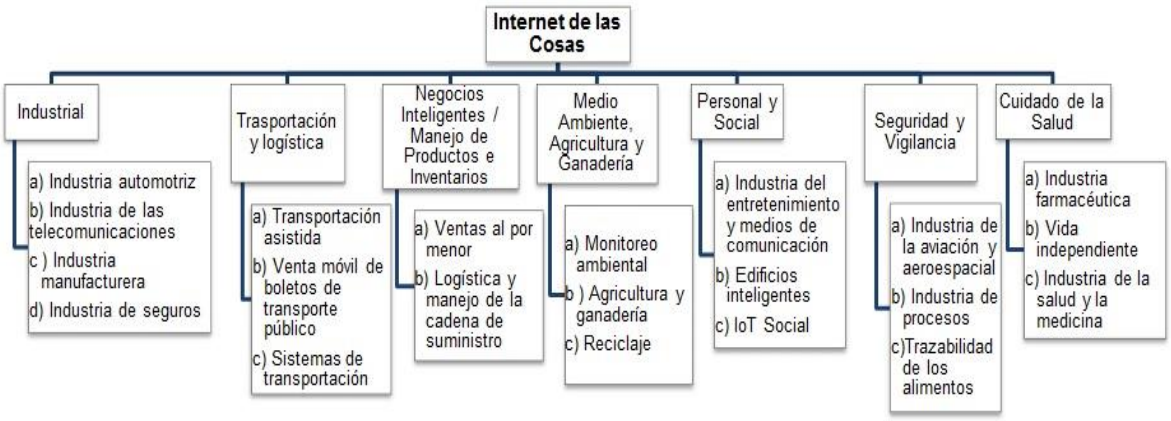


Figura 1.1 Dominios de aplicación del IoT.

Industrial. El IoT es capaz de mejorar las transacciones de negocios con redes de servicios inteligentes, los cuales mejoran la gestión de mejores aplicaciones y la eficiencia al procesar la información en tiempo real, tales como los pagos *online,* el almacenamiento de datos críticos, agregar calidad al servicio (QoS) y asociar los indicadores de funcionamiento. Adicionalmente, el IoT reduce la brecha entre los componentes dentro de la economía digital actual, donde la economía centrada en los servicios se realiza mediante redes de transacciones. Mientras tanto, los modelos de negocios se benefician del IoT en los niveles intra-organizacionales, además las empresas que utilizan el IoT se benefician de los productos de la competencia, en relación a la rentabilidad, contar con modelos de negocios más amigables con el medio ambiente, optimizar recursos y en relación al procesamiento de la información en tiempo real. Por otra parte, el IoT conectado globalmente proporciona a las empresas redes de servicios integrados y con ello los fabricantes obtienen beneficios, ya que el IoT proporciona a los socios de negocios integrar los diversos recursos de las empresas. Este dominio está integrado por los siguientes subdominios: Industria automotriz, Industria de las telecomunicaciones, Industria manufacturera e Industria de seguros.

Trasportación y logística. El IoT ofrece soluciones para sistemas de recolección de tarifas y peajes, control de los pasajeros y las mercancías que circulan por el sistema de carga internacional que apoyan las políticas en materia de seguridad de los gobiernos y la industria del transporte, para satisfacer la creciente demanda de seguridad en el mundo. Por otra parte, el monitoreo de los atascos de tráfico a través de los teléfonos celulares de los usuarios y el despliegue de sistemas de transporte inteligente (ITS) contribuyen a que el transporte de mercancías y personas sea más eficiente. Además el uso de las tecnologías del IoT para la

gestión del equipaje de los pasajeros en los aeropuertos y las operaciones de las aerolíneas permite el seguimiento automatizado, la clasificación, el control de los cobros por exceso de equipaje y una mayor seguridad.

Adicionalmente, el IoT brinda soluciones para transformar el sistema de transporte y el servicio de automóviles. Para ello, la integración de la nube con las tecnologías del IoT representan una prometedora oportunidad, por consiguiente una nueva generación de servicios de minería de datos en la nube se encuentran en desarrollo para obtener muchos beneficios en los negocios, tales como el incremento de la seguridad vial, reducción de la congestión de tráfico, manejo del tráfico y estacionamiento, realización de análisis de garantía y recomendación de fecha de mantenimiento del automóvil. Además, numerosos vehículos se encuentran equipados con poderosos sensores de gran alcance, para la creación de redes, la comunicación y con capacidad para procesar datos, intercambiando información con otros vehículos (V2V) o intercambiando información con la infraestructura carretera, tales como cámaras y luces de las calles (V2I) sobre varios protocolos tales como el protocolo de transferencia de hipertexto (HTTP), protocolo para la Transferencia Simple de Correo (SMTP), Protocolo de Control de Transmisión/Protocolo de Internet (TCP/IP), Protocolo de Aplicaciones Inalámbricas (WAP) y la siguiente generación de protocolos de telemática (NGTP).

Por otra parte, la adopción del IoT en la nube, promete en logística una nueva manera de servicio que está cambiando realmente el campo de los negocios. Además, permite nuevos escenarios interesantes facilitando el manejo automatizado de los flujos de mercancías entre el punto de consumo y el punto de origen, con el propósito de satisfacer los requisitos específicos expresados en términos de tiempo, costo o medio de transporte. Adicionalmente, gracias a las tecnologías de geolocalización, se realiza el seguimiento automático de las mercancías en tránsito. Este dominio está integrado por los siguientes subdominios: Transportación asistida, Venta móvil de boletos de transporte público y Sistemas de transportación.

Negocios Inteligentes/Manejo de Productos e Inventarios. Actualmente la tecnología RFID se utiliza en muchos sectores del manejo de inventarios, a través de la oferta y la cadena de suministro. Por otra parte, en relación a las ventas al por menor, las tecnológicas del IoT se utilizan para supervisar la disponibilidad del producto en tiempo real y mantener el inventario adecuado. Además juega un papel muy importante en la post-venta, debido a que los usuarios recuperan automáticamente todos los datos acerca de los productos comprados. Adicionalmente, las tecnologías de identificación ayudan a limitar los robos y la

falsificación, ofreciendo productos con un identificador único que incluye una descripción completa y de confianza del producto. Este dominio está integrado por los siguientes subdominios: Ventas al por menor y Logística y manejo de la cadena de suministro.

Medio Ambiente, Agricultura y Ganadería. El IoT se utiliza de forma adecuada en las aplicaciones del monitoreo ambiental, en este caso un papel fundamental lo juega la capacidad sensorial, para percibir de forma autónoma y distribuida, fenómenos y procesos naturales como la temperatura, el viento, la lluvia, el nivel de agua, entre otros aspectos; así como para integrar datos heterogéneos en aplicaciones globales, procesando la información en tiempo real en conjunto con un gran número de dispositivos para comunicarse entre ellos mismos, además proporciona una plataforma sólida para detectar y monitorear anomalías que ponen en peligro la vida humana y animal. El despliegue de pequeños dispositivos permite el acceso a ciertas áreas críticas, ya que en ciertos casos el ser humano no es una opción viable. Este dominio está integrado por los siguientes subdominios: Monitoreo ambiental, Agricultura y ganadería, y Reciclaje.

Personal y Social. Las aplicaciones que se encuentran en este dominio son las que permiten al usuario interactuar con otras personas para mantener y construir relaciones sociales. De hecho, las cosas desencadenan automáticamente mediante las redes sociales (Facebook, Twitter, Micro-blog y LinkedIn) la transmisión de mensajes a sus amigos para que ellos estén enterados de lo que él está haciendo o lo que ha hecho en el pasado, tales como desde salir de casa, de la oficina, salir de viaje, hasta para localizar compañeros comunes para jugar su deporte favorito, entre otros aspectos. Este dominio está integrado por los siguientes subdominios: Industria del entretenimiento y medios de comunicación, Edificios inteligentes, y el IoT Social.

Seguridad y vigilancia. En un modelo virtual del IoT, cada objeto físico encuentra una respuesta, la cual brinda servicios a los usuarios. Por ello, cada objeto está bien dirigido y etiquetado por el IoT, sin embargo, las interacciones entre las cosas necesitan contar con aspectos de seguridad para evitar ataques y un mal funcionamiento. En las redes tradicionales, tales como Internet, los protocolos de seguridad y la garantía de la privacidad son ampliamente utilizados en la comunicación y para proteger la privacidad, sin embargo, las téc-

nicas de seguridad aplicadas en las redes convencionales son insuficientes en el IoT, por ello se requiere que los protocolos y mecanismos de seguridad existentes se mejoren antes de ser aplicados en el contexto del IoT. Este dominio está integrado por los siguientes subdominios: Industria de la aviación y aeroespacial, Industria de procesos y Trazabilidad de los alimentos.

Cuidado de la Salud. La salud es una aplicación importante en el área del IoT, ya que se adapta para reducir los costos y mejorar la calidad en el servicio. En este sentido, se utiliza una serie de sensores médicos o dispositivos para monitorizar los parámetros médicos, tales como la presión arterial, nivel de glucosa en la sangre y la temperatura corporal. Los recientes avances en las tecnologías de procesamiento, comunicaciones inalámbricas y en los sensores son el impulso de la aplicación de sistemas en el IoT, en el cuidado de la salud. Recientemente la adopción de sensores corporales portátiles, mejor conocidos como *wearables* se desarrollan para supervisar las actividades de los pacientes o los parámetros de manera continua y en tiempo real. En este contexto, el IoT proporciona en los sistemas de salud una interconexión de los diversos dispositivos heterogéneos para obtener información precisa, completa y rápida de los parámetros de salud de un paciente. Este dominio está integrado por los siguientes subdominios: Industria farmacéutica, Vida independiente (bienestar, movilidad y el seguimiento de adultos mayores) e Industria de la salud y la medicina (red de área personal, monitoreo de parámetros, posicionamiento, sistemas de localización en tiempo real). En este contexto el IoT en este dominio de aplicación es apto para simplificar los procedimientos enfocados al cuidado de la salud y mejorar la calidad de los servicios médicos permitiendo la cooperación entre las diferentes entidades involucradas (personal de enfermería, médicos, pacientes, medicamentos, equipos, entre otros) a través de un control y seguimiento constante. Por otro lado, gracias a la conectividad global del IoT, toda la información relacionada con la asistencia sanitaria (tratamientos, medicación, recuperación, diagnósticos, gestión, las diversas actividades diarias, finanzas, logística, entre otras) se recolecta, gestiona y se comparte eficientemente. Por ejemplo, el ritmo cardiaco de un paciente se proporciona por un sensor de manera constante y posteriormente es enviado a la oficina del doctor, esto es posible mediante el uso de dispositivos de computación personal (computadora portátil, teléfono móvil, tablet) y al acceso móvil a Internet (Wi-Fi, 3G, evolución a largo plazo (LTE por sus siglas en inglés *Long Term Evolution*), entre otros), permiten que los servicios de salud basados en el IoT sean móviles y persona-

lizados. Así mismo, los ambientes de vida asistida, en particular, están enfocados a facilitar la vida diaria de las personas con condiciones médicas crónicas y/o con alguna discapacidad. Además, debido a la gestión eficiente de datos proporcionados por los sensores es posible brindar servicios de vida asistida en tiempo real. Al aplicar el IoT en el cuidado de la salud es posible suministrar muchos servicios innovadores, tales como: garantizar el acceso ubicuo o el uso compartido de los datos médicos como lo son los registros electrónicos de la salud, la entrega de datos en la nube de un centro médico para su almacenamiento y procesamiento, la recopilación de datos vitales de los pacientes a través de una red de sensores conectados a los dispositivos médicos y una adecuada gestión de la información proporcionada por los diversos sensores. Por otra parte, existen desafíos en este dominio de aplicación por abordar, tales como: fiabilidad, seguridad y privacidad de los datos de los pacientes (exposición de ataques de hackers, violación a los datos médicos con confidencialidad, bloqueo de datos y pérdida de la gobernabilidad, abuso de privilegios), mejora en la seguridad de los datos médicos, disponibilidad del servicio, redundancia, rendimiento impredecible (agotamiento de recursos, cuellos de botella en transferencia de datos, impacto en los servicios de tiempo real, transmisión de QoS) y aspectos legales (jurisdicción de datos, contratos, derechos de propiedad intelectual). Por consiguiente, son áreas de oportunidad para desarrollar e implementar el IoT en este dominio y así contribuir en mejorar el cuidado de la salud.

1.1.3 Arquitectura Orientada a Servicios

Un requisito indispensable en el IoT es que las cosas en la red estén interconectadas entre sí, por tanto una arquitectura para el IoT garantiza el funcionamiento propio de las cosas, además sirve como puente entre el mundo virtual y el mundo físico. Diseñar una arquitectura para el IoT implica muchos factores tales como los modelos de negocio y procesos, la comunicación, la creación de redes y la seguridad; adicionalmente se considera la interoperabilidad, escalabilidad y la extensibilidad entre los dispositivos heterogéneos y sus modelos de negocios. Debido a que las cosas en el IoT se desplazan geográficamente y a la necesidad de comunicación con otras cosas en tiempo real, una arquitectura del IoT es adaptable para permitir que los dispositivos se comuniquen con otras cosas de forma dinámica y ayuda a lograr una interacción inequívoca de los eventos, además el IoT tiene la característica de ser heterogéneo y descentralizado. En este sentido, en el IoT la arquitectura orientada al

servicio (SOA) es imperativa para los usuarios y para los proveedores de servicios, además garantiza la interoperabilidad entre los dispositivos heterogéneos de múltiples maneras. En la Figura 1.2, se presenta una arquitectura SOA que consiste en cuatro capas:

1. **Capa de detección**.- se integra con los objetos de hardware disponibles para recibir los estados de las cosas.
2. **Capa de red**.- es la infraestructura de apoyo a través de conexiones inalámbricas por cable entre las cosas.
3. **Capa de servicio**.- en esta capa se crean y gestionan los servicios requeridos por los usuarios o aplicaciones.
4. **Capa interfaz**.- consiste en los métodos de interacción con los usuarios o aplicaciones.

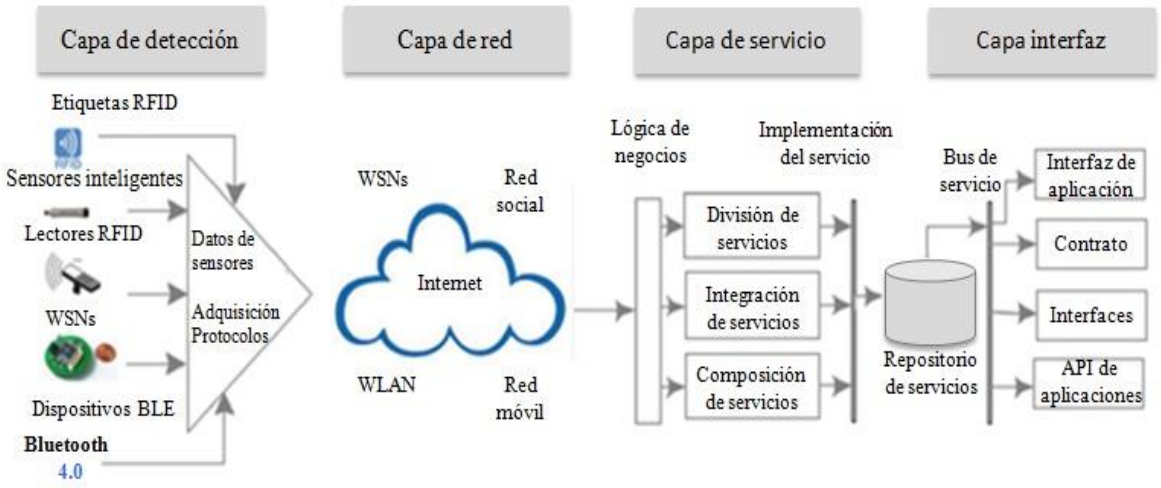


Figura 1.2 Arquitectura orientada al servicio (SOA) para el IoT (Li et al., 2015).

La SOA es un sistema complejo, un conjunto de objetos simples bien definidos o subsistemas, en donde estos son reutilizados y se mantienen de forma individual; por lo tanto los componentes de hardware y software en el IoT se reutilizan y actualizan de manera eficiente. Gracias a estas ventajas SOA se considera como la arquitectura principal de las redes de sensores inalámbricos, porque está diseñada principalmente para proporcionar interoperabilidad entre las cosas heterogéneas, modularidad, escalabilidad y ampliación; además las capacidades y funcionalidades se resumen en un conjunto común de servicios (Li et al., 2015).

1.1.4 Lenguajes de programación para el IoT

El IoT es un concepto moderno para crear ambientes para dispositivos inteligentes, los cuales están conectados entre sí y trabajan en un determinado conjunto de tareas, al igual que las computadoras en Internet. Estos dispositivos permiten recopilar, transferir, analizar datos y realizar cualquier acción o decisión, en función de estos datos. Por otra parte, las redes en el IoT incorporan casi cualquier tipo de recopilador de datos para mejorar el contexto dentro del IoT. Por consiguiente, en cada dispositivo conectado en el IoT, existe un fragmento de código que ejecuta y procesa diferentes entradas y/o salidas de información, haciendo uso de un lenguaje de programación específico (Qubit Labs, 2018).

Los lenguajes de programación para el IoT son similares a los que se utilizan en las aplicaciones de escritorio, aplicaciones móviles y servidores. Por ello, en ocasiones no se perciben ciertas diferencias, sin embargo, existen diversos aspectos y características diferentes en el uso de los lenguajes de programación en el IoT, principalmente la heterogeneidad de los dispositivos inteligentes y el proceso de ejecución de los datos que se realiza en tiempo real. En este sentido, elegir un lenguaje de programación en el IoT es similar a cualquier proyecto de desarrollo de software, porque su comportamiento no es diferente de una computadora portátil, tableta o servidor. Cuando se trata de sensores y concentradores con algún tipo de arquitectura de micro servicios, los datos se almacenan en una base de datos estándar (Intersog, 2018).

Por otra parte, en el IoT se están utilizando ciertos lenguajes de programación populares, sin embargo, los desarrolladores utilizan otros lenguajes para crear diversas cosas novedosas e interesantes. Es importante resaltar que el IoT no es dependiente de en un solo lenguaje de programación, sino al contrario utiliza diversos lenguajes de programación. Por consiguiente para desarrollar soluciones para el IoT se utilizan diferentes lenguajes de programación como: Java, C, JavaScript, Python, C++, PHP, C#, Assembler, Lua, Go, R, Swift, Ruby, Rust, entre otros lenguajes. Así mismo, existen características particulares con las que cuentan en general los diversos lenguajes de programación como son:

- **Abstracción.** Es la posibilidad de usar y definir operaciones o estructuras complejas sin considerar determinadas circunstancias, lo cual repercute en la escritura.

- **Simplicidad.** Consiste en ofrecer conceptos sencillos y fáciles de entender para aprenderlo e implementarlo con determinada comodidad, sin descuidar que el funcionamiento sea de calidad.
- **Estructuración.** Con base en los conocimientos previos relacionados con la programación estructurada, los programadores codifican sus programas cuidando de no incurrir en equivocaciones.
- **Localidad.** Está relacionada con la codificación de los programas con los que el programador trabaja en un determinado tiempo.
- **Naturalidad.** Este aspecto se refiere a que el lenguaje proporciona de manera natural sintaxis, estructuras y operadores para que los programadores desarrollen soluciones de manera eficiente.
- **Compacidad.** Permite describir las acciones a realizar de forma concisa, sin la necesidad de anotar muchas particularidades.
- **Eficiencia.** Se refiere a que el proceso de ejecución y traducción del lenguaje de programación no requiere demasiada inversión de tiempo ni abarcar mucha capacidad de la memoria (Chakray, 2020).

Además, otras características adicionales que se toman en cuenta al referirse a un lenguaje de programación son:

- **Inferencia de tipos.** Para impedir que los programadores realicen la declaración de variables de manera recurrente, estas se infieren de acuerdo al código fuente.
- ***Closures* y Lambdas.** Los *closures* son como fragmentos de código o bloques de código que se utilizan sin ser una clase o un método. Los *closures* acceden a variables no definidas en su lista de parámetros y también asignan variables. Una lambda es esencialmente una función que se define en línea en lugar del método estándar de declaración de funciones. Las lambdas se pasan con frecuencia como objetos.
- ***Defaults methods* y *Traits*.** Los *defaults methods* permiten agregar nuevas funciones a las interfaces existentes y garantizar la compatibilidad binaria con el código escrito para versiones anteriores de esas interfaces. Los *defaults methods* permiten agregar métodos que aceptan expresiones lambda como parámetros para interfaces existentes. Por otra parte, un *trait* es un concepto utilizado en la programación

orientada a objetos, que representa un conjunto de métodos que se utilizan para ampliar la funcionalidad de una clase.

- ***High Order Functions*.** Una función de orden superior toma una función como argumento o devuelve una función. Este tipo de función tiene implementaciones en muchos lenguajes de programación, incluidos Go, JavaScript, Python, entre otros.
- ***Currying.*** Es un proceso de programación funcional en el que se transforma una función con múltiples argumentos en una secuencia de funciones anidadas. Además, en este proceso se transforma una función con múltiples argumentos en una secuencia/serie de funciones, cada una de las cuales toma un solo argumento.
- ***Duck typing*.** Es un concepto utilizado en diversos lenguajes como PHP, JavaScript, Python y Groovy que consideran a las variables por su soporte de métodos y particularmente por sus principales características.
- ***Intersection types y Union types*.** Un tipo de intersección combina varios tipos en uno. Un tipo de unión describe un valor que es de varios tipos. El uso y la notación son bastante simples, el símbolo "&" se utiliza para construir una intersección mientras que el símbolo "|" se utiliza para la unión.
- ***Named arguments*.** Se refieren al soporte de un lenguaje para llamadas a funciones que indican claramente el nombre de cada parámetro dentro de la llamada a la función.
- ***Destructuring assignment*.** Es una expresión para descomprimir valores de matrices o propiedades de objetos en distintas variables, particularmente utilizada en JavaScript.
- ***Tuples*.** Es un conjunto ordenado de valores utilizados en lenguajes de programación como Python, Lisp, Linda, entre otros. Los usos comunes de la tupla como tipo de datos son para pasar una cadena de parámetros de un programa a otro, y para representar un conjunto de valores de atributos en una base de datos relacional (Picodotdev, 2020).

1.1.5 Plataformas para el IoT

Las plataformas en el IoT facilitan la comunicación entre personas, dispositivos, entornos y objetos virtuales que se conectan y requieren capacidades de interacción. De manera similar, tienen el potencial de apoyar a los fabricantes de hardware que producen dispositivos

físicos y sensores para alimentar el IoT. Las plataformas modernas del IoT son la base para interconectar dispositivos y generar sus propios ecosistemas. De manera similar, las plataformas en el IoT son software de soporte que conecta hardware de última generación, puntos de acceso y redes de datos a aplicaciones de usuario final a través de técnicas como software como Servicio (SaaS), Infraestructura como Servicio (IaaS) y Plataforma como Servicio (PaaS). Estos enfoques tienen dificultades para escalar los sistemas en el IoT, ya que cada plataforma gestiona las discrepancias y la traducción de protocolos de manera diferente. Por lo tanto, gracias a las plataformas en el IoT, las APIs (*Application Programming Interface*) REST (*Representational State Transfer*) permiten la descomposición del servicio y aseguran la conectividad inteligente de los objetos en la Web. Sin embargo, a menudo se desconoce cómo interactúan algunos objetos inteligentes, especialmente si no se basan en estas plataformas. Por lo tanto, este problema implica que se requiere el desarrollo de nuevos modelos o mecanismos de composición de servicios para resolver esta problemática (Han et al., 2016).

Las plataformas en el IoT gestionan y permiten visualizar datos en tiempo real para que los usuarios automaticen su entorno. Estas plataformas son necesarias para las soluciones de *middleware*, que a menudo se distribuyen en el IoT (Wortman et al., 2009). Por lo general, una solución del IoT o M2M es un *mashup* (Aplicación Web usada o llamada desde otra aplicación, con el propósito de reutilizar su contenido o funcionalidad) de múltiples proveedores que incluye sensores o controladores, un dispositivo de puerta de enlace para agregar y transmitir datos hacia y desde la red de datos, una red de comunicaciones para enviar datos que se recopilan de los sensores o controladores, software para el análisis y traducción de los datos, y el servicio de la aplicación final. Según (Mineraud et al., 2016), las aplicaciones para las plataformas en el IoT tienen tres propiedades operativas fundamentales:

1. Están basadas en la nube y tienen la capacidad de ser administradas, mantenidas y accesibles desde cualquier lugar.
2. Tienen conectividad. Las plataformas en el IoT proporcionan un enlace común entre los dispositivos y sus datos. Cada dispositivo tiene algo que decir a través de la información.
3. Son controlados por datos y los dispositivos conectados son tan útiles como la información que proporcionan. Las aplicaciones aprovechan la conectividad en la nube para tener puntos de datos del dispositivo y los traducen en información signifi-

cativa que ayuda a comprender qué están haciendo los dispositivos para tomar una decisión.

Las plataformas en el IoT son la clave para el desarrollo de aplicaciones, software escalable y servicios que interconectan el mundo real y el virtual. Por lo tanto, muchas empresas invierten en la prestación de diversos tipos de servicios. Algunas plataformas se centran en proporcionar infraestructuras de *back-end*, otros trabajan en la provisión de conectividad y otros proporcionan plataformas solo para dominios y dispositivos específicos. La industria en su conjunto sigue evolucionando, pero está lejos de generalizar la plataforma que proporcione todas las funcionalidades que se requieren en el IoT. Algunas plataformas disponibles para en IoT son Kaa, Device hive, OracleIoT, ThingSpeak, Microsoft IoT, Axeda, EVRYTHNG, ThingWorx, JasperloT, AWS, Microsoft Azure, IBM Watson, entre otras.

1.1.6 Protocolos de comunicación en el IoT

El IoT cubre una amplia gama de industrias y casos de uso que van desde un solo dispositivo restringido hasta implementaciones masivas multiplataforma de tecnologías integradas y sistemas en la nube que se conectan en tiempo real. Uniéndolos a todos son numerosos los protocolos de comunicación heredados y emergentes que permiten a los dispositivos y a los servidores comunicarse entre sí. Al mismo tiempo, docenas de alianzas y coaliciones se están formando con la esperanza de unificar el contexto de IoT. Por otra parte, de acuerdo al estudio sobre “Tendencias claves para desarrolladores en el IoT 2018” realizado por la fundación Eclipse, como se muestra en la Figura 1.3 los protocolos de comunicación más utilizados en el IoT de acuerdo al porcentaje de uso son: HTTP, MQTT, CoAP, HTTP/2, empresa propietaria, AMQP, XMPP, protocolo del vendedor del dispositivo, otros protocolos y DDS (Ian Skerrett, 2018).

HTTP. Se ha convertido en mucho más que la navegación entre páginas en Internet. Hoy en día, también se utiliza para la comunicación M2M, automatización y es el protocolo utilizado en el desarrollo de soluciones para el IoT, entre otras cosas. Actualmente, se realizan un sin número de actividades en Internet utilizando el protocolo HTTP, porque es de fácil acceso y es fácil relacionarse con él. Por otra parte, HTTP en algunos aspectos es un poco complejo, debido a que es un protocolo de solicitud / respuesta sin estado donde los clientes

solicitan información de un servidor y el servidor responde a estas solicitudes. Una solicitud se compone básicamente de un método, un recurso, algunos encabezados y algún contenido opcional. Una respuesta se compone de un código de estado de tres dígitos, algunos encabezados y algún contenido opcional. Adicionalmente, cada colección de documentos de hipertexto o documentos HTML, es identificado por un Localizador Uniforme de Recursos (URL). Los usuarios simplemente usan el método GET para solicitar un recurso del servidor. Además, los métodos PUT y DELETE permiten a los usuarios cargar y eliminar contenido del servidor, mientras que el método POST permite enviar datos a un recurso en el servidor, por ejemplo, en un formulario Web (Packtpub, 2018).

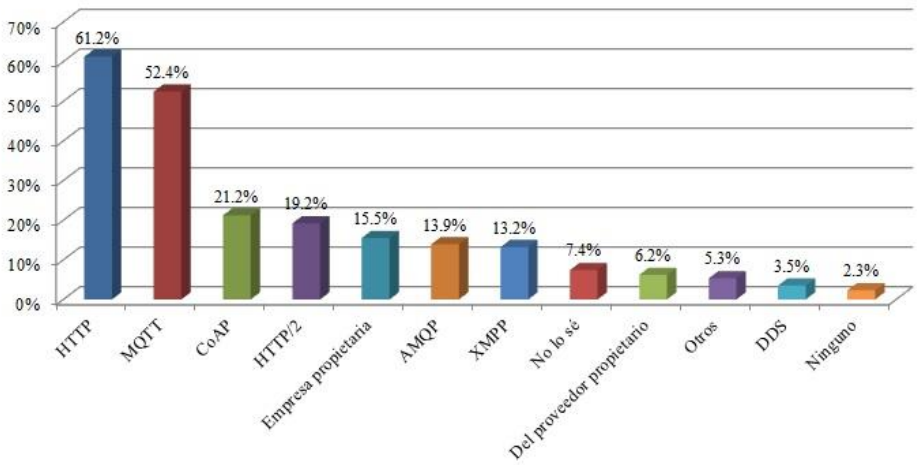


Figura 1.3 Protocolos de comunicación en el IoT (Ian Skerrett, 2018).

MQTT. El protocolo MQTT (*Message Queue Telemetry Transport*), fue creado hace aproximadamente 15 años para monitorear nodos de sensores remotos y está diseñado para conservar energía y memoria. Adicionalmente, mediante el uso de MQTT, un dispositivo conectado se suscribe a cualquier número de temas alojados por un agente MQTT. En general, es un protocolo ligero que se ejecuta en dispositivos integrados y plataformas móviles. El buen desempeño y la confiabilidad del protocolo MQTT se visualiza en su uso a través de Facebook Messenger, Amazon IoT (AWS-IoT), IBM Node-Red, entre otras organizaciones que lo utilizan para atender a millones de personas diariamente (Quora, 2018).

CoAP. El protocolo de aplicación restringida (CoAP - *Constrained Application Protocol*), es un protocolo de la capa de aplicación diseñado para dispositivos en Internet con recursos limitados, como lo son los nodos WSN. CoAP está diseñado para traducirse fácilmente documentos HTTP para una integración simplificada con la Web, al mismo tiempo que cum-

ple con requisitos especializados como el soporte de multidifusión, baja sobrecarga y simplicidad. Adicionalmente, otros aspectos que caracterizan al protocolo CoAP son el diseño de protocolos RESTful que minimiza la complejidad de la asignación con HTTP, la baja sobrecarga del encabezado y complejidad del análisis, soporte de URI y de tipo de contenido, el soporte para el descubrimiento de los recursos proporcionados por los servicios conocidos de CoAP, la suscripción simple a un recurso y el resultado a través de notificaciones *push* (Postscapes, 2018).

HTTP/2. La Web está a punto de abarcar completamente una revolución llamada protocolo HTTP / 2. El protocolo no fue diseñado específicamente para el IoT, pero los creadores de este nuevo protocolo tomaron en cuenta algunas de las necesidades del IoT, ya que se centra en una serie de mejoras sobre el protocolo HTTP tradicional. Por otra parte, permite respuestas multiplexadas: es decir, enviar respuestas en paralelo y que los clientes y los servidores utilicen una única conexión TCP en la que las solicitudes y respuestas se envían en secuencia. Por consiguiente, estas características son muy interesantes para la Web de las Cosas (WoT), ya que conduce a un uso más eficiente de las conexiones, lo que reduce la sobrecarga de HTTP, conduce a transmisiones más rápidas y, por lo tanto, a ahorros potenciales en términos de energía de la batería requerida para comunicarse. El protocolo HTTP/2 también introduce encabezados comprimidos utilizando un formato de compresión de memoria muy eficiente y bajo (HPACK, Compresión de encabezado para HTTP/2). Este nuevo formato reduce el tamaño de cada solicitud y respuesta HTTP. Además, si bien HTTP/1.1 era un protocolo ASCII, es decir, un protocolo que transmite caracteres ASCII, el protocolo HTTP/2 utiliza una estructura binaria, es decir, transmite flujos binarios de datos, ya que son más eficientes de analizar y más compactos. Adicionalmente, esto es interesante para la Web de recursos con recursos limitados, ya que significa que el tamaño de los paquetes es significativamente menor, lo que permite que los dispositivos con memoria RAM limitada trabajen con HTTP/2. Además, HTTP/2 introduce la noción de empuje del servidor, esto significa que el servidor proporciona contenido a los clientes sin tener que esperar a que envíen una solicitud (Webofthings, 2018).

AMQP. El Protocolo AMQP (*Advanced Message Queuing Protocol*) es un estándar de código abierto para mensajería asíncrona por cable. El protocolo AMQP permite la mensajería cifrada e interoperable entre las organizaciones y las aplicaciones. Así mismo, se utiliza en la mensajería cliente/servidor y en la gestión de dispositivos en el IoT. Adicionalmente,

AMPQ es eficiente, portátil, multicanal y seguro. El protocolo binario ofrece autenticación y cifrado por medio de SASL (*Simple Authentication and Security Layer*) o TLS (*Transport Layer Security*), confiando en un protocolo de transporte como el TCP. El protocolo de mensajería es rápido y ofrece una entrega garantizada con acuse de recibo de los mensajes recibidos. AMPQ funciona bien en entornos de múltiples clientes y proporciona un medio para delegar tareas y hacer que los servidores manejen las solicitudes inmediatas más rápido. Debido a que AMPQ es un sistema de mensajería binaria con un comportamiento de mensajería muy estricto, la interoperabilidad de los clientes de diferentes proveedores está asegurada. Por otro lado, AMQP permite que se envíen tres modos de mensajería garantizados especificando un mensaje: 1) A lo sumo una vez (enviado una vez con la posibilidad de ser perdido); 2) Por lo menos una vez (garantizando la entrega con la posibilidad de mensajes duplicados); 3) Exactamente una vez (garantizando una única entrega) (Whatis.techtarget, 2018).

XMPP. El protocolo XMPP (*Extensible Messaging and Presence Protocol*) está basado en el lenguaje de marcado extendido (XML), así mismo permite el intercambio rápido y casi en tiempo real de datos entre varias entidades en una red. Además de proporcionar capacidades de presencia y mensajería, también se utiliza en VoIP, juegos y en las aplicaciones del IoT. Adicionalmente, el protocolo XMPP cuenta con extensas bibliotecas de códigos y un SDK, además se ha utilizado ampliamente como una buena forma genérica de enviar fragmentos de datos entre sistemas, prácticamente es posible mover cualquier tipo de datos. Por otra parte, el protocolo XMPP es el protocolo de mensajería instantánea subyacente para *Google Hangouts*, *WhatsApp Messenger* y otras aplicaciones relacionadas con el chat. Dado que cada dispositivo que usa XMPP tiene un ID único, se considera un protocolo probado, fácil de configurar y fácil de escalar. Adicionalmente, debido a las raíces de XMPP en el mundo de las aplicaciones de chat, también implementa una lista de amigos que, en el caso de las aplicaciones del IoT, contribuye a implementar el control de acceso para los dispositivos conectados (IoT Agenda, 2018).

DDS. El protocolo DDS (*Data Distribution Service*) permite el rendimiento global escalable, en tiempo real, confiable, excesivo y las estadísticas interoperables mediante la técnica de envío-suscripción. Adicionalmente, el protocolo DDS utiliza la arquitectura sin intermediarios y la multidifusión para transmitir QoS de alta calidad a las aplicaciones. Así mismo, el protocolo DDS se implementa en plataformas que van desde dispositivos de bajo impac-

to en la nube y admiten el uso del ancho de banda, además de la ágil orquestación de los aditivos del sistema. Los protocolos DDS en el IoT tienen capas fundamentales tales como: hechos centrados en la suscripción de envío (dcps) y la capa de reconstrucción estadística local (dlrl). Los dcps desempeñan la tarea de entregar los hechos a los suscriptores y la capa dlrl presenta una interfaz para las funcionalidades de dcps, permitiendo el intercambio de datos distribuidos entre objetos habilitados en el IoT (Data-flair, 2018).

1.1.7 Composición de servicios

La composición de servicios (SC) es un principio básico de la computación orientada a servicios (SOC) (Pisching et al., 2015) donde los diferentes servicios se combinan para satisfacer los requisitos complejos de los usuarios. Existen dos perspectivas para la SC, la orquestación de servicios y la coreografía de servicios (Yang y Li, 2014). La Figura 1.4 muestra las relaciones de cada una de ellas en un alto nivel. La orquestación se refiere a un proceso ejecutable y la coreografía sigue la trayectoria de secuencias del mensaje entre las partes y las fuentes.

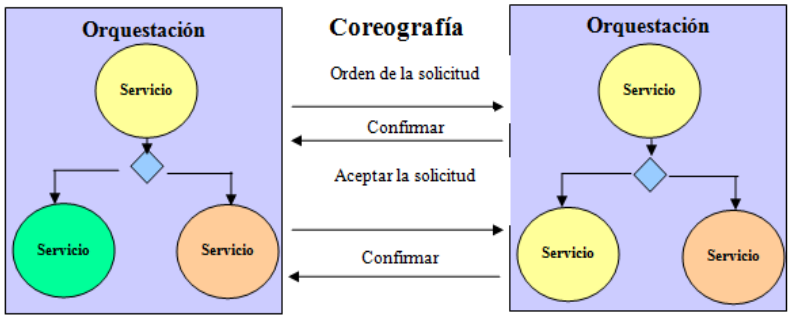


Figura 1.4 Orquestación y coreografía para la composición de servicios.

La SC se estudia frecuentemente en el contexto de servicios Web y procesos de negocios, donde se desarrollan y utilizan una serie de estándares en implementaciones del mundo real para respaldarlos. Sin embargo, las características actuales de los sistemas en el IoT (por ejemplo, los dispositivos basados en datos o eventos), así como problemáticas como la restricción de recursos, hacen que algunas de las técnicas desarrolladas para la SC Web sean inadecuadas cuando se aplican en el contexto del IoT, por tal motivo se requieren nuevos mecanismos de SC que consideren los diversos requisitos de los sistemas en el IoT.

1.1.7.1 Orquestación

La orquestación de servicios es un proceso centralizado para organizar interacciones entre los servicios de una actividad o proceso de negocios; sin embargo, los orquestadores involucrados en una misma tarea de orquestación de servicios rara vez se conocen entre sí. La orquestación se refiere a procesos de negocios ejecutables que interactúan con otros servicios internos y externos. Las interacciones ocurren en el nivel del mensaje. Ellos incluyen negocios lógicos y preguntan el orden de ejecución, y miden a las aplicaciones y organizaciones para definir una duración, una transacción, y un modelo de proceso de multiacceso (Macker y Taylor, 2017).

1.1.7.2 Coreografía

La coreografía de servicios es colaborativa, permite a cada parte involucrada describir su participación en la interacción. La coreografía sigue las secuencias de los mensajes entre las múltiples partes y obtiene típicamente los intercambios en los mensajes públicos que ocurren entre los servicios, en vez de un proceso de negocios específico que una sola parte ejecuta. En la coreografía, no hay un coordinador central. En su lugar, cada servicio implicado en dicha coreografía conoce exactamente cuándo ejecutar sus operaciones y con quién interactua. La coreografía es un esfuerzo colaborativo centrado en el intercambio de mensajes en procesos de negocio públicos. Todos los participantes en la coreografía necesitan estar informados del proceso de negocio, las operaciones a ejecutar, los mensajes a intercambiar, y el tiempo a invertir en dicho intercambio de mensajes (Chen y Englund, 2017).

1.1.8 Lenguajes composicionales

La SC requiere contar con lenguajes que permitan la integración y coordinación de servicios en forma de procesos de negocios o flujos de trabajo (*workflows*). Actualmente existen dos lenguajes composicionales utilizados en los últimos años para realizar la orquestación y la coreografía de servicios, estos son el lenguaje de ejecución de procesos de negocios para servicios *Web* (WS-BPEL o BPEL, por sus siglas en inglés *Web Services Business Process*

Execution Language) y el lenguaje de descripción de la coreografía de servicios *Web* (WSCDL, por sus siglas en inglés *Web Services Choreography Description Language*).

1.1.8.1 BPEL

BPEL es un lenguaje que se utiliza para la definición y ejecución de procesos de negocios que utilizan servicios web. Así mismo, BPEL permite la ejecución de arriba a abajo de la SOA a través de la composición, la orquestación y la coordinación de los servicios web. Adicionalmente, BPEL proporciona una forma relativamente fácil y directa de componer varios servicios en nuevos servicios compuestos llamados procesos de negocios. Además, BPEL es un lenguaje basado en XML.

Por otra parte, BPEL se utiliza para estandarizar la integración de aplicaciones empresariales, así como para extender la integración a sistemas previamente aislados. Entre empresas, BPEL permite una integración más fácil y efectiva con socios comerciales. Adicionalmente, BPEL estimula a las empresas a definir mejor sus procesos comerciales, lo que a su vez conduce a la optimización de los procesos comerciales, la reingeniería y la selección de los procesos más apropiados, optimizando aún más la organización. Las definiciones de los procesos comerciales descritos en BPEL no afectan los sistemas existentes, estimulando así las actualizaciones. Por tal motivo, BPEL es la tecnología clave en entornos donde las funcionalidades ya están expuestas a través de los servicios web (Juric, 2020).

1.1.8.2 WSCDL

WS-CDL es un lenguaje para especificar protocolos punto a punto y no existe ningún punto de centralización, no hay variables globales, condiciones o unidades de trabajo, permite variables y condiciones en varios lugares. Así mismo, WSCDL tiene la capacidad de sincronizar las variables que residen en un servicio con las variables que residen en otro servicio, para obtener un estado global. Además, en WS-CDL, todos los mensajes se describen como tipos de información y no tienen una importancia especial entre sí. Todo lo que WS-CDL describe son las reglas de ordenación de los mensajes que dictan el orden en que son observadas.

WS-CDL es un lenguaje basado en XML que se utiliza para describir el comportamiento común y colaborativo de múltiples servicios que necesitan interactuar para lograr algún

objetivo. Adicionalmente, WS-CDL es una descripción y no un lenguaje ejecutable, un lenguaje que se utiliza para describir inequívocamente colaboraciones de servicio o protocolos comerciales. Adicionalmente, WS-CDL se utiliza para describir los flujos de trabajo internos que involucran múltiples servicios (también llamados puntos finales) que constituyen un comportamiento colaborativo. Así mismo, WSCDL fomenta la conformidad de los servicios con una descripción de coreografía y mejora la interoperabilidad de los servicios a través de una descripción de coreografía. Esto no es más que describir un protocolo de negocios que define una colaboración entre servicios. Cuando el enfoque de WS-CDL está en los dominios de control, se utiliza para describir el orden de los intercambios de mensajes utilizando los diagramas UML (*Unified Modeling Language*) de secuencia (W3C, 2020).

1.1.9 Redes de Petri

Las Redes de Petri son un modelo creado por Carl Adam Petri, basándose en la teoría de autómatas modulares, en 1960 y desde entonces se utilizan para modelar y analizar varios sistemas, como los sistemas de comunicación, fabricación, transporte, entre otros (Ojo et al., 2018). Las Redes de Petri son una herramienta de modelado gráfica y matemática conformada por una serie de elementos denominados lugares, transiciones y arcos de expresiones; representados mediante círculos/elipses, rectángulos y flechas respectivamente. Son una herramienta prometedora para describir y estudiar las relaciones entre partes de un sistema que se caracterizan por ser concurrentes, asíncronos, distribuidos, paralelos, no deterministas y/o estocástico (Zhou y Reniers, 2017). Además del modelado de sistemas, se usan *tokens* o marcas para simular las actividades dinámicas y concurrentes de un sistema, así mismo el contenido de un *token* representa un recurso, persona, entre otros aspectos (Ojo et al., 2018).

Por otra parte, existe una variación de las Redes de Petri, que, en lugar de que las marcas representen objetos, estos tienen un valor que está afiliado a un tipo de dato, ya sea entero, booleano, entre otros. Esta variación recibe el nombre de Redes de Petri Coloreadas o CPN (por sus siglas en inglés, *Colored Petri Nets*) en donde los colores son los datos que contienen las marcas. Al ofrecer esta característica, las CPN resultan ser una gran herramienta para el modelado y visualización de comportamientos dentro un sistema, pues proporcionan: descripciones jerárquicas, simulaciones interactivas y una representación gráfica intuitiva y atractiva (Ojo et al., 2018).

1.2 Planteamiento del problema

El IoT es la evolución importante del Internet donde los dispositivos heterogéneos y las máquinas están siendo conectados a Internet, interconectándose entre ellos mismos y además con las personas. Con más de 10 millones de micro controladores que se fabrican cada año, en donde cada uno de ellos se conecta a través de Internet, una gran variedad de dispositivos inteligentes y conectados en red están haciéndose disponibles más frecuentemente, a partir de objetos mejorados digitalmente, sensores de movimiento, dispositivos de vigilancia de la salud, medidores eléctricos e incluso las luces de las calles. Estos dispositivos inteligentes se caracterizan por su capacidad de detección, procesamiento y por la creación de redes. Sin embargo, debido a la heterogeneidad de dispositivos inteligentes que se utilizan actualmente en el IoT, existen diversas problemáticas tales como la dificultad de integración y colaboración de dispositivos inteligentes heterogéneos, problemas en la representación de flujos de trabajo en el contexto del IoT y una limitada coordinación de servicios basados en el IoT. Por lo anterior, la interrogante no sólo es cómo hacer que los objetos inteligentes sean capaces de comunicarse a través de Internet, sino cómo sus servicios están compuestos para crear aplicaciones nuevas y creativas.

En este sentido, para la comunicación de los objetos inteligentes a través de Internet, actualmente se están enfocando algunos protocolos Web en tiempo real diseñados especialmente para los objetos inteligentes y compatibles con los estándares Web *open source*, como un perfil de los dispositivos para los WS y CoAP. Los servicios de este tipo de objetos inteligentes se acceden directamente en la Web e interactúan con una gran cantidad de WS convencionales para formar una nueva generación de aplicaciones ubicuas, sin embargo, existe un problema con respecto a la SC de los objetos inteligentes.

Por otra parte, la problemática de la SC de los objetos inteligentes, es uno de los principios básicos de la SOC, donde los diferentes servicios se combinan para satisfacer los requerimientos de los usuarios. Dos aspectos importantes de la SC son la orquestación y la coreografía de servicios. La SC se estudia ampliamente en el contexto de los servicios Web y procesos de negocios, donde se desarrollan y utilizan una serie de estándares para implementaciones en el mundo real. Sin embargo, las características actuales de los sistemas en el IoT (por ejemplo, los dispositivos basados en datos o eventos), así como las problemáticas de integración y colaboración de dispositivos inteligentes, de representación de flujos de trabajo, de coordinación de servicios y de la restricción de recursos, hacen que algunas

de las técnicas desarrolladas para la SC Web sean inadecuadas cuando se aplican en el contexto del IoT. Además, actualmente los dispositivos heterogéneos que se conectan en el IoT presentan problemas de conectividad (capacidad de un dispositivo para conectarse con una computadora personal o a cualquier otro dispositivo de forma autónoma), interoperabilidad (capacidad que tienen los sistemas y/o dispositivos no solo de intercambiar información si no de interpretarla y procesarla en un formato amigable para el usuario) y de integración (proceso de combinar datos heterogéneos de muchas fuentes diferentes en la forma y estructura de una única aplicación).

Adicionalmente, el soporte en tiempo real para los protocolos basados en la Web facilita la llegada de nuevas aplicaciones en el IoT. Además, la SC busca reutilizar varios servicios de componentes existentes al unirse a ellos de una manera creativa; la idea es que cuando se aplica al contexto del IoT, optimiza el desarrollo de las aplicaciones en el IoT. Del mismo modo, la SC aplicada en el IoT permite combinar servicios de múltiples objetos inteligentes para satisfacer las necesidades complejas de los usuarios en una amplia gama de dominios de aplicación y se utiliza para crear aplicaciones innovadoras de una manera más eficiente.

Ante este contexto, se requieren mecanismos avanzados que faciliten la integración y colaboración de dispositivos inteligentes heterogéneos, la representación de flujos de trabajo y la coordinación de servicios basados en el IoT, en donde los servicios se utilicen a través de la orquestación y coreografía de servicios en diferentes escenarios de la vida real para satisfacer las diversas necesidades de los usuario, para realmente aprovechar y explotar las ventajas que ofrece el IoT. Por ello, el IoT proporciona nuevas oportunidades de crear un mundo más inteligente que es evidente en la medida que el número de datos y eventos de diversos objetos inteligentes sean fáciles y dinámicamente compuestos para crear nuevas aplicaciones.

1.3 Objetivos

1.3.1 General

Desarrollar un mecanismo de composición de servicios bajo el enfoque del IoT que permita la orquestación y coreografía de servicios ofrecidos e invocados en tiempo real por dispositivos inteligentes que permitan la comunicación M2M y P2M a través del envío y recepción de datos o eventos.

1.3.2 Específicos

- Analizar plataformas de servicios del IoT para identificar la que presente las mejores ventajas y características.
- Analizar e identificar *wearables* y sensores que establezcan comunicación entre diversos dispositivos inteligentes compatibles con las plataformas de servicios para el IoT.
- Diseñar la orquestación de servicios de objetos inteligentes en un entorno inteligente local para comprobar la comunicación M2M y P2M.
- Desarrollar e implementar el mecanismo de composición de servicios para realizar la coreografía de servicios.
- Definir los casos de estudio basados en alguno de los dominios de aplicación del IoT para realizar las pruebas del mecanismo propuesto.
- Validar el mecanismo de composición en los escenarios propuestos del IoT para garantizar la funcionalidad del mecanismo de composición de servicios bajo el enfoque del IoT.

1.4 Hipótesis

A través del desarrollo de un mecanismo de composición de servicios bajo el enfoque del IoT se permitirá realizar la orquestación y coreografía de servicios invocados y ofrecidos en tiempo real por objetos inteligentes heterogéneos (*wearables* o sensores) equipados con las tecnologías RFID, WSN, NFC, Sistema de Posicionamiento Global (GPS), wifi o *bluetooth* lo cual permitirá la cooperación y colaboración entre los dispositivos mediante la comunicación M2M o P2M representada a través de flujos de trabajo.

Existe la posibilidad de generar hipótesis alternas a la planteada en esta tesis, sin embargo esta hipótesis representa la idea central que se presenta en este trabajo de tesis doctoral.

1.5 Justificación

La SC se estudia ampliamente en el contexto de los WS y los procesos de negocios, en donde una serie de normas se desarrollan y utilizan en las implementaciones del mundo real para apoyar la SC. Sin embargo, las problemáticas en los sistemas en el IoT, tales como las limitaciones de los recursos, los dispositivos basados en datos/eventos, la integración y colaboración de dispositivos inteligentes heterogéneos, la representación de flujos de trabajo y la coordinación de servicios basados en el IoT, hacen que algunas de las técnicas desarrolladas para la SC Web sean inadecuadas, por lo tanto, existe la necesidad de modelos o mecanismos de composición de servicios con respecto a los nuevos requerimientos de los sistemas en el IoT. Por otra parte, el apoyo de protocolos Web en tiempo real, es un paso para la llegada de las aplicaciones del IoT, pero cómo agregar servicios a los objetos inteligentes para crear nuevas aplicaciones, en este sentido y ante esta interrogante una implementación exitosa de servicios de objetos inteligentes creará oportunidades para que los desarrolladores realicen una nueva generación de aplicaciones en el IoT con la facilidad de trabajar con las aplicaciones Web que existen hoy en día. Además, la SC tiene como objetivo reutilizar varios servicios de los componentes existentes uniéndolos de forma creativa, la idea es que al aplicarse al contexto del IoT promete generar una aceleración en la creación de aplicaciones para el IoT. Por otra parte, la SC en el IoT permite la agregación de servicios de objetos inteligentes para satisfacer las necesidades complejas de varios dominios de aplicación, además se utiliza para crear aplicaciones innovadoras de una manera más eficiente. El desarrollo de un mecanismo de composición de servicios, hace posible soportar aplicaciones en un entorno de red dinámica, esta es la clave para fomentar el desarrollo de aplicaciones en el IoT. En la Figura 1.5 se observan las aplicaciones en Internet soportadas por la SC Web tradicionales para redes de computadoras y los servicios de los objetos inteligentes en el IoT, en donde para el proceso de composición se involucran dos tipos de servicios: servicios atómicos y servicios compuestos.

Además, aunque la idea original del IoT se deriva de las tecnologías RFID y WSN, en relación a la SC en estas tecnologías, se considera que los estudios y trabajos actualmente se encuentran en la fase inicial en cuanto a las pruebas de campo, por tal motivo el IoT es un área activa de investigación en diversos aspectos, particularmente en materia de integración

y coordinación de dispositivos inteligentes heterogéneos, así como en la composición de servicios utilizados en el IoT.

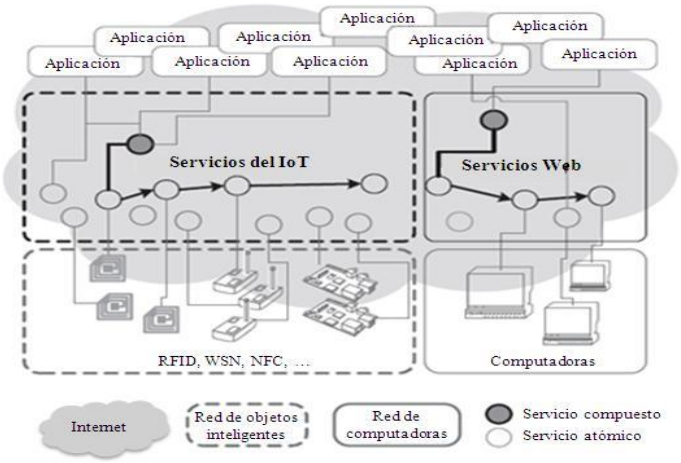


Figura 1.5 Aplicaciones en Internet (Han et al., 2016).

Por otro lado, dado que los servicios de objetos inteligentes en el IoT tienen muchas características diferentes en comparación con los WS tradicionales, la problemática de la SC en el IoT tiene muchos retos y desafíos que involucran el rediseño de los modelos o mecanismos de composición de servicios que aprovechen plenamente su potencial en el IoT. En este sentido, cuando el IoT se extienda a muchos otros objetos conectados en red, la SC evolucionará (Han et al., 2016).

El mecanismo de composición de servicios bajo el enfoque del IoT que se presenta en esta tesis doctoral se desarrolló utilizando herramientas *open source*, sin embargo, fue necesario adquirir sensores y *wearables* para las pruebas y validación del mecanismo en escenarios de los dominios de aplicación del IoT del cuidado de la salud y la domótica.

1.6 Contribución al conocimiento

La principal contribución del desarrollo del mecanismo de composición de servicios bajo el enfoque del IoT es:

- Un lenguaje para la composición de servicios bajo el enfoque del IoT que realiza la orquestación y coreografía de servicios ofrecidos por dispositivos inteligentes que permiten la comunicación M2M y P2M a través del envío y recepción de datos o eventos.

Capítulo 2
Estado del arte

En este capítulo se presentan una serie de trabajos relacionados que describen el área de oportunidad que se aborda en esta tesis doctoral. Dichos trabajos se encuentran divididos en cuatro secciones: 2.1 Composición de servicios en el IoT, 2.2 Orquestación de servicios en el IoT, 2.3 Coreografía de servicios en el IoT y 2.4 Coordinación de servicios en el IoT. Se consideró esta clasificación para identificar y ordenar los trabajos que presentan alguna relación con estos aspectos fundamentales que se abordan en esta tesis doctoral y para una mejor presentación del análisis comparativo del estado del arte.

2.1 Composición de servicios en el IoT

En Rodríguez et al. (2012) se presentó un mecanismo de fusión de datos basado en el modelo de SC para el IoT, ya que la relevancia del IoT en el ámbito de la computación ubicua está creciendo y el mundo de los dispositivos en la vida diaria está aumentando exponencialmente tanto en casa, en el trabajo e incluso en la relación de las personas. Se propuso un nuevo método para realizar la fusión de datos distribuida utilizando el modelo de SC del *middleware* DOHA (*Dynamic Open Home-Automation*) el cual está basado en SOA y cuenta con un mapa de SC basado en el XML. El carácter distribuido de esta propuesta hace el modelo de servicio muy dinámico y escalable, aspectos importantes en el desarrollo de aplicaciones que se ven en dispositivos embebidos. Para ilustrar el modelo de SC en la plataforma DOHA, se presentaron diversos escenarios probados en algunas áreas de aplicación del IoT, tales como: automatización del hogar, vida cotidiana asistida y la industria.

Además, Yang y Li (2014) presentaron una estrategia eficiente desde el punto de vista de la selección de los datos sensoriales y la agregación en la SC de información del IoT, en donde la selección de los servicios candidatos se discutió por la modelización y la evaluación de la calidad del servicio (QoS) en el IoT. Los autores utilizaron como método de optimización global un algoritmo genético binario (GA) mejorado, este algoritmo se encargó de encontrar las óptimas soluciones en la SC. Finalmente los resultados experimentales demostraron la viabilidad de la optimización basada en la mejora del GA, la eficacia del criterio de optimización basado en la QoS y el valor funcional del servicio.

Salle et al. (2016) presentaron un enfoque biológico inspirado preliminarmente para aprovechar las peculiaridades del sistema inmunológico de tal manera que permita la composición de software y servicios de una manera dinámica y fiable, con el fin de proporcionar una herramienta capaz de gestionar la combinación y selección de software y servicios en el contexto del IoT. La composición se generó para satisfacer la necesidad de fiabilidad y teniendo en cuenta la necesidad de escribir aplicaciones basadas en grandes datos (Big Data) de las cuales el IoT es un gran productor. Imitar el comportamiento de la naturaleza no es nuevo en el campo de la informática, ejemplos clásicos son las redes neuronales, el algoritmo genético, el mismo sistema inmunológico y más el cálculo de inspiración biológica, los cuales tienen ciertas características como la propiedad distributiva, el dinamismo, la fiabilidad y la capacidad de recuperación; todos los nuevos retos que los dispositivos móviles presentan todos los días. Por ello se recomendó adoptar el uso del paradigma basado en la programación funcional porque ofrece una técnica eficaz para los desafíos en el IoT.

Además, Liu et al. (2013) desarrollaron un algoritmo basado en la evolución cooperativa que tuvo como base la optimización de la nube de partículas, para resolver la problemática de la SC en el IoT impulsado por la calidad del servicio (QoS). Así mismo, presentaron estrategias efectivas para resolver este problema en la simulación del proceso de la evolución genética biológica, el cual incluyó una mejor estrategia local y global que introdujo las perturbaciones. Los autores consideraron al mismo tiempo la diversidad de la población y la presión de selección. Además, se propusieron doce pasos para la implementación de este algoritmo y se indicó que para empezar a diferenciar la SC tradicionales en el dominio virtual de la información, es necesaria la SC en el IoT para procesar datos en tiempo real, obtenidos de los dispositivos electrónicos que trabajan conjuntamente en el mundo real.

Boa et al. (2012) presentaron un protocolo de gestión de la confianza escalable para el IoT, el cual toma en cuenta las relaciones sociales y considera el uso de tres propiedades de confianza: honestidad, cooperación e interés para evaluar la confianza. Los autores demostraron que utilizando más las nuevas observaciones directas sobre la información pasada, se incrementa la exactitud de evaluación y la velocidad de convergencia de la confianza; además se demostró que al utilizar más recomendaciones indirectas sobre la información pasada, se incrementa la velocidad de convergencia de la confianza, sin embargo disminuye la precisión de ataques de recomendación falsos de nodos maliciosos. Las pruebas demostraron que el protocolo de evaluación proporciona confianza al actual estado del nodo y demostraron la eficacia del mismo en una aplicación de la SC en entornos del IoT. Finalmente

los resultados demostraron que la SC basados en la confianza supera a la SC al azar y se aproxima al rendimiento máximo del mundo real.

Por otra parte, Atzori et al. (2012) presentaron una integración de las redes sociales con los objetos del IoT, que es a lo que se le conoce como el paradigma del Internet social de las cosas (SIoT), el cual tiene el potencial para soportar aplicaciones novedosas y servicios de red para el IoT de una manera eficaz y eficiente. Además se propuso una posible arquitectura que incluye las funcionalidades requeridas para integrar las cosas dentro de la red social. Otra contribución de este trabajo fue la identificación de políticas adecuadas para el establecimiento y la gestión de las relaciones sociales entre los objetos de tal manera que la red social resultante es navegable. Por otra parte se analizaron las características de la estructura de la red SIoT por medio de las salidas del simulador de movilidad SWIM (*Small World In Motion*), en donde los resultados mostraron que las distribuciones de probabilidad de la distancia entre los nodos que están vinculados por una relación social dependen del tipo de relación.

Además, Zhou et al. (2013) presentaron la arquitectura "CloudThings" la cual es un enfoque común para integrar el IoT y la computación en la nube, en donde se examinó un escenario de una casa inteligente habilitada para analizar los requisitos de las aplicaciones del IoT. Además de la arquitectura se propuso una plataforma del IoT basada en la nube con capacidad para integrar el IaaS, PaaS y SaaS de las cosas en la nube para acelerar la aplicación, desarrollo y gestión del IoT. Adicionalmente se utilizó el sensor de temperatura LM35 para detectar la temperatura del ambiente en la casa, además se utilizó el sensor analógico LDR (*Light Dependent Resistor*) para detectar la luminosidad de la luz en la casa, por otro lado se utilizó el cable Ethernet para conectar Arduino a Internet y a través del protocolo HTTP se enviaron datos entre las cosas del IoT compatibles con Arduino y la aplicación en la nube, así mismo se utilizó el servicio de una Google Apps para alojar la aplicación en la nube que almacenó las lecturas del sensor y las visualizaba, finalmente se utilizó Paraimpu un servicio del IoT basado en la nube para conectar los sensores compatibles con Arduino y compartir las lecturas de los sensores con amigos.

Chen et al. (2015) presentaron el diseño y análisis de la adaptación y supervivencia de un protocolo de gestión de la confianza para sistemas del IoT centrados en el usuario, en donde el usuario realizó la evaluación de confianza basada en sus experiencias de satisfacción pasadas y en las evaluaciones de confianza de otros usuarios que compartieron intereses sociales similares. Se consideraron tres tipos de relaciones sociales: la amistad, el contacto social y la comunidad de interés, para la medición de las evaluaciones de similitud y la con-

fianza del filtrado social basado en la similitud social. Además se desarrolló una técnica de filtrado adaptativo a través del cual se determinó que la mejor manera de combinar la confianza directa y la retroalimentación indirecta es de forma dinámica, permitiendo que cada nodo seleccione de forma adaptativa su mejor parámetro de confianza para reducir al mínimo el tiempo de convergencia y el sesgo de confianza. La aplicabilidad del protocolo se demostró utilizándolo en una aplicación de la SC en sistemas de IoT basado en SOA, en donde el resultado fue que el protocolo es capaz de acercarse al óptimo desempeño y supera significativamente al protocolo de selección aleatorio no basado en la confianza.

Cassar et al. (2013) desarrollaron un algoritmo de divide y vencerás sobre la SC semánticos en entornos ubicuos como lo es el IoT. El algoritmo propuesto se utilizó en la ejecución del servicio para dividir repetidamente una solicitud de SC en algunas sencillas sub-solicitudes. El algoritmo se repite hasta que cada sub-solicitud se encuentra con al menos un servicio atómico que cumpla con los requisitos de la sub-solicitud, entonces los servicios atómicos identificados son utilizados para crear una SC. Adicionalmente se evaluó la propuesta con base en un conjunto de solicitudes de servicios compuestos en diferentes dominios incluyendo: negocios, ciudad, comercio, geografía, militar, oficina, tecnología, viajes y el clima, en donde se observó que el algoritmo propuesto realizó eficientemente la descomposición de una solicitud de servicios compuestos de un número de sub-solicitudes y encontró componentes de servicios que atendieron la petición de la SC.

Pang et al. (2015) presentaron una metodología de co-diseño tecnológico de negocios aplicado al diseño de una estación de cuidado de la salud en casa (IHHS). El núcleo de la metodología fue la alineación de tres elementos: 1) el modelo de negocios (MN); 2) la arquitectura de integración de servicios y dispositivos (DSIA); 3) la arquitectura de integración de servicios de información (ISIA). En relación al MN se formuló un ecosistema cooperativo de salud en el IoT mediante la deconstrucción y reconstrucción de la medicina tradicional y las cadenas de valor del Internet móvil, en donde todo está integrado en una nube cooperativa de la salud y extendida a la casa de los pacientes a través de una IHHS. Además para cumplir con los requisitos de ISIA y MN, se utilizaron los principios de diseño de una solución IHHS incluyendo la reutilización de la plataforma 3C, una efectiva SC, entre otros aspectos. Para verificar la propuesta, se desarrolló un prototipo de solución IHHS denominado iMedBox, el cual es una caja de medicina inteligente basada en la metodología propuesta y con un alto rendimiento.

Adicionalmente, en Glova et al. (2014) se demostró la importancia y la utilidad de aplicar un enfoque basado en valores al modelado de negocios de nuevas soluciones basadas en el

IoT. Los autores se centraron en los servicios de atención médica (Tele-HealthCare System para el control de pacientes con diabetes o pacientes ancianos) y se aplicó el valor e3 para demostrar cómo se desarrolla un modelo de negocio sostenible para una plataforma en el IoT y cómo se utiliza el valor del análisis. Además, se consideraron los nuevos objetos de valor y se descubrió cómo la aparición de nuevos objetos de valor invoca a actores nuevos al sistema empresarial para mejorar la sostenibilidad y el rendimiento del modelo.

Además, en Dijkman et al. (2015) se presentó el desarrollo de un marco para el desarrollo de modelos comerciales para aplicaciones en el IoT. El marco se creó a partir de una encuesta bibliográfica sobre marcos de modelos de negocios existentes y posteriormente, los marcos se adaptaron a partir de entrevistas con 11 empresas que desarrollan aplicaciones para el IoT. Finalmente, la importancia relativa de las diferentes partes del marco para las aplicaciones en el IoT se determinó a través de una encuesta a 300 profesionales en el IoT que generó 72 observaciones. Por lo tanto, la contribución de los autores fue un nuevo marco de modelo de negocios para aplicaciones en el IoT que se basó en la literatura, entrevistas y una encuesta entre profesionales del IoT.

Además, Rapti et al. (2015) propusieron un modelo de composición de servicios descentralizado basado en campos de potencial artificial (APF), que condujo el proceso de composición de servicios a través del equilibrio de fuerzas aplicado entre las solicitudes de servicio y los nodos de servicio. Los APF se formaron teniendo en cuenta el porcentaje de servicios solicitados por el usuario, ofrecidos por los nodos de prestación de servicios, así como la disponibilidad de los nodos de servicios. La aplicabilidad del enfoque propuesto se usó en un escenario sobre la composición dinámica y personalizada de un servicio de guía virtual audiovisual en una red del IoT de una feria comercial.

Por otra parte, Vidyasankar (2016) propuso un modelo de transacción y un criterio de corrección para la ejecución de la composición de servicios en el IoT. El modelo definió la atomicidad relajada y las propiedades de aislamiento para transacciones de una manera flexible y se adaptó para una variedad de aplicaciones en el IoT. Además, el modelo se probó en un caso de estudio del sistema de vida asistida en el dominio del cuidado de la salud.

Adicionalmente, Ju et al. (2016) propusieron un marco de modelo de negocios genérico para el IoT a través del análisis de la literatura y entrevistas. Asimismo, se identificaron los elementos esenciales para los modelos de negocios del IoT y se establecieron los componentes básicos de un modelo de negocios en el IoT con base en el lienzo del modelo de negocios. El modelo se probó en algunos casos de estudios de empresas que actualmente están desarrollando productos o servicios para el IoT. Además, el modelo de negocios genéri-

co para el IoT sirvió como punto de partida para cuando los profesionales diseñan y desarrollan modelos de negocios en el entorno del IoT. Es importante que las empresas identifiquen los elementos críticos en su modelo de negocio para crear valor agregado para los servicios en el IoT que se ofrece a los usuarios.

Por otro lado, Chen et al. (2016) presentaron el diseño y análisis de la adaptación y supervivencia de un protocolo de administración de confianza para sistemas en el IoT centrados en el usuario, donde el usuario realizó la evaluación de confianza basada en sus experiencias de satisfacción y en las evaluaciones de confianza de otros usuarios que compartieron experiencias sociales similares. Se consideraron tres tipos de relaciones sociales: la amistad, el contacto social y la comunidad de interés, para la medición de las evaluaciones de similitud y la confianza del filtrado social basado en la similitud social. Además, se desarrolló una técnica de filtrado adaptativo mediante la cual se determinó que la mejor forma de combinar la confianza directa y la retroalimentación indirecta es dinámicamente, lo que permite a cada nodo seleccionar de forma adaptativa su mejor parámetro de confianza para minimizar el tiempo de convergencia y el sesgo de confianza. La aplicabilidad del protocolo se demostró aplicándolo a una aplicación de SC en sistemas del IoT basados en SOA, donde el protocolo es capaz de aproximarse al rendimiento óptimo y excede significativamente el protocolo de selección aleatoria no basado en la confianza.

De nuevo, Baker et al. (2017) presentaron y evaluaron un algoritmo novedoso de composición de servicios en el IoT multi-nube sensible a la energía que genera planes de composición de eficiencia energética al integrar el menor número posible de servicios de proveedores de servicios dispersos en todo el mundo. Además, abordó un problema clave y emergente del IoT que consiste en adoptar enfoques de eficiencia energética para servicios y aplicaciones basados en la nube.

Además, Gierej et al. (2017) propusieron desarrollar el concepto de un modelo de negocios dedicado a las empresas que implementan tecnologías del Internet Industrial de las Cosas. El concepto propuesto se desarrolló para apoyar a las empresas tradicionales en la transición al mercado digital. El estudio se basó en la literatura disponible sobre el impacto que el Internet industrial de las cosas tiene en la economía y los modelos de negocios.

Lee et al. (2015) propusieron una arquitectura de servicios *Web* para entornos de servicios domésticos. Tres capas dan forma a la arquitectura: (1) capa de información, (2) capa de gestión y (3) capa de presentación. La red de superposición de servicios se utilizó en este trabajo para generar una nueva composición de servicios en el contexto de IoT. Adicionalmente, Baker et al. (2017) introdujeron y probaron el algoritmo E2C2 (*Energy Efficient*

multiple Cloud Computing), una composición de servicios de IoT de múltiples nubes con conciencia energética que genera propuestas de composición de eficiencia energética al agregar servicios de proveedores de servicios que están dispersos a nivel mundial. Además, Sun et al. (2019) propusieron un mecanismo de eficiencia energética que optimiza la composición de servicios en el IoT para admitir solicitudes concurrentes. El mecanismo reduce el consumo de energía en la red y mejora el intercambio de servicios en el IoT entre solicitudes concurrentes.

2.2 Orquestación de servicios en el IoT

En Gama et al., (2012) desarrollaron un *middleware* para el IoT llamado *suite* de RFID que se diseñó en una arquitectura de múltiples capas con la presencia en cada etapa de la SOC. Las novedades y contribuciones que se identificaron son: la arquitectura orientada a servicios y la multicapa en el contexto del RFID, el dinamismo y la flexibilidad del enfoque de la SOC en la capa de recolección de datos, el protocolo de flexibilidad introducido en la capa intermedia y el servicio de nombres de objetos orientados al servicio Web. El middleware se probó inicialmente en una aplicación en la industria para el seguimiento y monitoreo de los objetos de la cadena de suministro y luego se extendió a probarse en otros dominios de aplicación más amplios.

Por otro lado, Liu et al. (2012) propusieron una arquitectura de SC basada en la nube para el IoT, adicionalmente se exploraron varias tecnologías claves tales como la semántica ligera de servicio, mecanismos de descubrimiento de servicios basados en el contexto sensible y un modelo de SC adaptado. En la arquitectura de SC basada en la nube, se combinaron tanto la computación en la nube como el IoT. Finalmente el principal objetivo fue apoyar eficientemente a los servicios que utilizan la tecnología de la nube de diferentes tipos de objetos, particularmente en el enfoque del IoT.

Stelmach (2013) presentó una discusión de los escenarios de la SC en el paradigma del IoT, particularmente en el ámbito del transporte, en donde la SC se define con un enfoque holístico. Además se describieron los pasos para el proceso de la propuesta de composición: definición de requerimientos, requisitos de descomposición o agregación haciendo uso de la ontología de dominio, construcción compuesta de la estructura del servicio, el descubrimiento de los servicios, la estructura y la optimización del plan de servicio. Los autores utilizaron la plataforma PlaTel para la composición y la gestión de los servicios de teleco-

municaciones como parte del componente de comportamiento dinámico en el motor de ejecución de servicios compuestos. Finalmente se discutió la aplicación de la SC con diversos escenarios.

Cubo et al. (2014) presentaron la plataforma DEEP para el manejo de la integración y orquestación consciente del comportamiento de dispositivos heterogéneos como servicios, almacenados y accedidos a través de la nube. Adicionalmente se describió un ligero modelo para especificar el comportamiento de los dispositivos para determinar el orden de la secuencia del intercambio de los mensajes durante la composición de dispositivos; también se definió una arquitectura común utilizando un ambiente estándar orientado a servicios para integrar los dispositivos heterogéneos por medio de sus interfaces, a través de una puerta de enlace para orquestarlos de acuerdo a sus comportamientos; por otra parte, se diseñó una plataforma basada en la tecnología de la computación en la nube, conectando la puerta de enlace encargada de la adquisición de datos de los dispositivos con la plataforma en la nube, para acceder remotamente y monitorear los datos en tiempo real en situaciones de emergencia; finalmente se validó la propuesta en un conjunto de enfoques en escenarios reales para su aplicación en un ambiente específico de la vida asistida.

Por otra parte, Li et al. (2014) propusieron un modelo de planificación de la QoS de tres capas para el IoT orientado al servicio: 1) capa de aplicación, 2) capa de red y 3) capa de detección. En la primera capa, el esquema de planificación de la QoS exploró la óptima SC que considera la QoS utilizando el conocimiento de cada uno de los componentes. En la segunda capa, el modelo tuvo como objetivo hacer frente a la programación del entorno de redes heterogéneas. En la tercera capa, se abordó la planificación de adquisición de información y la asignación de recursos a los diferentes servicios. Adicionalmente se consideró la QoS para la arquitectura del IoT orientada al servicio en vista de que fue viable para optimizar el rendimiento de la red de programación del IoT y reducir los costos de los recursos. Además los algoritmos de optimización para la QoS, dependieron de los requerimientos específicos del servicio.

Por otro lado, Tektonidis et al. (2014) presentaron una interfaz del usuario intuitiva para contribuir a aumentar la adopción del IoT y los servicios del contenido del Internet a través de proporcionar un mejor apoyo a los usuarios con discapacidades y deficiencias desde la comodidad de su hogar, proporcionándoles un conjunto de servicios escalables que se ofrecieron gratuitamente o con algún costo a través de alguna manera centralizada de un repositorio de servicios. Los autores presentaron tres escenarios: una casa inteligente, un hospital inteligente y una ciudad inteligente, en donde se plantearon apoyos o requerimientos para

usuarios con alguna discapacidad o deficiencia, planteando la necesidad de proporcionarles servicios de asesoramiento, servicios de alerta, confianza en los servicios, una interfaz accesible e inteligente, orquestación de servicios y servicios a terceros. En este contexto se identificó que es necesario mejorar el modelo de servicios para que se consideren las especificaciones adicionales relacionadas con los problemas de accesibilidad, la semántica para enriquecer los servicios y un mercado de servicios y aplicaciones personalizadas en donde los desarrolladores los publiquen y en donde los usuarios localicen los servicios o aplicaciones de acuerdo a su perfil.

Adicionalmente, en Prince et al. (2014) se presentó la literatura sobre innovación estratégica e innovación de redes para desarrollar nuevas ideas en el área de redes de innovación estratégica. Además, se exploró cómo se organizan las redes de innovación para desarrollar una iniciativa de innovación estratégica en el IoT. También, se identificó cómo un actor central reúne a un grupo de actores diversos para crear primero y luego organizar la red de innovación estratégica mediante el uso de tres estrategias de diálogo: proyección persuasiva, desarrollo reflexivo y control de definición. Además, mostró cómo se establecen diferentes tipos de legitimidad a través de diversas estrategias de diálogo en la organización de redes de innovación estratégica.

Showkat et al. (2014) propusieron un módulo funcional semántico para el usuario y un algoritmo centrado en la SC para la plataforma de la Web de los Objetos (WoO), la cual tiene como objetivo simplificar al objeto y el desarrollo de aplicaciones, el mantenimiento y la operación de la infraestructura del IoT, además ofrece un servicio del IoT centrado en el usuario permitiendo la virtualización del objeto y la SC basados en ontologías semánticas. Además, se diseñó un modelo de ontología para objetos virtuales (ViO), el cual es la representación física de los objetos del mundo real. Las ontologías describieron la relación entre los objetos, servicios y reglas para componer nuevos servicios de forma dinámica. Finalmente se evaluó el trabajo de los autores al compararlo con los enfoques de la SC existentes y los que se encontraban desarrollando la aplicación del usuario con el fin de garantizar la funcionalidad completa.

Además, Swiatek (2015) presentó el desarrollo de la plataforma ComSS que se diseñó para ser un *middleware* para el funcionamiento y el manejo del flujo de servicios compuestos en el paradigma del IoT, en donde su rendimiento depende de diversos factores, incluyendo el número de servicios disponibles, sus instancias y datos, y los formatos de flujo utilizados. La consulta de los servicios compuestos fue simple o se refería a las aplicaciones de servicios complejos que incluyen muchos servicios. En este contexto, el tiempo de la ejecución

de la composición es el indicador clave de rendimiento de una plataforma de este tipo y determina la rapidez con la que se responde y se despliega la aplicación del servicio requerido por un usuario. Dentro de los resultados analizados, se identificó que en la solicitud de estructuras de servicios simples, la plataforma ComSS no mostró un resultado significativo ya que el tiempo de respuesta fue poco, sin embargo, al aumentar el número de nodos o conexiones el tiempo de respuesta aumenta de un segundo a dos minutos.

Por otro lado, Shehu et al. (2015) desarrollaron dos algoritmos evolutivos que llevan a cabo la SC en el IoT considerando la red. El objetivo de los algoritmos fue la búsqueda de servicios compuestos con un costo óptimo, tiempo de respuesta, reputación y la latencia de la red en la QoS. El primer algoritmo VPSO (*Evolutionary Particle Swarm*) utilizó técnicas evolutivas tales como tipo no-dominado y de múltiples poblaciones en su funcionamiento. El segundo es un algoritmo N-Genético o NGA que utilizó un algoritmo de agrupamiento k-means para clasificar los servicios de la IoT en grupos, en función de su distancia de tiempo de ida y vuelta a otros servicios y posteriormente trata de mutar individuos con otros individuos en el mismo clúster. A partir de los resultados de la experimentación se identificó que el NGA fue mejor en términos de calidad de forma física, mientras que el algoritmo VPSO fue mejor en términos de velocidad de cálculo. Además el algoritmo VPSO es más eficiente en comparación con el NGA.

Pisching et al. (2015) presentaron un estudio sobre la SC de fabricación basada en la nube de la industria 4.0, considerando que en este contexto todos los objetos, características y recursos que representaron sus estados, información y modo de operación se consideraron como servicios; en donde los servicios se describieron, publicaron, localizaron e invocaron en una red, la cual dio respuesta a solicitudes entre los consumidores y proveedores de servicios. Adicionalmente se indicó que se espera un aumento del número de miembros en la cadena de producción en un entorno de colaboración, en la que la ubicación sea indiferente y se tenga acceso a todos los recursos a través de sistemas ubicuos desde cualquier lugar del mundo. De la misma manera se señaló que se espera que todos los objetos involucrados en los procesos de producción estén disponibles como servicios y que se comuniquen e intercambien información entre ellos. Las expectativas y la complejidad en la industria 4.0, son el gran número de máquinas interconectadas, productos, procesos y personas que sin duda incrementan los retos y amplían las oportunidades de investigación como la seguridad de los datos, la normalización y la interoperabilidad de los servicios.

Qu et al. (2016) presentaron un modelo de especificación de servicios dinámicos para entidades en el IoT, en este modelo la información del estado de la entidad se emitió en tiempo

real por la estructura extendida y se liberó a los solicitantes como servicios dinámicos. De esta manera, las transacciones en el IoT se construyeron y ejecutaron de forma inteligente según fue necesario. La Web Semántica es una tecnología eficaz para la inteligencia del IoT, donde los WS se utilizan comúnmente para describir las funciones de la entidad en el proceso de la transacción, sin embargo, en una especificación para el procesamiento de la información, los WS convencionales no satisfacen plenamente las necesidades de ejecución y control de las transacciones en el IoT. Finalmente en este modelo se amplió la ontología OWL-S (*Semantic Markup for Web Services*) con el estado del servicio para describir la información tanto de la cola de espera, como el estado actual de las entidades que están involucradas en los servicios, además los resultados experimentales demostraron la eficacia del modelo.

Yu et al. (2016) presentaron una plataforma adaptada para la convergencia del IoT y la WoT, la cual fue esencial en la implementación de redes inteligentes a través de la fusión de elementos dinámicos sin la intervención del usuario. El trabajo de los autores es un nuevo tipo de plataforma que proporcionó inter-compatibilidad para ayudar a los usuarios a comunicarse fácilmente entre sí mediante la conexión a través de redes y además ayudó en la comunicación entre los usuarios y las cosas, mediante la fusión de las cosas conectadas a la Web. La propuesta de los autores garantizó una eficiente plataforma de gestión del IoT y de la WoT, la sincronización adaptable entre las cosas, un entorno de plataforma escalable y la creación de nuevos servicios. Además se validó la propuesta realizando pruebas mediante experimentos que verificaron que las simulaciones fueron satisfactorias.

Por otra parte, Urbieta et al. (2017) presentaron un marco de composición de servicios adaptable que respalda el razonamiento dinámico basado en wEASEL (*contExt Aware web Service dEscription Language*), y un modelo de servicio abstracto que representa servicios y tareas de usuario en términos de su firma, especificación (es decir, condiciones previas sensibles al contexto, condiciones posteriores y efectos) y conversación (es decir, comportamiento con flujo de datos relacionados y restricciones de flujo de contexto). Además, para evaluar el marco de la composición, los autores desarrollaron un nuevo banco de pruebas basado en OWLS-TC4 mediante la combinación de servicios simples y compuestos. La evaluación mostró que el sistema basado en wEASEL realiza una composición más precisa y permite a los usuarios finales descubrir e investigar más oportunidades de composición que otros enfoques.

Por otro lado, Bergesio et al. (2017) propusieron un modelo que se utilizó para definir un conjunto de interfaces estándar adecuadas para cada objeto inteligente. Los dispositivos que

se adhieren al mismo modelo se controlan y colocan fácilmente en relación con ellos, creando comportamientos de objetos múltiples para un espacio inteligente. El modelo se inspiró en la programación orientada a objetos, reinterpretando características como la herencia y el polimorfismo al mundo real, por lo que fue posible proporcionar un sistema de software capaz de adaptar los comportamientos existentes a los nuevos espacios. Además, el uso del modelo se ejemplificó con dos prototipos de espacios inteligentes.

Adicionalmente, Wen et al. (2017) propusieron un marco para orquestar los entornos de computación en la nube, en los que permiten una planificación y optimización adaptables, la composición de la aplicación se gestionó en paralelo bajo una amplia gama de restricciones. Además, se implementó un marco paralelo basado en algoritmos genéticos (GA-Par) en Spark para administrar escenarios de orquestación que involucran la composición de un gran conjunto de aplicaciones de IoT.

Además, Macker et al. (2017) proporcionaron un conjunto de requisitos de flujo de trabajo para aplicaciones grupales en MANETs (*Mobile Ad Hoc Networks*), para respaldar la toma de decisiones descentralizada, para respaldar la comunicación basada en grupo (uno a muchos) y para admitir múltiples transportes diferentes de manera extensible. Además, describió la arquitectura e implementación de Newt que cumple con estos requisitos. Además, aplicó Newt a dos casos de uso: un caso de uso de comunicación basado en un grupo representativo para demostrar que Newt cumplía con los requisitos antes mencionados; y para orquestar Hamlet de William Shakespeare distribuyendo a los actores a través de un entorno inalámbrico emulado, donde los actores intercambiaron sus líneas entre sí.

Por otro lado, Pahl et al. (2018) presentaron un patrón arquitectónico con sus principios subyacentes. El patrón combina la orquestación de bordes en el IoT con un mecanismo de procedencia, que se basa en la cadena de bloques para la administración de orquestaciones de confianza (TOM) en la nube.

Además, Ren et al. (2018) presentaron un modelo de selección de servicios que destaca el efecto de sinergia global basado en los requisitos de colaboración. La validez y las ventajas del modelo y el algoritmo se probaron a través del experimento de simulación de fabricación de autos inteligentes en la nube.

2.3 Coreografía de servicios en el IoT

Dar et al. (2011) desarrollaron un modelo de SC para el paradigma del IoT conformado por dos niveles complementarios: 1) proceso de orquestación local; 2) proceso de coreografía

mundial. Se utilizó la notación del modelado de procesos de negocios (BPMN) 2.0 para definir el esquema de orquestación, el cual se soporta por el lenguaje de ejecución de procesos de negocios para servicios Web (WS-BPEL - Web Services - Business Process Execution Language). Sin embargo en la parte experimental a causa de la escasez de recursos de dispositivos inteligentes, utilizaron REST para el intercambio de mensajes. Para el proceso de coreografía utilizaron las notaciones BPMN 2.0, el cual se traduce además en un código WS-BPEL, además para el proceso de orquestación y coreografía utilizaron la plataforma Eclipse. Finalmente mediante un caso de estudio de un sistema de vivienda asistida enfocado al cuidado de la salud evaluaron la propuesta en un entorno del mundo real.

Por otro lado, Stavropoulos et al. (2013) realizaron una revisión comparativa de los sistemas que realizan la SC en entornos de inteligencia ambiental que cumplen con las directrices de la computación ubicua, utilizando los sistemas de Ambientes Inteligentes (AmI) para la orientación de los servicios, los cuales emplean la tecnología de WS para facilitar la interoperabilidad. A consecuencia de que la SC en los sistemas ambientales se realiza manualmente, lo cual genera una cierta molestia para el usuario, existe la necesidad de desarrollar métodos o mecanismos que automaticen este proceso. Se identificó que la propagación de dispositivos multimedia ha dado lugar a un amplio uso de sistemas multimedia residenciales en AmI tales como oficinas inteligentes, salas de reuniones, la teleconferencia y la salud. Además los enfoques que se adoptaron en los últimos años varían ampliamente en aspectos tales como el dominio de aplicación, el modelado de los servicios, el método de composición, la representación del conocimiento y las interfaces.

Además, Sulistyo (2013) presentó el método AMG (abstraer, modelar y generar) para el desarrollo de sistemas compuestos, en vista de que en el IoT billones de dispositivos dirigidos por software y redes son conectados en Internet, estos dispositivos se comunican y cooperan entre ellos mismos para funcionar como un sistema compuesto. Con el método AMG el desarrollo de aplicaciones de software se realizó de una manera automática, reduciendo el costo y el tiempo de desarrollo. Para la SC, el método AMG consideró solo servicios concretos (servicios en tiempo real). Por otra parte la SC se realizó en un escenario de una casa inteligente, utilizando modelos en niveles de abstracción diferentes, mientras la ejecución de servicios compuestos se generó automáticamente.

Rodríguez et al. (2014) presentaron un nuevo método para implementar una adquisición fusionada de datos distribuidos utilizando un modelo de SC de pesos ligeros, lo que garantizó la exactitud de colaboraciones sin un comportamiento cíclico, permitiendo trabajar con los datos de manera distribuida y descentralizada. Igualmente el método resume la típica

complejidad en los escenarios del IoT de acuerdo a su heterogeneidad de los dispositivos, sin embargo, en vista del al alto nivel de abstracción del método, los desarrolladores utilizaron el concepto de servicio y la interacción entre los servicios para diseñar los escenarios del IoT. Finalmente se validó este trabajo, determinando la predicción del tiempo local a partir de la medición de las condiciones atmosféricas de una ubicación específica, en donde para lograr esto, diversos dispositivos se implementaron proporcionando la información para conocer las condiciones atmosféricas de temperatura, presión y humedad.

Dar et al. (2015) desarrollaron la arquitectura ROA (*Resource Oriented Architecture*), un diseño e implementación de un modelo de arquitectura genérica que proporcionó los componentes básicos para una integración de los sistemas extremo a extremo en el IoT, con especial atención a la corriente de los procesos de negocios (PN) del IoT. La arquitectura fue estándar y compatible para integrar dispositivos con recursos limitados dentro de los procesos basados en notaciones de modelado de procesos de negocios (BPMN). Además la arquitectura se caracterizó por sus interfaces de programación de aplicaciones necesarias para invocar los servicios en el IoT que facilitaron la vida de los programadores en los PN del IoT, un modelo de integración basado en eventos construidos sobre un mecanismo de publicación-suscripción de servicios dinámico en caso de algún fallo de los dispositivos del IoT y la ejecución descentralizada de los PN. El trabajo de los autores es una alternativa para desarrollar pequeños servicios en el IoT los cuales son accesibles fácilmente dentro del entorno de desarrollo de los PN.

En Duhart et al. (2016) se presentó el marco del agente de gestión y supervisión del entorno (EMMA), que se basa en un conjunto de elementos para diseñar la coreografía de arquitecturas distribuidas para entornos receptivos. Los autores utilizaron la arquitectura orientada a recursos (ROA). Adicionalmente, Han et al. (2016) presentaron una explicación de la viabilidad del futuro del IP-IoT completo con protocolos Web en tiempo real y analizó los desafíos de investigación de la composición de servicios. Además, explicó que la composición de servicios en el IoT promueve la idea de reunir servicios de objetos inteligentes de manera novedosa y creativa en los servicios compuestos que se utilizan en múltiples dominios de aplicación para aumentar el poder del IoT. Por lo tanto, se determinó que la composición del servicio es la clave para estimular el desarrollo de futuras aplicaciones en el IoT para construir un mundo más inteligente.

Por otro lado, Chen y Englund (2017) analizaron una plataforma de coreografía para servicios orientados a Internet, que realiza la coreografía de servicios heterogéneos por medio de una síntesis automática de diagramas de coreografía. El enfoque se propuso particularmente

para los Sistemas Cooperativos de Transporte Inteligente (C-ITS), donde los vehículos, la infraestructura y los servicios en la nube están interconectados y cooperan para lograr soluciones de transporte eficientes.

Además, Seeger et al. (2018) presentaron un diseño para la gestión de coreografías dinámicas en el IoT con el objetivo de extender la evaluación y las facetas de implementación para mantener la funcionalidad de los sistemas de automatización de edificios, de modo que los nuevos dispositivos involucrados en la coreografía participen de manera activa o permanezcan inactivos. Así mismo, se utilizó la tecnología de red definida por software (SDN) para reemplazar las ofertas que sufrieron una falla de enlace con otras ofertas que no comparten esos mismos enlaces para la comunicación. Adicionalmente, se propuso ampliar las capacidades de la descripción de la oferta, no solo para describir coreografías de servicios Web, sino también para ofertas de cómputo con características comparables a la tecnología de la computación en la niebla, es decir, para desplegar ofertas de cómputo a una "nube" local y ser capaz de migrar sin problemas dichos cálculos entre nodos. Además, en el nivel de evaluación, se identificó extender las pruebas para abarcar un modelo físico de comunicación para medir la eficiencia energética de un enfoque distribuido para diferentes tecnologías de red subyacentes o una red de malla.

Montali y Plebani (2017) propusieron un enfoque para IoT adoptando la implementación de procesos comerciales en varios dominios de aplicación (logística, fabricación y salud). Además, propusieron el uso de dispositivos inteligentes para interconectarse con las diferentes partes involucradas en un proceso. Además, trabajaron con el IoT Multi-party (IoT Multipartes) que se definió para monitorear y verificar el cumplimiento entre los procesos y dispositivos inteligentes, basado en un diagrama de Modelo y Notación de Procesos de Negocio (BPMN), en el que la notación era la indicada, lo cual permitió mostrar el proceso de una manera simple y clara.

Además, Cherrier et al. (2016) presentaron una arquitectura completa para diseñar aplicaciones en el IoT, así como la propuesta D-LITe que permitió el acceso universal al procesamiento interno de objetos y tener el poder de cómputo. Además, las características de cada objeto que descubrieron se programaron a través de la red, sin ningún acceso físico. Por otro lado, D-LITe llegó con el lenguaje SALT (*Simple Application Logic Description using Transducers*), que se refiere al comportamiento lógico para incluir los objetos del usuario en una solicitud de IoT, la comunicación se basó en la arquitectura REST, reuniendo todas las unidades lógicas en una composición global, y de ese modo, se creó una coreografía de servicios, en la que cada objeto tenía su propia tarea.

Blanc et al. (2016) propusieron el uso de la coreografía de procesos, un paradigma basado en SOA combinado con redes inalámbricas de sensores y actuadores (WSAN) para obtener la virtualización de WSAN y usarlas en el IoT. Los autores propusieron una arquitectura para la integración inalámbrica de la WSAN en el IoT, en la cual la implementación de un motor de coreografía se llevó a cabo con base en trabajos previos que consideraron la implementación de la coreografía. Además, la arquitectura abordó tres requisitos importantes: (1) Virtualización a nivel de nodo, capacidad de priorizar la conexión de Internet a WSAN. (2) La virtualización de la red a través de la conexión de los diferentes elementos de la WSAN a la coreografía. (3) Soporte prioritario de aplicación / servicio.

2.4 Coordinación de servicios en el IoT

Chen et al. (2016) presentaron la ropa inteligente como un sistema innovador de monitoreo de la salud que incorpora las nuevas técnicas de fabricación textil para superar las deficiencias de los dispositivos portátiles tradicionales en las aplicaciones de monitoreo de la salud, como bajo nivel de comodidad, baja precisión, operación compleja y monitoreo inadecuado a largo plazo.

Adicionalmente, Belkeziz y Jarir (2017) presentaron un estudio para describir el comportamiento de las capas de una arquitectura flexible y múltiple para la coordinación de servicios de IoT de manera eficiente. Además, describieron la arquitectura de tres capas: (1) La capa de percepción, que se refiere a diferentes tipos de sensores para identificar objetos en la red, (2) La capa de red, para procesar los datos recopilados de los sensores de la capa de percepción, (3) La capa de aplicación, que se refiere a una combinación entre el IoT y la demanda en la industria. Por otro lado, para lograr la coordinación de los servicios, utilizaron la orquestación o la coreografía de los servicios, sin embargo, esto representó un cierto nivel de complejidad, ya que la coordinación de los servicios representa varios desafíos, como la heterogeneidad, la accesibilidad, el conocimiento del contexto, el descubrimiento y la toma de decisiones.

Además, Giang et al. (2016) presentaron el proceso de desarrollo de aplicaciones en ciudades inteligentes en el IoT desde una perspectiva donde un modelo de coordinación distribuida se supervisó por un grupo de componentes para la construcción de aplicaciones. Asimismo, las aplicaciones se presentaron como un componente clave en el IoT para ciu-

dades inteligentes, lo que permitió una mayor eficiencia y nuevos servicios que mejoran la calidad de vida de las personas.

Cano et al. (2014) presentaron un caso de estudio para el diseño de aplicaciones seguras en el IoT para relacionar la semántica de las reglas de evento-condición-acción (ECA) basadas en los modelos de autómatas utilizados para la verificación y el control de la compilación en tiempo de ejecución basado en reglas. El enfoque se basó en la traducción de los lenguajes de reglas de ECA a un lenguaje reactivo llamado Heptagon / BZR (lenguaje de flujo de datos síncrono usando la extensión BZR). Asimismo, el enfoque se validó en un caso de estudio de una plataforma de sensores y actuadores, donde la integración se describió en la plataforma sensiNact.

Además, Giang et al. (2015) propusieron un modelo de programación de flujo de datos distribuidos (DDF) para el IoT, que utilizó la nube y la infraestructura de la computación en la nube. Del mismo modo, la propuesta se evaluó mediante la implementación de un marco DDF basado en Node-RED (nodo de red distribuido - D-NR), una herramienta de programación visual basada en un modelo de flujo para crear aplicaciones para el IoT. Además, se identificó que el enfoque facilitó el proceso de desarrollo y se utilizó para la construcción de varias aplicaciones en el IoT que operaban eficientemente en la niebla.

Análisis comparativo del estado del arte

Una vez presentados y descritos los diversos trabajos relacionados sobre el IoT, en la Tabla 2.1 se presenta un análisis comparativo, en donde se describen diversos aspectos, los cuales fueron considerados para identificar las similitudes y particularmente las principales diferencias con el mecanismo de composición de servicios en el contexto del IoT que se presenta esta tesis doctoral.

- Autores (es) – Año.- se coloca el primer autor y el año de publicación.
- Trabajo relacionado.- se realiza una descripción del trabajo relacionado.
- Objetos destino o tecnología.- se indican los objetos y/o tecnología utilizados.
- Restricción de recursos.- se indica si los objetos tuvieron restricciones de velocidad de procesamiento y almacenamiento de información.
- Batería eficiente.- se indica si los objetos utilizaron una batería eficiente.
- Datos/Eventos.- se indica si en el trabajo relacionado se utilizaron datos o eventos.
- Asíncrono.- se indica el uso de comunicación asíncrona entre los objetos.

- QoS.- se indica si en el trabajo relacionado se consideró la QoS.
- Wearables.- se indica si en el trabajo relacionado se utilizaron *wearables*.
- Orquestación y/o Coreografía. Indica si el trabajo considera la orquestación y/o coreografía de servicios en el IoT.
- Dominio de aplicación. Indica el dominio de aplicación del IoT con el que se relaciona el trabajo: (1) Industrial; (2) Transportación y logística; (3) Negocios inteligentes / Manejo de productos e inventario; (4) Medio ambiente, agricultura y ganadería; (5) Personal y social; (6) Seguridad y vigilancia; (7) Cuidado de la salud.

Además, en el último registro de la tabla se presentan los aspectos que se consideran en esta tesis doctoral.

El resultado del análisis comparativo indica que la mayoría de los trabajos relacionados no se basan en servicios *Web* tradicionales o *wearables* para la composición de servicios IoT. Así mismo, se identificó que los objetos destino más utilizados son el RFID, WSN, NFC, USN (*Ubiquitous Sensor Network*) y Sensores. Adicionalmente, algunos trabajos consideran el uso de WS y REST, además en la mayoría de los trabajos relacionados existieron restricciones de velocidad de procesamiento y almacenamiento de información, por otra parte no en todos los trabajos relacionados los objetos contaron con una batería eficiente, adicionalmente en todos trabajos relacionados se utilizaron datos o eventos, así mismo solo en algunas propuestas se consideró la comunicación asíncrona y la QoS.

Por otra parte, pocos trabajos relacionados utilizaron un dispositivo *wearable* o sensores, sin embargo, algunos de los dispositivos se fabricaron en un laboratorio de pruebas adaptando algunos sensores de uso básico para el monitoreo y envío de la información de forma local, careciendo de la tecnología con la que cuentan actualmente la nueva generación de dispositivos *wearables* o sensores inteligentes.

Así mismo, son pocos los trabajos que consideraron de manera conjunta la orquestación y la coreografía, sin embargo, por el momento son más los trabajos que consideran la orquestación y pocos los que consideran la coreografía en el paradigma del IoT. Además, la mayoría de los trabajos relacionados presentados anteriormente fueron validados o probados en alguno de los dominios de aplicación del IoT.

Autor (es) - Año	Trabajo relacionado	Objetos destino o tecnología	Restricción de recursos	Batería eficiente	Datos/ Eventos	Asíncrono	QoS	Wearables	Orquestación y/o coreografía	Dominio de aplicación
Dar et al., 2011	Modelo de composición de servicios para el paradigma del IoT, conformado por dos niveles complementarios: proceso de orquestación local y proceso de coreografía mundial.	WS	SÍ	SÍ	SÍ	NO	NO	NO	Orquestación y coreografía	7
Rodríguez et al., 2012	Mecanismo de fusión de datos basado en el modelo de composición de servicios para el IoT, utilizando el *middleware* DOHA y la composición de servicios basada en XML.	RFID	SÍ	SÍ	SÍ	SÍ	NO	NO	No especificado	1
Liu, et al,. 2012	Arquitectura de composición de servicios basada en la nube para el IoT incluyendo la semántica ligera de servicio, mecanismos de descubrimiento de servicios en un contexto sensible y un modelo de composición de servicios adaptado.	RFID	SÍ	NO	SÍ	SÍ	SÍ	NO	Orquestación	3
Gama et al., 2012	*Middleware* para el IoT llamado *suite* de RFID que se diseñó en una arquitectura de múltiples capas con la presencia en cada etapa de la computación orientada al servicio (COS).	RFID NFC	SÍ	SÍ	SÍ	SÍ	NO	NO	Orquestación	1
Boa y Chen, 2012	Protocolo de gestión de la confianza escalable para el IoT, que considera las relaciones sociales y el uso de propiedades para evaluar la confianza: honestidad, cooperación e interés.	NFC	SÍ	NO	SÍ	SÍ	SÍ	NO	No especificado	5
Atzori et al., 2012	Integración de las redes sociales con los objetos del IoT, conocido como el paradigma del Internet social de las cosas (SIoT).	RFID NFC WSN	SÍ	SÍ	SÍ	SÍ	NO	NO	No especificado	5
Stavropoulos et al., 2013	Comparativa de sistemas de composición de servicios en entornos de inteligencia ambiental con directrices de la computación ubicua, utilizando AmI y servicios Web.	RFID	SÍ	NO	SÍ	SÍ	SÍ	NO	Coreografía	5
Cassar et al., 2013	Algoritmo de divide y vencerás sobre la composición de servicios semánticos en entornos ubicuos como el IoT.	RFID	SÍ	SÍ	SÍ	SÍ	NO	NO	No especificado	6
Liu et al., 2013	Algoritmo basado en la evolución cooperativa con optimización en la nube, para la composición de servicios en el IoT impulsado por la calidad del servicio (QoS).	WS	SÍ	NO	SÍ	SÍ	SÍ	NO	No especificado	4
Stelmach, 2013	Discusión de escenarios de la composición de servicios en el paradigma del IoT, en el ámbito del transporte, utilizando la plataforma PlaTel.	WS	SÍ	NO	SÍ	NO	SÍ	NO	Orquestación	2
Zhou et al., 2013	Arquitectura “CloudThings”, un enfoque común para integrar el IoT y la computación en la nube.	RFID WSN	SÍ	SÍ	SÍ	SÍ	NO	NO	No especificado	5
Sulistyo, 2013	Método AMG (abstraer, modelar y generar) para el desarrollo de sistemas compuestos en el IoT, para la comunicación y colaboración entre los dispositivos.	WS	NO	NO	SÍ	SÍ	NO	NO	Orquestación y coreografía	5

Cubo et al., 2014	Plataforma DEEP para el manejo de la integración y orquestación consciente del comportamiento de dispositivos heterogéneos como servicios, almacenamiento y accediendo a través de la nube.	WSN	SÍ	SÍ	SÍ	SÍ	SÍ	NO	Orquestación	7
Showkat et al., 2014	Módulo funcional semántico para el usuario y un algoritmo centrado en la composición de servicios para la plataforma de la WoO.	WSN	SÍ	SÍ	SÍ	SÍ	NO	NO	Orquestación	3
Rodríguez et al., 2014	Método para implementar una adquisición fusionada de datos distribuidos utilizando un modelo de composición de servicios de pesos ligeros en el IoT, permite trabajar con los datos de manera distribuida y descentralizada.	WSN	SÍ	SÍ	SÍ	SÍ	NO	NO	Orquestación y coreografía	4
Tektonidis et al., 2014	Interfaz del usuario intuitiva para aumentar la adopción del IoT y los servicios de Internet a usuarios discapacitados o con alguna deficiencia.	RFID NFC WSN	SÍ	SÍ	SÍ	SÍ	NO	NO	Orquestación	7
Yang y Li, 2014	Estrategia eficiente desde el punto de vista de la selección de los datos sensoriales y la agregación en la composición de servicios de información del IoT.	WSN	SÍ	SÍ	SÍ	SÍ	SÍ	NO	No especificado	3
Li et al., 2014	Modelo de planificación de la QoS de tres capas para el IoT orientado al servicio: 1) capa de aplicación, 2) capa de red, 3) capa de detección.	RFID WSN	SÍ	SÍ	SÍ	SÍ	SÍ	NO	Orquestación	3
Dar et al., 2015	Arquitectura ROA, un diseño e implementación de un modelo de arquitectura genérica que proporciona los componentes básicos para una integración de los sistemas extremo a extremo en el IoT	WSN	SÍ	SÍ	SÍ	SÍ	NO	NO	Orquestación y coreografía	3
Pang, 2015	Metodología de co-diseño tecnológico de negocios aplicado al diseño de una IHHS en el IoT. El núcleo de la metodología es la alineación de tres elementos: MN; DSIA e ISIA.	RFID WSN NFC	SÍ	SÍ	SÍ	SÍ	NO	SI	No especificado	7
Swiatek, 2015	Plataforma ComSS, un *middleware* para el funcionamiento y el manejo del flujo de servicios compuestos en el paradigma del IoT	WSN	SÍ	SÍ	SÍ	SÍ	NO	NO	Orquestación	3
Pisching et al., 2015	Estudio sobre la composición de servicios de fabricación basada en la nube de la industria 4.0.	RFID WSN	SÍ	SÍ	SÍ	SÍ	NO	NO	Orquestación	1
Shehu et al., 2015	Uso de los algoritmos VPSO y N-Genético o NGA para llevar a cabo la composición de servicios en el IoT considerando la red.	WSN NFC	SÍ	SÍ	SÍ	SÍ	SÍ	NO	Orquestación	4
Chen et al., 2015	Diseño y análisis de la adaptación y supervivencia de un protocolo de gestión de la confianza para sistemas del IoT centrados en el usuario.	NFC WSN	SÍ	SÍ	SÍ	SÍ	NO	NO	No especificado	5
Salle et al., 2016	Enfoque biológico para la composición de software y servicios dinámicos y fiables, una herramienta capaz de gestionar la combinación y selección de software y servicios en el IoT.	WS	NO	NO	SÍ	SÍ	NO	NO	No especificado	4
Yu et al., 2016	Plataforma adaptada para la convergencia del IoT y la WoT, esencial en la implementación de redes inteligentes sin la intervención del usuario.	RFID WSN USN	SÍ	SÍ	SÍ	SÍ	NO	NO	Orquestación	5

Qu et al., 2016	Modelo de especificación de servicios dinámicos para entidades en el IoT, haciendo uso de la ontología OWL-S.	WSN	SÍ	SÍ	SÍ	SÍ	SÍ	NO	Orquestación	3
Glova et al., 2014	Servicios de atención médica (Sistema Tele-HealthCare para el control de pacientes con diabetes o pacientes ancianos) y el valor e3 para el modelo de negocio sostenible y una plataforma en el IoT.	WS M2M	SÍ	SÍ	SÍ	SÍ	NO	NO	No especificado	7
Prince et al., 2014	Orquestación de redes de innovación en el desarrollo de una iniciativa de innovación estratégica en torno al IoT.	RFID	SÍ	SÍ	SÍ	SÍ	NO	NO	Orquestación	3
Dijkman et al., 2015	Marco de modelo de negocios para aplicaciones en el IoT a través de un estudio de la literatura, entrevistas y una encuesta entre profesionales.	WS	SÍ	SÍ	SÍ	NO	NO	NO	No especificado	3
Rapti et al. 2015	Modelo descentralizado de composición de servicios basado en APF, que condujo el proceso de composición de servicios a través del equilibrio de fuerzas aplicado entre las solicitudes de servicios y los nodos de servicios.	RFID	SÍ	SÍ	SÍ	SÍ	NO	NO	No especificado	3
Vidyasankar, 2016	Modelo de transacción y un criterio de corrección para la ejecución de la composición de servicios en el IoT.	WS	SÍ	SÍ	SÍ	SÍ	NO	SÍ	No especificado	7
Ju et al., 2016	Marco genérico de modelo de en el contexto del IoT, que identifica elementos y establece componentes básicos de un modelo de negocios en el IoT basado en el lienzo del modelo de negocio.	RFID	SÍ	SÍ	SÍ	SÍ	NO	NO	No especificado	2
Han et al. 2016	Explicación de la viabilidad del futuro del IP-IoT completo con protocolos Web en tiempo real y análisis de los desafíos de investigación de la composición de servicios.	RFID WSN	SÍ	SÍ	SÍ	SÍ	SÍ	NO	Orquestación y coreografía	5
Chen et al., 2016	Administración de confianza adaptable y escalable para admitir aplicaciones de composición de servicios en sistemas del IoT basados en SOA, técnica basada en filtrado colaborativo distribuido, técnica de filtrado adaptativo y análisis contra el algoritmo EigenTrust y el *framework PeerTrust*.	WS	SÍ	SÍ	SÍ	NO	SÍ	NO	No especificado	6
Urbieta et al., 2017	Marco de composición de servicios adaptable que admite el razonamiento dinámico basado en wEASEL y un modelo de servicio abstracto que representa servicios y tareas de usuario en términos de su firma, especificación y conversación.	WS	SÍ	SÍ	SÍ	NO	NO	NO	No especificado	5
Baker et al., 2017	Nuevo algoritmo de composición de servicios multi-nube en el IoT con conciencia energética que genera planes de composición de eficiencia energética al integrar el menor número posible de servicios de proveedores de servicios dispersos en todo el mundo.	WS	SÍ	SÍ	SÍ	NO	SÍ	NO	No especificado	3
Bergesio et al., 2017	Modelo orientado a objetos capaz de describir objetos inteligentes y que es utilizado por un sistema autónomo para ayudar a los usuarios a personalizar un espacio inteligente y para proporcionar una adaptación automática de una "personalización", cuando se mueven a otro entorno.	NFC	SÍ	SÍ	SÍ	SÍ	NO	NO	Orquestación	5

Wen et al., 2017	Marco que organiza entornos de computación en la nube, permite la planificación, la optimización adaptativa y una composición con restricciones que utilizan algoritmos genéticos (GA-Par) en Spark para administrar escenarios de orquestación en el IoT.	WSN	SÍ	SÍ	SÍ	SÍ	NO	NO	Orquestación	3
Macker et al. 2017	Conjunto de requisitos de flujo de trabajo para aplicaciones basadas en grupos en MANET para admitir la toma de decisiones descentralizada, la comunicación basada en grupos (uno a muchos) y transportes diferentes de forma extensible.	WS	SÍ	NO	SÍ	NO	NO	NO	Orquestación	4
Gierej et al., 2017	Concepto de un modelo de negocio dedicado a las empresas que implementan tecnologías del Internet Industrial de las Cosas, desarrollado para apoyar a las empresas tradicionales en la transición al mercado digital y basado en la literatura disponible sobre el impacto que el Internet de las Cosas Industrial tiene en la economía y los modelos de negocios.	WS	SÍ	NO	SÍ	NO	NO	NO	No especificado	3
Duhart et al., 2016	EMMA - basado en un conjunto de elementos para diseñar la coreografía de arquitecturas distribuidas para entornos receptivos.	WSAN REST	SÍ	SÍ	SÍ	SÍ	NO	NO	Coreografía	3
Chen y Englund, 2017	Plataforma de coreografía para servicios orientados a Internet, que realiza la coreografía de servicios heterogéneos por medio de una síntesis automática de diagramas de coreografía.	REST	SÍ	SÍ	SÍ	SÍ	NO	NO	Orquestación y coreografía	2
Seeger et al., 2018	Diseño para la gestión de coreografías dinámicas en el IoT para extender la evaluación y las facetas de implementación.	WS REST	SÍ	SÍ	SÍ	SÍ	NO	SÍ	Coreografía	5
Pahl et al., 2018	Patrón arquitectónico que combina la orquestación en el IoT con un mecanismo de procedencia, que se basa en la cadena de bloques para la TOM en la nube.	RFID Sensores	SÍ	SÍ	SÍ	SÍ	NO	NO	Orquestación	3
Ren et al., 2018	Modelo de selección de servicios que destaca el efecto de sinergia global basado en los requisitos de colaboración.	WSN WS	SÍ	NO	SÍ	SÍ	SÍ	NO	Orquestación	1
Montali y Plebani, 2017	Enfoque para IoT adoptando la implementación de procesos comerciales en varios dominios de aplicación (logística, fabricación y salud). Propusieron el uso de dispositivos inteligentes para la interconexión y trabajaron con el IoT Multi-party (IoT Multipartes).	Sensores	SÍ	SÍ	SÍ	SÍ	NO	NO	Coreografía	1 y 2

Cherrier et al., 2016	Arquitectura completa para diseñar aplicaciones en el IoT, así como la propuesta D-LITe que permitió el acceso universal al procesamiento interno de objetos y tener el poder de cómputo.	WS, REST Sensores	SÍ	SÍ	SÍ	SÍ	NO	SÍ	Orquestación y coreografía	1 y 5
Blanc et al., 2016	Uso de la coreografía de procesos, un paradigma basado en SOA combinado con WSAN para obtener la virtualización de WSAN y usarlas en el IoT, y además la propuesta de una arquitectura para la integración inalámbrica de la WSAN en el IoT.	WS, REST Sensores	SÍ	SÍ	SÍ	SÍ	NO	SÍ	No especificado	5
Chen et al., 2016	Plataforma de servicios para IoT basada en SOA, que permite el diseño de servicios en IoT orientados a eventos y servicios, presentación de un SEDL y de un modelo confiable de distribución de datos en tiempo real para datos sensoriales entre proveedores de información y consumidores.	WS REST	SÍ	SÍ	SÍ	SÍ	NO	SÍ	No especificado	1 y 6
Belkeziz y Jarir, 2017	Estudio para describir el comportamiento de las capas de una arquitectura flexible y múltiple de tres capas para la coordinación de servicios de IoT de manera eficiente.	Sensores	SÍ	SÍ	SÍ	SÍ	NO	SÍ	Orquestación y coreografía	4
Giang et al., 2016	Proceso de desarrollo de aplicaciones en ciudades inteligentes en el IoT desde una perspectiva donde un modelo de coordinación distribuida fue supervisado por un grupo de componentes para la construcción de aplicaciones.	WSN WS REST Sensores	SÍ	SÍ	SÍ	SÍ	NO	SÍ	No especificado	5
Cano et al., 2014	Diseño de aplicaciones seguras en el IoT para relacionar la semántica de las reglas de ECA basadas en los modelos de autómatas utilizados para la verificación y el control de la compilación en tiempo de ejecución basado en reglas.	REST Sensores	SÍ	SÍ	SÍ	SÍ	SÍ	NO	No especificado	5
Giang et al., 2015	Modelo de programación de flujo de datos distribuidos para el IoT, que utilizó la nube y la infraestructuras de la computación en la nube.	WS WSN Sensores	SÍ	SÍ	SÍ	SÍ	SÍ	NO	No especificado	5
Lee et al., 2015	Arquitectura de servicios Web para entornos de servicios domésticos. Tres capas dan forma a la arquitectura: 1) capa de información, 2) capa de gestión y 3) capa de presentación.	WS Sensores	SÍ	SÍ	SÍ	SÍ	NO	NO	Orquestación	5
Baker et al., 2017	Introducción y pruebas del algoritmo E2C2, para una composición de servicios de IoT de múltiples nubes con conciencia energética.	WS	SÍ	NO	SÍ	NO	NO	NO	No especificado	3 y 5
Sun et al., 2019	Mecanismo de eficiencia energética que optimiza la composición de servicios en el IoT para admitir solicitudes concurrentes.	WSN Sensores	SÍ	SÍ	SÍ	SÍ	SÍ	NO	No especificado	5
TESIS DOCTORAL: Mecanismo de Composición de Servicios bajo el enfoque del Internet de las Cosas (IoT)		RFID WSN NFC WS REST	SÍ	SÍ	SÍ	SÍ	SÍ	SÍ	Orquestación y coreografía	5 y 7

Tabla 2.1 Análisis comparativo de trabajos relacionados

Capítulo 3
Aplicación de la metodología

El contenido del presente capítulo se enfoca en describir las etapas que conformaron la metodología de investigación empleada para desarrollar la presente tesis doctoral. La metodología de investigación se basó en el método científico inductivo que comprende cuatro etapas: análisis, diseño, desarrollo y pruebas. El seguimiento de las cuatro etapas permitió alcanzar la solución del problema propuesto.

3.1 Metodología de investigación

Esta sección presenta la metodología de investigación utilizada para el desarrollo de la tesis doctoral la cual se basó en el método científico inductivo y se plasmó en cuatro etapas: 1) Análisis, 2) Diseño, 3) Desarrollo y 4) Pruebas. El seguimiento de cada uno de estas etapas permitió llegar a la solución del problema propuesto: desarrollo de un mecanismo de composición de servicios bajo el enfoque del IoT. A continuación se presenta en la Figura 3.1 la metodología de investigación utilizada para el desarrollo del mecanismo.

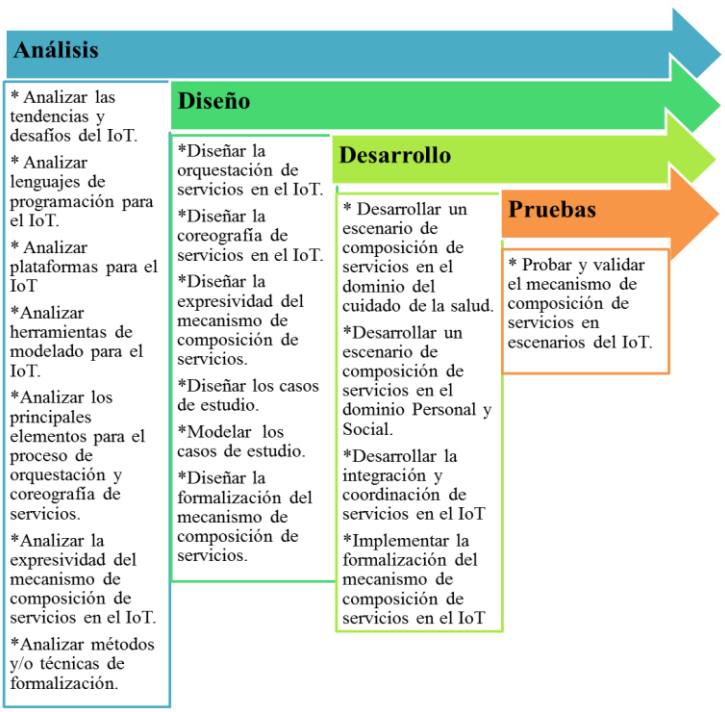


Figura 3.1 Etapas de la metodología de investigación

3.2 Etapa de Análisis

En esta etapa se llevó a cabo el análisis de las tendencias y desafíos del IoT, el análisis y descripción de los lenguajes de programación que se utilizan en el IoT, así mismo se realizó la identificación, clasificación y análisis de plataformas para el IoT. Por otra parte, se realizó el análisis de las herramientas de modelado para el IoT, se describieron los principales elementos para el proceso de orquestación y coreografía de servicios. Además, se describieron los elementos para la expresividad del mecanismo de composición de servicios en el IoT y se analizaron los métodos de formalización.

3.2.1 Análisis de las tendencias y desafíos del IoT

De acuerdo con la revisión del estado del arte, algunos dominios de aplicación comparten desafíos y tendencias para el IoT muy similares en relación a la composición, orquestación y/o coreografía de servicios, especialmente debido a las características que tienen en común sus subdominios correspondientes. En este sentido, se identificó que las soluciones propuestas hasta el momento para el IoT generalmente se validan cuando se implementan en un dominio de aplicación del IoT. Además, la confiabilidad del trabajo aumenta a medida que se garantiza su aplicabilidad e interoperabilidad en múltiples dominios de aplicación del IoT. En este sentido, en la Figura 3.2 se presenta una distribución de las tendencias y desafíos actuales para el IoT de los siete dominios de aplicación del IoT, en donde el dominio de aplicación Personal y Social reporta el mayor número de trabajos (19), mientras que el dominio Seguridad y Vigilancia reporta el menor número (3). Además, se identificaron 7 trabajos en el dominio Industrial, 4 en el dominio de Transportación y Logística, y 16 en el dominio Negocios Inteligentes/Manejo de Productos e Inventarios. Finalmente, se identificaron 6 trabajos en el dominio del Cuidado de la Salud y 6 trabajos en el dominio de Medio Ambiente, Agricultura y Ganadería. Por otra parte, se presenta una discusión para cada uno de los dominios de aplicación del IoT.

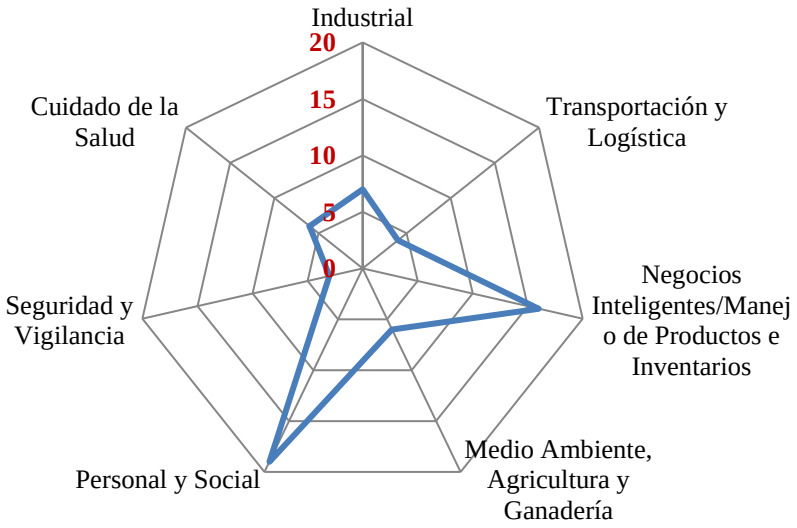


Figura 3.2 Tendencias y desafíos del IoT

Industrial. El rápido crecimiento industrial conlleva la necesidad de colaborar y adoptar nuevas tecnologías del IoT. Hoy en día, las industrias colaboran entre ellas para lograr objetivos comunes y así obtener mejores resultados. Sin embargo, para explotar el potencial del IoT en entornos industriales, la estandarización es importante. La estandarización de los procesos ayuda a resolver el problema actual de la interoperabilidad. Sin embargo, para este fin, las empresas comparten toda la información sobre los productos o servicios que ofrecen, lo que implica un gran desafío en términos de seguridad y confianza.

Transportación y logística. Los sensores se emplean generalmente para monitorear los vehículos de transporte, ya que ofrecen información en tiempo real sobre sus condiciones. En caso de una falla en un vehículo, estos sensores envían una advertencia preventiva al conductor para informarle de la falla detectada, lo que reduce los riesgos de accidentes. Sin embargo, la seguridad sigue siendo un desafío importante a superar en este dominio, ya que hay muchas fuentes de información involucradas. Es decir, generalmente es necesario conjugar la información obtenida del sensor con la obtenida tanto de la infraestructura donde opera el vehículo como de los otros vehículos que están involucrados directa o indirectamente.

Negocios inteligentes / Manejo de productos e Inventario. Los principales desafíos en este dominio están relacionados con la heterogeneidad del hardware que proporciona flexibilidad en términos de uso de recursos debido al intercambio de datos y a la asignación dinámica. Otros dos desafíos son la virtualización logística y la elección del servicio, donde quienes solicitan o brindan un servicio coordinan los datos que se utilizarán en la interacción y ofrecen la opción de un servicio Web que se ejecuta con pautas no funcionales y funcionales. Finalmente, otro desafío importante es aumentar la vida útil de la batería del dispositivo inteligente durante la manipulación del producto, ya que se espera que sea igual a la vida útil de un producto o superior a su fecha de vencimiento.

Medio ambiente, Agricultura y Ganadería. En este dominio de aplicación, el desafío para el IoT es aprovechar todos los dispositivos inteligentes que están estratégicamente instalados en bosques, selvas, valles y campos, entre otros, para prevenir o reducir los efectos adversos de los incendios. El objetivo es enviar advertencias preventivas una vez que comience el fuego para actuar con rapidez y evitar pérdidas de flora o fauna. Otro desafío de las aplicaciones en el IoT es controlar las plagas de los cultivos y controlar el crecimiento y la maduración de los cultivos. Finalmente, con respecto a la ganadería, el desafío es prevenir, monitorear y controlar las enfermedades, el crecimiento y la reproducción del ganado.

Personal y Social. En entornos personales y sociales, se espera que la tecnología 5G impulse aún más la adopción del IoT. En lo que respecta al uso de la Web semántica, el desafío sigue siendo proporcionar un entorno más consciente del IoT. Además, el IoT se combina eficazmente con otras tecnologías, como la computación en la nube y las redes de radio cognitivas, para dar lugar a nuevos servicios en el IoT. Esto va de la mano con la garantía de áreas urbanas sostenibles, que requieren una nueva tecnología que sea eficiente y fácil de usar. También hay problemas en términos de puntualidad, heterogeneidad, escalabilidad, confiabilidad y seguridad que revelan una falta de confiabilidad y una escasez de estándares para la interacción basada en el IoT. Finalmente, es importante superar las fallas de comunicación y garantizar la QoS y la confiabilidad al mejorar las capacidades de reconocimiento de dispositivos de los sistemas nacionales basados en el IoT.

Seguridad y Vigilancia. Las tendencias de integración de la computación en la nube en el IoT traen nuevos desafíos. Los proveedores de servicios rara vez son completamente con-

fiables, ya que tienen acceso a la ubicación e información confidencial de cualquier persona. Además, es más probable que los sistemas sufran ataques cibernéticos como resultado de su vulnerabilidad en términos de seguridad de datos. En este sentido, el desafío a superar por las aplicaciones del IoT basadas en la nube es particularmente importante pero se aborda parcialmente mediante la estandarización de los procesos en el IoT para garantizar la confiabilidad y la seguridad de la información.

Cuidado de la Salud. Las aplicaciones en el IoT se esfuerzan por proteger la seguridad y privacidad de los datos del paciente y abordar los problemas de imprevisibilidad del rendimiento y los problemas legales. Además, al igual que en el dominio de Seguridad y Vigilancia, la integración de la computación en la nube en el IoT implica superar desafíos importantes en términos de confiabilidad, privacidad y seguridad del servicio al mejorar la seguridad de los datos médicos y la disponibilidad y redundancia del servicio. Además, la gran popularidad de los dispositivos de salud inteligentes, como sensores y dispositivos portátiles, revelan la necesidad de confiar en los mecanismos de composición del servicio que aseguran la colaboración y la coordinación del servicio desde múltiples objetos inteligentes que cuidan la salud del paciente en todo momento sin la necesidad de la intervención de los humanos.

En general, no parece haber una diferencia significativa con respecto a la distribución de trabajos en los dominios de la aplicación del IoT, lo que implica que todos los dominios se esfuerzan por aprovechar el potencial del IoT. Sin embargo, es probable que los desafíos actuales en las aplicaciones del IoT, especialmente en los dominios del Cuidado de la Salud, Seguridad y Vigilancia, fomenten una nueva ola de proyectos para el IoT en los próximos años, que sin duda tendrán impacto en diversos aspectos como la salud, la confiabilidad, la facilidad de uso y la seguridad.

3.2.2 Análisis y descripción de los lenguajes de programación para el IoT

La fundación Eclipse realizó un estudio de desarrolladores en el IoT titulado "*IoT Developer Survey 2020*", en donde los lenguajes de programación C, C ++, Java, Python y JavaScript son los lenguajes de programación más utilizados para el desarrollo de soluciones en el IoT, ya que dominan el mercado de aplicaciones en las tecnologías de la información.

Además, el lenguaje de programación C es el número uno para dispositivos restringidos, sin embargo, el lenguaje de programación Java es el número uno en *Edge* y en la nube (Eclipse IoT, 2020). Por otra parte, la Figura 3.3 presenta los lenguajes de programación más utilizados en el IoT, los cuales son: Java, C, JavaScript, Python, C++, PHP, C#, Assembler, Lua, Go, Otros lenguajes, R, Swift, Ruby y Rust (IoT for all, 2018).

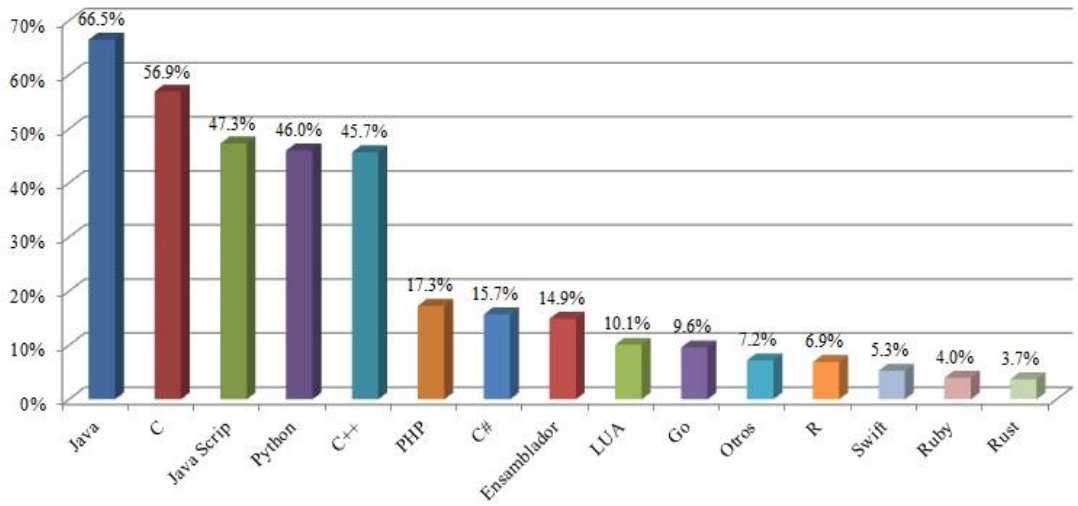


Figura 3.3 Lenguajes de programación utilizados en el IoT (IoT for all, 2018)

Java. Es el conocido lenguaje de programación utilizado por los expertos. Se considera que es la mejor opción para el desarrollo en el IoT, ya que se ejecuta en cualquier lugar y se escribe solo una vez. Así mismo, los desarrolladores que lo utilizan producen y depuran código en su computadora de una manera sencilla. Adicionalmente se transfiere a cualquier chip utilizando la Máquina Virtual de Java (JVM), se ejecuta en lugares donde se utiliza una JVM y en cualquier otra máquina. Java incorpora técnicas de codificación desde los lenguajes como Mesa, Eiffel, C y C++. Por otra parte, Java tiene las capacidades integradas, lo que lo hace orientado a objetos y portátil con la menor dependencia de hardware. Además, Java tiene bibliotecas de soporte de hardware que permiten el acceso al código genérico.

C. Es uno de los lenguajes de programación más importantes en los sistemas del IoT. Se encuentra en la capa más baja de software que está cerca del hardware. C es la base de muchos otros lenguajes de codificación, esto hace que su conocimiento sea una necesidad básica para el desarrollo de proyectos en el IoT. El lenguaje de programación C, no requiere

mucha potencia de procesamiento. Adicionalmente, está disponible en casi todas las plataformas de sistemas integrados avanzados. Además, es procedimental en lugar de orientado a objetos, ya que no tiene capacidades integradas. Este lenguaje de programación es compilado, lo que lo hace ideal para proyectos en el IoT.

JavaScript. Se utiliza como lenguaje de programación en todos los navegadores Web y en el Lenguaje de Marcado para Hipertextos (HTML). Este es un lenguaje de programación que comparte sus bibliotecas con los lenguajes. JavaScript facilita las cosas, ya que hace que los dispositivos sean interoperables. La mayor parte del trabajo se centra en los servidores y centros que recopilan información y luego la almacenan.

Python. Se utiliza principalmente para escribir aplicaciones Web, pero ahora es ampliamente utilizado en los sistema del IoT. Es un lenguaje interpretado que ofrece legibilidad con sintaxis sin comprometer el tamaño. Este lenguaje tiene una gran cantidad de bibliotecas, hace más cosas con menos códigos. La sintaxis de Python es adecuada para la disposición de la base de datos, en el caso de que una aplicación necesite que los datos se ordenen en un formato de base de datos o utilice tablas (Techahead, 2018).

C++. Es un lenguaje de programación orientado a objetos y tiene mayor potencia de procesamiento que C. Adicionalmente, se utiliza como un preprocesador para C para permitirle ejecutar lenguajes de nivel superior. Por otra parte, es uno de los lenguajes de programación favoritos entre los programadores. En los proyectos de Linux más comunes y la programación integrada, permite capas de objetos, abstracciones y capas. Es ideal para los desarrolladores que buscan extender su código de programación para el IoT y código incrustado. Además, C ++ contribuye a la utilización de otros lenguajes, incluidos C #, D, Java y Python, entre otros (Medium corporation, 2018).

PHP. Este lenguaje de programación, está siendo cada vez más utilizado por los desarrolladores en el IoT. Adicionalmente, convierte lo más bajo de Internet en un servidor Web completo. Con la ayuda de PHP, las aplicaciones se desarrollan utilizando los datos del GPS de los dispositivos en el IoT (Techahead, 2018).

C#. Es uno de los leguajes de programación que se dirigen a la *Microsoft Common Language Runtime* (CLR). Los idiomas que se dirigen a CLR se benefician de características como la integración en varios lenguajes y el manejo de excepciones, seguridad mejorada, un modelo simplificado para la interacción de componentes y servicios de depuración y perfilado. De los lenguajes CLR que existen actualmente, C # es el más utilizado para proyectos de desarrollo profesional y complejo que se dirigen a los entornos de escritorio, dispositivos móviles, servidores de Windows y en el contexto del IoT. Por otra parte C# es un lenguaje orientado a objetos, tiene una comprobación de tipo estricta, tanto en tiempo de compilación como en tiempo de ejecución. Además, permite que la mayoría de los errores típicos de programación en C # se informen lo antes posible y sus ubicaciones se identifican con bastante precisión, esto disminuye el tiempo de programación (Toptal, 2018).

Ensamblador. Es lo primero en lo que se piensa cuando se requiere programar pequeños dispositivos. Ensamblador es la piedra angular de la programación en el IoT y es una interfaz directa entre un ingeniero y un dispositivo. Sin embargo, ensamblador es comparativamente difícil de dominar, ya que es un lenguaje de bajo nivel y de hecho, no es un lenguaje en absoluto. Regularmente cualquier tipo de chip en el IoT tiene su propio ensamblador que es diferente de otros, lo que lo convierte en la mejor opción si se desea trabajar directamente con los chips de los dispositivos, sin embargo, es la opción más difícil para la programación en el IoT (Qubit Labs, 2018).

Lua. Es un lenguaje de secuencias de comandos potente, eficiente, liviano e integrable. Admite programación de procedimientos, programación orientada a objetos, programación funcional, programación basada en datos y descripción de datos. Adicionalmente, combina una sintaxis de procedimiento simple con construcciones de descripción de datos potentes basadas en matrices asociativas y semántica extensible. Así mismo, se escribe dinámicamente, se ejecuta al interpretar el código de bytes con una máquina virtual basada en registros y tiene una administración automática de memoria con recolección de basura incremental, lo que lo hace ideal para configuración, creación de *scripts* y creación rápida de prototipos (Lua, 2018).

Go. Compartiendo muchas similitudes con C, Go es un lenguaje de programación incorporado desarrollado por Google. Lo relevante de Go es que es más potente que C y permite

que los dispositivos trabajen juntos para enviar y recibir datos en muchos canales simultáneamente. Sin embargo, todavía hay una desventaja importante, ya que existe una alta posibilidad de pérdida de datos o errores si no se administra correctamente durante la fase de codificación. Se espera que a medida que el lenguaje vaya evolucionando, las cosas cambien en el corto plazo (Medium corporation, 2018).

R. Es un lenguaje y entorno para computación estadística y gráficos. Así mismo, es un proyecto GNU (*Not Unix*) que es similar al lenguaje y el entorno de S desarrollado en los Laboratorios Bell. El lenguaje R se considera como una implementación diferente de S. Hay algunas diferencias importantes, pero gran parte del código escrito para S se ejecuta sin ningún problema en R. Adicionalmente, R proporciona una amplia variedad de técnicas estadísticas (modelado lineal y no lineal, pruebas estadísticas clásicas, análisis de series de tiempo, clasificación, agrupación, entre otras) y técnicas gráficas, además es altamente extensible. El lenguaje S es el vehículo elegido para la investigación en metodología estadística y el lenguaje R proporciona una ruta de código abierto para participar en esa actividad. Por ello, una de las fortalezas de R es la facilidad con la que producen gráficos de calidad de publicación bien diseñados, incluyendo símbolos matemáticos y fórmulas donde sea necesario. Los valores predeterminados reciben gran atención para las opciones de diseño menores en gráficos, pero el usuario conserva el control total. Adicionalmente, el lenguaje R está disponible como software libre bajo los términos de la *Free Software Foundation 's* Licencia Pública General de GNU en forma de código fuente. Además, se compila y se ejecuta en una amplia variedad de plataformas UNIX y sistemas similares (incluyendo FreeBSD y Linux), Windows y MacOS (R, 2018).

Swift. Es el lenguaje de programación que se utiliza para crear las aplicaciones para MacOS o dispositivos iOS de Apple. Si se desea interactuar con los iPhones e iPads con su centro de inicio central, Swift es el lenguaje de programación viable para lograrlo. Swift está ganando más fama como lenguaje de programación que por su procesador Objective-C. Por otra parte, Apple para lograr convertirse en líder en el IoT particularmente en la domótica, está construyendo diversas bibliotecas. Estas bibliotecas manejan una gran parte del trabajo, lo que facilita que los desarrolladores se centren en las tareas específicas, mientras que la plataforma HomeKit se encarga de manejar la integración de los dispositivos (Techahead, 2018).

Ruby. Es un lenguaje de código abierto que está orientado principalmente hacia la programación orientada a objetos, pero también se aplica a proyectos funcionales y de procedimiento. Adicionalmente, Ruby está ganando popularidad en entornos de simulación avanzada, robótica y otros entornos complejos, lo que significa que probablemente tendrá un impacto en el IoT una vez que haya alcanzado un nivel crítico de escala y complejidad. Por otra parte, Ruby se ejecuta en máquinas Windows, Linux, Mac, Solaris y es compatible con numerosos entornos de desarrollo integrado (IDE) que suavizan muchas de las funciones de codificación, depuración y ejecución (Tecnopedia, 2018).

Rust. Es un lenguaje de programación de sistemas de código abierto que se centra en la velocidad, la seguridad de la memoria y el paralelismo. Los desarrolladores están utilizando Rust para crear una amplia gama de nuevas aplicaciones de software, como motores de juegos, sistemas operativos, sistemas de archivos, componentes de navegador y motores de simulación para la realidad virtual (Mozilla, 2018).

3.2.3 Identificación, clasificación y análisis de plataformas para el IoT

Actualmente, más de 400 plataformas para el IoT están disponibles e implementan diferentes estrategias para expandirse y crecer. Por ello, se presenta un análisis comparativo de las 48 plataformas de IoT identificadas en una revisión. Estas plataformas se clasifican en dos grupos: plataformas de código abierto y plataformas propietarias. El análisis comparativo se resume en tablas que contienen la siguiente información:

- Nombre de la Plataforma IoT.
- Dominio de aplicación en el IoT: dominio de aplicación para el cual se propone una plataforma en el IoT.
- Casos de éxito: situaciones específicas en las que se empleó alguna plataforma en el IoT para resolver un problema o mejorar un proceso.
- Lenguajes: los lenguajes de programación con los que es compatible una plataforma en el IoT.
- Comunicación: la forma en que los dispositivos están conectados para transmitir datos.

Plataformas *Open Source*

Las plataformas *open source* se distribuyen y desarrollan libremente. En las Tablas 3.1, 3.2 y 3.3 se presentan los resultados de un análisis comparativo realizado para 20 plataformas *open source* identificadas en la literatura. Como se observa, siete plataformas tienen aplicaciones exitosas en los dominios Industrial y Negocios Inteligentes/Manejo de Productos e Inventarios, en ocho plataformas se identificaron casos de éxito relacionados con los dominios del cuidado de la Salud y el dominio de Transportación y Logística, mientras que en cuatro plataformas se identificaron casos de éxito relacionados con el dominio de Seguridad y Vigilancia. Finalmente, diez plataformas *open source* reportan aplicaciones exitosas en el dominio del Medio Ambiente, Agricultura y Ganadería, mientras que 12 casos de éxito se identificaron en el dominio Personal y Social.

En lo que respecta a los enfoques de comunicación, ocho plataformas *open source* dependen de M2M, mientras que solo dos consideran la comunicación P2P. De manera similar, los enfoques de comunicación P2M y GPaaS (*Gartner Platform as a Service*) solo se identificaron en una plataforma en cada caso. En cuanto a los lenguajes de programación compatibles con las plataformas, se identificó que Java es el más compatible (12 plataformas), y luego le siguen C, C ++ y JavaScript, soportados en ocho plataformas cada uno. Además, C # es compatible con siete plataformas *open source*, Android es compatible con cinco y Objective C con cuatro. Así mismo, tres plataformas se basan en el lenguaje Python y otras dos admiten Ruby, iOS y Swift. Finalmente, Scala, Spring Framework, Matlab, Visual Basic y Groovy solo son compatibles con una plataforma de *open source* cada una.

Por otra parte, se identificó que Kaa es la mejor plataforma *open source* para desarrollar proyectos en el IoT. Kaa es una de las plataformas en la nube más eficientes y con más funciones, ya que administra un número ilimitado de dispositivos conectados, garantiza la interoperabilidad entre estos dispositivos, permite el monitoreo en tiempo real a través de Apache Cassandra y Apache Zappelin, y recopila y analiza datos de sensores. De manera similar, Kaa realiza el aprovisionamiento y configuración de dispositivos remotos, entrega notificaciones específicas y también permite a los usuarios crear servicios en la nube para productos y servicios inteligentes. En este sentido, cualquier persona o empresa tiene una forma gratuita de materializar sus conceptos de productos inteligentes.

Plataforma en el IoT	Dominio de aplicación en el IoT	Casos de éxito	Lenguajes soportados	Enfoque de Comunicación
Kaa, 2017	Industrial	Monitorización remota de línea de producción Seguimiento de la fuerza de trabajo	C C++ Java Objective C	M2M P2M
	Transportación y Logística	Etiquetas inteligentes Visibilidad de extremo a extremo en el proceso de entrega		
	Negocios Inteligentes / Manejo de Productos e Inventario	Pantallas de señalización digital interactiva Análisis de preferencias de clientes		
	Medio Ambiente, Agricultura y Ganadería	Monitoreo y previsión del clima Estadísticas sobre la alimentación y la producción de ganado Mapeo de campos y recursos basado en sensores		
	Personal y Social	Monitorear y analizar las actividades de los dispositivos / usuarios / grupos Configurar eventos para una interacción inteligente		
	Cuidado de la Salud	Monitoreo remoto de las estadísticas de salud del paciente Gestión de activos hospitalarios		
Device hive, 2017	Industrial	Sistema de monitoreo de clústeres OpenStack basado en algoritmos de aprendizaje automático (Canonical, Reino Unido)	Java Python JavaScript C++ C# Scala Ruby Android Objective C Spring Framework	M2M GPaaS
	Transportación y Logística	Sistema Merchmanager (Andrus Logistics)		
	Negocios Inteligentes / Manejo de Productos e Inventario	Solución de reportes y análisis basado en la nube (GuestMetrics)		
	Medio Ambiente, Agricultura y Ganadería	Termostato inteligente		
	Personal y Social	Modelo de predicción del consumo de energía con Spark MLib Ambiente inteligente controlado por voz para hoteles Plataforma Doméstica IoT LeisureLink		
	Seguridad y Vigilancia	Triometric y FUJITSU		
	Cuidado de la Salud	Nova Seek y Zesty Strategic Medical Solutions		
Sofia 2, 2017	Transportación y Logística	Comunicación de transportes Entrega segura de mensajes	Java JavaScript Python Android iOS C# C C++	M2M
	Negocios Inteligentes / Manejo de Productos e Inventario	Gestión inteligente de supermercados Analítica *online* de productos más solicitados		
	Personal y Social	Coruña *Smart City* Social Media, Herramienta de monitorización y escucha activa e inteligente de redes sociales		
	Cuidado de la Salud	SISENS, Sistema de control de estado clínico y evolución del paciente (Televés e Indra) Hogar Digital Asistencial		
OracleIoT, 2017	Industrial	Telefónica España	Java	M2M
	Transportación y Logística	XEROX Seguimiento de activos		
	Negocios Inteligentes / Manejo de Productos e Inventario	Solución de movilidad IFC (avanttic)		
	Personal y Social	Sistema Verisure (Securitas Direct)		
	Cuidado de la Salud	ICO Instituto Catalán de Oncología		
Open.Sen.se., 2017	Transportación y Logística	Sistema de monitorización de bienes en tiempo real para servicio de transporte	JavaScript C C++	M2M
	Medio Ambiente, Agricultura y Ganadería	Sistema de control de calidad de leche Aplicación de aparcamiento inteligente para la ciudad de Londres		
	Cuidado de la Salud	Sistema de medición de presión arterial para pacientes con problema cardíacos		

Tabla 2.1 Plataformas en el IoT *open source* (a)

Plataforma en el IoT	Dominio de aplicación en el IoT	Casos de éxito	Lenguajes soportados	Enfoque de Comunicación
OpenRemote, 2017	Industrial	Plataforma Prodrive	Java iOS Android	M2M
	Transportación y Logística	Control fronterizo de Schiphol (Royal Marechaus) Sistema Beatrix Canal		
	Medio Ambiente, Agricultura y Ganadería	Sistema de alcantarillado de la ciudad de Brujas Cannabis Farm para monitorización de temperatura y humedad		
	Personal y Social	Markthal(Universidad de Eindhoven) Gestión de multitudes con Stratumseind Control de iluminación con Skislope		
	Cuidado de la Salud	Social Alert Sistema de entretenimiento y luces controlado con parpadeo		
Paraimpu, 2017	Medio Ambiente, Agricultura y Ganadería	Control de luces urbanas	C C++ JavaScript	M2M
	Personal y Social	Sistema de monitorización del consumo de energía Tlight		
	Cuidado de la Salud	Recordatorio para tomar medicinas		
SensorCloud, 2017	Industrial	Vehículos aéreos no tripulados	Python Java C#	M2M
	Medio Ambiente, Agricultura y Ganadería	Agricultura de precisión en cualquier escala		
	Seguridad y Vigilancia	Minería inteligente Monitoreo de máquinas en la Industria de Petróleo y Gas		
ThingSpeak, 2017	Industrial	Conteo de autos y control de tráfico	C, Java, MATLAB, C#, JavaScript, Ruby y Python	M2M
	Medio Ambiente, Agricultura y Ganadería	Sistema meteorológico MathWorks Alertas de marea con Twitter en tiempo real		
	Personal y Social	Medición del consumo de energía (Cadmus)		
Microsoft IoT, 2017	Personal y Social	Aplicación de alertas de los dispositivos Z-Wave Aplicación para el control de cámara de video Aplicación para el control de luces SoftUPS: Virtualización de la solución de UPS en el hogar para permitir un uso eficiente de energía Instituto de Ciencia y Tecnología Daegu Gyeongbuk (DGIST) Universidad de Maryland, Baltimore Colegio Universitario de Londres	C#	M2M
	Seguridad y Vigilancia	Vigilancia Digital de Vecindarios		
	Cuidado de la Salud	Aplicación para el cuidado de personas con discapacidad SoundChoice, un sistema de reloj inteligente que traduce los sonidos en el hogar a señales táctiles y visuales Monitorización de la movilidad entre los adultos mayores de la comunidad Sistema basado en Kinect para pacientes con enfermedad de Parkinson SOLACE, Sistema para el cuidado de adultos mayores		
InfoBright, 2017	Industrial	Mavenir Systems Viavi Solutions	No especificado	M2M
	Negocios Inteligentes / Manejo de Productos e Inventario	Fuseforward Optimización de la gestión de cadenas de suministro (IntegriChain) Polystar		
	Cuidado de la Salud	Sistema para el tratamiento de enfermedades cardiacas (Preventice Solutions)		

Tabla 3.2 Plataformas en el IoT *open source* (b)

Plataforma en el IoT	Dominio de aplicación en el IoT	Casos de éxito	Lenguajes soportados	Enfoque de Comunicación
2lemetry, 2017	Transportación y Logística	Automatización de envíos de mercancía	JavaScript C++	M2M
	Negocios Inteligentes / Manejo de Productos e Inventario	Sistema de análisis y gestión de cliente complejos		
	Medio Ambiente, Agricultura y Ganadería	Sistema para la gestión de propiedades y medición de agua		
Carriots, 2017	Transportación y Logística	Transporte inteligente en Boyacá Cadena de Suministro Inteligente	Java Groovy JavaScript Android	M2M
	Negocios Inteligentes/ Manejo de Productos e Inventario	Venta Inteligente de Cerveza y Dispensadores Automáticos Inteligentes Cartelería Digital Inteligente Seguimiento del Suministro de Repuestos		
	Medio Ambiente, Agricultura y Ganadería	Pozuelo Smart City SINTELUR (Wairbut)		
	Personal y Social	Ascensores Inteligentes Sistema Inteligente de Calefacción, Ventilación y Aire Acondicionado		
	Seguridad y Vigilancia	Energy Smart Generation (Unatec) Gas Natural Unión Fenosa y CETASA Gestión Inteligente de Tuberías y Seguimiento Inteligente de Recursos		
NanoService, 2017	Personal y Social	Aplicación de iluminación inteligente	Python Java	M2M
NewAer, 2017	Personal y Social	Aplicación para mensajes personalizados interactivos que atraigan a los visitantes con recompensas, información o participación en tiempo real Aplicación para intercambiar archivos entre cualquier computadora portátil, *smartphone* o tableta	Objective C Swift Java	P2P
Nimbits, 2017	No especificado	No especificado	Java	M2M
AllJoyn, 2017	Personal y Social	Sistema de control de puerta de garaje Sistema de automatización del hogar y del exterior	C C++ Java JavaScript Android Objective C Swift C# Visual Basic	P2P
HarvestGeek, 2017	Medio Ambiente, Agricultura y Ganadería	Sistema de monitorización y automatización de invernaderos de código abierto e inalámbrico para jardines y granjas	C C++	M2M
XOBXOB, 2017	No especificado	No especificado	Java	M2M
Contiki, 2017	No especificado	No especificado	C	M2M

Tabla 3.3 Plataformas en el IoT *open source* (c)

Adicionalmente, Kaa permite la administración de datos para objetos conectados y su infraestructura de *back-end* al proporcionar los componentes de servidor y SDK de punto final. Además, los usuarios integran cualquier plataforma en particular, mientras que para la recopilación de datos, Kaa utiliza los protocolos MQTT, CoAP, XMPP, TCP y HTTP. Por otro lado, las plataformas ThingSpeak y DeviceHive son otras dos plataformas en el IoT que se vuelven cada vez más populares gracias a sus características. Por un lado, los usuarios de ThingSpeak analizan y visualizan datos en MATLAB sin tener que comprar una licencia de Mathworks. Del mismo modo, la plataforma recopila y almacena datos de sensores en la nube y desarrolla aplicaciones en el IoT, recopila datos en canales privados, comparte datos con canales públicos tales como API RESTful y MQTT, envía cartas y programa eventos. Por otro lado, DeviceHive es una plataforma con diversas características distribuidas bajo la licencia Apache 2.0, fácil de usar y cambiar, así mismo proporciona opciones de implementación de *docker* y *kubernetes*. Además, DeviceHive se descarga y utiliza tanto en la nube pública como en la privada y también se escala desde una sola máquina virtual al clúster de nivel empresarial. Finalmente, DeviceHive se conecta a cualquier dispositivo o tablero de *hackers* a través de la API REST, WebSockets o MQTT.

Plataformas propietarias

Las plataformas propietarias ofrecen servicios que son accesibles después de pagar por una licencia. En las Tablas 3.4, 3.5, 3.6, 3.7 y 3.8 se presenta el análisis comparativo de las 28 plataformas propietarias de IoT identificadas en la literatura. Así mismo, en las tablas se muestran los dominios de aplicación y los casos de éxito identificados en cada uno de ellos. En este sentido, el resultado del análisis indicó que en 14 plataformas se implementaron casos de éxito en el dominio industrial, mientras que en 16 plataformas se encontraron casos de éxito en el dominio de aplicación Transportación y Logística. Además, en tres dominios se identificaron 12 casos de éxito de la implementación de las plataforma en el IoT: el dominio Negocios Inteligentes / Manejo de Productos e Inventario, el dominio de Seguridad y vigilancia, y el dominio del cuidado de la Salud. Adicionalmente, se identificó que diez plataformas propietarias presentaron casos de éxito en el IoT en el dominio del Medio ambiente, agricultura y ganadería, además que se encontraron 13 casos de éxito para el dominio de aplicación Personal y social. Por otro lado, en cuanto a los enfoques de comunicación, 26 plataformas se utilizan el enfoque M2M y solo dos el enfoque B2B.

Plataforma en el IoT	Dominio de aplicación en el IoT	Casos de éxito	Lenguajes soportados	Enfoque de Comunicación
Xively (Pachube), 2017	Industrial	SATO Monitorización en tiempo real de impresoras para detección de problemas de impresión (SATO)	Android Java C Objective-C Python PHP Ruby JavaScript	M2M
	Medio Ambiente, Agricultura y Ganadería	Aplicación Farmhand Connect para supervisión del crecimiento de plantas(Freight Farms) Sistemas Halo y Halo + para la extracción de datos meteorológicos y envío de alertas en caso de desastres naturales (Halo Smart Labs) Termostato SunStat Connect (Watts Water) Aceleración de la investigación del ADN con congeladores de enzimas de autoservicio (New England Biolab) Jardín de Cocina NATUFIA para el control de cultivos de forma remota		
	Personal y Social	Calentador de líquidos con energía eléctrica (Heatworks Technologies) Control inalámbrico de iluminación (Lutron Electronics)		
	Seguridad y Vigilancia	Aplicación Notifo para el envío de notificaciones en tiempo real al teléfono móvil		
	Cuidado de la Salud	Aplicación SureFlap para el cuidado de las mascotas		
Axeda, 2017	Transportación y Logística	Gestión de inventario a través de una red global, creación de tipos de ubicación, pronóstico de flujos de demanda y automatización del proceso de reposición de existencias (Aeronáutica Embraer)	Groovy Java	M2M
	Seguridad y Vigilancia	Sistema para el mantenimiento de vagones de ferrocarril y locomotoras sobre una base de 24/7 (NedTrain)		
mnubo, 2017	Negocios Inteligentes / Manejo de Productos e Inventario	Refrigerador inteligente de cerveza y reabastecimiento predictivo(Buzz Connect) Optimización de servicios y aumento de las ventas en Smart Product Manufacturer (SPM)	Python JavaScript Java iOS C# Android	M2M
	Medio Ambiente, Agricultura y Ganadería	Rendimiento de cosechas con análisis en tiempo real de las condiciones óptimas de humedad y tensión del suelo para los productores (Hortau)		
	Personal y Social	Retroalimentación inteligente del uso doméstico de una casa (Icontrol Networks) Análisis de la utilización y las tendencias de consumo de energía (CaSA)		
BUGswarm, 2017	Industrial	Monitorización básica de hardware (BUGstats)	Python JavaScript	M2M
	Negocios Inteligentes / Manejo de Productos e Inventario	HNTR una aplicación para lista de compras compartida		
Digi (Etherios), 2017	Transportación y Logística	Enrutamiento de mensajes de control de trenes y las comunicaciones inalámbricas, para el mantenimiento del sistema, la configuración y la gestión de la red a través de celular (SEPTA)	Python Java Android C JavaScript	M2M
	Negocios Inteligentes / Manejo de Productos e Inventario	Estación de carga inalámbrica de baterías de teléfonos inteligentes (Powermat)		
	Medio Ambiente, Agricultura y Ganadería	Sistema de gestión remota de las operaciones de secado de granos para el desarrollo de negocios (Nebraska Engineering Company)		
	Personal y Social	Redes de energía inteligente (IDigi Energy Base) Monitorización de almacenamiento de sisternas (IDigi Tank) Aplicación para la detección de problemas de mantenimiento de bombillas a través de alertas móviles (Enlight) Gestión de paneles solares bifaciales para el ahorro de energía en alumbrado público (Mirabella Energy) Gestión de energía de hoteles basada en sensores infrarrojo (Lodging Technology)		
	Seguridad y Vigilancia	Sistema de seguridad alimentaria (Pizza Ranch) Sistema de alojamiento directo de huéspedes conectado a puertas inteligentes de habitaciones (DIRECT-IN)		
	Cuidado de la Salud	Sistema de monitorización en el hogar de pacientes y traslado a centros de atención médica a través de Internet (Orange Business Services) Seguimiento de lavado de manos a través recordatorios de voz de sensores montados en dispensadores de jabón y alcohol (Clean Hands Safe Hands)		

Tabla 3.4 Plataformas en el IoT propietarias (a)

Plataforma en el IoT	Dominio de aplicación en el IoT	Casos de éxito	Lenguajes soportados	Enfoque de Comunicación
ioBridge, 2017	Negocios Inteligentes / Manejo de Productos e Inventario	Optimización en la fabricación de cantidades limitadas de cerveza (Small Batch)	iOS Android	M2M
	Personal y Social	CheerLights un sistema de sincronización del color de luces de navidad mediante Twitter		
	Seguridad y Vigilancia	Sistema de seguimiento de nieve automatizado e inalámbrico (Laurel Highlands Snowmobile Club)		
EVRYTHNG, 2017	Industrial	Plataforma operativa para proporcionar a los fabricantes de iluminación un marco orientado a servicios para desarrollar aplicaciones IoT (Gooee)	JavaScript C C++ Java PHP	B2B
	Transportación y Logística	Aplicaciones de ciclismo urbano inteligente (Brompton)		
	Negocios Inteligentes / Manejo de Productos e Inventario	La botella inteligente JohnnieWalker Blue Label para el control de ofertas promocionales mediante la tecnología NFC (DIAGEO) Conexión digital de productos físicos a la Web para impulsar las ventas y empaquetado inteligente para activar productos como medios digitales (*The Coca-Cola Company)* Comercio electrónico de productos inteligentes (MARS)		
	Personal y Social	Enchufe inteligente iSP5 SmartPlug para controlar aparatos menores de 1800 volts como: luces, aire acondicionado, ventiladores y calentadores portátiles (iHome)		
ThingWorx, 2017	Transportación y Logística	Análisis de seguridad del tráfico y servicios de minería de datos (All Traffic Solutions)	Java C# C iOS Android	M2M
	Negocios Inteligentes / Manejo de Productos e Inventario	Soporte a sistemas de kioscos ubicados en entornos comerciales para actualización de software de forma automática (ecoATM)		
	Medio Ambiente, Agricultura y Ganadería	Detección de problemas en los cultivos y visualización de activos agrícolas mediante mapas (OnFarm) Aplicación para el cuidado de los cultivos de almendras (Z-Farms)		
	Seguridad y Vigilancia	Conexión entre sistemas de fabricación, calidad, mantenimiento y ERP, y para la creación de tableros de mandos y aplicaciones interactivas para el soporte de decisiones basadas en roles (ATI Specialty Materials) Ampliación del nivel de oferta de servicio y soporte para cajeros automáticos de forma remota (Diebold) Conectividad segura de equipamiento médico para proporcionar servicio y soporte (Sysmex)		
	Cuidado de la Salud	Reducción de la tasa de readmisiones para los pacientes con cardiopatía isquémica		
GroveStreams, 2017	Transportación y Logística	Sistema para el rastreo de paquetes y transporte Control de mantenimiento de motores y programación de servicios	Python Java	M2M
	Negocios Inteligentes / Manejo de Productos e Inventario	Verizon		
	Medio Ambiente, Agricultura y Ganadería	Monitorización de la salud y el bienestar animal en el ambiente del granero Monitorización agrícola, ambiental y de investigación (SenseTerra) Mejoramiento de la rentabilidad en los sistemas de producción animal y sistemas de producción de forraje (Feedworks)		
	Personal y Social	Sistema de alertas y detección del uso excesivo de energía para optimizar el consumo Sistema de control y detección de fugas de agua en lugares remotos		
	Seguridad y Vigilancia	Sistema de vigilancia de horarios de llegada y salida de empleados		
	Cuidado de la Salud	Aplicación de alertas basado en indicadores de salud críticos Optimización del rendimiento de la cadena de suministro mediante el monitoreo del uso de consumibles en las ubicaciones de los pacientes		

Tabla 3.5 Plataformas en el IoT propietarias (b)

Plataforma en el IoT	Dominio de aplicación en el IoT	Casos de éxito	Lenguajes soportados	Enfoque de Comunicación
Zatar, 2017	Industrial	Sistema de automatización y monitorización para fábricas	Objective C Swift C C++ Python JavaScript	M2M
	Transportación y Logística	Sistema de monitorización para analizar datos precisos en tiempo real sobre los activos en cualquier punto de una cadena de suministro		
	Negocios Inteligentes / Manejo de Productos e Inventario	Sistema de recomendaciones para una tienda de vinos que permita mostrar sugerencias de complementos a los clientes Sistema de monitorización de refrigeradores en tiempo real para proveedores de productos perecederos		
	Cuidado de la Salud	Aplicación para seguimiento de pacientes con infarto agudo de miocardio (IAM) para planificar e informar a los cardiólogos intervencionistas y otros médicos sobre el estado de salud paciente en todo momento		
Yaler, 2017	Industrial	Recuperación, procesamiento y análisis de datos de cualquier tipo de dispositivo o red	C,C++, Java, Python, JavaScript, Android,iOS	M2M
	Transportación y Logística	Monitorización en tiempo real de dispositivos para facilitar su implementación en múltiples ubicaciones (decentLab)		
	Cuidado de la Salud	Desarrollo de una solución de tele-audiología (Phonak)		
SAP, 2017	Industrial	Aplicación para analizar datos de Trenitalia para mejorar los procesos de mantenimiento Aplicación móvil Dommelstroom para analizar la productividad de centrales hidroeléctricas pequeñas	JavaScript Swift Java	M2M
	Transportación y Logística	Sistema de logística portuaria inteligente del Puerto de Hamburgo para aumentar la capacidad y aumentar la eficacia general Aplicación para analizar e integrar datos telemáticos automotores para mejorar los servicios y crear nuevos modelos y oportunidades de negocio.		
	Negocios Inteligentes / Manejo de Productos e Inventario	Aplicación para ofrecer promociones en tiempo real de subastas y ventas cruzadas para personalizar la experiencia del usuario y promover la fidelidad a la marca		
	Medio Ambiente, Agricultura y Ganadería	Sistema de prevención de inundaciones en tiempo real con sensores inteligentes Plataforma Vivaldi para la evaluación y cobertura de riesgos de índices climáticos para agricultores		
	Seguridad y vigilancia	Tablero para visualizar los datos analíticos en tiempo real de sensores de un auto de carreras		
Arrayent, 2017	Industrial	Sistema de comunicación y gestión de lavanderías basado en la web para aumentar los beneficios de los propietarios de equipos (Maytag)	Android Objective C Swift	M2M
	Personal y Social	Sistema de gestión de puertas de garaje (Chamberlain) Sistemas de gestión de luz para uso residencial y comercial (OSRAM y SYLVANIA) Aplicación móvil para administrar y operar un termostato para mejorar la usabilidad y capacidades de ahorro de energía de los sistemas de calefacción y aire acondicionado (SALUS Controls)		
Sine Wave, 2017	Transportación y Logística	Sistemas de control de tráfico terrestre Sistemas de gestión de buques	C Objective C Swift JavaScript	M2M
	Seguridad y Vigilancia	Sistema de seguridad para monitorizar y apoyar las operaciones mineras subterráneas		
Ayla Networks, 2017	Industrial	Administración y mantenimiento de Sistemas de purificación de agua	Ruby Objective C	M2M
	Personal y Social	Automatización del sistema de aire acondicionado Sistema de alertas y control de incendios en viviendas		
	Seguridad y Vigilancia	Sistema de iluminación y seguridad para empresas comerciales Sistema de supervisión y control de propiedades de forma remota		
	Cuidado de la Salud	Aplicación de alertas Owlet Smart para monitorización de la frecuencia cardiaca, la calidad de sueño, el oxígeno en la sangre y la temperatura de un bebé		
arm Mbed, 2017	No especificado	No especificado	C, C++ , JavaScript, Python, Shell	M2M

Tabla 3.6 Plataformas en el IoT propietarias (c)

Plataforma en el IoT	Dominio de aplicación en el IoT	Casos de éxito	Lenguajes soportados	Enfoque de Comunicación
Echelon, 2017	Industrial	Monitorización y mantenimiento de los sistemas de sonido para obtener un mejor rendimiento	PHP	M2M
	Transportación y Logística	Sistema de automatización de frenado de trenes muy largos de mercancías Implementación de la programación dinámica para los sistemas de pasajeros de trenes		
	Negocios Inteligentes / Manejo de Productos e Inventario	Sistema de seguimiento de activos en minibares de un hotel		
	Personal y Social	Sistemas de iluminación adaptable para ahorro de energía Sistema de iluminación pública para aumentar la seguridad de los ciudadanos		
	Cuidado de la Salud	Sistema de localización de recién nacidos para evitar robos Sistema de rastreo de pacientes con demencia o trastornos neurocognitivos para evitar extravíos		
Exosite, 2017	Industrial	Aplicación Parker para optimizar y mejorar el rendimiento de un sistema de aire comprimido	Python Erlang JavaScript Go C C++	M2M
	Personal y Social	Sistema de control de acceso remoto en tiempo real Aladdin Connect para el historial de actividad de puertas de garaje Sistema de notificación remota Victor Kill-@lert para el seguimiento del rendimiento de trampas de ratones		
Marvell, 2017	Industrial	Sistema avanzado de asistencia al conductor para la prevención de colisiones y detección de puntos ciegos Sistema de sincronización de audio de habitaciones con mínimos requerimientos energéticos	C C++ Python	M2M
	Transportación y Logística	Aplicación para el control de congestión urbana		
	Seguridad y Vigilancia	Sistema de vigilancia para la detección de movimiento en video y gestión de alertas		
arkessa, 2017	Transportación y Logística	Sistema de monitorización de tiempos de viaje y obras viales para mejorar la seguridad de los conductores y de los trabajadores de construcción Plataforma Tracknstop para la monitorización y rastreo en tiempo real de vehículos Sistema Jenotik para el control de flujo de tráfico y la reducción de accidentes Sistema inteligente Clearview para la gestión de estacionamientos	PHP Java C C++	M2M
Wovyn, 2017	Negocios Inteligentes / Manejo de Productos e Inventario	Sistema de seguimiento de los activos de un restaurante	C C++ JavaScript	M2M
	Medio Ambiente, Agricultura y Ganadería	Sistema inteligente de gestión de invernadero		
	Seguridad y Vigilancia	Sistema de seguridad para propiedades en renta		
	Personal y Social	Sistema automático de iluminación y ventilación para el hogar		
InterDigital, 2017	Industrial	Sistema de monitorización del estado estructural para facilitar el mantenimiento predictivo y preventivo para las industrias de gas y petróleo	No especificado	M2M
	Transportación y Logística	Sistema integral para una agencia de autopistas		
	Medio Ambiente, Agricultura y Ganadería	Sistema de monitorización ambiental de la calidad del aire, agua y manejo de desechos		
Wind River, 2017	Industrial	Sistemas de control para refinería y planta de energía Airbus Group; BAE Systems; Boeing; BMW y General Motors	C, Perl, C++, Python	M2M
	Cuidado de la Salud	Sistema de monitorización de pacientes y medicina predictiva		
Linkafy, 2017	Personal y Social	Aplicación móvil para el control de aparatos electrodomésticos	PHP JavaScript	M2M

Tabla 3.7 Plataformas en el IoT propietarias (d)

Plataforma en el IoT	Dominio de aplicación en el IoT	Casos de éxito	Lenguajes soportados	Enfoque de Comunicación
JasperIoT, 2017	Industrial	Sistema de servicios de limpieza automática de vehículos a través de teléfono inteligente (Daimler) Aplicación de arranque de vehículos, bloqueo/desbloqueo remoto y localización (Ford)	JavaScript Python C C++	M2M
	Transportación y Logística	Sistema para la gestión de conectividad y análisis del comportamiento del conductor, diagnóstico del motor y ubicación del vehículo (Alamo) Sistema para el seguimiento en tiempo real de los vehículos conectados y activos de tránsito (Enterprise) Sistema de rastreo en tiempo real de vehículos para aumentar la eficiencia operativa y acelerar los procesos de recolección y entrega (DHL) Sistema de seguimiento y recuperación de automóviles robados (Guidepoint) Sistema de visibilidad de tráfico, clima y precios de combustible (TomTom) Sistema de gestión de diagnóstico remoto para optimizar el rendimiento de aviones y equipos de carga conectados (Virgin Airlines)		
	Negocios Inteligentes / Manejo de Productos e Inventario	Plataforma para ofrecer contenido de libros electrónicos y acceso a internet en cualquier lugar (Amazon Kindle) Aplicación de localización de estudiantes e instructores de pistas de esquiadores (Flaik) Sistema de monitorización de la calidad de bebidas y nivel de inventario en barriles en lugares públicos (Heineken) Aplicación de conectividad móvil para garantizar el tiempo de actividad continuo para apoyar las transacciones de la tienda en caso de que no haya red (Starbucks)		
	Medio Ambiente, Agricultura y Ganadería	Sistema de seguimiento y gestión automatizada del ganado para aumentar ingresos a agricultores y mejorar la salud de los animales (Litams) Sistema para monitorización del estado de salud y actividades de parto ganadero (Motech) Sistema de gestión del agua para el ahorro de la misma y la reducción de costos operativos (Observant) Sistema de riego y control de plagas y heladas para cultivos (Semios)		
	Seguridad y Vigilancia	Sistema de redes de monitorización de alarmas y seguridad (SCSI) Sistema de seguridad en el hogar (Vivint) Solución de seguridad, señalización de alarmas y control remoto (WebWay)		
	Cuidado de la Salud	Sistema de llamadas de enfermeras 7/24 para uso en hospitales, casas hogares de ancianos y centros similares de cuidados intensivos (Acetek Systems) Aplicación de monitorización de pacientes con marcapasos (Boston Scientific) Sistema inteligente para el cuidado de la salud de pacientes con Parkinson (Great Lakes) Sistema integral de salud y servicios de emergencia (Jupl/Samsung)		
IOTA, 2017	No especificado	No especificado	JavaScript, Python, Java, Go,C	B2B
Sequans, 2017	Transportación y Logística	Sistema de seguimiento de mercancía en todas las etapas de la entrega (Monarch)	C C++ JavaScript Python	M2M
	Seguridad y Vigilancia	Sistema de seguridad doméstico con un consumo de energía eficiente		
	Personal y Social	Sistema de medición de gas y agua para uso doméstico		
	Cuidado de la Salud	Sistema de monitorización del ritmo cardiaco y signos vitales de pacientes		
BoschSI, 2017	Transportación y Logística	Sistema de control de calidad del plátano para garantizar que llegue en óptimas condiciones a su destino Sistema de monitorización de trenes de carga para predicción de tiempos de entrega	Java C Ruby Scala	M2M
	Medio Ambiente, Agricultura y Ganadería	Sistema para el control de temperatura de cultivos de espárragos Aplicación para el control de una cortadora de césped		

Tabla 3.8 Plataformas en el IoT propietarias (e)

Por otra parte, la mayoría de las plataformas propietarias en el IoT son compatibles con C y JavaScript, adicionalmente 17 plataformas soportan Java, mientras que 16 plataformas soportan JavaScript. Del mismo modo, Python es compatible con 14 plataformas, Java con 12 plataformas y C ++ con 10 plataformas. Por el contrario, los lenguajes menos comunes son Android (siete plataformas), Objective-C y PHP (cinco plataformas), Swift (cuatro plataformas), iOS y Ruby (tres plataformas), C # y Go (dos plataformas), Scala, Groovy, Erlang, Shell y Perl (una plataforma). Adicionalmente, es importante tener en cuenta que una sola plataforma en el IoT no satisface todas las necesidades de los usuarios. En consecuencia, existe una amplia gama de plataformas en el IoT que se utilizan dependiendo de los requisitos funcionales exigidos por los usuarios y grupos de usuarios. En este sentido, las plataformas propietarias Etherios, GroveStreams y JasperIoT tienen presencia exitosa en la mayoría de los dominios de aplicación del IoT. Por un lado, Etherios ofrece productos y servicios para apoyar a las empresas totalmente conectadas.

Adicionalmente, la nube de Etherios es una solución PaaS que permite a las empresas conectar cualquier producto o dispositivo y obtener visibilidad de activos en tiempo real. Además, Etherios ofrece soluciones personalizadas, conecta cualquier dispositivo a la nube, administra todos los dispositivos conectados desde una sola interfaz, administra los dispositivos de control y monitoreo en tiempo real, proporciona una infraestructura segura y escalable e integra los datos haciendo uso del SaaS.

Por otro lado, GroveStreams ofrece múltiples soluciones industriales en el IoT, incluida la tecnología de sensores para monitoreo ambiental. La plataforma recopila grandes cantidades de datos y realiza análisis en tiempo real para tomar decisiones inteligentes. Además, GroveStreams almacena más de 60 millones de puntos de datos, admite muchos tipos de datos, proporciona tiempos de muestra exactos en milisegundos, genera flujos derivados, detecta intervalos vacíos para el monitoreo de la calidad de los datos y proporciona seguridad de acceso basada en roles.

Finalmente, JasperloT administra y monitorea dispositivos conectados y desarrolla potentes aplicaciones para el IoT. La plataforma se adapta a las necesidades operativas específicas, los modelos de negocios y los requisitos de las industrias de todo el mundo. Además, ofrece visibilidad completa de la red a través de los dispositivos conectados, así como monitoreo en tiempo real para un control preciso y soporte para la toma de decisiones. Finalmente, JasperIoT analiza patrones de comportamiento y rendimiento, integra la infraestructura de las tecnologías de la información existentes e identifica problemas en tiempo real.

Además, se identificó que las plataformas Amazon Web Services (AWS), Microsoft Azure e IBM Watson, actualmente están teniendo una gran popularidad en el desarrollo de proyectos en el IoT.

3.2.4 Análisis de herramientas de modelado para el IoT

Las herramientas de modelado son aquellas que se utilizan para la creación de modelos de sistemas que ya existen o que apenas se van a desarrollar. Adicionalmente, permiten crear un "simulacro" del sistema, a bajo costo y con poco riesgo. A bajo costo porque, es un conjunto de gráficos y textos que representan el sistema, pero no son el sistema físico real (el cual es más costoso). Adicionalmente, minimizan los riesgos, porque los posibles cambios (por errores o cambios en los requerimientos), se realizan más fácil y rápidamente sobre el modelo que sobre el sistema ya implementado. Además, las herramientas de modelado, permiten concentrarse en ciertas características importantes del sistema, prestando menos atención a otras. Por ello, los modelos resultados, son una buena forma de determinar si están representados todos los requerimientos del sistema, así como también saber si el analista comprendió la funcionalidad del sistema.

Por otra parte, en el contexto del IoT se identificaron algunas herramientas de modelado de las cuales se realizó una descripción y se elaboró un estudio comparativo para identificar sus principales características a considerar para el modelado de los casos de estudio.

UML4IoT

UML4IoT es un enfoque basado en Lenguaje de Modelado Unificado (UML) para explotar la ingeniería dirigida por modelos en el desarrollo de sistemas en el IoT. Se basa en el uso de un perfil UML (el perfil UML4IoT) para automatizar el proceso de generación de la capa compatible con el IoT que se requiere para que los componentes cibernéticos se integren efectivamente en el entorno del IoT.

UML4IoT utiliza una aplicación Java (LiqueurPlantV21) la cual demuestra el enfoque basado en componentes adoptado en la construcción de sistemas de fabricación. Los componentes heredados o nuevos componentes cibernéticos se modelan con SysML y UML que tiene una API convencional orientada a objetos (OO), que se expresa como un conjunto de clases con sus métodos asociados. Esta interfaz conduce a un alto acoplamiento entre los

componentes del sistema. Los objetos inteligentes de aplicación de la alianza móvil abierta (OMA-Open Mobile Alliance) Máquina ligera a máquina (LWM2M-Lightweight Machine to Machine) y los objetos inteligentes definidos por el protocolo interno para la alianza de objetos inteligentes (IPSO), se utilizan para desarrollar una capa sobre una API convencional OO para transformar el componente ciberfísico en un componente compatible con el IoT, es decir, una especie de automatización industrial llamada capa IoTwrapper.

El enfoque UML4IoT describe un proceso de desarrollo sobre un modelo basado en un perfil UML para automatizar la generación en la capa IoTwrapper. La capa IoTwrapper es necesaria para que el componente ciberfísico se integre de manera efectiva en el moderno entorno de fabricación del IoT.

El enfoque UML4IoT se aplica en la especificación de nivel de código fuente del componente en caso de que no esté disponible una especificación de diseño UML. El desarrollador solo tiene que anotar el código fuente del componente físico-cibernético con anotaciones específicas. Posteriormente, un transformador de modelo a modelo adecuado genera la automatización. El perfil UML4IoT contiene construcciones de claves básicas independientes del dominio de la aplicación para que sean lo suficientemente generales como para ser aplicables en otros dominios de aplicación (Thramboulidis y Christoulakis, 2016).

ARIS

ARIS es un nuevo método de modelado que soporta proyectos en el IoT desde la fase de diseño hasta su implementación y gestión. ARIS permite a los diseñadores en el IoT enriquecer los procesos de negocio con objetos inteligentes, sensores y actuadores inteligentes. Más allá de la documentación, ARIS también permite el análisis del impacto en los modelos y procesos empresariales. ARIS utiliza un lenguaje de modelado conocido como *Event Driven Process Chain* (EPC), que reúne múltiples aspectos del modelado empresarial utilizando el marco de ARIS *House* de la ingeniería de los negocios. ARIS es un acercamiento al modelado de la empresa. Ofrece métodos para analizar procesos y tomar una visión holística del flujo de trabajo de gestión de diseño de procesos y procesamiento de aplicaciones. El enfoque ARIS proporciona un marco metodológico bien documentado para la Gestión de Procesos de Negocio (BPM- *Business Process Management*)

Adicionalmente, ARIS permite a la una empresa: entender, analizar y transformar procesos de negocio; descubrir y analizar las variaciones de los procesos; mejorar el rendimiento de

los procesos de negocios; y reducir los costes de implementación de tecnologías de la información con procesos claramente documentados. Por otra parte, ARIS presenta productos completamente nuevos, así como nuevas características, convenciones de modelado y muchas mejoras técnicas y de usabilidad. La versión propietaria ARIS 10 contempla aspectos destacados como presentar un análisis visual, que proporciona información rápida en un panel de control, ARIS Aware viene con casos de uso originales y proporciona una visualización innovadora, sensible al contexto y narrativa de los indicadores claves del desempeño, y un análisis de datos de manera transparente e integrada en ARIS *Connect*.

El rendimiento del proceso y la minería es el enfoque principal de ARIS *Process Performance Manager* (PPM) con una visualización mejorada y una integración profunda con otros componentes de ARIS, Por otro lado, ARIS PPM 10 proporciona nuevas funcionalidades de análisis en los paneles de análisis de procesos y mejoras visuales de la representación del proceso. Además de su diseño atractivo, le permiten filtrar interactivamente las rutas del proceso (por ejemplo, en función del proceso o la frecuencia de transición) en la visualización del flujo del proceso. Junto con ARIS Aware, ARIS PPM utiliza por completo sus funcionalidades de análisis para proporcionar los indicadores clave de rendimiento (KPI) medidos en paralelo con sus procesos documentados a seguir.

Además, en el lado de los métodos, ARIS 10 cuenta con nuevos tipos de modelos y metodologías:

- Respaldando el diseño, la planificación y la gestión de escenarios en el IoT, incluidos sensores y objetos inteligentes, modelo de decisión y notación (DMN) para un mejor modelado de los procesos de decisión, así como el soporte del cumplimiento del reglamento de protección de datos general (GDPR) de los Estados Unidos.
- ARIS para soluciones SAP (*Systems, Applications, Products in Data Processing*), ofrece soporte extendido para el manejo de la solución SAP 7.2, incluido el diseño de prueba para las mejores soluciones SAP de su clase, lo que es claramente un gran diferenciador para ARIS frente a la competencia.
- La administración de Gobierno, Riesgo y Cumplimiento (GRC), se beneficia de las funciones mejoradas de generación de informes y utilizando una oferta completa disponible en la nube.
- La integración de ARIS en la plataforma de negocios digitales de Software AG sigue siendo un enfoque estratégico, clave para la transformación exitosa e integrada de negocios (ARIS, 2018).

MDD para aplicaciones en el IoT

El principal propósito que cumple el Desarrollo Dirigido por Modelos (MDD) es tratar de minimizar los costos y tiempo de desarrollo de las aplicaciones de software, en la búsqueda de mejorar la calidad de las aplicaciones, independiente de la plataforma donde el software se ejecuta. El MDD es un paradigma emergente que resuelve numerosos problemas asociados con la composición e integración de sistemas a gran escala basado en el uso de modelos, soportado por potentes herramientas que tienen como objetivo reducir el tiempo de desarrollo y mejorar la calidad de los productos, separando el diseño de la arquitectura. Con el objetivo principal de permitir aumentar la productividad y reducir los costos del desarrollo. Los modelos se van generando desde la parte más abstracta a lo más concreto, a través de transformaciones y/o refinamientos. Los puntos claves de la iniciativa del MDD son la abstracción, automatización y estándares, trayendo consigo beneficios como la adaptación de los cambios tecnológicos, requisitos, reutilización y mejora la comunicación tanto para los usuarios como para los desarrolladores.

La utilidad del modelo se centra en definir lenguajes de modelado sin ambigüedades, contando con herramientas de transformación para leer y entender los modelos, en tener reglas de transformación claras que describan cómo un modelo en un lenguaje fuente se transforma a un modelo en un lenguaje destino y en el uso de definiciones formales obtenidas por la sintaxis de los lenguajes, facilitando su automatización (Sosa-Reyna et al., 2018).

Papyrus

Papyrus para el IoT es un entorno de modelado que permite especificar, diseñar y desplegar sistemas complejos en el IoT utilizando una metodología. También permite supervisar el sistema en ejecución usando la tecnología *Models @ Runtime* que busca extender la aplicabilidad de los modelos producidos en los enfoques de ingeniería dirigida por modelos (MDE) al entorno de ejecución. Dos conceptos centrales de la MDE son la abstracción y la automatización, los cuales son la mejor solución para los dos primeros desafíos. La abstracción facilita la especificación y el diseño de sistemas complejos en el IoT a través de la actividad de modelado. La automatización permite administrar tecnologías heterogéneas a través de transformaciones automáticas de modelos y generación de código para platafor-

mas tecnológicas específicas. Además de estos dos principios, Papyrus cuenta con un enfoque arquitectónico común para el empoderamiento del IoT: el modelo de referencia de arquitectura en el IoT-A fue diseñado para extrapolar elementos comunes y definir una capa de abstracción que es común a todas las arquitecturas existentes relacionadas con el IoT.
Papyrus para el IoT utiliza una metodología basada en IoT-A, para guiar al diseñador de sistemas en el IoT durante el desarrollo del sistema. La metodología es genérica e independiente de cualquier producto. Se compone de cinco pasos:
Paso 1: Se diseña el propósito y los requisitos del sistema utilizando diagramas de casos de uso UML y diagramas de requisitos del Lenguaje de modelado de sistemas (SysML- *System Modeling Lenguaje*).
Paso 2: Definir la especificación del proceso. Los casos de uso del sistema en el IoT se describen formalmente, se basan y derivan del propósito y la especificación de requisitos.
Paso 3: Definir la arquitectura funcional del sistema basada en el modelo del IoT.
Paso 4: Se define la plataforma operativa en la que se ejecutará el sistema funcional. La vista operativa del sistema contiene información sobre dispositivos informáticos (p. ej., Raspberry PI, Intel Edison, ST Cortex, entre otros), sensores, actuadores y protocolos de comunicación (p. ej., Wifi, Ethernet, Zigbee, entre otros).
Paso 5: Definir los planes de implementación que incluyen información sobre la asignación de bloques funcionales (paso 3) a los operativos (paso 4). Después de definir los planes de implementación, Papyrus para sistemas en el IoT permite la generación automática de código para Vortex de Prismtech, que proporciona una solución basada en el servicio de distribución de datos (DDS) para desplegar dinámicamente microservicios en dispositivos del IoT, descubrir nuevos dispositivos y migrar microservicios en tiempo de ejecución.
Para abordar un dominio en el IoT, cada parte de Papyrus se personaliza: un perfil UML, explorador de modelos, estilo y notación de diagramas, vistas de propiedades, menús de creación y paleta. Así mismo, Papyrus permite técnicas basadas en modelos: simulación basada en modelos, pruebas formales basadas en modelos, análisis de seguridad, análisis de rendimiento / compensaciones y exploración de arquitectura (Eclipse.org, 2018).

TORTE

TORTE es una herramienta para editar y verificar modelos de arquitectura del sistema en el IoT. La herramienta está compuesta por un editor UML para permitir una fácil edición de

modelos arquitectónicos, así como su fácil verificación, utilizando la programación lógica. TORTE es un método de modelado para arquitecturas de sistemas en el IoT. En TORTE, los objetos arquitectónicos, sus relaciones y sus propiedades son capturados con el fin de modelar una arquitectura de sistema en el IoT, mapea cada relación correspondiente en una capa única y captura ampliamente tales interacciones entre objetos. Cada tipo de relación tiene un espacio de descripción respectiva llamada capa, por lo que la complejidad se suaviza incluso si existen muchas relaciones, las que son diferentes entre sí están definidas. Una capa también se coloca en una vista arquitectónica en una etapa del desarrollo conforme al modelo de picos gemelos.

TORTE es una herramienta novedosa que proporciona dos características principales para el soporte que son: la edición y verificación de los modelos. TORTE es una herramienta para editar y verificar modelos. Además, la parte del verificador de la herramienta proporciona diversas características como: se generan automáticamente cláusulas en Prolog, se proporciona un formulario de entrada para dar una consulta a la programación, el programa se ejecuta utilizando JIProlog, una implementación de Prolog en Java, como un componente de comprobación y el resultado de la ejecución se resalta en el modelo para dar una retroalimentación al desarrollador (Ogata et al., 2019).

VDSML

El lenguaje de modelado visual específico para el IoT (LMSDV) es un lenguaje de modelado desarrollado para el IoT y demostrado que es lo suficientemente potente para el profesional y al mismo tiempo, lo suficientemente simple para el uso de los usuarios no técnicos. Es un enfoque mixto que fue diseñado con UML, sistemas en el IoT y usuarios finales técnicos / no técnicos.

El mayor desafío en el diseño del lenguaje fue cómo hacer una compensación entre la variedad / expresión del lenguaje en el contexto del IoT y la simplicidad para el no desarrollador, el usuario final. El lenguaje se construye a partir de varios elementos: objeto, especificación, anotación, encapsulación y subsistemas – grupos, ítems, círculo y notación de semicírculo, estereotipos de la interfaz, lista de interfaz de elementos, puertos y estructura interna y reglas (Eterovic et al., 2015).

IoTSec

IoTSec es una extensión / perfil UML para guiar a los desarrolladores a lo largo del ciclo de vida del diseño de sistemas en el IoT, esto se refiere a los requisitos de seguridad en cada etapa. Este enfoque propone una representación gráfica de los módulos de seguridad, una nomenclatura que encapsula los problemas de seguridad en el IoT y extensiones de diagramas UML. Se propone una extensión UML llamada IoTSec que es un subconjunto de UML y SysML. Aplica mecanismos estereotipos UML, diagramas UML / SySML y estereotipos UMLSec. Su objetivo es modelar la seguridad y problemas en los sistemas del IoT. Principalmente extiende UML, por lo tanto también extiende SysML e incluye algunos estereotipos propuestos en UMLSec.

IoTSec propone una nomenclatura con preocupaciones de seguridad dentro de cada elemento, algunas de ellas son tomadas de la propuesta del IoT, sin embargo, en esta obra la nomenclatura es más amplia que los módulos de seguridad del IoT. La nomenclatura comprende quince elementos: 1) Autenticación, 2) Autorización, 3) Cifrar, 4) Descifrar, 5) Almacenamiento seguro, 6) Comunicación segura, 7) IoT Broker o puente, 8) Confianza y reputación, 9) Gestión de claves, 10) Gestión de la identidad, 11) Seudónimo, 12) Autoridad de certificación, 13) Autoridad de registro, 14) Protección contra manipulación y 15) Control personalizado (Robles et al., 2017).

Modelado del sistema de protección de fronteras basado en IoT

El modelado del sistema de protección de fronteras basado en el IoT propone trabajar con sensores para proporcionar un monitoreo continuo en la frontera. Las puertas de enlace y los elementos fronterizos se asumen en este modelo. Incluso si un intruso no es identificado, es detectado, informado y capturado mediante el despliegue de este sistema después de su implementación. En primer lugar, el modelo basado se desarrolla en grafos, los grafos son efectivos para los sistemas de comunicación con sensores, los *gateways* y los elementos fronterizos se asumen como nodos y la comunicación entre ellos se describe a través de los arcos.

Se utiliza el modelado basado en UML para que se comprenda el desarrollo del sistema. El caso de uso se define para representar requisitos funcionales y los diagramas de secuencia

se desarrollan para describir la comunicación en el sistema. Después de desarrollar modelos semiformales, es decir, modelos basados en grafos y basados en UML. Se propone el algoritmo para superar los inconvenientes de simulaciones, se utilizan métodos formales basados en lenguajes de especificación, es decir, VDM-SL (Lenguaje de especificación del método de desarrollo de Viena) para especificar el algoritmo. La especificación formal propuesta se analiza a través de VDMSL la cual es una herramienta que identifica varios errores en las primeras etapas del desarrollo del sistema (Afzaal et al., 2017).

MARTE++

Un perfil UML para MARTE, es el estándar actual del grupo de manejo de objetos (OMG) para el modelado y análisis de sistemas integrados en tiempo real. MARTE fue probado en varios dominios de aplicación y enfoques de validación. Los aspectos útiles incluyen plataformas modernas como Multi-núcleo, muchos núcleos y las unidades de procesamiento de gráficos (GPUs), redes para dominios más amplios como es caso del IoT. Marte es un modelado con nuevas áreas, herramientas, tecnologías y especificaciones relacionadas, lo que sugiere la necesidad de su adaptación a los entornos de los profesionales modernos.
MARTE es la especificación de plataformas hardware que incluyen múltiples núcleos. Esta extensión incluye no solo la descripción del hardware, sino también los algoritmos de programación modernos para multi-núcleo, incluida la capacidad de las tareas para migrar de un núcleo a otro, así como los mecanismos para realizar la asignación del núcleo (afinidad) a las tareas (Medina y Villar, 2017).

Redes de petrí coloreadas (CPN)

Las CPN son el lenguaje gráfico para construir modelos de sistemas concurrentes y analizar sus propiedades. Adicionalmente, las CPN son una red Petri clásica extendida con datos, jerarquía y tiempo; un valor de datos adjunto se llama color de *token*, además el conjunto de colores es un lugar que contiene *tokens*. Así mismo, las CPN proporcionan las primitivas para la descripción de la sincronización de los procesos concurrentes, mientras que el lenguaje de programación ML de las CPN, proporciona las primitivas para la definición de tipos de datos y la manipulación de valores de datos. Además, las CPN permiten al diseña-

dor enfocarse en los modelos problemáticos utilizando una herramienta de CPN para editar, simular y analizar el modelo de redes de Petri coloreadas (Dechsupa et al., 2016).

Por otra parte, CPN Tools es una herramienta para editar, simular y analizar redes de Petri coloreadas. La interfaz gráfica del usuario (GUI) se basa en técnicas de interacción avanzadas, como herramientas, menús de marcado e interacción bi-manual, entre otras. Adicionalmente, las funciones de retroalimentación proporcionan mensajes de error contextuales e indican relaciones de dependencia entre elementos netos. Así mismo, el informe de espacio de estado estándar contiene información como propiedades de acotación y propiedades de vida. La herramienta presenta una verificación incremental de sintaxis y generación de código, que tiene lugar mientras se construye una red. Adicionalmente, un simulador rápido maneja eficientemente redes no cronometradas y cronometradas. Además, se generan y analizan espacios de estados completos y parciales, y un informe de espacio de estado estándar contiene información, como propiedades de acotación y propiedades de vida. CPN Tools utiliza el lenguaje CPN ML para especificar declaraciones e inscripciones netas. Este lenguaje es una extensión del lenguaje de programación funcional Standard ML (CPN Tools, 2020).

Las herramientas de CPN consisten básicamente en dos componentes: un editor gráfico y un simulador. El editor permite a los usuarios construir interactivamente un modelo de CPN que se transmite al simulador, que lo comprueba en busca de errores sintácticos y genera código específico del modelo para simular el modelo de CPN. Así mismo, el editor invoca el código del simulador generado y presenta los resultados gráficamente. Además, el editor soporta y guarda modelos usando un formato XML (Westergaard y Kristensen, 2009).

Análisis comparativo de herramientas de modelado para el IoT

Una vez descritas las herramientas de modelado de servicios para el IoT, en la Tabla 3.9 se presenta un análisis comparativo de las herramientas de modelado que considera diversos criterios con el propósito de identificar la herramienta de modelado que presenta las mejores características y se adapte al modelado de escenarios de la vida real en el contexto del IoT, los aspectos considerados son:

Nombre	Representación gráfica	Generación de código	Ingeniería inversa	Lenguajes	Sistemas operativos	Tipo de licencia
UML4IoT	Sí	NE	NE	Java	Microsoft Windows	Propietaria
ARIS	Sí	No	No	NE	Multiplaforma	Propietaria ARIS Cloud for Students
MDD para aplicaciones en el IoT	Sí	Sí	NE	NE	Microsoft Windows, Mac OS X	Propietaria
Papyrus	Sí	Sí	Sí	SysML	Microsoft Windows, Mac OS X	Propietaria
TORTE	Sí	Sí	NE	Prolog y Java	Microsoft Windows, Mac OS X	Propietaria
VDSML	Sí	Sí	NE	Java	Multiplaforma	Libre
IoTSec	Sí	NE	NE	SysML	Multiplaforma	Propietaria
Modelado del sistema de protección de fronteras basado en IoT	Sí	Sí	NE	Viena, VDM-SL	Multiplaforma	Propietaria
MARTE++	Sí	Sí	NE	SysML	Microsoft Windows, Mac OS X	Propietaria
CPN	Sí	Sí	Sí	Multilenguaje	Multiplaforma	Libre

Tabla 3.9 Comparativa de herramientas de modelado para el IoT.

- Nombre. Indica el nombre de la herramienta de modelado para el IoT.
- Representación gráfica. Indica si la propuesta considera y/o permite la representación gráfica (Diagramas o esquemas).
- Generación de código. Indica si la propuesta permite la generación de código.
- Ingeniería inversa. Indica si la propuesta considera la aplicación de la ingeniería inversa.
- Lenguajes. Indica los lenguajes que soporta la herramienta.

- Sistemas operativos. Indica la compatibilidad de la propuesta con los sistemas operativos que existen actualmente o indica si es multiplataforma.
- Tipo de licencia. Indica si la propuesta es libre o propietaria.

Adicionalmente, se indica con un "Sí" o un "No", cuando considera o no el aspecto que corresponda, los aspectos no especificados se indican con un "NE". El resultado del análisis comparativo de las herramientas de modelado disponibles para el IoT, indica que todas las herramientas analizadas consideran una representación gráfica del modelado, siete herramientas consideran la generación de código, solo Papyrus y CPN consideran la ingeniería inversa.

Así mismo, se observa que son diversos los lenguajes soportados por cada herramienta, sin embargo, Java es el lenguaje que presentan mayor compatibilidad con las herramientas de modelado (cuatro herramientas), además es relevante indicar que solo las CPN tienen compatibilidad con cualquier lenguaje, es decir, es multilenguaje. Adicionalmente, cinco herramientas son multiplataforma, solo dos herramientas son de acceso libre y la mayoría son propietarias, sin embargo, ARIS cuenta con una licencia para estudiantes limitada a 30 días de acceso. Además, se identificó que siete de las diez herramientas analizadas hacen uso o están basadas en UML. Por tal motivo, para el modelado de los casos de estudio propuestos en esta tesis doctoral se decidió utilizar las CPN debido a que es la herramienta de modelado que presenta las mejores características, es multilenguaje, multiplataforma, es de acceso libre y particularmente permite modelar de manera precisa cualquier escenario en el contexto del IoT, entre otros aspectos. Además, el modelado de casos de estudio en CPN permite identificar claramente el proceso de orquestación y coreografía a través de la vinculación con los lenguajes composicionales BPEL y WSCDL.

3.2.5 Principales elementos para el proceso de orquestación y coreografía de servicios

En esta sección se presentan los principales elementos identificados en la literatura para realizar el proceso de orquestación y coreografía de servicios, con base en un análisis de diversos lenguajes composicionales que permiten la orquestación y coreografía de servicios, los cuales se basan en XML y con los requerimientos actuales que se requieren en el IoT.

Elementos para el proceso de la orquestación de servicios

El proceso de orquestación de servicios es un proceso central (otro servicio) que lleva el control de los servicios implicados en la realización de una tarea y coordina la ejecución de las diferentes operaciones sobre dichos servicios. Los servicios orquestados no "conocen" (y no necesitan conocer) que están implicados en un proceso de composición y que forman parte de un proceso de negocio de nivel más alto. Solamente el coordinador central de la orquestación es "consciente" de la meta a conseguir, por lo que la orquestación se centraliza mediante definiciones explícitas de las operaciones y del orden en el que se invocan los servicios. Además, la orquestación se utiliza normalmente en procesos de negocio privados.

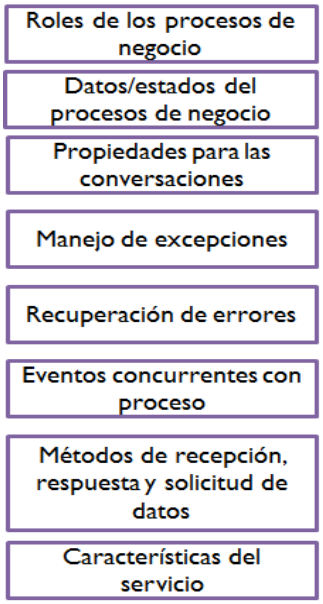


Figura 3.4 Elementos para la orquestación de servicios.

En la Figura 3.4 se presentan los elementos identificados para realizar un proceso de orquestación de servicios los cuales son: roles de los procesos de negocios, datos/estado de los procesos de negocios, propiedades para las conversaciones entre los participantes en el proceso de orquestación de servicios, manejo de excepciones, recuperación de errores, un elemento para la recuperación de errores en caso de ser necesario, manejo de eventos concurrentes con un proceso, método de solicitud, método de recepción, método de respuesta y elementos que consideren las características de los dispositivos utilizados en el IoT (marca, modelo, estado del dispositivo, nivel de la batería, peso, tamaño, entre otros).

Elementos para el proceso de coreografía de servicios

En el proceso de la coreografía de servicios, no existe un coordinador central. En su lugar, se utiliza la colaboración entre servicios como herramienta para describir la comunicación e identificar el intercambio de mensajes entre los servicios participantes, en donde cada servicio implicado en dicho proceso de coreografía de servicios "conoce" exactamente cuándo ejecutar sus operaciones y con quién interactuar. La coreografía es un esfuerzo colaborativo centrado en el intercambio de mensajes en procesos de negocio públicos. Además, la colaboración entre servicios permite que todos los participantes en un proceso de coreografía de servicios se mantengan "informados" del proceso de negocios, las operaciones a ejecutar, los mensajes a intercambiar y el tiempo a invertir en dicho intercambio de mensajes.

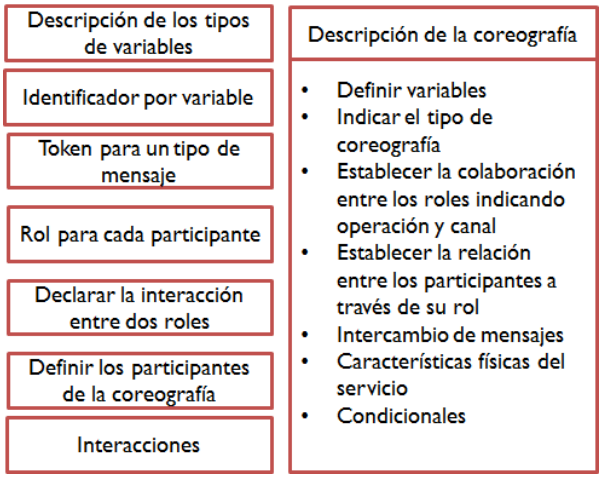


Figura 3.5 Elementos para la coreografía de servicios.

En la Figura 3.5 se presentan los elementos identificados para realizar un proceso de coreografía de servicios los cuales son: una descripción de los tipos de variables a utilizar, un identificador por cada variable, se requieren *tokens* para los tipos de mensajes, un rol por cada participante, se requiere declarar la interacción entre dos roles, se definen los participantes en el proceso de la coreografía de servicios, establecer las interacciones y describir la coreografía, es decir, dentro de este elemento se considera definir las variables a utilizar, indicar el tipo de coreografía, establecer la colaboración entre los roles indicando la operación y el canal a utilizar, establecer la relación entre los participantes mediante de su rol correspondiente, se establece el intercambio de los mensajes, se establecen las condiciona-

les a utilizar y se consideran las características físicas del servicio que participa en el proceso de la coreografía de servicios.

3.2.6 Expresividad del mecanismo de composición de servicios en el IoT

La expresividad es la capacidad de expresar la solución de un problema en la forma más cercana a la formulación original del problema, una forma clara, natural, intuitiva y concisa y en términos de otros (sub) problemas resueltos. Por ello, cuanto más expresivo es un mecanismo, mayor es la variedad y cantidad de ideas que se utilizan para representar y resolver un problema. En otras palabras, la expresividad de un mecanismo es la capacidad con la que cuenta para expresar determinadas ideas. Si un mecanismo expresa más y de mejor forma determinadas ideas del usuario, entonces se le considera más expresivo que otros métodos o mecanismos. Por ello, la expresividad es muy importante si se desea definir o crear un método o un mecanismo que ayude a los usuarios a resolver de una mejor manera las problemáticas de un determinado contexto. La expresividad de un mecanismo, entonces, no es una propiedad intrínseca del mismo, sino una propiedad que surge de la utilización que se le da en un contexto específico. Por lo tanto, es muy importante definir qué es lo que se desea expresar, para así diseñar un mecanismo que efectivamente sea capaz de comunicarlo. Además, en el contexto de los lenguajes de programación se sabe que los lenguajes declarativos son más expresivos que los lenguajes imperativos.

Ante este contexto, el mecanismo de composición de servicios bajo el enfoque del IoT propuesto en esta tesis es declarativo y contiene la mayor cantidad de elementos o características que le permiten ser lo más expresivo posible, tales como: identificadores, tipos de datos, operadores, variables y constantes, palabras reservadas, condicionales, funciones, ciclos, excepciones, la concurrencia, comentarios y que permita realizar la orquestación y coreografía de servicios en el contexto del IoT. En la Figura 3.6 se presentan los elementos considerados para la expresividad del mecanismo de composición de servicios bajo el enfoque del IoT, los cuales se describen a continuación:

- **Identificadores**. Se consideró incluir identificadores porque son un conjunto de caracteres alfanuméricos de cualquier longitud para identificar las entidades de la solución de un problema (clases, funciones, variables, tipos compuestos). Los identificadores son combinaciones de letras y números. Cada mecanismo establece las reglas que definen como son construidos. Cuando un identificador se asocia a una en-

tidad concreta, entonces es el "nombre" de dicha entidad y en adelante la representa durante la ejecución. Además, nombrar las entidades hace posible referirse a las mismas, lo cual es esencial para cualquier tipo de procesamiento simbólico.

- **Tipos de datos**. Según su uso y función los datos con los que trabaja el mecanismo son principalmente: 1) Entero: para representar números enteros; 2) Real: para representar números con punto decimal, y 3) Cadena: para datos de tipo texto o carácter.

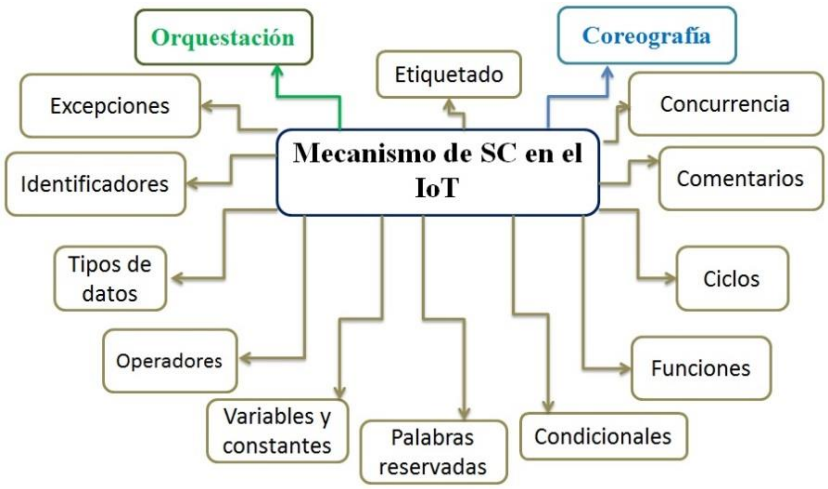


Figura 3.6 Elementos para la expresividad del mecanismo de composición de servicios

- **Operadores.** Se consideró utilizarlos para crear instrucciones realizando cálculos matemáticos, comparaciones u operaciones lógicas. Se consideraron tres tipos de operadores: aritméticos, relacionales y lógicos.
- **Variables y constantes.** Ambos se consideraron definidos por un identificador para hacer referencia a ellos durante la composición de servicios. Las constantes son datos que no cambian durante la ejecución. Las variables por el contrario son datos que cambian durante el proceso de ejecución.
- **Palabras reservadas.** Conjunto de palabras que se consideraron propias de su sintaxis, y fueron empleadas para construir instrucciones. Este tipo de palabras no se utilizan, por ejemplo, para crear o nombrar variables.
- **Estructuras de control.** Según su función el mecanismo cuenta con ciclos para repetir un bloque de instrucciones un número de veces determinado según se requiera. Además, se consideró incluir condicionales las cuales a raíz de realizar una comparación lógica, ejecuten una o varias instrucciones.

- **Funciones.** Se consideró incluir funciones ya que son una especie de variables, definidas por el usuario o pertenecientes al mecanismo, que al llamarlas ejecutan un subprograma con el objetivo de simplificar la solución de un problema en caso de invocarlas en varias ocasiones.
- **Orquestación.** En este caso se consideró incluir los elementos identificados en el proceso de orquestación de servicios.
- **Coreografía.** En este caso se consideró incluir los elementos identificados en el proceso de la coreografía de servicios.
- **Comentarios.** Los comentarios se consideraron necesarios para colocar instrucciones que no se ejecutan, su función es la de comentar o aclarar partes de la solución de un problema para facilitar su seguimiento.
- **Concurrencia.** Se consideró que el mecanismo permita la concurrencia, es decir, que se capaz de realizar varias actividades, solicitudes o eventos simultáneamente.
- **Excepciones.** Se consideró que el mecanismo contenga excepciones, es decir, indicar o dar un aviso cuando la ejecución de un método, actividades, solicitudes o evento no termina correctamente, sino que termina de manera excepcional como consecuencia de una situación no esperada.
- **Etiquetado.** Se consideró que al igual que un documento XML el mecanismo utilice la característica del etiquetado, es decir, que toda etiqueta que se abre se debe cerrar.

Es relevante mencionar que el mecanismo de composición de servicios bajo el enfoque de IoT generado es de tipo declarativo con la mayor expresividad posible, ya que los mecanismos declarativos son ampliamente aceptados y son más expresivos que los imperativos.

3.2.7 Métodos y/o técnicas de formalización

Existen diferentes técnicas que se utilizan para el proceso de formalización, entre las que se destacan las álgebras de procesos, los autómatas temporizados, las reglas ECA (evento-condición-acción), las CPN, entre otras. A continuación, se presenta una descripción de cada una de ellas:

- **Álgebra de procesos**. El álgebra de procesos es el estudio de sistemas distribuidos o paralelos por medios algebraicos. La palabra "proceso" se refiere al comporta-

miento de un sistema. Un sistema es cualquier cosa que muestre un comportamiento, como la ejecución de un sistema de software, las acciones de una máquina o incluso las acciones de un ser humano. El comportamiento es el total de eventos, acciones o evoluciones que realiza un sistema, el orden en que se ejecuta y quizás otros aspectos de esta ejecución como el tiempo, las probabilidades o los aspectos continuos. Siempre, la atención está en ciertos aspectos de la conducta, sin tener en cuenta otros aspectos, por lo que se considera una abstracción o idealización de la conducta "real". En lugar de considerar el comportamiento, se considera una observación del comportamiento, donde una acción es la unidad de observación elegida. Como el origen del álgebra de procesos está en la informática, se suele pensar que las acciones son discretas: la ocurrencia ocurre en algún momento y las diferentes acciones se separan en el tiempo. Esta es la razón por la que un proceso a veces también se denomina sistema de eventos discretos. Además, el álgebra de procesos se ha extendido en los últimos años para abarcar no solo sistemas de eventos discretos, sino también fenómenos en continua evolución, lo que da como resultado las denominadas álgebras de procesos híbridos (Baeten, 2005).

- **Autómatas temporizados**. Fueron introducidos como una notación formal para modelar el comportamiento de los sistemas en tiempo real. Su definición proporciona una forma sencilla de anotar gráficos de transición de estado con restricciones de tiempo utilizando un número finito de variables de reloj de valor real. El análisis automatizado de autómatas temporizados se basa en la construcción de un cociente finito del espacio infinito de las valoraciones del reloj. A lo largo de los años, el formalismo fue estudiado exhaustivamente y se obtuvieron muchos resultados que establecen conexiones con los circuitos y la lógica, logrando un gran avance en el desarrollo de algoritmos de verificación, heurísticas y herramientas (Alur, 2003).
- **Reglas ECA.** Son utilizadas en entornos dinámicos y están diseñadas para los sistemas que necesitan una respuesta automática a determinadas condiciones o eventos. Los cambios de las condiciones ambientales durante el tiempo son factores importantes que impactan en una reducción de la efectividad de estas reglas que están implícitas en las demandas cambiantes de los usuarios de los sistemas que varían con el tiempo. La regla ECA es una de las estructuras de representación del conocimiento para eventos que también se denominan regla activa (Isazadeh et al., 2014). Una regla ECA tiene tres partes: un evento, una condición y una acción. La semántica de una regla ECA es cuando se detecta un evento, evalúa la condición y,

si se cumple, ejecuta la acción. Por tal motivo, una regla ECA permite a los sistemas responder a eventos que ocurren en un orden arbitrario, así como a combinaciones de eventos (Cacciagrano y Culmone, 2020). Además, las reglas ECA se originaron en bases de datos activas y desde entonces son utilizadas en áreas que incluyen personalización, gestión de big data y automatización de procesos comerciales. En la actualidad las reglas ECA están siendo exploradas para redes M2M, el Internet de las cosas, computación cognitiva y la web semántica (Isazadeh et al., 2014).

- **CPN.** En el contexto del IoT, los entornos inteligentes son sistemas que reaccionan a las condiciones cambiantes y al comportamiento de los usuarios, los cuales se especifican fácilmente con las CPN. Por otra parte, las CPN son el enfoque correcto para validar determinadas expresiones, tienen la ventaja de verificar y validar los requisitos no funcionales de un sistema. Así mismo, las CPN proporcionan un modelo ejecutable a través del cual se conoce si el sistema modelado funciona correctamente o no. En las CPN, se utilizan estructuras de datos que modelan el objeto y sus atributos. Adicionalmente, las CPN verifican la corrección del sistema diseñado que se realiza con herramientas CPN. Además, las CPN son un lenguaje de modelado orientado a gráficos que se utiliza para el diseño, especificación, simulación y verificación de sistemas. Las CPN constan de lugares, transiciones, arcos, inscripciones de arco y fichas. Los lugares se dibujan como elipses que representan el estado de una red de Petri coloreada. Las transiciones que se dibujan como rectángulos representan las acciones. Cada lugar está asociado con un tipo de datos que determina el tipo de datos que contiene. Cada lugar contiene una serie de fichas que llevan el valor de los datos. Cuando se dispara una transición, el estado cambia, este cambio en el estado de los lugares depende de las inscripciones de arco que muestran el número exacto de fichas. Por otro lado, la herramienta CPN *Tools* se utiliza para validar modelos diseñados mediante simulaciones. CPN *Tools* se utiliza para validar el modelo de red de Petri coloreada. Da un impacto visual incorporando todas las limitaciones para diseñar un buen marco. Además, CPN *Tools* también se utiliza para validar modelos estocásticos y probabilísticos con redes de Petri coloreadas jerárquicas (Sharaff y Rath, 2020).

Para el proceso de formalización del mecanismo de composición de servicios en el contexto del IoT se decidió utilizar las CPN debido a que son un formalismo ampliamente utilizado

para describir sistemas concurrentes como protocolos de red y sistemas de flujos de trabajo o *workflow* (Westergaard y Kristensen, 2009).

3.3 Etapa de Diseño

Las principales actividades durante esta etapa fueron el diseño de la orquestación y la coreografía de servicios en el IoT. Así mismo, en esta etapa se incluyó el diseño de la expresividad para el mecanismo de composición de servicios en el IoT, el diseño de los casos de estudio, particularmente en el dominio del cuidado de la salud, en el dominio de aplicación del IoT Personal y Social, y un caso de estudio para la integración y coordinación de servicios en el contexto del IoT. Adicionalmente, en esta etapa se consideró el modelado de los casos de estudio y el diseño de la formalización del mecanismo de composición de servicios en el contexto del IoT.

3.3.1 Diseño de la orquestación de servicios en el IoT

La orquestación de servicios es un proceso centralizado para organizar interacciones entre los servicios de una actividad o proceso de negocio; sin embargo, los orquestadores involucrados en una misma tarea de orquestación de servicios rara vez se conocen entre sí. La orquestación se refiere a procesos empresariales ejecutables que interactúan con otros servicios internos y externos. Las interacciones ocurren a nivel de mensaje. Incluyen negocios lógicos y solicitan el orden de ejecución, y miden las aplicaciones y organizaciones para definir la duración, una transacción y un modelo de proceso de acceso múltiple (Macker y Taylor, 2017).

Los servicios orquestados no "saben" (y no necesitan saber) que están involucrados en un proceso de composición y que son parte de un proceso comercial de nivel superior. Solo el coordinador central de la orquestación es "consciente" del objetivo a alcanzar, por lo que la orquestación se centraliza mediante definiciones explícitas de las operaciones y el orden en que se invocan los servicios. Además, la orquestación se usa normalmente en procesos comerciales privados.

La orquestación de servicios se basó en el lenguaje composicional BPEL, pero enfocado y dirigido a la orquestación de servicios en el IoT. Del mismo modo, se tuvo en cuenta que la organización de servicios se diseñó para organizar tanto los servicios Web tradicionales

descritos a través del WSDL, así como para organizar los servicios REST (transferencia estatal de representación) que actualmente utilizan los diferentes proveedores de dispositivos portátiles dentro del IoT.

Por ello, se presentan elementos en el lenguaje composicional BPEL que son necesarios para realizar la orquestación de servicios en el contexto del IoT. Las etiquetas consideradas para la orquestación de servicios son:

- **process**: Etiqueta utilizada para el principio y el final de la orquestación de servicios en la IoT.
- **partnersLinks**: Etiqueta para definir procesos de negocios y sus respectivos roles.
- **variables**: Etiqueta que indica los datos / estados que se utilizan en los procesos de negocio.
- **correlationSets**: Etiqueta para las conversaciones en el proceso de negocios.
- **faultHandlers**: Etiqueta utilizada para el manejo de las excepciones.
- **compensationHandlers**: Etiqueta considerada para la recuperación de errores.
- **eventHandlers**: Etiqueta para a utilizar en los eventos concurrentes.
- **invoke**: Etiqueta que representa el método para invocar datos o eventos del dispositivo inteligente.
- **receive**: Etiqueta que representa el método para recibir datos o eventos del dispositivo inteligente.
- **reply**: Etiqueta que representa el método para responder a datos o eventos desde el dispositivo inteligente.

Es importante indicar en el mecanismo de composición del servicio bajo el enfoque del IoT, es posible realizar la orquestación de servicios en diferentes escenarios, no solo en el ámbito médico, sino también en otros contextos, como la domótica y el sector industrial.

3.3.2 Diseño de la coreografía de servicios en el IoT

La coreografía de servicios es colaborativa, lo que permite a cada parte involucrada describir su participación en la interacción. La coreografía sigue las secuencias de los mensajes entre las múltiples partes y generalmente obtiene los intercambios en los mensajes públicos que ocurren entre los servicios, en lugar de un proceso comercial específico que ejecuta una

sola parte, en la coreografía, no hay coordinador central. En cambio, cada servicio involucrado en esa coreografía sabe exactamente cuándo ejecutar sus operaciones y con quién interactuar. La coreografía es un esfuerzo de colaboración centrado en el intercambio de mensajes en los procesos comerciales públicos. Todos los participantes en la coreografía están informados del proceso comercial, las operaciones a ejecutar, los mensajes a intercambiar y el tiempo a invertir en dicho intercambio de mensajes (Chen y Englund, 2017).
La coreografía de servicios se basó en el lenguaje composicional WSCDL, pero enfocado y dirigido a la coreografía de servicios en el IoT. En este contexto, los elementos de la coreografía de servicios se explican a continuación:

- **package.** Etiqueta que indica el comienzo y el final del proceso de coreografía de servicios.
- **variableType.** Describe los tipos de variables a utilizar.
- **informationType**. Etiqueta para describir los tipos de las variables a usar en una coreografía. Además, se utiliza para describir los tipos de mensajes entre roles en una interacción y es útil para los *tokens*, *token locators* y *channel types*.
- **token.** Etiqueta para definir un alias para cada variableType.
- **tokenLocator.** Etiqueta para ubicar un *token* dentro de un tipo de mensaje.
- **roleType.** Etiqueta para definir el rol o comportamiento de cada participante.
- **relationshipType.** Etiqueta para declarar la interacción del rol o comportamiento entre dos participantes.
- **participantType.** Etiqueta para definir los participantes de la coreografía.
- **channelType**. Etiqueta para hacer las interacciones y/o colaboraciones entre los participantes.
- **choreography**. Etiqueta para indicar el comienzo y el final de la descripción de la coreografía.
- **variableDefinitions**. Etiqueta para definir las diferentes variables a utilizar dentro de los servicios de coreografía.
- **sequence, parallel y choice**. Etiquetas para indicar que la descripción de la actividad dentro de la coreografía es secuencial, en paralelo o de elección.
- **interaction**. Etiqueta para establecer la colaboración entre los participantes, indicando la operación y el tipo de colaboración que se desea utilizar.
- **participate.** Etiqueta para establecer la relación entre los participantes a través de su rol o comportamiento dentro de la coreografía.

- **participateData**. Se identificó la necesidad de esta etiqueta para indicar los datos del participante dentro de la coreografía (*token* de acceso y características de los servicios ofrecidos por participante).
- **exchange.** Etiqueta para intercambiar mensajes dentro de la coreografía, a través de messageType y la acción (*request / response*).
- **send.** Etiqueta para enviar mensajes dentro de la coreografía.
- **receive**. Etiqueta para recibir mensajes dentro de la coreografía.

Es importante mencionar que es posible que la coreografía de servicios en el mecanismo de composición de servicios de IoT se represente en otros escenarios y dominios de aplicación del IoT.

3.3.3 Diseño de la expresividad del mecanismo de composición de servicios

En esta sección se presenta la expresividad del mecanismo de composición de servicios en el IoT, establecido como un modelo y llamado SCM-IoT. Este modelo integra la expresividad de los lenguajes composicionales BPEL y WSCDL, los cuales contienen los principales elementos descritos en la sección 3.2.6 (identificadores, tipos de datos, operadores, variables y constantes, palabras reservadas, condicionales, funciones, ciclos, excepciones, la concurrencia, comentarios, entre otros), así como la orquestación y coreografía que se realiza con cada uno de ellos, además el modelo cuenta con tres extensiones en BPEL (de selección, actualización y notificación) que permiten que SCM-IoT sea un modelo con mayor expresividad.

Por otra parte SCM-IoT es un modelo que se propuso para desarrollar aplicaciones en el IoT y tiene una arquitectura orientada a datos lógicamente centralizada en oposición a la arquitectura distribuida orientada a servicios en la que se definen los lenguajes composicionales BPEL y WS-CDL, la cual se fundamenta en el intercambio de mensajes sobre una infraestructura de red. En consecuencia, SCM-IoT permite extender o simplificar el desarrollo de muchas aplicaciones que son propias de la inteligencia artificial o del aprendizaje automático. Dichas aplicaciones involucran la colaboración de agentes inteligentes que intercambian información a través de un medio de comunicación compartido como, por

ejemplo, un repositorio o la pizarra (*blackboard*). Además, entre las diversas aplicaciones de SCM-IoT, se presentan las siguientes:

- Sistema en domótica para la asistencia de personas con discapacidades.
- Sistema de asistencia en el área del cuidado de la salud para contribuir en el control de sobrepeso y obesidad.
- Integración de servicios en el IoT en un sistema que permite el seguimiento de actividades programadas como aquellas que son observadas en pacientes con algún tipo de discapacidad física.

SCM-IoT incluye un lenguaje de marcado incorporado a BPEL. Las nuevas etiquetas se traducen a construcciones escritas enteramente en BPEL, las cuales permiten orquestar las actividades de agentes interesados en acceder a datos relevantes del sistema en el IoT.
Por otra parte, SCM-IoT corresponde a un modelo orientado a datos (*data-centred*) similar al modelo de las bases de datos distribuidas, en donde los datos se encuentran localizados en diferentes espacios de almacenamiento distribuidos lógica o geográficamente, pero interconectados por una infraestructura de red que asegura la recuperación e intercambio de información entre las aplicaciones y los espacios de almacenamiento.
No obstante, para hacer posible su definición sobre una arquitectura orientada a servicios, las colecciones de datos se encapsulan a través de servicios compartidos proporcionados por un administrador. Los espacios de almacenamiento tienen colecciones de datos representados como documentos XML, designados como libros de notas (*notebooks*). Los libros constituyen el medio de coordinación y comunicación entre los procesos participantes. El contenido de las notas es administrado exclusivamente por un proceso administrador de notas. El propósito del administrador es el de preservar la consistencia e integridad de los datos almacenados en las notas, actuando como prestador de servicios de consulta, actualización y notificación. Cada nota consiste en un conjunto fijo de campos entre los cuales uno de ellos permite identificar de manera única a la nota en su conjunto. La actualización de una nota es una modificación de uno solo de sus campos. Para su actualización, el administrador obtiene una copia de la nota solicitada la cual localiza y recupera del libro mediante una búsqueda de su identificador. Concluida la modificación de la copia, el administrador se encarga de substituir la nota original por su copia modificada. A su vez, una actualización desencadena una serie de notificaciones a procesos cliente interesados en determi-

nar si se alcanzó alguna o algunas de las condiciones establecidas sobre el contenido de las notas.

Adicionalmente, el diseño de las extensiones al modelo SCM-IoT y al lenguaje de marcado de BPEL incorpora conceptos como notas, campos, libros, así como de operaciones de selección, actualización y de notificación de actualizaciones como nuevos elementos XML. La evolución del comportamiento de un sistema que usa el modelo SCM-IoT se caracteriza por los siguientes elementos:

- Los clientes desarrollan sus actividades independientemente entre sí, pero interactuando a través de los servicios que presta el administrador.
- Los servicios se distinguen por la dirección que toma el flujo de notas con respecto al libro, que es, o bien hacia el libro mediante una actualización, o bien desde el libro mediante una notificación.
- A los procesos que actualizan los libros se les conoce como productores y a los que reciben notificaciones se les conoce como subscriptores.
- Después de que ha ocurrido alguna actualización por un productor, el administrador revisa el estado del contenido del libro probando las condiciones de una lista de avisos, en donde cada una de las condiciones de aviso fue proporcionada por al menos un subscriptor.
- Una vez identificadas las condiciones que se verifican de la lista, el administrador selecciona alguna de ellas de manera no determinística, enviando un mensaje de notificación al subscritor interesado en ella.

En el contexto del IoT, particularmente de la domótica, sensores y actuadores se identifican respectivamente con productores y subscritores ya que, en los primeros, se originan los datos con los que se actualiza el libro de notas y, en los segundos, se destinan las notificaciones detectadas por el administrador. Así, una aplicación en el IoT se desarrolla como un flujo de datos que transita de los sensores hacia los actuadores mediando el libro de notas para su registro y transformación. SCM-IoT simplifica la programación de actividades dado que las interacciones entre los participantes son generalmente muy complejas. En SCM-IoT, los sensores y actuadores se agrupan por el tipo de variable ambiental a la que están dedicados, por ejemplo, temperatura, humedad o intensidad de luz. Estas variables ambientales se designan en el modelo SCM-IoT como tópicos. Entre las ventajas de señalar un

tópico es que permite simplificar la automatización de condiciones preestablecidas las cuales se aplican en caso de su omisión en la especificación.
Por otra parte, la información contenida en el libro tiene varias propiedades entre las que se destacan las siguientes:

- Mediante sus servicios, los participantes acceden al libro de manera secuencial lo que elimina los problemas de acceso concurrente a una memoria compartida como aquellos que se observan en las bases de datos distribuidas.
- El libro contiene en todo momento la información más reciente conocida hasta que ocurre una nueva actualización proveniente de algún productor.
- Una actualización desencadena una serie de modificaciones en el contenido del libro debido a los suscriptores. Dichas modificaciones siempre terminan eventualmente dejando el libro en un estado estable.
- En el momento que ocurre una notificación, el libro contiene la información más reciente conocida.
- Una notificación induce la participación de un subscriptor el cual garantiza al final de su participación que se preservan las condiciones de consistencia e integridad y otras condiciones de invarianza sobre el contenido del libro.

3.3.4 Diseño de los casos de estudio

En esta sección se presenta el diseño de los casos de estudio planteados en el dominio del cuidado de la salud y en el dominio Personal y Social basados en el paradigma del IoT, así como un caso de estudio para la integración y coordinación de servicios en el contexto del IoT, los cuales permitieron realizar la validación del mecanismo de composición de servicios bajo el enfoque del IoT.

3.3.4.1 Caso de estudio 1: Dominio de aplicación del cuidado de la salud

Este caso de estudio se presenta para representar la composición de servicios dentro de un mecanismo de composición de servicios basado en IoT. El caso de estudio aborda la orquestación y la coreografía de los servicios de salud para pacientes con sobrepeso u obesidad. El escenario del caso de estudio es el siguiente:

- Los adultos mayores con obesidad necesitan saber la cantidad de calorías ingeridas, las calorías quemadas, el sueño, la frecuencia cardíaca, la cantidad de agua consumida, los pasos diarios y la actividad física necesaria para lograr una pérdida de peso gradual sin complicaciones de salud, lo que ayuda a evitar un infarto de miocardio.

La Figura 3.7 representa el escenario explicado anteriormente. Como se observa, los parámetros médicos del paciente (es decir, calorías quemadas, actividad física, frecuencia cardíaca, peso, índice de masa corporal – IMC, entre otros) se recopilan mediante un dispositivo *wearable* y una báscula inteligente, que se sincronizan mediante un teléfono inteligente. El paciente utiliza una aplicación para visualizar los datos de orquestación de los servicios. Para realizar la orquestación de los servicios (es decir, calorías quemadas, actividad física, frecuencia cardíaca, peso, IMC), el mecanismo de composición de servicios se basa en servicios REST para solicitar los parámetros médicos tanto al proveedor del *wearable* como al proveedor de la báscula inteligente.

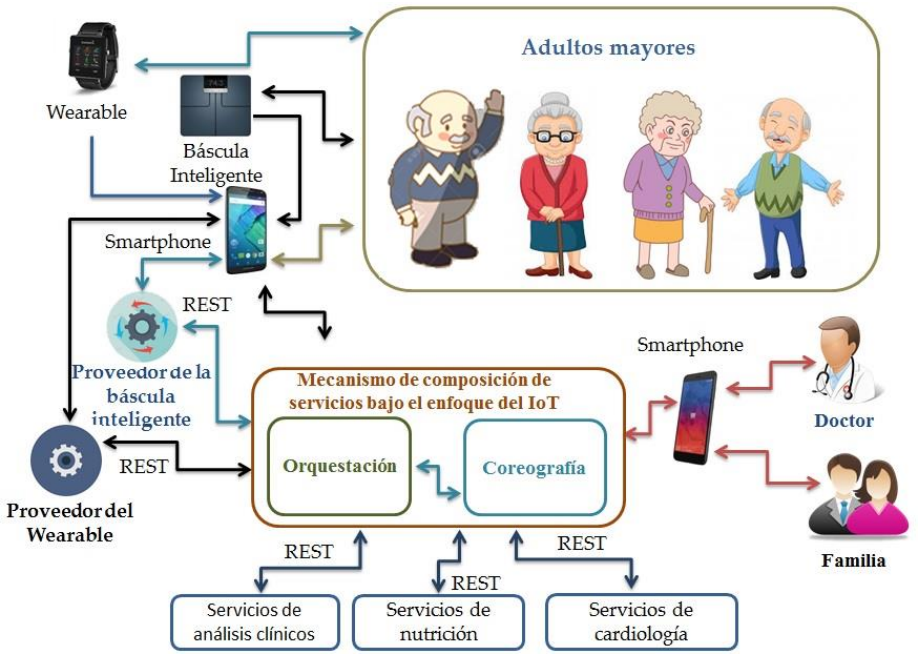


Figura 3.7 Esquema del caso de estudio en el dominio del cuidado de la salud.

El mecanismo de composición de servicios hace esto para coordinar estos servicios y establecer un orden de invocación, para que el paciente visualice la información de forma di-

námica y en tiempo real. Luego, se realiza la coreografía de los servicios Web externos (servicios de análisis clínicos, servicios de nutrición y servicios de cardiología). Para este fin, el mecanismo de composición de servicios se basa en el correspondiente WSDL para solicitar información sobre los servicios Web externos (por ejemplo, disponibilidad, precio, tiempo, ubicación). Luego, utilizando esta información, el paciente o el familiar selecciona de las propuestas presentadas: el laboratorio de análisis clínico más confiable, el mejor plan de nutrición y/o el cardiólogo más conveniente, de acuerdo con sus preferencias y criterios. Además, los datos de monitorización diaria, así como el historial del paciente son consultados por el médico o familiar vinculados con el paciente cuando alguno de ellos lo requiera.

3.3.4.2 Caso de estudio 2: Dominio de aplicación personal y social

Este caso de estudio se presenta para representar la composición de servicios dentro de un mecanismo de composición de servicios basado en IoT. El caso de estudio aborda la orquestación y coreografía de servicios para el confort, seguridad y el ahorro de energía en una casa inteligente. El escenario del caso de estudio es el siguiente:

- Una casa equipada con sensores tales como: sensor de flujo de agua, sensor de control de energía, sensor de presencia o de movimiento, sensor de temperatura, sensor de gas y sensor de sonido, requiere ser monitorizada para garantizar que brinde a sus residentes el confort y seguridad adecuados, cuidando simultáneamente la disminución en el consumo de energía.

En la Figura 3.8 se muestra el escenario para el confort, seguridad y ahorro de energía de una casa inteligente, en donde se observa que los datos y/o eventos del hogar se monitorean a través de los seis sensores, que envían los datos y/o eventos a un *gateway* con el cual están sincronizados. Este mecanismo garantiza que se envíe la información adecuada a cada proveedor de los sensores. Posteriormente, el mecanismo de composición de servicios solicita los datos de la casa a cada proveedor de los sensores para garantizar la comodidad interior y emitir recomendaciones de ahorro de energía de acuerdo con las preferencias de los residentes.

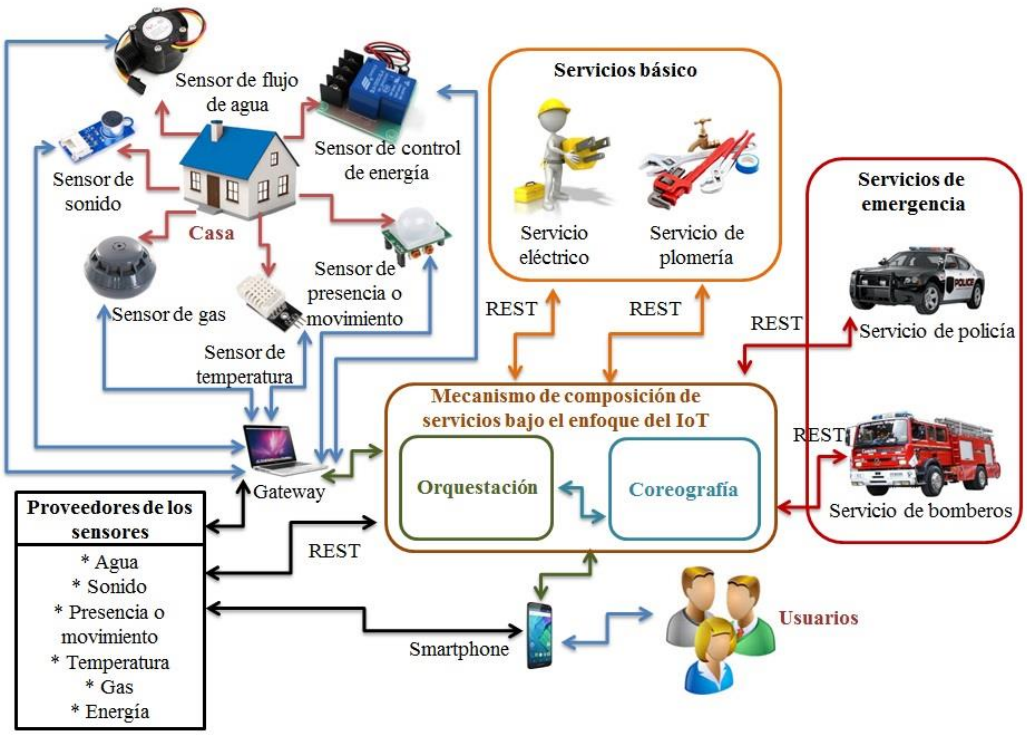


Figura 3.8 Esquema del caso de estudio en el dominio personal y social.

El mecanismo de composición de servicios recibe y analiza los datos para identificar: 1) patrones de consumo de energía; 2) posibles problemas dentro del funcionamiento normal de la casa, como fugas de agua o gas y fallas eléctricas (que probablemente aumenten el consumo de energía) y, 3) posibles emergencias que comprometan la seguridad y la integridad del hogar (por ejemplo, inundación, incendio, intento de robo). Si un problema requiere solicitar un servicio básico, el mecanismo de composición localiza los proveedores de servicios básicos disponibles para atender la solicitud.

Luego, el sistema notifica al usuario sobre el problema y muestra información básica de cada proveedor (es decir, nombre, dirección, número de teléfono, horario de atención, costos y referencias, entre otros). El usuario elige su proveedor preferido y envía la confirmación del servicio al sistema para notificar al proveedor correspondiente, de modo que se responda la solicitud del servicio. Por el contrario, si el mecanismo de composición de servicios identifica una emergencia de seguridad, invoca automáticamente a un proveedor de servicios de emergencia (servicio de policía o servicio de bomberos) enviando datos básicos sobre la casa (por ejemplo, dirección, tipo de emergencia, los datos y/o eventos emitidos por los sensores, propietario).

El mecanismo de composición de servicios notifica simultáneamente a los residentes de la casa de la emergencia para ayudarlos a tomar las medidas necesarias, dependiendo de si están en la casa o no.

3.3.4.3 Caso de estudio 3: Integración y coordinación de servicios en el contexto del IoT

Este caso de estudio se presenta para la integración de los dominios de aplicación del cuidado de la salud y el dominio personal y social, así mismo para representar la composición de servicios dentro de un mecanismo de composición de servicios basado en IoT. El caso de estudio aborda la orquestación y la coreografía de los servicios proporcionados por los sensores de una casa inteligente, servicios de salud, servicios de emergencia y servicios básicos para el confort y seguridad del paciente, monitoreo de parámetros médicos, definición de rangos aceptables de los parámetros médicos y la vigilancia médica de un paciente con discapacidad física y sobrepeso. El escenario del caso de estudio es el siguiente:

- Un paciente con amputación traumática de una extremidad inferior y con sobrepeso requiere apoyo para realizar sus actividades básicas dentro de su hogar (apagar o encender luces, cerrar o abrir cortinas, encender o apagar el calentador, entre otras) debido a su discapacidad. Además, necesita saber la cantidad de calorías ingeridas y calorías quemadas, las horas de sueño, la frecuencia cardíaca, la cantidad de agua consumida y la actividad física de cada día con el propósito de controlar su peso y gradualmente reducirlo para contribuir en evitar algún otro tipo de complicación para su salud (problemas en la rodilla de la extremidad sana, diabetes, hipertensión, entre otras).

La Figura 3.9 representa el escenario explicado anteriormente. Como se observa, un paciente que ha sufrido la amputación traumática de una extremidad inferior requiere de la ayuda de la domótica para realizar sus actividades básicas diariamente dentro de su hogar. En este sentido los datos y/o eventos en la casa se monitorean a través de cinco sensores (sensor de flujo de agua, sensor de luz natural, sensor de presencia o de movimiento, sensor de temperatura y sensor de gas), los cuales envían la información a un *gateway* con el cual están sincronizados, además los sensores se configuran con las preferencias o necesidades de con-

fort que el paciente requiere. Este mecanismo garantiza que se envíe la información adecuada a cada proveedor de los sensores y sobre todo la comodidad del paciente.

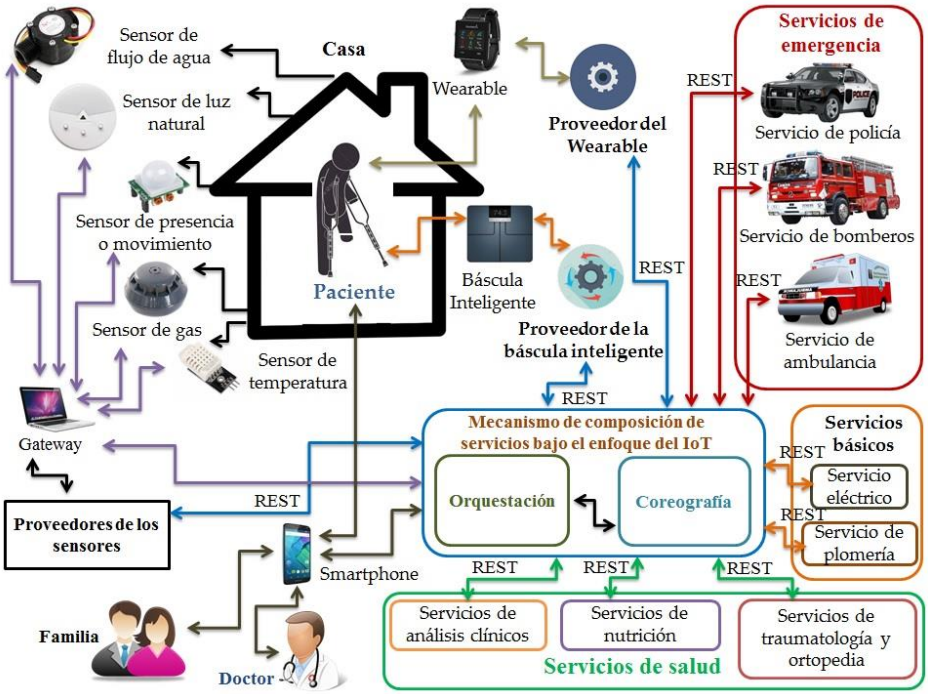


Figura 3.9 Esquema del caso de estudio de un paciente con discapacidad física y sobrepeso.

Así mismo, los parámetros médicos del paciente (es decir, calorías ingeridas, calorías quemadas, sueño, agua, pasos, minutos de actividad física, peso e IMC) se recopilan mediante un dispositivo *wearable* y una báscula inteligente, los cuales envían los datos del paciente a sus respectivos proveedores. Esto permite la monitorización de los parámetros del paciente para controlar el incremento de peso y contribuir a disminuirlo de manera gradual para que tenga una mejor recuperación y evitar otro tipo de complicaciones (problemas en la rodilla de la extremidad sana, diabetes, hipertensión, entre otras).

El mecanismo de composición de servicios se basa en servicios REST para solicitar los parámetros médicos del paciente al proveedor del *wearable* y al proveedor de la báscula inteligente. Así mismo, utilizando REST el mecanismo de composición de servicios en el IoT solicita los datos de la casa a cada uno de los proveedores de los sensores. Estas solicitudes de información las realiza el mecanismo de composición de servicios para coordinar los servicios disponibles y garantizar la comodidad del paciente, realizar el monitoreo y vigilancia médica del paciente, así como para establecer un orden de invocación según se requiera, para que el paciente y los familiares visualicen por medio de un *Smartphone* la in-

formación de la casa y el estado de salud del paciente de una forma dinámica y en tiempo real, en este aspecto también el doctor visualiza los datos médicos diarios del paciente.

Por otra parte, el mecanismo de composición de servicios recibe y analiza los datos enviados por los proveedores de los dispositivos (*wearable* y báscula inteligente) y de los sensores para identificar: 1) variables críticas fuera de los valores normales (por ejemplo, el ritmo cardíaco, peso, IMC, entre otros), 2) posibles problemas dentro de la casa, como fugas de agua o gas y fallas eléctricas (que dificulten la comodidad del paciente y/o su vigilancia médica) y, 3) posibles emergencias que comprometan la integridad física del paciente (por ejemplo, inundación, incendio, intento de robo, caída o estado de salud delicado del paciente). Si alguno de los dos primeros casos requiere solicitar un servicio de salud (servicios de análisis clínicos, servicios de nutrición o servicios de traumatología y ortopedia) o un servicio básico (servicios de plomería o servicio eléctrico), el mecanismo de composición localiza los proveedores de servicios disponibles para atender la solicitud.

Luego, el sistema a través de un *Smartphone* en tiempo real notifica al usuario, al doctor y al familiar sobre el estado de salud del paciente o si es un problema identificado en la casa se notifica solo al paciente y al familiar, en ambos casos se muestra la información básica de cada proveedor (es decir, nombre, dirección, número de teléfono, horario de atención, tipo de servicio, fechas y horarios disponibles, costos y referencias, entre otros). El paciente o familiar selecciona al proveedor del servicio que se requiere de acuerdo a sus criterios, preferencias y costo, y envía la confirmación del servicio elegido al sistema para notificar al proveedor correspondiente, de modo que se atienda la solicitud del servicio en la fecha y horario establecido en la confirmación.

Además, si el mecanismo de composición de servicios identifica alguna emergencia (caída o estado de salud delicado del paciente, incendio, inundación, fuga de gas mayor o intento de robo), invoca automáticamente a uno o varios proveedores de servicios de emergencia (servicio de ambulancia , servicio de bomberos o servicio de policía) enviando los datos de la casa (por ejemplo dirección, tipo de emergencia, los datos y/o eventos emitidos por los sensores, propietario, la ubicación del paciente, entre otros) y la información médica del paciente (nombre, sexo, edad, tipo de discapacidad, estado de salud y los parámetros médicos del paciente recolectados por el *wearable*). El mecanismo de composición de servicios notifica simultáneamente al paciente y a sus familiares del tipo de emergencia, así mismo, proporciona el estado de salud del paciente y los datos del proveedor o proveedores de servicios que confirmaron atender la emergencia para ayudar al paciente a tomar las medidas necesarias si se encuentra dentro o fuera de la casa.

Por otra parte, como dato adicional es importante indicar que posterior a la cirugía de una amputación traumática de alguna extremidad inferior de un paciente, regularmente permanece hospitalizado de 48 a 72 hrs. dependiendo de su estado de salud. Después de salir del hospital el paciente continúa con su recuperación en casa, una vez transcurridos de 7 a 10 días y si todo se encuentra de manera normal, se sugiere acudir con su médico para que le retiren los puntos de la operación y volver nuevamente a casa para seguir recuperándose.
Posteriormente, al pasar un mes es necesario acudir con su médico para un revisión y verificar que la herida este completamente sana, dependiendo del estado de su recuperación, el médico programa revisiones mensuales o bimestrales durante el primer año y después el paciente se sugiere acudir al menos una vez al año para una revisión.
Finalmente, es relevante indicar que los pacientes que sufren de una amputación traumática de una extremidad inferior, tienden a subir de peso debido a la disminución de actividad física, depresión (al volverse en ocasiones pacientes dependientes) o por ansiedad (consumo excesivo de alimentos), lo cual es un riesgo para su salud.

3.3.5 Modelación de los casos de estudio

En este apartado se presenta el modelado de los casos de estudio para la composición de servicios en el dominio de aplicación del IoT del cuidado de la salud, la composición de servicios en el dominio de aplicación del IoT personal y social, y la integración y coordinación de servicios en el contexto del IoT, utilizando la herramienta de modelado CPN Tools 4.0.1.

3.3.5.1 Caso de estudio 1: Composición de servicios de salud

El modelado del caso de estudio de la composición de servicios de salud se realizó con base en sus principales actividades tales como:

- Coordinar el seguimiento de las diversas actividades que realiza durante el día de acuerdo con lo establecido en una agenda que además le indica el tipo de equipo a utilizar en cada caso.
- Coordinar el seguimiento de la agenda de citas médicas e ingesta de medicamentos para su oportuna administración.

- Reportar periódicamente la frecuencia cardiaca, el número de pasos dados, calorías quemadas, calorías ingeridas, el sueño, entre otras variables al proveedor del dispositivo *wearable* para su registro y posterior consulta.
- Para el control de peso, avisar al paciente cuando tiene que pesarse en la báscula inteligente con el propósito de llevar un registro de su peso.
- Avisar al nutriólogo en caso de acumular un número preestablecido de incrementos consecutivos de peso, o bien, en caso de que su peso se sitúe dentro de los límites de obesidad de acuerdo con las clasificaciones médicas aceptables.

El sistema de salud se caracteriza por estar organizado en torno a una persona o, más aún, a un paciente que presenta varios tipos y niveles de incapacidad. Por ello, es importante mencionar que es bien sabido que las personas tienden a manifestar un comportamiento errático, el cual, en el mejor de los casos, solamente se aproxima al comportamiento de un modelo ideal de un paciente. En consecuencia, el sistema de salud está guiado casi enteramente por el paciente, lo que se traduce en una red donde es más difícil identificar un medio centralizado organizado por libros. En la Figura 3.10 se presenta el modelado en CPN de la composición de servicios de salud.

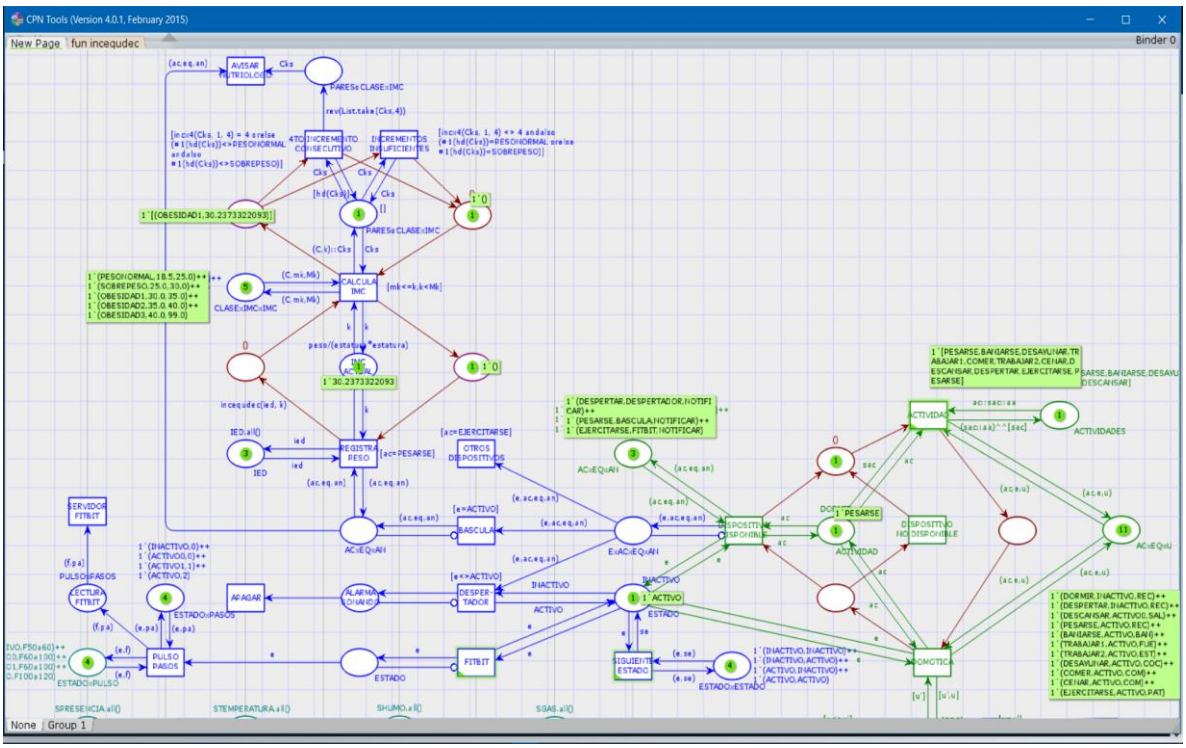


Figura 3.10 Modelado en CPN de la composición de servicios de salud

En la Figura 3.10 aparece la transición ACTIVIDAD que indica la actividad a realizar obtenida de una lista de actividades programadas en el día para el paciente. Dadas las diferentes condiciones de incapacidad que un paciente tenga como una amputación traumática de una extremidad inferior y con sobrepeso, así como las actividades obligadas o recomendadas que requiera su tratamiento, la lista incluye actividades como: DORMIR, DESPERTAR, EJERCITARSE, PESARSE, COMER, entre otras. Estas actividades se especializan dependiendo del estado o condición del paciente, por ejemplo, para pacientes con diabetes, la actividad COMER tiene que revelar el contenido de azúcares y carbohidratos que la comida contiene y de este modo, quitarla para introducir otras como COMERSINAZUCAR, COMERPOCAAZUCAR, entre otras.

Por otra parte, puesto que muchas de las actividades que realiza un paciente con discapacidad requieren de algún dispositivo para coadyuvar su desarrollo, inicio o término, la transición DISPOSITIVOSDISPONIBLES determina cuál es el dispositivo requerido para la actividad a realizar. Para la selección del dispositivo, esta transición usa la información conocida sobre el estado del paciente, como activo o inactivo. Por ejemplo, puesto que la mayoría de las actividades requieren que el paciente se encuentre en estado ACTIVO, el sistema considera el uso de dispositivos que induzcan un cambio sin riesgos en el estado del paciente, como un despertador. En este caso, si la agenda tiene prevista que es hora de realizar la actividad DESAYUNAR, la cual solo se realiza en estado activo pero el paciente no lo está, entonces el sistema hace uso del dispositivo DESPERTADOR para inducir el cambio de estado requerido. La intervención de este dispositivo queda representada por la transición DESPERTADOR que se aplica solo en el caso de que el paciente se encuentre en estado INACTIVO.

Así mismo, determinar el estado del paciente con suficiente precisión es uno de los problemas por resolver en la asistencia automatizada a pacientes con discapacidades. En la solución propuesta para este caso de estudio se sugiere el uso de dispositivos *wearables* como fitbit que son cada vez más robustos, confiables y precisos, cuyos costos los hacen cada vez más asequibles a la población. Adicionalmente, fitbit proporciona varios tipos de biometría como frecuencia cardiaca, temperatura, intensidad de actividad por el número de pasos dados, peso y nivel de estrés. Aunque de enorme utilidad, esta información resulta aún insuficiente para determinar con precisión la condición general del paciente y con ello sugerir el tipo de actividad a realizar, de acuerdo con las indicadas en la agenda dictada por el médico o terapeuta. La observancia del estado del paciente es importante para determinar

su progreso en su tratamiento, pero sobre todo para llevar un registro sobre la evolución de su enfermedad y/o discapacidad.

Entre las actividades previstas con pacientes con sobrepeso y obesidad se encuentra el de revisar su peso en forma periódica. La revisión periódica consiste en medir el peso y calcular el IMC del paciente mediante una báscula inteligente. Las mediciones generalmente se toman semanalmente a una hora y un día preestablecidos.

En la transición BASCULA, la cual representa al dispositivo del mismo nombre, se tienen programadas una serie de pasos a seguir para informar al médico del estado del paciente. El primer paso está representado por la transición REGISTROPESO para la cual, su disparo representa el momento en que el paciente se ha subido a la báscula y ésta obtiene el peso. Una vez obtenido el peso, el dispositivo calcula el IMC que le corresponde y lo envía al sistema de salud para su registro en una lista. El sistema compara este valor del IMC con el último registrado y contabiliza el número de incrementos sucesivos del IMC en la lista. Cuando el número de incrementos sucesivos haya excedido a un umbral predefinido, se elabora un reporte con estos registros, mismo que se envía al médico del paciente solicitando una cita. También se solicita la cita tan pronto como el último valor obtenido de su IMC lo sitúe en el nivel superior siguiente en la clasificación de sobrepeso y obesidad. Esta decisión se realiza aun cuando no se haya superado el umbral de incrementos consecutivos porque presupone un aumento en los riesgos a su salud. Además, el sistema de salud fue modelado para poder incorporarse en un sistema integral que brinde mayores atenciones para la vigilancia de los cuidados de salud de pacientes en situación agravada por alguna determinada enfermedad y/o discapacidad.

3.3.5.2 Caso de estudio 2: Composición de servicio en la domótica

El modelado del caso de estudio de la composición de servicios en la domótica, a diferencia del modelado del caso de estudio de la composición de servicios de salud, se caracteriza por no estar organizado en función de actividades programadas de una persona, si no, más bien está organizado para que los habitantes de una casa realicen sus actividades dentro del hogar sin estar limitados a una agenda de actividades. En la domótica es común encontrar sistemas que controlan el encendido y apagado de las luces o la automatización de otras actividades que se realizan dentro del hogar, por ejemplo, el apagado automático de determinados dispositivos (TV, aire acondicionado, entre otros), además son pocos los sistemas que

consideran el confort o seguridad de los habitantes y en algunos casos el ahorro de energía, sin embargo, en la mayoría de los casos no se consideran los tres aspectos, los cuales en el modelado de este caso de estudio sí se consideran. En la Figura 3.11 se presenta el modelado en CPN de la composición de servicios en la domótica.

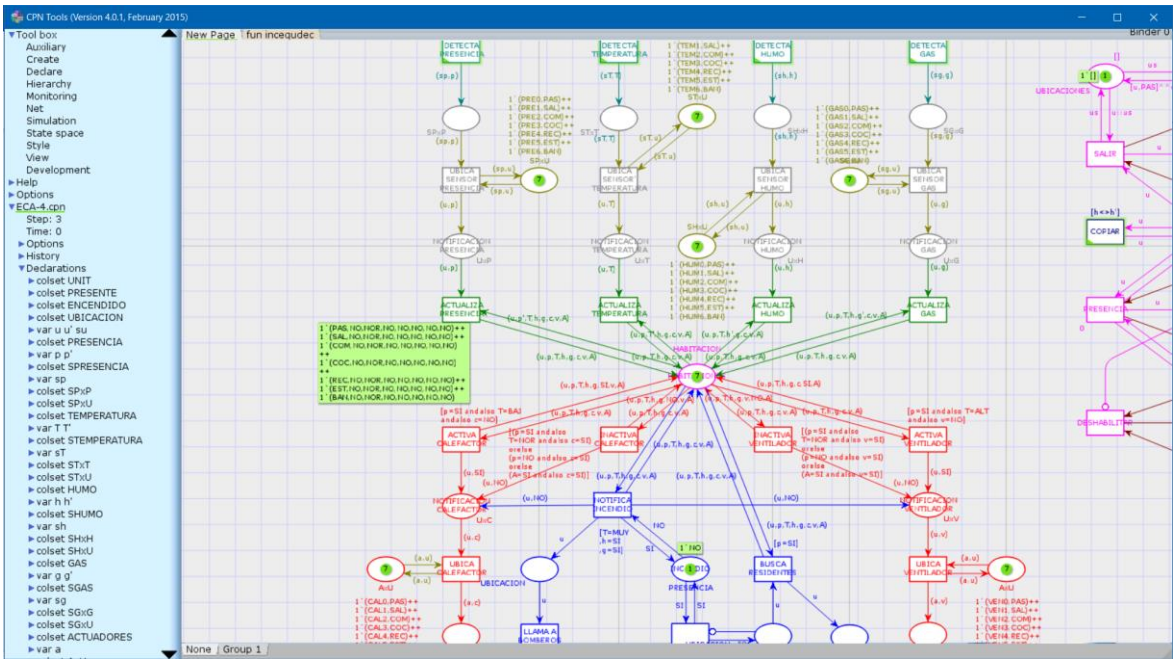


Figura 3.11 Modelado en CPN de la composición de servicios en la domótica

Para contribuir el confort y seguridad de los habitantes, en la Figura 3.11 se identifica la transición ACTUALIZAPRESENCIA, la cual identifica la presencia de un habitante de la casa en alguna de las habitaciones previamente registradas (SALA, COMEDOR, COCINA, BAÑO, PASILLO, RECAMARA, ESTUDIO, PATIO, AFUERA), la información de la presencia de la persona es fundamental para el control del encendido y apagado de las luces de la casa para contribuir en el ahorro de energía. Adicionalmente, se identifica la transición ACTUALIZATEMPERATURA, la cual identifica el nivel de temperatura de alguna habitación de la casa (BAJA, NORMAL, ALTA, MUY ALTA), esta información es muy importe para mantener una temperatura agradable para los residentes de acuerdo a las condiciones del medio ambiente a una previa configuración de preferencias de los usuarios y sobre todo para identificar si existe una temperatura muy alta que es una de las características de un posible incendio. También se identifican las transiciones ACTUALIZAHUMO y ACTUALIZAGAS, las cuales identifican la presencia de humo y gas tóxico en alguna habitación, aspectos relevantes que se utilizan para identificar un posible incendio. Los eventos

que ocurren en estas transiciones se envían al lugar HABITACIONES, el cual funge como intermediario con las transiciones ACTIVACALEFACTOR, ACTIVAVENTILADOR, INACTIVACALEFACTOR e INACTIVAVENTILADOR, las cuales realizan una determinada acción de acuerdo al evento identificado para contribuir en la comodidad de los habitantes de la casa.

Por otra parte, el lugar HABITACIONES se vincula con la transición NOTIFICAINCENDIO, la cual una vez que se cumplen los tres eventos de incendio (presencia de gas, humo y temperatura muy alta) invoca al servicio de bomberos. Así mismo, el lugar HABITACIONES se vincula con la transición BUSCARESIDENTES para que en el caso de un incendio se identifique la última ubicación de cada residente antes de que exista un fallo en el sistema y esta información se le proporcione a los rescatistas en el caso de que alguno de los residentes se encuentre dentro de la casa. Lo anterior permite identificar que en el modelado del caso de estudio de la composición de servicios en la domótica se consideró el confort y la seguridad de los residentes de una casa y además se consideró el ahorro de energía.

3.3.5.3 Caso de estudio 3: Integración y coordinación de servicios en el contexto del IoT

Para la integración y coordinación de servicios en el IoT el diseño y la modelación consistió de cuatro principales módulos los cuales se describen a continuación:

1. Módulo de control de peso e invocación de servicios de salud. Este módulo es el encargado del control de peso y la monitorización de los datos médicos del paciente. En la Figura 3.12 se presenta el modelado de este módulo en donde se utiliza un dispositivo *wearable* para la monitorización de los datos médicos del paciente (frecuencia cardíaca, pasos, calorías quemadas, calorías ingeridas, distancia recorrida, minutos de actividad física, sueño, consumo de agua y ejercicio), por otra parte, para el registro y el seguimiento del peso e IMC se utilizó una báscula inteligente. Además, se clasifica al paciente de acuerdo a su peso que se vaya registrando en normal, sobrepeso, obesidad 1, obesidad 2 y obesidad 3 según sea el caso. Es importante indicar que es posible incorporar otros dispositivos que el paciente necesite o requiera para una determinada actividad o rehabilitación en casa.

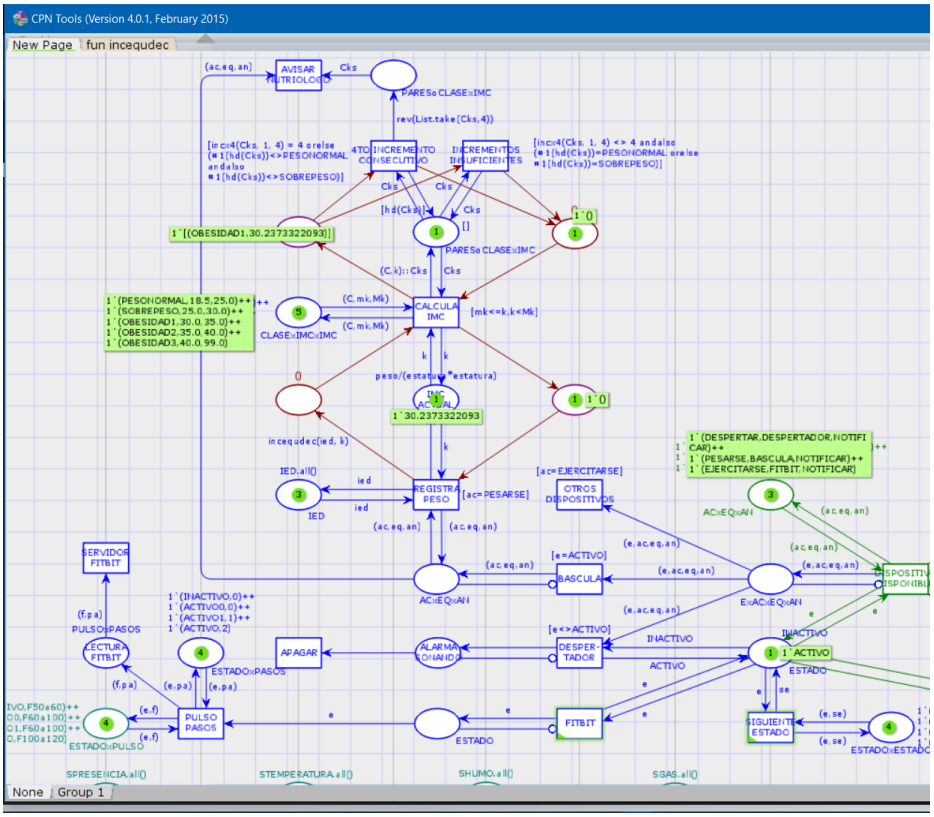


Figura 3.12 Módulo de control de peso e invocación de servicios de salud

Además, este módulo es el encargado de solicitar alguno de los servicios de salud (servicio de nutriólogo, cardiólogo, por ejemplo) en caso de ser necesario y de acuerdo a los datos de monitorización del paciente.

2. **Módulo de control de luces.** Este módulo es el encargado del control del encendido y apago inteligente de las luces de cada una de las habitaciones con el propósito de optimizar el consumo de energía eléctrica y reducir costos. En la Figura 3.13 se presenta el modelado del control de luces del caso de estudio propuesto de la composición de servicios en el contexto del IoT para un paciente con discapacidad física.

   En el modelado se indica si existe la presencia del paciente en alguna de las habitaciones (recámara, pasillo, baño, cocina, comedor, sala o patio) y si el sensor de luz natural indica que existe poca visibilidad, de manera automática se encienden las luces de la habitación en la que se encuentre el paciente, o por el contrario se apagan las luces de la habitación en caso de percibir la ausencia del paciente.

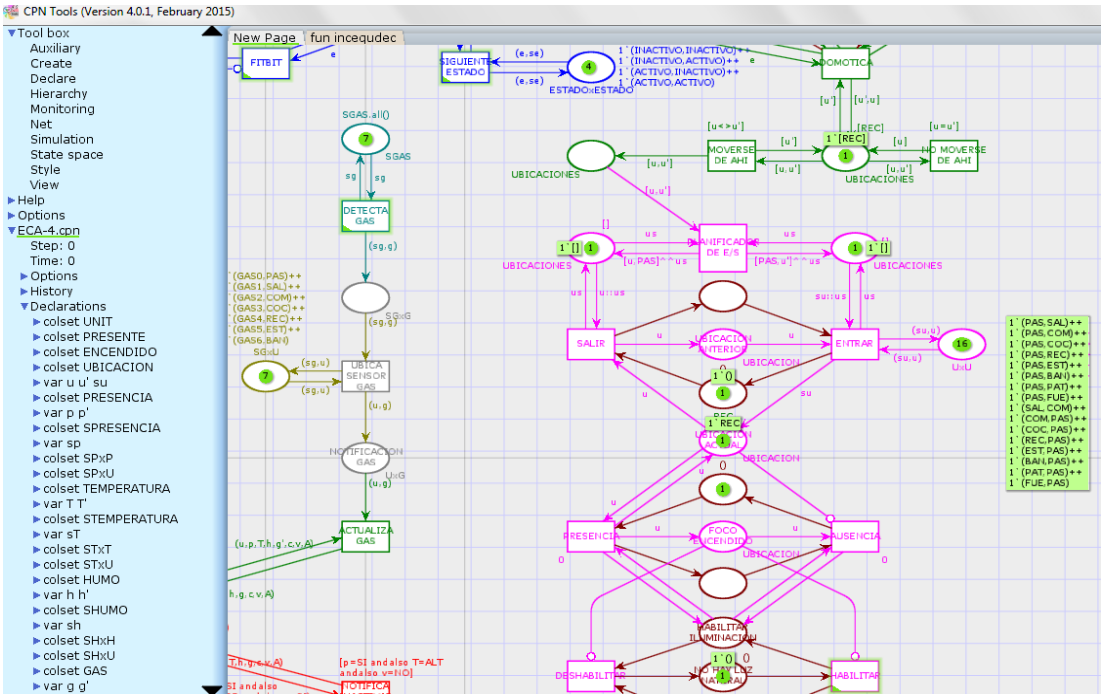


Figura 3.13 Módulo de control de luces.

3. **Módulo de domótica e invocación de servicios de emergencia.** Este módulo es el encargado de la seguridad de la casa ante un posible incendio o fuga de gas y sobre la integridad del paciente. En la Figura 3.14 se presenta el modelado de este módulo el cual monitoriza y vigila la temperatura de las habitaciones, la presencia de humo y/o gas que son factores principales en la generación de un incendio.

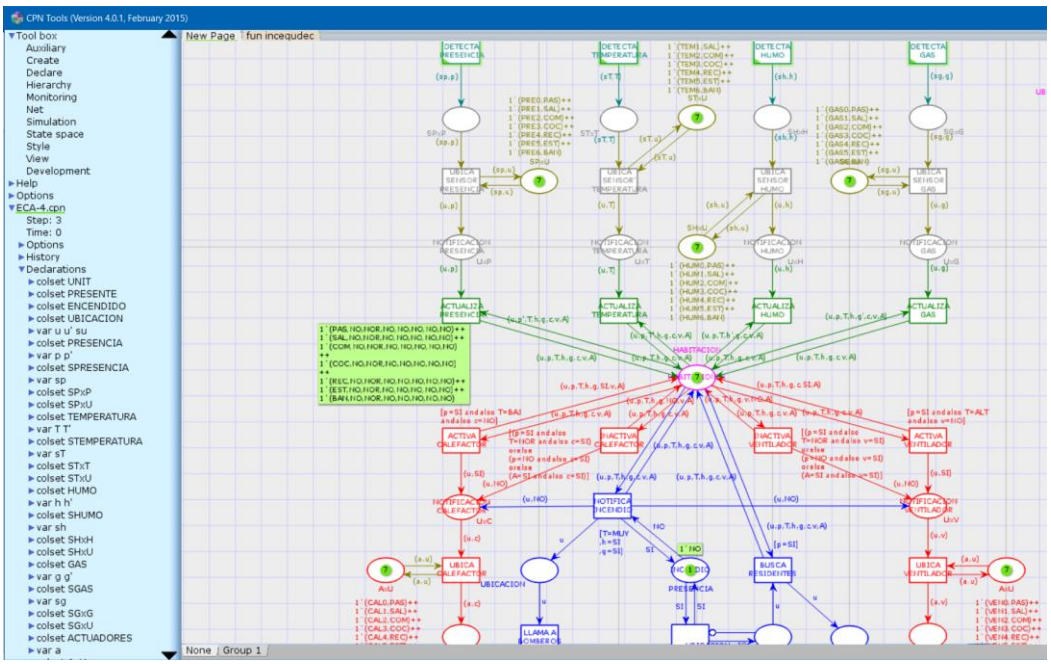


Figura 3.14 Módulo de domótica e invocación de servicios de emergencia

En caso de detectarse la generación de un incendio, este módulo es el encargado de solicitar de manera automática los servicios de emergencia, ya sea el servicio de bomberos para atender la emergencia del incendio y/o el servicio de ambulancia si el paciente se encuentra dentro de la casa durante el incendio.

4. **Módulo coordinador de servicios en el IoT.** Este módulo es el encargado de coordinar los otros módulos descritos anteriormente de acuerdo a las actividades y necesidades del paciente. En la Figura 3.15 se presenta el modelado de este módulo en donde, de acuerdo a las actividades a realizar por el paciente, este módulo se vincula con el módulo de control de luces y el módulo de domótica e invocación de servicios de emergencia de acuerdo a la habitación en la que se encuentre el paciente.

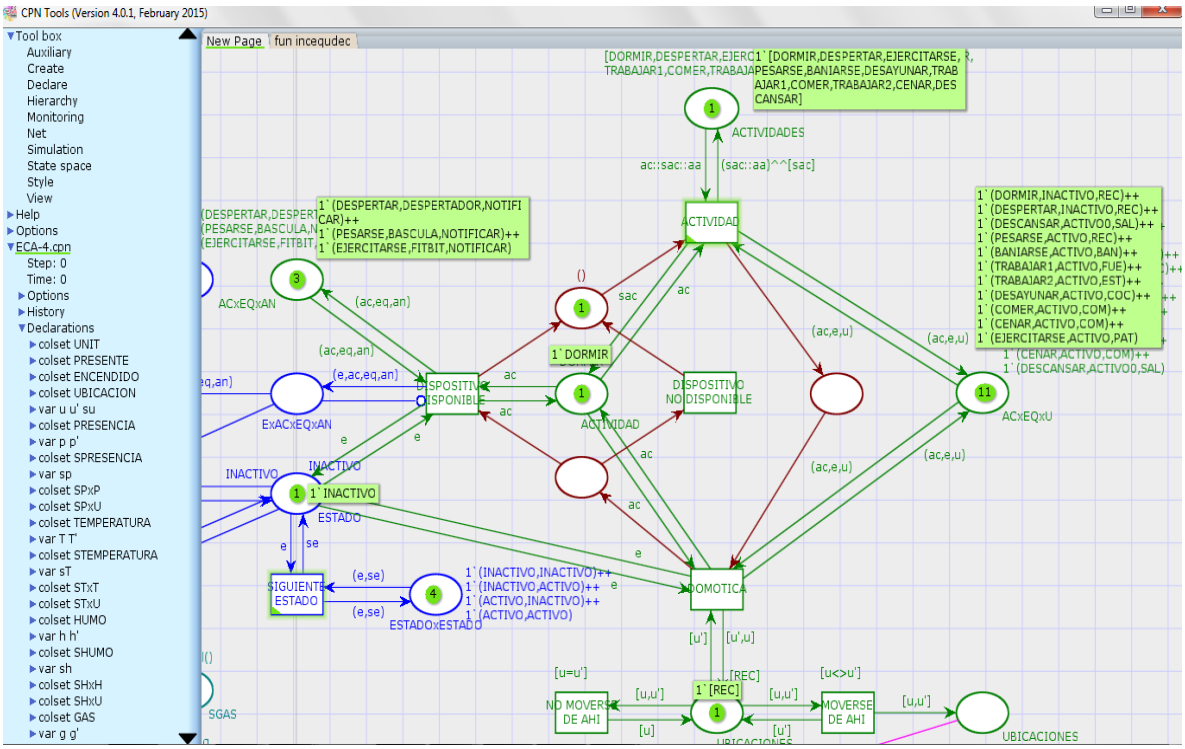


Figura 3.15 Módulo coordinador de servicios en el IoT

Así mismo, se vincula con el módulo de control de peso e invocación de servicios de salud para que de acuerdo a la actividad que realice el paciente, realice la monitorización de los datos médicos, peso o invocación de un servicio de salud en caso de ser necesario.

3.3.6 Diseño de la formalización del mecanismo de composición de servicios

Las extensiones a BPEL del modelo SCM-IoT se representaron mediante CPN, lo que permite visualizar, validar y verificar los sistemas construidos con este modelo. Una CPN consiste en un grafo bipartito dirigido con anotaciones sobre sus elementos. En una CPN, los nodos son de dos tipos distintos, conocidos como *lugares y transiciones*, los cuales se visualizan mediante elipses y rectángulos, respectivamente. Los arcos entre los nodos del grafo dirigido cumplen la restricción de que solamente conectan a nodos de distinto tipo. A los lugares y a sus arcos respectivos que inciden en una transición se les conoce como *lugares y arcos de entrada* de la transición, respectivamente. De la misma forma, a los lugares y a los arcos respectivos que inciden en ellos, partiendo de una transición, se les conoce como *lugares y arcos de salida* de la transición. Para modelar un sistema concurrente, los lugares fueron concebidos para almacenar recursos, mientras que las transiciones fueron concebidas para mover recursos de unos lugares a otros, posiblemente transformándolos durante su movimiento. Los arcos de entrada y salida de una transición establecen entonces la dirección del movimiento y transformación de recursos, llevándolos desde los lugares de entrada hacia los lugares de salida de la transición.

Sobre los elementos de una red de Petri se escriben anotaciones en algún lenguaje formal, como la aritmética, o de programación, como el lenguaje funcional SML (*Standard Meta Lenguaje*), de acuerdo con reglas sintácticas y semánticas bien definidas. A la anotación sobre los lugares se le conoce como *marcado*. El marcado de un lugar se visualiza como una colección de *marcas* que corresponden a los elementos contenidos en ese lugar. Por otra parte, las anotaciones sobre los arcos son expresiones que usan variables para describir marcas genéricas las cuales se concretan cuando a dichas variables se les vincula un valor. Para una transición, una anotación sobre un arco de entrada expresa las características generales que tienen algunas de las marcas contenidas en su lugar de entrada, mientras que una anotación sobre un arco de salida establece similarmente las características generales de las marcas que son transformados y/o depositados por la transición. En la Figura 3.16 se muestra una CPN que representa un ejemplo de formalización en donde se percibe el funcionamiento y vinculación de transiciones, lugares y arcos en el entorno de desarrollo CPN Tools.

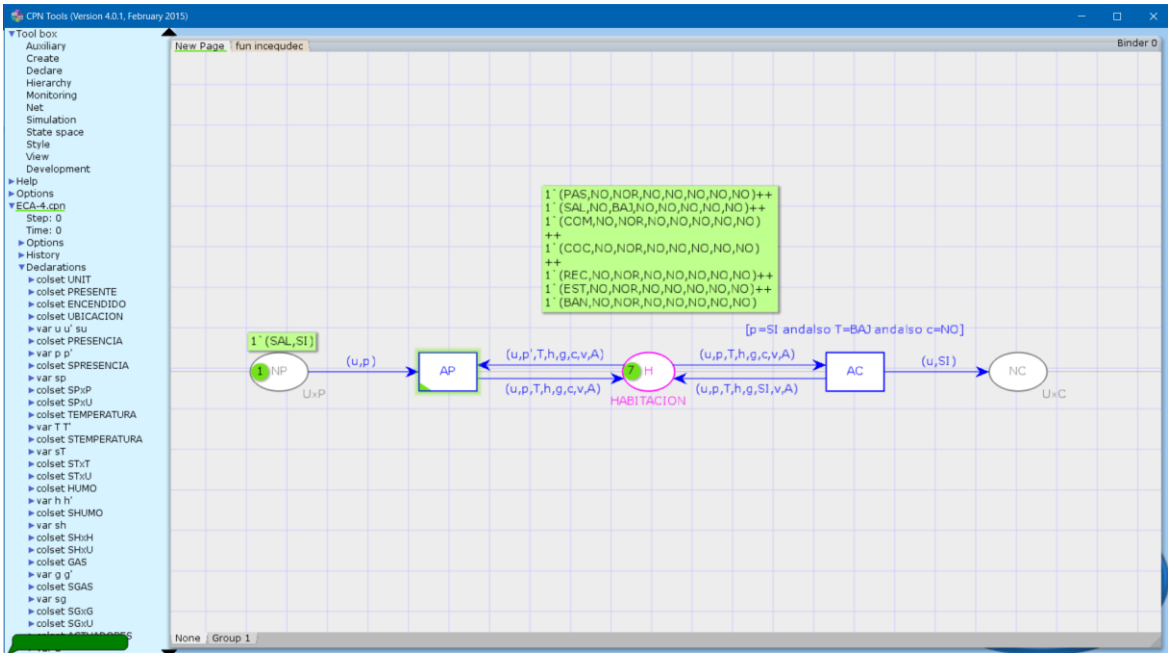


Figura 3.16 Formalización en una CPN

Por otra parte, en la Figura 3.16 se muestra una red de Petri coloreada formada por:

- **Tres lugares con nombres, NP, H y NC, de tipo UxP, HABITACION y UxC, respectivamente.** El tipo determina la estructura de las marcas que contiene este lugar. El tipo UxP está formado por todos los pares de UBICACION y PRESENCIA. El tipo UBICACIÓN a su vez corresponde con los nombres de las habitaciones, mientras que el tipo PRESENCIA está formado por los valores de presencia SI, NO. La marca 1`(SAL,SI) indica que este lugar solo contiene una marca con valor (SAL,SI). El tipo HABITACION está formado por las tuplas de ocho elementos que describen el estado de una habitación. El lugar H contiene ocho marcas, una de los cuales es (SAL,NO,BAJ,NO,NO,NO,NO,NO) que describe el estado de la sala.
- **Dos transiciones con nombres AP y AC.** La transición AP tiene dos lugares de entrada, NP y H, un lugar de salida, H. Observe que H es lugar de entrada y salida a la vez. Similarmente, la transición AC tiene a su vez un lugar de entrada, H, y dos lugares de salida H y NC. Observar también que H pertenece a los lugares de entrada de ambas transiciones por lo que se encuentran en conflicto.
- Para la transición AP, el arco de entrada que sale del lugar NP tiene la anotación (u,p), en donde u y p son variables de tipo UBICACIÓN y PRESENCIA, mientras

que el arco de entrada que sale del lugar H tiene inscrita la marca genérica (u,p',T,h,g,c,v,A), en donde u,p,T,h,g,c,v y A son todas variables de sus tipos respectivos. Observar que ambas anotaciones usan la misma variable u. Esta restricción sirve para indicar que el sensor se encuentra en la habitación representada por la marca (u,p',T,h,g,c,v,A). Así mismo, es importante visualizar que las substituciones u=SAL, p'=NO, T=BAJ, h=NO, g=NO, c=NO, v=NO, A=NO aplicadas a la marca genérica (u,p',T,h,g,c,v,A) producen la marca (SAL,NO,BAJ,NO,NO,NO,NO,NO), la cual describe el estado de la sala.

Una transición está *habilitada* cuando cada uno de sus lugares de entrada contiene las marcas indicadas en las anotaciones de sus arcos de entrada correspondientes. Una transición habilitada es *disparada* eventualmente en tanto permanezca habilitada. El *disparo* de una transición produce un cambio en la distribución en el marcado de los lugares de entrada y de salida de la transición. El disparo de una transición ocurre en dos etapas, asumiendo que la transición está habilitada:

- **Selección:** para cada lugar y arco de entrada de la transición, obtener las marcas del lugar que cumplen la condición anotada en el arco correspondiente, las cuales en su conjunto son las que habilitan la transición. De este conjunto se selecciona una marca de cada lugar de modo que esta selección de marcas sea habilitadora de la transición
- **Substitución:** para cada arco de entrada, determinar la substitución de las variables que aparecen en su anotación de modo que permitan seleccionar a las mismas marcas seleccionadas
- **Eliminación:** para cada lugar y arco de entrada de la transición, eliminar las marcas seleccionadas del lugar
- **Inserción:** para cada lugar y arco de salida de la transición, crear e insertar las marcas que se indican en la anotación del arco correspondiente.

Dichas etapas se realizan de manera indivisible, es decir, ambas etapas, o bien se realizan de manera completa sobre todos los lugares de entrada y salida involucradas, o bien no se realiza ninguna.

3.4 Etapa de Desarrollo

En esta etapa se presenta el desarrollo de un escenario de composición de servicios en el dominio de aplicación del IoT del cuidado de la salud, el desarrollo de un escenario de composición de servicios en el dominios de aplicación del IoT personal y social, el desarrollo de la integración y coordinación de servicios en el contexto del IoT, la implementación de la formalización del mecanismo de composición de servicios en el IoT, las pruebas y la validación del mecanismo de composición de servicios en un escenario del IoT.

3.4.1 Desarrollo del escenario de composición de servicios de salud en el IoT

Para el escenario de la composición de servicios de salud del caso de estudio en el dominio de aplicación del IoT del cuidado de la salud en donde los adultos mayores con obesidad necesitan saber la cantidad de calorías ingeridas, las calorías quemadas, el sueño, la frecuencia cardíaca, la cantidad de agua consumida, los pasos diarios y la actividad física necesaria para lograr una pérdida de peso gradual sin complicaciones de salud, lo que ayuda a evitar un infarto de miocardio, se desarrolló una aplicación para dispositivos móviles llamada PISIoT: plataforma de salud inteligente basada en IoT y el aprendizaje automático (*Machine Learning*) para el control del sobrepeso u obesidad), la cual pretende motivar a las personas al auto cuidado para contribuir en la prevención y control del sobrepeso u obesidad. El desarrollo de PISIoT refleja la coordinación de servicios del escenario de composición de servicios en el dominio de aplicación del cuidado de la salud. La motivación del desarrollo de PISIoT fue debido a que las aplicaciones sanitarias basadas en IoT mejoran la calidad de la atención de un paciente a un bajo costo. Actualmente, los dispositivos inteligentes en el IoT se utilizan para monitorear las variables biomédicas de los pacientes. Además, el progreso en las telecomunicaciones ha facilitado significativamente el uso de soluciones basadas en IoT en el sobrepeso y la obesidad (Ouaddah et al., 2017). Del mismo modo, los dispositivos portátiles permiten recopilar variables biomédicas en tiempo real (por ejemplo, niveles de estrés, calidad del sueño, niveles de azúcar en la sangre, frecuencia cardíaca y calorías quemadas) independientemente de dónde se encuentre el paciente (en

casa o en el hospital) (Goudos et al., 2017). Desde esta perspectiva, la atención médica se beneficia enormemente del IoT porque diferentes soluciones están vinculadas a sensores o dispositivos inteligentes, mejorando la atención del paciente y la eficiencia del personal de salud (Trappey et al., 2017).

PISIoT es una plataforma de salud inteligente que utiliza dispositivos basados en IoT y técnicas de aprendizaje automático para ayudar a controlar el peso y la obesidad. Adicionalmente, PISIoT permite el monitoreo en tiempo real de las variables biomédicas de un paciente a través de dispositivos portátiles y dispositivos inteligentes. Todos los datos recopilados se procesan y analizan utilizando algoritmos de aprendizaje automático para identificar variables críticas y hacer recomendaciones para la pérdida o el control del peso de los pacientes.

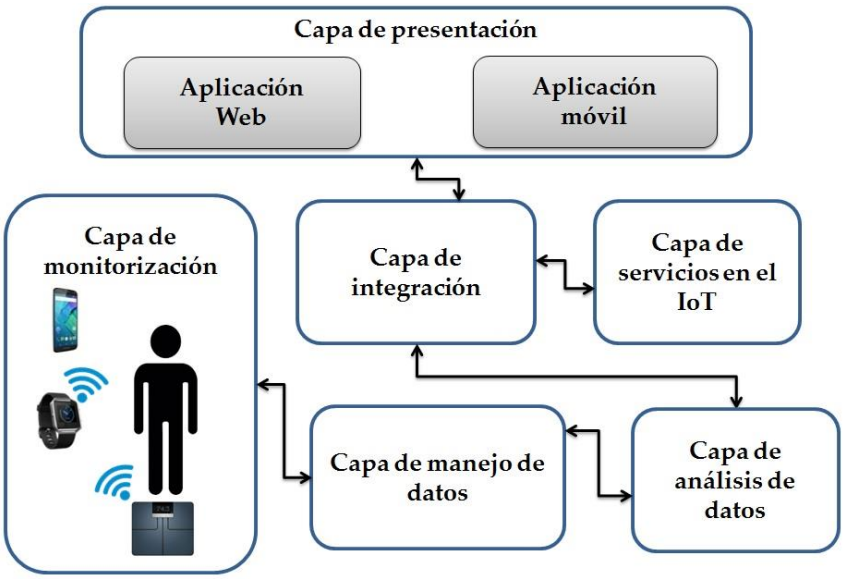


Figura 3.17 Arquitectura general de PISIoT

La PISIoT se basa en una arquitectura en capas, que proporciona una definición clara y una descripción de las actividades y funciones de cada módulo. Esto facilita el mantenimiento y permite una alta escalabilidad. La Figura 3.17 presenta la arquitectura PISIoT, que se compone de seis capas: la capa de presentación, la capa monitorización, la capa de integración, la capa de manejo de datos, la capa de análisis de datos y la capa de servicios basados en IoT. Cada capa está compuesta de varios componentes con una funcionalidad y relación específica. A continuación se presenta una descripción general de las capas.

Capa de monitorización

Esta capa está compuesta por diferentes dispositivos basados en IoT, como dispositivos portátiles e inteligentes vinculados a equipos de telecomunicaciones, que tienen interfaces de comunicación que facilitan el intercambio de información. Estos dispositivos permiten recopilar información sobre las variables biomédicas del paciente (frecuencia cardíaca, calorías quemadas, sueño, minutos de actividad física y peso), en función de las actividades realizadas durante un día. Toda la información recopilada se envía a la capa de gestión de datos para su posterior procesamiento y análisis.

Es importante considerar la disponibilidad de una plataforma confiable, las características particulares del dispositivo (marca, modelo y duración de la batería) y los permisos de acceso a datos por parte de los proveedores para garantizar un monitoreo óptimo de las variables biomédicas de los pacientes. Por otro lado, la adquisición de datos por dispositivos inteligentes depende de varios factores, como la compatibilidad entre los dispositivos utilizados, la portabilidad del dispositivo portátil durante el día, el posicionamiento correcto del dispositivo portátil por la noche, el informe continuo del peso y la honestidad de los pacientes en informar los alimentos y la cantidad de agua ingeridos. Por esta razón, se requiere una red de monitoreo que permita a la PISIoT monitorear y recolectar las variables biomédicas de los pacientes e identificar hábitos alimenticios diarios y actividad física con el propósito de motivar a los pacientes a mantener o reducir su peso, contribuyendo así a reducir la prevalencia de sobrepeso y obesidad. Por lo tanto, PISIoT utiliza una red de monitoreo basada en IoT que permite a los dispositivos inteligentes conectarse y comunicarse. El objetivo principal de la red de monitoreo es recolectar variables biomédicas y otras variables de los pacientes desde dispositivos portátiles. En particular, la red de monitoreo está compuesta por dispositivos inteligentes basados en IoT que se dividen en tres categorías:

1. Dispositivos portátiles o *wearables*. Los dispositivos portátiles son responsables de monitorear y recopilar la mayoría de las variables biomédicas, así como otras variables generadas a partir de las acciones del paciente. Los dispositivos portátiles en IoT proporcionan una infraestructura de alta tecnología que permite la comunicación y los enlaces entre sensores portátiles para monitorear las actividades de una persona, incluidas, entre otras, las variables biomédicas, el comportamiento y el bienestar de la persona, con el fin de mejorar la calidad de vida (Hiremath et al.,

2014). Los dispositivos portátiles se clasifican según su apariencia, funcionalidad, portabilidad en el cuerpo, características y capacidad funcional para proporcionar una mejor descripción de los diferentes sectores. Por esta razón, (Mardonova y Choi, 2018) clasificaron los dispositivos portátiles en smartwatch, gafas inteligentes, rastreador de actividad física, ropa inteligente, cámara portátil y dispositivo biomédico portátil. La PISIoT funciona con dispositivos de algunos proveedores, pero su escalabilidad permite considerar dispositivos portátiles de otros proveedores.

2. Dispositivos inteligentes. Estos dispositivos son responsables de la recolección inteligente de peso u otras variables no identificadas por dispositivos portátiles (por ejemplo, báscula inteligente, sensor de temperatura, sensor de movimiento). Los dispositivos inteligentes son la fuerza impulsora del IoT porque proporcionan información rápida y precisa en tiempo real e identifican varios patrones. En este sentido, la PISIoT incluye algunos proveedores de dispositivos inteligentes, pero garantiza la incorporación de otros proveedores de dispositivos inteligentes.
3. Teléfonos inteligentes o *Smartphones* Los teléfonos inteligentes se encargan de mantener la comunicación entre los dispositivos, los proveedores y la PISIoT. Los dispositivos portátiles utilizados en IoT carecen de un sistema operativo como tal; por lo tanto, requieren estar conectados a un teléfono inteligente o una computadora con acceso a Internet porque la información recopilada por los dispositivos necesita estar protegida por el esquema de almacenamiento y soporte de la PISIoT y de sus proveedores. La PISIoT es multiplataforma; sin embargo, de acuerdo al modelo y el proveedor del dispositivo portátil que desee utilizar, respeta las reglas de compatibilidad entre cada modelo con la variedad de teléfonos inteligentes.

Capa de manejo de datos

Esta capa es responsable del almacenamiento y la copia de seguridad del historial médico del paciente y de los datos recopilados por los dispositivos utilizados.

Capa de análisis de datos

Esta capa es responsable de identificar variables críticas y generar recomendaciones médicas. Además, esta capa protege la información útil para la PISIoT y el historial médico de

los pacientes. Esta capa identifica variables críticas y clasifica a los pacientes según su nivel de obesidad a través de la API Weka basada en la versión 3.8 de Java, ya que esta es la última versión estable y es un software de código abierto que es fácil de integrar con la PISIoT y cumple con los parámetros de funcionalidad requeridos. Además, para realizar la clasificación del IMC, identificar variables críticas y generar recomendaciones médicas, PISIoT utiliza el algoritmo de aprendizaje automático J48, que es una implementación de código abierto en Java del algoritmo C4.5. Esto significa que tanto C4.5 como J48 son algoritmos de clasificación utilizados para generar árboles de decisión. Los algoritmos de clasificación basados en árboles de decisión son útiles para el diagnóstico de hepatitis (Sathyadevi, 2011), cáncer del tracto biliar (Pattanapairoj et al., 2015) y cáncer de pulmón (Tartar et al., 2013); mientras que las máquinas de vectores de soporte son utilizadas para la predicción del crecimiento del cáncer (Chen et al., 2010) y el cáncer de ovario (Shoaip et al., 2017); el algoritmo Bayesiano ingenuo para la predicción de enfermedades cardíacas (Subbalakshmi et al., 2011) y la clasificación de enfermedades oculares (Fageeri et al., 2017); las redes neuronales artificiales para diferenciar los nódulos pulmonares malignos, benignos y avanzados (Hashemi et al., 2013), y en la clasificación de tumores en mamografías digitales (Abdalla et al., 2011) y diagnóstico de cáncer de páncreas (Yang et al., 2014). Además, se seleccionó J48 porque se ha utilizado para construir modelos predictivos en problemas de clasificación similares y ha mostrado un mejor rendimiento que otros algoritmos. Por ejemplo, en (Kureshi et al., 2016), J48 produjo predicciones significativas y útiles con un mejor rendimiento (área más grande debajo de la curva) que otros algoritmos de árbol de decisión como *random forest* y modelos CART (árbol de clasificación y regresión). Además, en (Sood y Mahajan, 2018), se utilizó un árbol de decisión J48 para clasificar a los usuarios en diferentes categorías, y los resultados experimentales mostraron que el clasificador del árbol de decisión J48 tiene mayor precisión y un menor tiempo de respuesta para determinar la categoría de un usuario en comparación con otros algoritmos de clasificación como *REPTree*, *fuzzy C-means* y *random tree*. Por otro lado, PISIoT se desarrolló e implementó de manera modular y genérica, con miras a un alto rendimiento, facilidad de implementación y una mejor extensibilidad de la aplicación. La Figura 3.18 presenta las cuatro clases desarrolladas para el módulo de aprendizaje automático.

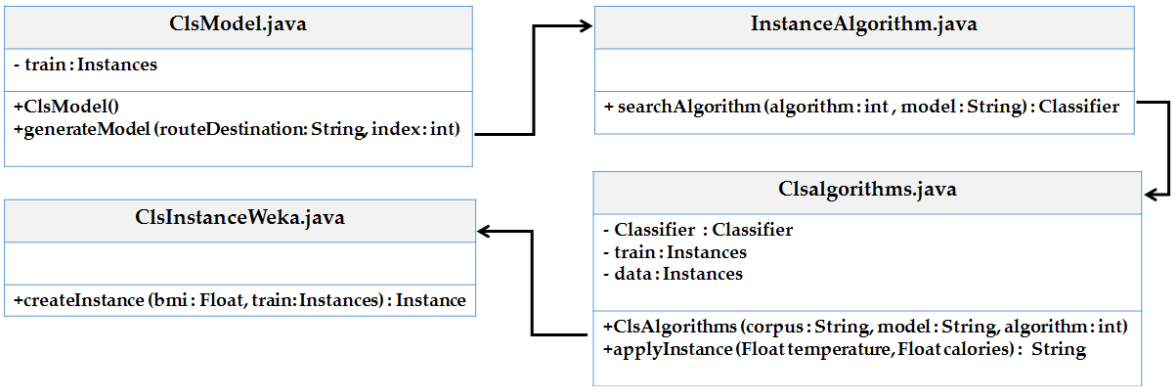


Figura 3.18 Clases del módulo de aprendizaje automático.

1. ClsModel.java sirve principalmente para generar el modelo a partir del conjunto de entrenamiento. Además, contiene dos métodos, uno para recibir el conjunto de datos y otro para generar el modelo basado en el conjunto de datos recibido. La clase comienza con una instancia del tipo "*Instances*" llamada "*train*". Asimismo, se envía el constructor ClsModel que, a través del método generateModel, indica el destino y el algoritmo utilizado. Además, se invoca el método searchAlgorithm.
2. InstanceAlgorithm.java contiene la selección del algoritmo a utilizar. El método searchAlgorithm recibe la selección del algoritmo y la ruta del modelo.
3. Clsalgorithms.java procesa el conjunto de datos, el modelo y la información entrante para aplicar el algoritmo correspondiente, además de proporcionar una respuesta oportuna a la solicitud. Esta clase comienza con un clasificador del tipo "*Classifier*", un *train* del tipo "*Instances*" y un objeto de datos del tipo "*Instances*". Además, utiliza el método applyInstance para aplicar los cambios realizados y designar la información a analizar. El método ClsAlgorithms también recibe el modelo, el conjunto de datos y el algoritmo para procesar la información y generar una respuesta.
4. ClsInstanceWeka.java genera una instancia basada en la información entrante; esta clase es responsable de seleccionar el conjunto de parámetros a procesar. El método CreateInstance usa el objeto *Instances* y la cantidad de elementos contenidos en el conjunto de datos, ignorando la etiqueta de la clase.

Por otra parte, para llevar a cabo las recomendaciones, se consideran los valores diarios y los promedios semanales de cada variable. Esto permite establecer las reglas (valores máximos y mínimos permitidos por variable) en el algoritmo. Por lo tanto, cada día, el paciente se clasifica según el IMC como normal, sobrepeso, obesidad 1, obesidad 2 u obesidad 3.

Una vez que se ha identificado el tipo de clasificación utilizando J48, las reglas establecidas se seleccionan según el tipo de obesidad. Además, el peso ideal de la persona se calcula en función de la edad y la altura, a fin de generar recomendaciones avaladas por profesionales de la salud de acuerdo con normas preestablecidas. Las reglas generadas identifican y describen cómo funciona el proceso de recomendación, que tiene como objetivo mejorar la calidad de vida de una manera simple, progresiva y no invasiva utilizando los valores obtenidos por los dispositivos e ingresados por el paciente. Apache Mahout y RuleML se utilizaron para generar recomendaciones médicas. Apache Mahout es una biblioteca de software gratuita que ofrece implementaciones escalables de algoritmos de aprendizaje automático. RuleML es un lenguaje basado en XML que sirve para especificar el intercambio inmediato de reglas. Además, junto con las recomendaciones, los servicios médicos sugeridos se muestran según el progreso del paciente o el estado de salud.

Capa de servicios basados en IoT

Esta capa es responsable de vincular, invocar, seleccionar y confirmar la disponibilidad de servicios basados en IoT. Adicionalmente, esta capa describe el conjunto de servicios REST desarrollados con el propósito de proporcionar a la plataforma información para quienes la solicitan y tienen permisos de acceso. Esto facilita el desarrollo de futuras aplicaciones utilizando la información proporcionada por la plataforma. Los servicios se dividen en información descargable de proveedores de dispositivos portátiles y servicios de variables biomédicas, recomendaciones, servicios basados en el IoT y servicios asociados con otras variables de pacientes almacenadas en la plataforma.

Los principales servicios REST desarrollados para descargar datos de los proveedores fueron "downloadSleep" para el sueño, "downloadWeight" para el peso, "downloadSteps" para los pasos y "downloadHeart" para la frecuencia cardíaca. Del mismo modo, se desarrollaron otros servicios REST para descargar los datos de cada variable monitoreada por los dispositivos inteligentes. Para invocar los servicios, la biblioteca jQuery se usa para llamadas asíncronas, lo que genera una respuesta en formato JSON que se compone de dos nodos. Un nodo corresponde al error y toma el valor de "0" si todo fue satisfactorio y "1" si ocurrió un error. El segundo es un nodo de mensaje que indica por qué se produjo el error, si lo hubo, o proporciona un aviso de proceso completo cuando la operación es satisfactoria. En la Figura 3.19 se presentan los diversos servicios REST que proporciona la PISIoT, los cua-

les son la base para realizar el proceso de orquestación y coreografía de servicios bajo el contexto del IoT.

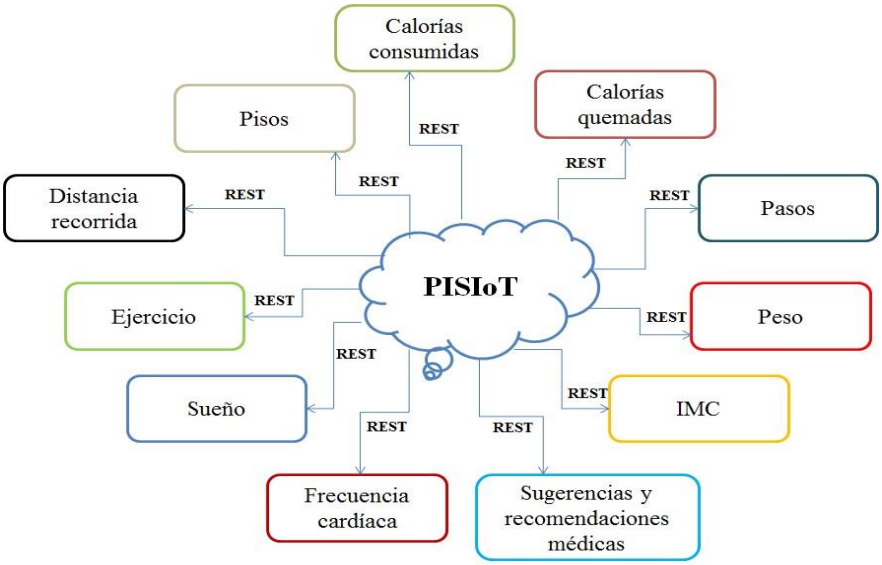


Figura 3.19 Servicios REST proporcionados por la PISIoT.

Por otra parte, la PISIoT se desarrolló para vincularse en primera instancia con los WS tradicionales pero sobre todo con los servicios en IoT, con los cuales colabora para ofrecer mayores alternativas y un mejor servicio a los usuarios. Adicionalmente, la PISIoT permite visualizar el historial de los datos del paciente a través de gráficos y tablas dinámicas, gracias a que se desarrolló en HTML5.

Capa de integración

Esta capa está compuesta por los proveedores de dispositivos portátiles e inteligentes y recibe los datos y consultas del paciente para generar las respuestas solicitadas. Además, esta capa es responsable de solicitar servicios de la capa de servicios basada en IoT con base en las recomendaciones de PISIoT. Adicionalmente, esta capa es responsable de hacer la solicitud de monitoreo de datos obtenidos por el dispositivo portátil o el dispositivo inteligente al proveedor correspondiente. Esto se logra a través de un *token* de acceso y siguiendo las políticas de acceso y permisos de cada proveedor, para luego enviar los datos a la capa de gestión de datos para su análisis y almacenamiento. La Figura 3.20 muestra el flujo de trabajo general de PISIoT implementado en esta capa.

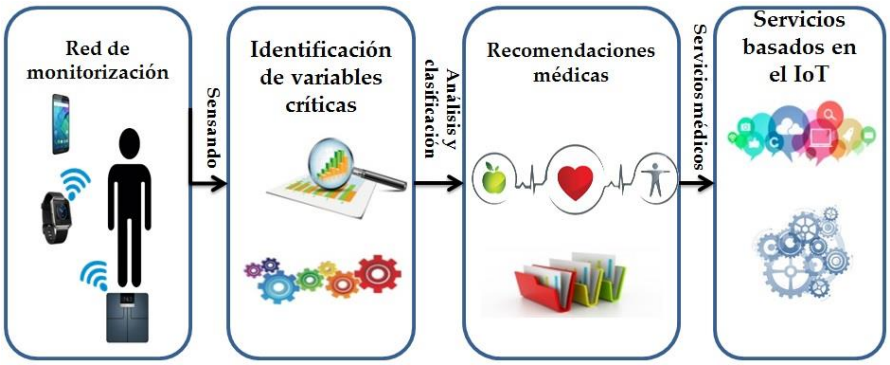


Figura 3.20 Flujo de trabajo general implementado en PISIoT.

Como se observa, los dispositivos portátiles juegan un papel importante en el monitoreo de varias variables, particularmente variables biomédicas, ya que recopilan datos en tiempo real y de manera no invasiva, permitiendo el análisis e identificación de variables críticas para luego generar recomendaciones médicas e invocar servicios médicos (análisis clínico, nutrición, cardiólogo y médico general). Los pacientes visualizan, en cualquier momento, su progreso o los datos del día y ajustan y optimizan su consumo de alimentos o actividad física de acuerdo con las recomendaciones proporcionadas por la plataforma. Además, en esta capa, se reciben las consultas del paciente, lo que significa que cada vez que el paciente accede a la plataforma PISIoT, se genera una solicitud de información para visualizar las variables biomédicas y todas las demás variables monitoreadas por los dispositivos portátiles y los dispositivos inteligentes. Además, según las recomendaciones de los servicios médicos realizados en la capa de manejo de datos, esta capa se encarga de solicitar los servicios médicos disponibles de la capa de servicios basados en el IoT para mostrarlos a los pacientes para que seleccionen los más adecuados según la disponibilidad de tiempo, confianza y costo.

Capa de presentación

A través de esta capa, la comunicación se realiza entre el paciente y la plataforma, lo que facilita una aplicación *Web* y móvil a través de la cual los pacientes visualizan y siguen sus variables biomédicas, los servicios disponibles basados en el IoT, el historial médico y las recomendaciones. Además, la plataforma permite la entrada manual de agua natural y los alimentos consumidos durante el día. Cuando los pacientes ingresan alimentos o bebidas consumidos utilizando un teléfono inteligente o una computadora con acceso a Internet vin-

culado al dispositivo portátil, las calorías, los carbohidratos, las grasas y las proteínas se obtienen automáticamente de la base de datos del proveedor del dispositivo. Sin embargo, en PISIoT, es posible ingresar alimentos o bebidas no incluidos en la base de datos del proveedor, además de cualquier actividad física realizada por el paciente.

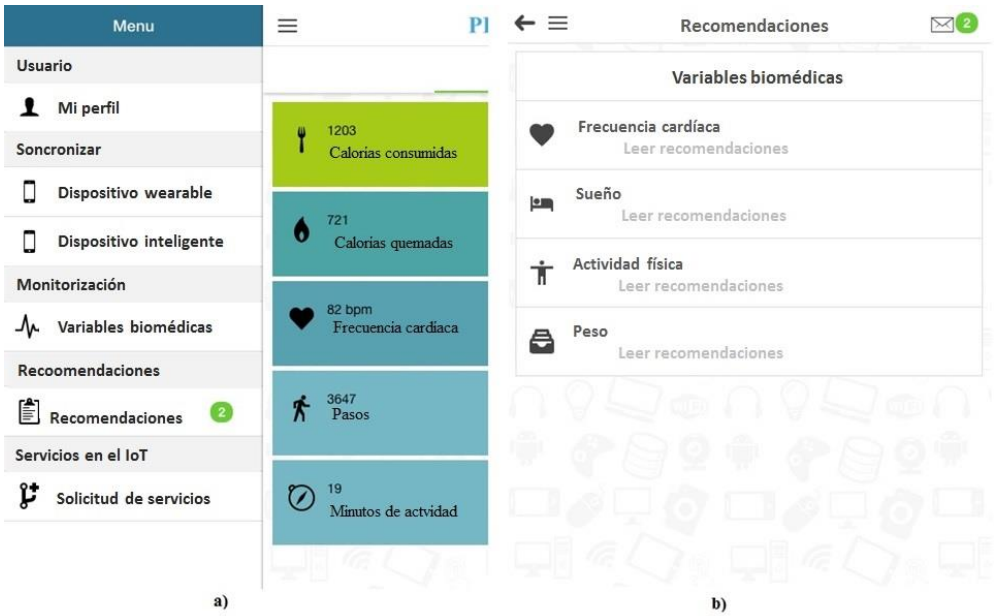


Figura 3.21 Principales interfaces de usuario de PISIoT: (a) índice; (b) recomendaciones.

Adicionalmente, PISIoT proporciona una aplicación para dispositivos móviles que permite a los pacientes interactuar con la plataforma. La Figura 3.21 inciso a) muestra la interfaz principal donde se observan las variables de las calorías consumidas, las calorías quemadas, la frecuencia cardíaca, los pasos y los minutos de actividad. Además, tiene un menú para acceder al perfil del paciente, los tipos de dispositivos sincronizados o la opción de agregar uno nuevo, una opción para ver las variables biomédicas monitoreadas, recomendaciones y servicios basados en el IoT. Además en la misma Figura 3.21 inciso b) se muestran las recomendaciones generadas para cada una de las variables biomédicas (frecuencia cardíaca, sueño, calorías quemadas, minutos de actividad física y peso).

Además, en la Figura 3.22 inciso a) se presenta un gráfico que muestra la correlación de las variables biomédicas (frecuencia cardíaca, sueño, calorías quemadas, peso y minutos de actividad física) con otras variables recopiladas por el dispositivo portátil y el dispositivo inteligente (pasos, pisos y agua consumida). Adicionalmente, se muestra un botón para ingresar al servicio médico del paciente. Así mismo, en la misma Figura 3.22 inciso b) se muestran los servicios médicos disponibles (análisis clínicos, nutriólogo, cardiólogo y mé-

dico general). Gracias al uso de tecnologías en el IoT y a las técnicas de aprendizaje automático, la PISIoT permite a los pacientes visualizar de manera sencilla, fácil de usar y en tiempo real las variables biomédicas y su clasificación, junto con recomendaciones médicas para la pérdida o el control de peso. Además, la aplicación para dispositivos móviles también proporciona un conjunto de interfaces de usuario que permiten a los pacientes ver y solicitar servicios médicos disponibles en el IoT, como servicios de análisis clínicos o servicios de nutrición, cardiólogo o médico general. Además, los pacientes son monitoreados por su médico de confianza y un familiar vinculado con su perfil.

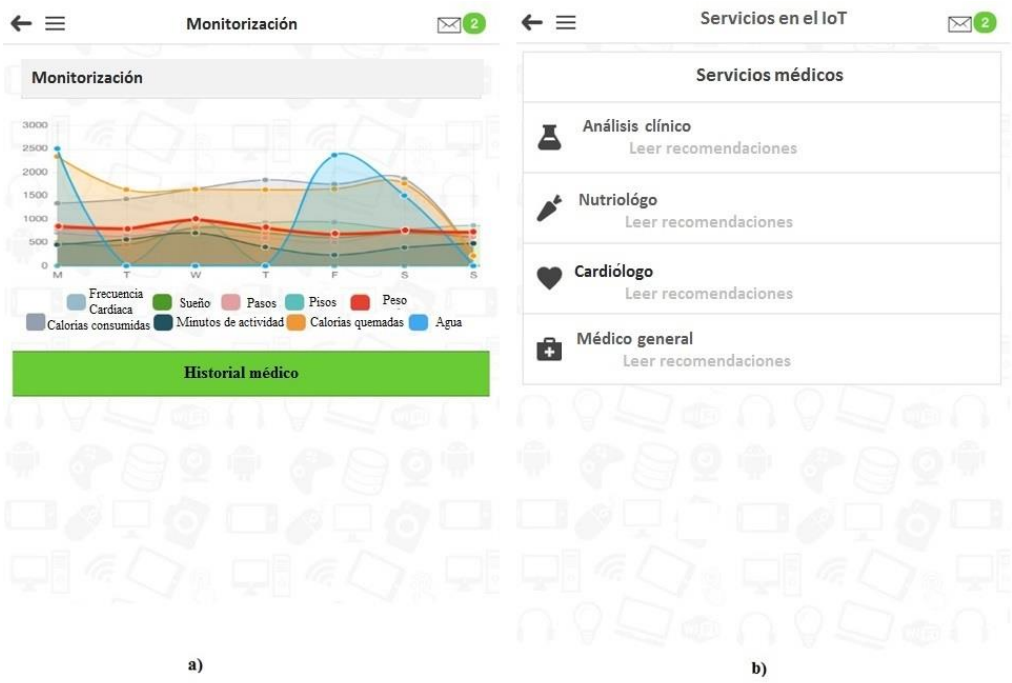


Figura 3.22 Principales interfaces de usuario de PISIoT: (a) monitorización; (b) Servicios médicos basados en el IoT.

El caso de estudio de la composición de servicios de salud en el IoT, se realizó para controlar a 40 personas mayores obesas de 60 a 80 años (20 mujeres y 20 hombres) que mostraron síntomas asociados con un infarto de miocardio (dolor en el centro del tórax, dificultad para respirar, entumecimiento o dolor en el brazo derecho, sudoración, palidez, mareos) o ya habían experimentado un infarto de miocardio y necesitaban perder peso. Se asignó un dispositivo portátil a cada paciente para obtener las variables biomédicas (frecuencia cardíaca, sueño, calorías quemadas, peso y minutos de actividad física) y otras variables (pasos, pisos, calorías consumidas, distancia recorrida, agua consumida y ejercicio). Además, se asignó una báscula inteligente a cada paciente para registrar periódicamente el peso. Para

los fines de este caso de estudio, fue necesario obtener información sobre los hábitos alimenticios, el consumo de agua, los minutos de actividad, las calorías quemadas, el sueño, la cantidad de pasos por día y la frecuencia cardíaca, tanto antes como después de la implementación de PISIoT. Por esta razón, se siguieron los pasos que se describe a continuación:

- El monitoreo inicial se realizó durante un período de dos meses (agosto a septiembre de 2018) utilizando el dispositivo portátil; al final de ese período, se obtuvo el peso con la báscula inteligente. Ambos dispositivos se vincularon a un teléfono inteligente para enviar los datos al proveedor del dispositivo. Un familiar del paciente registró el consumo de alimentos y agua utilizando la plataforma del proveedor del dispositivo portátil. Durante este período, solo se utilizó la aplicación para dispositivos móviles para el dispositivo portátil y la báscula inteligente de su correspondiente proveedor. Estos se enfocaron solo en monitorear las variables biomédicas del paciente y no proporcionaron ningún tipo de alerta o recomendación médica para perder peso.
- Implementación de PISIoT. Posteriormente, se ejecutó un segundo período de monitoreo durante cuatro meses (octubre de 2018 a enero de 2019); las variables biomédicas de las personas mayores se solicitaron a los proveedores de los dispositivos. Posteriormente, el IMC de los pacientes se calculó utilizando la fórmula IMC = peso (kg) ÷ altura² (metros) (WHO-1, 2019). Luego, los pacientes se clasificaron para predecir la obesidad con las clases ClsModel.java, Clsalgorithms.java, ClsInstanceWeka.java e InstanceAlgorithm.java utilizando el algoritmo de aprendizaje automático J48. La Tabla 3.10 muestra la clasificación utilizada para determinar el tipo de obesidad según la Organización Mundial de la Salud (OMS) (WHO-2, 2019). Además, las variables biomédicas del paciente (frecuencia cardíaca, calorías quemadas, sueño, minutos de actividad física y peso) y otras variables (pasos, pisos, calorías consumidas, distancia recorrida, agua consumida y ejercicio) se consideraron en la clasificación para describir el comportamiento de los pacientes con obesidad. Con los datos obtenidos por el dispositivo portátil y la báscula inteligente, se creó un conjunto de datos con 17 atributos predictores (calorías consumidas, calorías quemadas, carbohidratos, grasas, proteínas, agua consumida, duración del ejercicio, frecuencia cardíaca durante el ejercicio, frecuencia cardíaca en reposo, minutos de actividad física a nivel máximo, minutos a nivel cardiovascular, minutos a nivel de quema grasas, pasos, pisos, distancia recorrida, duración del sueño y peso), un atri-

buto de etiqueta de clase (obesidad) y 7200 instancias; se registraron datos diarios para cada paciente y se clasificaron de acuerdo al tipo de obesidad que los pacientes presentaron.

Clasificación	Valor Mínimo	Valor Máximo
Normal	18.50	24.99
Sobrepeso	25.00	29.99
Obesidad 1	30.00	34.99
Obesidad 2	35.00	39.99
Obesidad 3	40.00	+

Tabla 3.10 Regla de clasificación según el IMC

- Esto permitió realizar un análisis para identificar posibles variables críticas que influyen en el surgimiento de la obesidad en las personas mayores, generar recomendaciones y proponer los servicios médicos basados en el IoT que los pacientes requieren para perder peso. El algoritmo J48 se adoptó en PISIoT utilizando el conjunto de datos para obtener el modelo predictivo con la técnica de validación cruzada 10 veces. Este tipo de validación se utilizó porque, en general, se recomienda para la precisión de la estimación (incluso si el poder computacional permite el uso de más pliegues) debido a su sesgo y varianza relativamente bajos (Han et al., 2012). Además, se seleccionó el algoritmo J48, ya que se demostró en estudios anteriores que se desempeñó mejor que otros algoritmos (Dugan et al., 2015) (Suca et al., 2016) (Daud et al., 2018) (De-La-Hoz-Correa et al., 2019).
- Una vez que se ha identificado el tipo de clasificación, se seleccionan las reglas establecidas según el tipo de obesidad. Del mismo modo, con la fórmula peso = (altura-40) / 2 (D'Hyver y Gutiérrez, 2014), se identifica el peso ideal del paciente, que sirve como base para identificar variables críticas que son mayores o menores que los valores permitidos en cada clasificación. La Figura 3.23 muestra las reglas de clasificación para la obesidad 1 y también incluye recomendaciones basadas en las variables críticas detectadas diariamente.

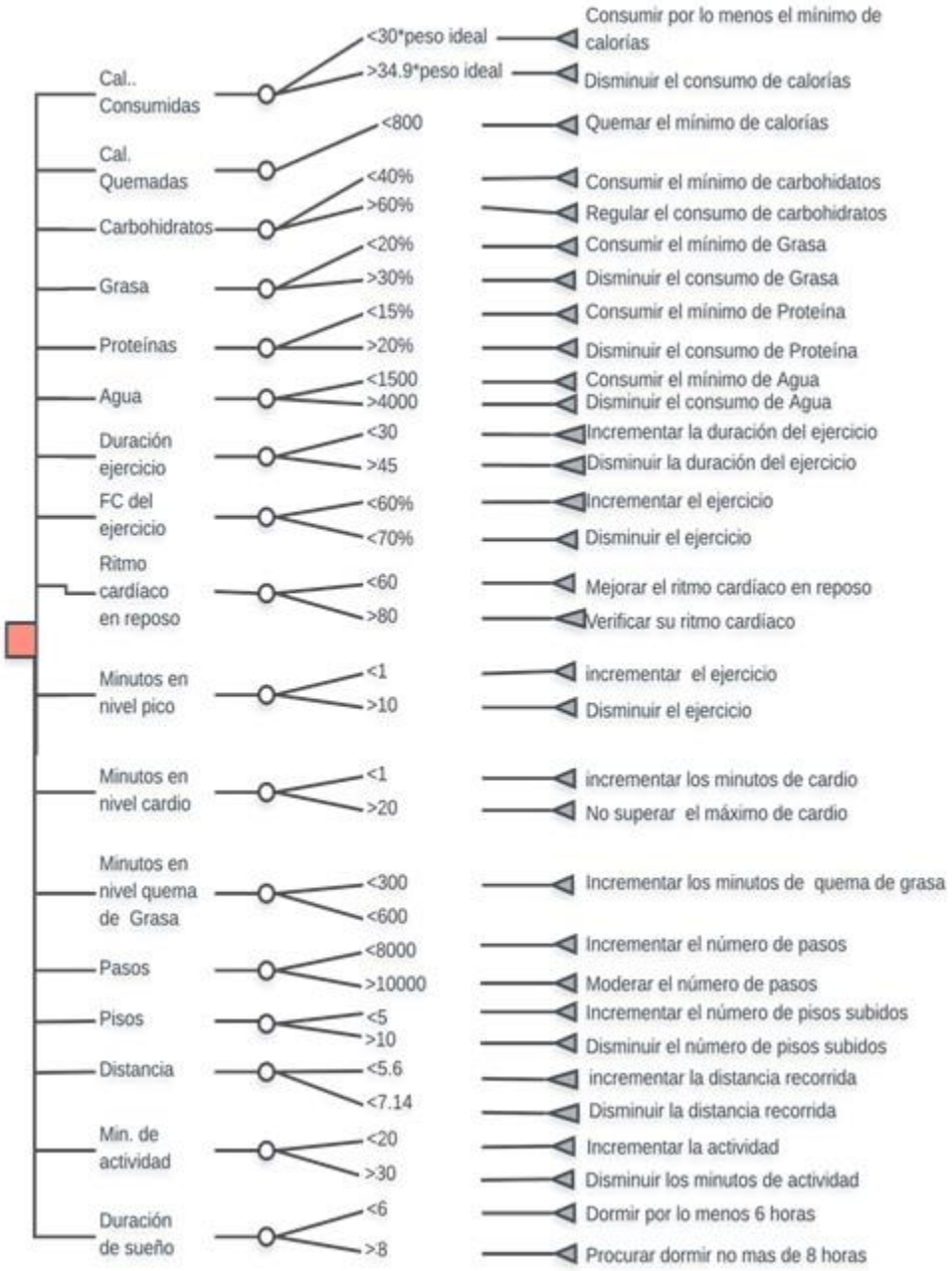


Figura 3.23 Árbol de reglas de recomendaciones para pacientes con obesidad 1.

- La sección de recomendaciones médicas es compleja, ya que influyen diferentes factores. Por esta razón, la disminución de calorías es gradual, es decir, porciones de 500 kcal cada 2 semanas para evitar cualquier descompensación. Se sugiere no intentar de acelerar el proceso o exceder los límites establecidos por la PISIoT.
- Si el paciente no sigue las recomendaciones diarias emitidas por la PISIoT, después de un período de tiempo, se genera una recomendación médica mayor. La Tabla 3.11 presenta las reglas para recomendaciones de pacientes con cualquier tipo de obesidad, que fueron hechas por personal de salud especializado (dos médicos, una

nutrióloga con una maestría en salud alimentaria y nutricional, y una enfermera con una maestría en salud pública). La tabla incluye la columna "Variable", que proporciona la variable utilizada; “Regla", que especifica la regla para la recomendación; "Frecuencia", que es la frecuencia con la que se realiza el análisis; y "Recomendación", que contiene la recomendación que se da si no se cumple la regla. Las reglas descritas anteriormente identifican y describen el funcionamiento del proceso de recomendación médica, cuyo objetivo es mejorar la calidad de vida de una manera simple, gradual y no invasiva, y el uso de los valores obtenidos por los dispositivos inteligentes y aquellos ingresados por el paciente o el familiar del paciente.

Variable	Regla	Frecuencia	Recomendación
Calorías quemadas	>(20 * peso ideal)	3 veces por semana	Cuidar la ingesta de alimentos debido al posible aumento en el IMC
Calorías consumidas	<1500	3 veces por semana	Solicitar servicio de análisis clínico y solicitar evaluación médica
Ejercicio	<75	Por semana	Realizar ejercicio
Frecuencia cardíaca	>100 ó <60	3 veces por semana	Ir a evaluación médica
Minutos en nivel máximo	>10	3 veces por semana	Ir a evaluación médica
Frecuencia cardíaca en reposo	>120 ó < 40	En todo momento	Solicitar ambulancia; posible taquicardia o infarto. Solicitar servicio de cardiología
Peso	(Peso) ^2 * 34.9	Por mes	Solicitar un servicio de nutriólogo para un nuevo plan nutricional.
Peso	(Peso)^2 * 30	Por mes	Solicitar un servicio de nutriólogo para un nuevo plan nutricional.
Sueño	<6	Cada 3 días	Es necesario dormir al menos 8 horas al día
Sueño	>8	Cada 3 días	Mantenerse activo

Tabla 3.11 Reglas de recomendaciones para pacientes con obesidad.

- Estas recomendaciones también incluyen los servicios médicos que los pacientes requieren para lograr su objetivo de control o pérdida de peso (análisis clínico, nutriólogo o cardiólogo). Las recomendaciones médicas y los servicios médicos sugeridos fueron validados por personal sanitario especializado. Los pacientes de edad avanzada monitoreados en este segundo período mostraron progreso en la pérdida de peso al usar la PISIoT, que realiza monitoreo en tiempo real, identifica variables críticas que conducen al aumento de peso, analiza variables biomédicas a través de

técnicas de aprendizaje automático, proporciona recomendaciones para la pérdida de peso, el cual es monitoreado y analizado por expertos en el área de la salud.

- Finalmente, los ancianos fueron monitoreados nuevamente durante cuatro meses más (febrero-mayo de 2019) para evaluar la contribución y el impacto del uso de la PISIoT en la pérdida de peso y la salud de los pacientes. Este tercer período fue propuesto para verificar la contribución de la PISIoT a la pérdida de peso y/o a una mejora en la salud de las personas, reduciendo el riesgo de un posible infarto de miocardio o enfermedades relacionadas con la obesidad. Además, en este período, se evaluó el aumento de la perspectiva y la calidad de vida de las personas mayores siguiendo las recomendaciones proporcionadas por la PISIoT.

3.4.2 Desarrollo del escenario de composición de servicios en la domótica

Para el escenario de la composición de servicios en la domótica, que es parte del dominio de aplicación del IoT Personal y Social, en donde una casa equipada con sensores tales como: sensor de flujo de agua, sensor de control de energía, sensor de presencia o de movimiento, sensor de temperatura, sensor de gas y sensor de sonido, requiere ser monitorizada para garantizar que brinde a sus residentes el confort y seguridad adecuados, cuidando simultáneamente la disminución en el consumo de energía, se desarrolló una aplicación para dispositivos móviles llamada HEMS-IoT: un sistema de casa inteligente basado en *Big Data* y *Machine Learning* para la comodidad, seguridad y ahorro de energía en el hogar en el contexto del IoT. El desarrollo de HEMS-IoT refleja la coordinación de servicios del escenario de composición de servicios en el dominio de aplicación personal y social, particularmente en la domótica. La motivación del desarrollo de HEMS-IoT fue debido a que la automatización del hogar ofrece muchas oportunidades para desarrollar nuevas aplicaciones útiles (Machorro et al., 2018). La automatización del hogar se refiere a un conjunto de técnicas destinadas a automatizar un hogar, que integran la tecnología en la seguridad, la gestión de la energía y el bienestar. La comodidad es esencial en un sistema de automatización del hogar y comprende todas las acciones realizadas para mejorar la forma en que los residentes se sienten en sus hogares (Matsui, 2018). Sin embargo, la automatización del hogar aplicada en IoT se enfrenta a un gran desafío: pocos sistemas de comunicación aseguran la interconectividad entre dispositivos (Risteska y Trivodaliev, 2017). Este problema surge

por la falta de protocolos unificados y los diferentes estilos de vida de los usuarios que coexisten en ciudades inteligentes (Hui et al., 2016).

Por otra parte, HEMS-IoT es un sistema de administración de energía para el hogar que busca contribuir a la comodidad y seguridad de los hogares inteligentes, al mismo tiempo que ayuda a los residentes a ahorrar energía. Con este fin, HEMS-IoT utiliza dispositivos del IoT, tecnologías de big data y aprendizaje automático para administrar el consumo de energía. Además, HEMS-IoT permite la monitorización en tiempo real de dispositivos domésticos y sensores. Toda la información obtenida se analiza y procesa utilizando algoritmos de aprendizaje automático para conocer los patrones de consumo de energía y los patrones de comportamiento del usuario y hacer recomendaciones relevantes para reducir el consumo de energía. HEMS-IoT tiene una arquitectura de siete capas, que facilita el mantenimiento del sistema y permite una alta escalabilidad. La arquitectura de HEMS-IoT se presenta en la Figura 3.24 e integra la capa de presentación, la capa de servicios IoT, la capa de seguridad, la capa de administración, la capa de comunicación, la capa de datos y la capa del dispositivo. Cada capa tiene una función claramente definida e incluye múltiples componentes que tienen una funcionalidad específica dentro de su capa correspondiente.

Figura 3.24 Arquitectura de HEMS-IoT.

Capa de dispositivos

Esta capa facilita el enlace y la recepción de datos desde varios dispositivos domóticos. Además, la capa de dispositivos controla actuadores y dispositivos de automatización del hogar. Por otra parte, es relevante indicar que un sistema eficiente de ahorro de energía en el hogar tiene en cuenta las preferencias de los residentes para controlar con éxito cómo funcionan los dispositivos domóticos en el hogar. Sin embargo, el consumo de energía en el hogar también depende de otros factores, como el entorno externo, la ventilación natural, las variaciones de temperatura según las estaciones del año y la edad de los residentes, por mencionar solo algunos. Los dispositivos considerados en la capa de dispositivos de HEMS-IoT se describen a continuación:

- ***Gateways***. Estos objetos permiten que los dispositivos inteligentes de automatización del hogar permanezcan interconectados. Del mismo modo, los *gateways* son el enlace entre las redes externas y los dispositivos domóticos instalados en la casa inteligente y hacen que sea más fácil controlar los dispositivos domóticos de forma remota y local. Los *gateways* utilizados en las casas inteligentes son dispositivos fronterizos que permiten el acceso entre redes externas y locales dentro de una casa. Debido a que los diferentes dispositivos de automatización del hogar vinculados a la casa inteligente se conectan a otras redes o incluso a Internet, los *gateways* manejan el acceso de comunicación principal entre esas redes.
- **Sensores**. Recopilan datos sobre diferentes parámetros de la casa inteligente, como el riesgo de robo, fugas de gas o agua, y temperatura ambiente, entre otros.
- **Actuadores**. Generalmente son de varios tipos y se instalan en toda la casa. Los actuadores se utilizan para cambiar el estado de los dispositivos domóticos y algunas instalaciones domésticas. Por ejemplo, los actuadores son capaces de interrumpir el suministro de agua y gas, emitir advertencias de falla o riesgo, aumentar o disminuir la temperatura del aire acondicionado o ajustar la intensidad de la luz.
- **Controladores**. Estos dispositivos permiten a los usuarios controlar dispositivos domóticos con respecto a los parámetros elegidos. HEMS-IoT recupera datos de varios sensores instalados en una casa y procesa los datos por medio de un algoritmo. Luego, el sistema prepara las reglas necesarias para invocar los actuadores. Del

mismo modo, HEMS-IoT permite a los usuarios monitorear el estado de los dispositivos domóticos disponibles, lo que hace que los residentes de la casa se involucren completamente en el proceso. Los usuarios también tienen la capacidad de controlar y programar los actuadores y sensores instalados en la casa inteligente a través de un sistema de control centralizado y usando una pantalla táctil, un teclado o una interfaz de voz, entre otros.

Capa de datos

Esta capa guarda los datos generados en la capa de dispositivos. Es decir, la capa de datos se basa en módulos para administrar cinco tipos de información: recomendaciones, perfiles de servicio, datos detectados, perfiles de dispositivo y perfiles de usuario. El módulo de recomendaciones es responsable de gestionar las recomendaciones de confort y ahorro de energía. A su vez, el perfil de servicio gestiona datos sobre la provisión de servicios del sistema. El módulo de datos detectados guarda y gestiona toda la información recopilada por la capa de dispositivos de la casa inteligente, como el consumo de gas / agua / energía y la temperatura ambiente, entre otros. El módulo de perfil de dispositivo maneja datos en dispositivos domóticos, como su estado y ubicación, entre otros. Finalmente, el módulo de perfil de usuario gestiona información del usuario, como dirección completa, nombre y género, entre otros.

Capa de comunicación

Esta capa considera elementos como un conjunto de sensores, HTTP y TCP/IP, y comunicación 4G para establecer los protocolos de comunicación para cada dispositivo domótico. Otras capas en la arquitectura se comunican entre sí a través de la capa de comunicación. Por otro lado, las necesidades del hogar varían de un hogar a otro, pero los más comunes incluyen refrigeración o calefacción en interiores, agua caliente e iluminación. HEMS-IoT necesita recurrir a redes de comunicación confiables para recuperar con éxito los datos de consumo de energía y el comportamiento del usuario. Algunos de los protocolos utilizados en la capa de comunicación de HEMS-IoT son ZigBee, TCP/IP y HTTP/IP.

La capa de comunicación ayuda a los habitantes de la casa a controlar los dispositivos domésticos inteligentes, de manera inteligente y eficiente. El objetivo en esta capa es recopilar

información de los diferentes dispositivos basados en el IoT instalados en la casa inteligente. Estos datos proporcionan información sobre el hogar, como el consumo de energía, la temperatura ambiente, el movimiento y la temperatura del aire acondicionado, por mencionar solo algunos. La Figura 3.25 ilustra el flujo de trabajo general de HEMS-IoT en la capa de comunicación.

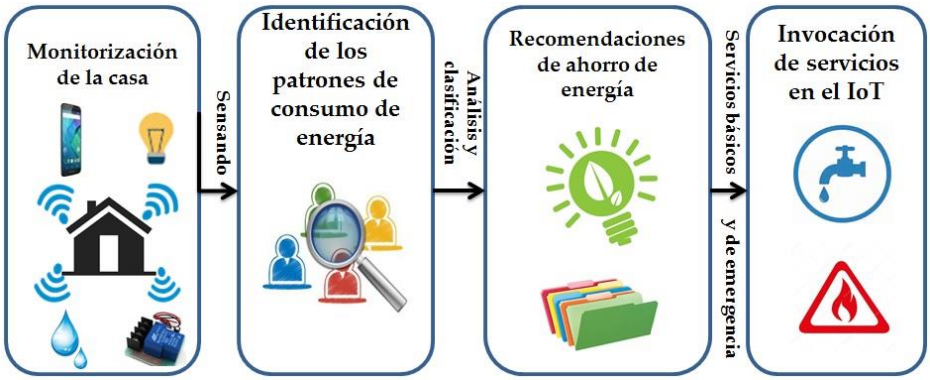


Figura 3.25 Flujo de trabajo de HEMS-IoT.

Como se ve en la Figura 3.25, los sensores son un elemento clave en la monitorización de la casa inteligente. Al recopilar información en tiempo real, los sensores permiten que HEMS-IoT analice e identifique cuánta energía consumen los electrodomésticos y dispositivos inteligentes. Los usuarios visualizan y monitorean esta información diariamente para hacer los ajustes necesarios para ahorrar energía. Además, gracias a los sensores, HEMS-IoT proporciona recomendaciones apropiadas de ahorro de energía e invoca tanto servicios básicos (por ejemplo, plomería y reparaciones eléctricas) como servicios de emergencia (por ejemplo, servicio de policía o bomberos) cuando sea necesario.

Capa de administración

Esta capa realiza y gestiona las acciones para cumplir con los requisitos del usuario solicitados en la capa de presentación. Para este fin, la capa de servicios en el IoT utiliza la API REST para garantizar la comunicación entre la capa de presentación y la capa de administración. Esta capa utiliza el aprendizaje automático y las tecnologías de almacenamiento de big data para administrar y analizar fácilmente la información recopilada en la capa de dispositivos. Del mismo modo, los comandos de acceso a la información están encapsulados por la capa de administración para garantizar la seguridad de la información. La capa de

administración también identifica los patrones de comportamiento del usuario y clasifica los hogares de acuerdo con sus patrones de consumo de energía utilizando la API de Weka 3.8, que es la versión estable del software de código abierto y ayuda con éxito a HEMS-IoT a cumplir con las funcionalidades requeridas. Para clasificar datos y generar recomendaciones de ahorro de energía, HEMS-IoT utiliza una implementación Java de código abierto del algoritmo C4.5, que es el algoritmo de aprendizaje automático J48. C4.5 y J48 se utilizan para generar árboles de decisión, y son algoritmos de clasificación. Los algoritmos de clasificación se utilizan ampliamente en la asistencia sanitaria y presentan resultados sobresalientes en el diagnóstico de diversas enfermedades (hepatitis, cáncer, enfermedades cardíacas, enfermedades oculares y tumores en mamografías digitales). Además, J48 tiene un mejor rendimiento que otros algoritmos (*random forest, CART, random tree, fuzzy C-means*, y *REPTree*).

HEMS-IoT se desarrolló e implementó de forma modular y genérica, con una mejor extensibilidad de la aplicación, facilidad de implementación y con vistas a un alto rendimiento. Las tareas realizadas por la capa de administración se dividieron en cuatro grupos:

1. **Gestión de usuarios.** Comprende acciones como eliminar y editar perfiles de usuario y registro de usuarios, entre otros.
2. **Administración del hogar.** Abarca acciones de eliminación de datos, edición de datos y gestión de dispositivos domóticos, entre otros. En este submódulo, se desarrolló una antología de recursos domóticos, como se recomienda en múltiples trabajos de investigación (Valiente-rocha y Lozano-tello, 2010) (Wongpatikaseree et al., 2012) (Comptona et al., 2012) (Sezer et al., 2015) (Lyazidi y Mouline, 2015) (Alirezaie et al., 2017). Para este fin, se utilizó una subversión del lenguaje de ontología *Web* (OWL), el OWL-DL se basa en la lógica de descripción SHI2. Del mismo modo, OWL-DL tiene un vocabulario amplio y una mayor expresividad que RDFS. La Figura 3.26 muestra un fragmento de la ontología desarrollada, que presenta los principales conceptos domóticos, como la actividad de la casa, el entorno, la entidad y la estadía, entre otros aspectos.

   También se guió el método *Methontology* para desarrollar la ontología de automatización del hogar (Fernández-López et al., 1997). Es decir, se realizaron diferentes operaciones, cada una basada en aspectos particulares del modelo conceptual de conocimiento: relaciones, términos, axiomas, taxonomía, reglas y aproximaciones matemáticas de elementos.

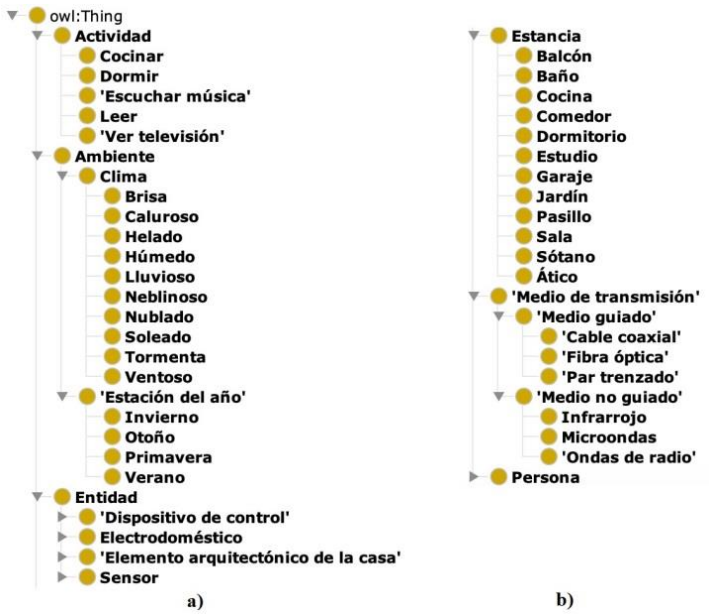


Figura 3.26 Ontología domótica: a) Fragmento 1, b) Fragmento 2.

3. **Sistema de recomendación.** Este sistema emite recomendaciones para el ahorro de energía y la comodidad del hogar en función de los patrones de comportamiento de los residentes del hogar. Para este fin, el sistema tiene en cuenta los valores de consumo de energía diarios y el promedio de cada dispositivo domótico, lo que permite que el sistema genere las reglas en el algoritmo. Por lo tanto, la casa inteligente se clasifica diariamente con respecto a cuatro categorías de consumo de energía: normal, baja, media y alta. Una vez que la clasificación se realiza con el algoritmo J48, las reglas se establecen de acuerdo con las categorías de consumo de energía. En este sentido, el consumo de energía se calcula con respecto al número de dispositivos domóticos conectados en un hogar, el número promedio de residentes y la temporada (primavera, verano, otoño e invierno). Las reglas del algoritmo identifican e indican cómo funciona el proceso de recomendación de ahorro de energía. También se utilizó Apache Mahout y RuleML para generar recomendaciones de ahorro de energía. Finalmente, el sistema de recomendación de HEMS-IoT no solo emite recomendaciones, sino que también sugiere servicios en el IoT para resolver problemas de seguridad.
4. **Tablero (*Dashboard*).** Este muestra representaciones gráficas de los principales indicadores de la casa inteligente (consumo de electricidad, gas o agua) y hábitos de los residentes.

Las tecnologías utilizadas en el sistema son muy importantes, porque permiten que HEMS-IoT analice y muestre representaciones visuales del consumo de energía. A través de estos gráficos, los usuarios toman decisiones acertadas de ahorro de energía y así optimizan el consumo de energía en sus hogares.

Capa de servicios de IoT

Esta capa sirve como enlace entre la capa de presentación y la capa de administración. Además, esta capa proporciona varios servicios REST, lo que permite a los usuarios explotar completamente las funcionalidades de HEMS-IoT. Los componentes principales de esta capa se describen a continuación:

- API REST. REST recopila información o realiza operaciones sobre dicha información en todos los formatos posibles, como JSON y XML, utilizando HTTP. REST es una buena opción si se compara con otros protocolos para el intercambio de información, como el SOAP, que tiene una alta capacidad pero es complicado.
- Selector de servicios. Este módulo valida los parámetros transmitidos por la capa de presentación y la elección de los servicios requeridos. Del mismo modo, el selector de servicios tiene la capacidad de dar o denegar servicios, de acuerdo con los datos de autenticación y los parámetros recibidos.

Capa de seguridad

Esta capa garantiza la seguridad de la información y, por lo tanto, garantiza la confidencialidad y la recopilación segura de datos de la capa de dispositivos. La comunicación entre la capa de dispositivos y la capa de seguridad no es directa, ya que se comunican a través de la capa de comunicación y la capa de administración. Adicionalmente, esta capa garantiza la confidencialidad de los datos y es capaz de recuperar datos protegidos tanto por la capa de dispositivos como por los usuarios finales. La capa de comunicación y la capa de administración facilitan la comunicación entre la capa de seguridad y la capa de dispositivos. Esta capa considera dos componentes de seguridad, autorización y autenticación, que se describen a continuación:

- **Autenticación.** Esto se refiere al acto de validar con evidencia de que algo/alguien es lo/quien dicen ser. La autenticación de objeto/dispositivo implica asegurar su origen, mientras que confirmar la identidad del usuario generalmente implica la autenticación del usuario. En HEMS-IoT, la autenticación del usuario requiere garantizar que el usuario que desea interactuar con el sistema sea realmente quien dice ser. Cuando este es el caso, HEMS-IoT autoriza a dicho usuario a acceder al sistema.
- **Autorización.** Esto ocurre después de que el sistema autentica la identidad del usuario. El objetivo de los mecanismos de autorización es proteger la información del usuario y evitar que usuarios no autorizados o no identificados accedan a los datos o realicen tareas particulares. La autorización y la autenticación son diferentes, ya que la autorización implica las tareas que los usuarios son capaces de realizar o la información a la que acceden una vez que se confirma su identidad. La autorización del usuario se aplica a elementos individuales o a un conjunto de ellos. En los sistemas de administración inteligente del hogar, cada elemento se relaciona con una actividad a ejecutar.

Capa de presentación

Esta capa garantiza la vinculación entre el usuario y el sistema a través de una aplicación para dispositivos móviles o una aplicación *Web*. En la capa de presentación, los usuarios visualizan los datos de consumo de energía, servicios en el IoT disponibles, historial de consumo de energía y recomendaciones. Como aplicación *Web*, HEMS-IoT recibe información y permite a los usuarios manipular y controlar dispositivos domóticos a través de varios dispositivos. Como una aplicación para dispositivos móviles, HEMS-IoT funciona en el sistema operativo *Android* para que los usuarios administren y controlen dispositivos domóticos. Además, con la aplicación para dispositivos móviles, los usuarios monitorean los dispositivos domóticos del hogar, incorporan nuevas habitaciones, eliminan o agregan dispositivos domóticos de una habitación particular de la casa y obtienen recomendaciones de ahorro de energía. La aplicación también muestra gráficos para consultar los patrones de consumo de energía diariamente, semanalmente y mensualmente, y ofrece a los usuarios recomendaciones de ahorro de energía para sus hogares inteligentes.

La Figura 3.27 inciso a) ilustra la lista del menú de la aplicación en la capa de presentación. Las opciones del menú principal incluyen inicio (es decir, volver a la pantalla inicial), favoritos (acceder rápidamente a habitaciones preferidas), estadísticas (visualizar patrones de consumo de energía), habitaciones (ver habitaciones de inicio) y dispositivos (visualizar dispositivos domóticos y sensores conectados a la casa inteligente). Además, el menú de la aplicación permite a los usuarios agregar y eliminar dispositivos o habitaciones, ver los perfiles de usuario existentes, consultar y cambiar la configuración de la aplicación, y salir de la aplicación. La Figura 3.27 inciso b) ilustra ejemplos de recomendaciones de confort, seguridad y ahorro de energía emitidas por HEMS-IoT.

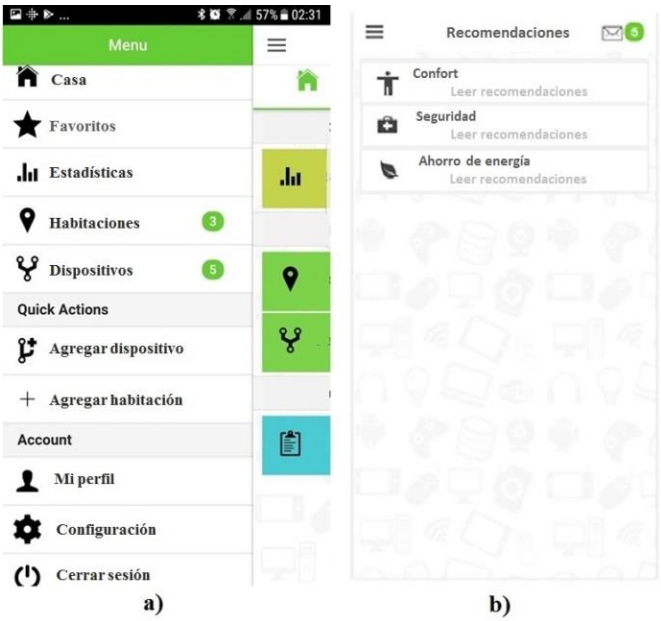


Figura 3.27 Interfaces de HEMS-IoT: a) Menú, b) Recomendaciones

Por otra parte, la Figura 3.28 inciso a) muestra un gráfico del consumo de un año y un gráfico del consumo semanal generado por HEMS-IoT. Los gráficos permiten a los usuarios de la aplicación visualizar los datos de consumo de energía diariamente, semanalmente, mensualmente o anualmente. Además, como lo ilustra la Figura 3.28 inciso b), la aplicación para dispositivos móviles HEMS-IoT proporciona un conjunto de interfaces de usuario para que los residentes de la casa visualicen y soliciten los servicios básicos disponibles en el IoT (por ejemplo, plomería y reparaciones eléctricas).

Figura 3.28 Interfaces de HEMS-IoT: a) Estadísticas, b) Servicios de IoT.

Gracias al uso de tecnologías en el IoT y a las técnicas de aprendizaje automático, HEMS-IoT ayuda a los residentes a monitorear sus casas inteligentes fácilmente y en tiempo real y brinda recomendaciones de ahorro de energía. Además, si el sistema detecta un incendio o un problema de seguridad, solicita de manera inteligente los servicios de emergencia (policía o bomberos), notificando a los residentes del hogar sobre el posible riesgo.

Por otra parte, el objetivo principal del caso estudio fue calcular el consumo de energía en hogares inteligentes y clasificar las tasas de consumo con respecto a lo que HEMS-IoT establece como un nivel de consumo de energía normal. El caso de estudio se realizó entre diez casas de un complejo residencial. Las casas eran de dos tipos, como se indica en la Tabla 3.12. Como se observa, el diseño de la primera casa es más pequeño que el segundo, y esta diferencia radica en el número de habitaciones: las casas con el primer diseño tenían dos habitaciones, mientras que aquellos con el segundo diseño tenían tres habitaciones.

El diseño de la primera casa es adecuado para tres residentes, mientras que el diseño de la segunda casa es apropiado para cuatro personas. Todas las casas tenían al menos dos enchufes inteligentes. Las diez casas estaban equipadas con sensores para visualizar y documentar datos como el movimiento y la ubicación de la habitación, la iluminación y la temperatura. Los sensores recopilaron y enviaron los datos a la capa de datos cada 30 segundos, generando así más de 2500 datos por día.

	Casa inteligente 1	Casa inteligente 2
Habitación 1	Dos lámparas, un foco, un televisor y un aire acondicionado	Dos lámparas, un foco, un televisor y un aire acondicionado
Habitación 2	Una computadora, un televisor, una plancha y un foco	Una computadora, un televisor, un ventilador de techo y un foco
Habitación 3	No aplica	Dos focos y una plancha
Sala	Un ventilador de techo, dos focos y un televisor	Un ventilador de techo, una consola de video-juegos, un televisor y dos focos
Comedor	Dos focos y un ventilador de techo	Un ventilador de techo y dos focos
Cocina	Una estufa eléctrica, un refrigera-dor, una licuadora y dos focos	Una estufa eléctrica, un refrigerador, un horno de microondas, una licuadora y dos focos

Tabla 3.12 Diseño y características de las casas.

Dado que era importante determinar el consumo de energía en los hogares inteligentes antes y después de usar HEMS-IoT, se aplicaron los siguientes pasos:

- El primer período de monitorización (antes de usar HEMS-IoT) duró ocho meses, desde mediados de Enero hasta mediados de Septiembre de 2018. A todos los habitantes de las casa inteligentes se les pidió que interactuaran con sus dispositivos domóticos normalmente sin prestar especial atención al consumo de energía o al uso de HEMS-IoT. En esta etapa, el consumo de energía en cada hogar se observó como se presenta en los recibos de electricidad emitidas por la comisión federal de electricidad (CFE).
- Los residentes de las casas inteligentes recibieron el manual HEMS-IoT para aprender a usar la aplicación. El manual es una guía escrita de la aplicación HEMS-IoT, ya que describe todas las funcionalidades del sistema que ayudan a controlar dispositivos domóticos y visualizar patrones de consumo de energía.
- El segundo período de monitorización (mientras se utilizó HEMS-IoT) también duró ocho meses, desde mediados de Enero hasta mediados de Septiembre de 2019. Durante este período, los residentes de las casas inteligentes usaron sus dispositivos domóticos normalmente, pero también se les pidió que siguieran las recomendaciones de ahorro de energía del sistema, el cual consideró las preferencias de confort interior de cada habitante. Es necesario tener en cuenta que cada vez que se acepta una recomendación en HEMS-IoT, el sistema ejecuta las operaciones necesarias para controlar y programar los dispositivos domóticos.

- Durante el segundo período de monitorización, se recopilaron datos sobre el consumo de energía gracias a la capa de dispositivos de HEMS-IoT, pero también a través de los recibos de la CFE. Los datos recopilados en este segundo período ayudaron a identificar patrones de consumo de energía después del uso de HEMS-IoT y sus recomendaciones.
- Los datos se analizaron a través de tecnologías de análisis de big data para reconocer patrones de uso en dispositivos domóticos, preferencias de confort en el hogar y problemas de la casa o riesgos de seguridad. Luego, gracias al algoritmo de aprendizaje automático J48, los residentes, los dispositivos domóticos y las casas inteligentes se clasificaron con respecto a los niveles de consumo de energía. HEMS-IoT se basa en el algoritmo de aprendizaje automático J48 que utiliza la técnica de validación cruzada de 10 pliegues para lograr un modelo predictivo. Finalmente, para generar las recomendaciones de ahorro de energía, HEMS-IoT utiliza Apache Mahout y RuleML. Se debe tener en cuenta que algunas de las recomendaciones del sistema son en forma de solicitudes de servicios en el IoT (servicios básicos o de emergencia).
- Para determinar si HEMS-IoT realmente logró reducir el consumo de energía en las diez casas inteligentes, se compararon los datos recopilados durante el primer período de monitorización (mediados de Enero a mediados de Septiembre de 2018) con los recopilados en el segundo período (mediados de Enero a mediados de Septiembre de 2019).

3.4.3 Desarrollo de la integración y coordinación de servicios en el IoT

Para la integración y coordinación de servicios en el contexto del IoT se planteó un escenario en donde un paciente con amputación traumática de una extremidad inferior y con sobrepeso requiere apoyo para realizar sus actividades básicas dentro de su hogar (apagar o encender luces, cerrar o abrir cortinas, encender o apagar el calentador, entre otras actividades), para brindarle mayor comodidad sin descuidar el ahorro de energía y sobre seguridad debido a su discapacidad. Además, necesita saber la cantidad de calorías ingeridas y calorías quemadas, las horas de sueño, la frecuencia cardíaca, la cantidad de agua consumida y la actividad física de cada día con el propósito de controlar su peso y gradualmente reducirlo para contribuir en evitar algún otro tipo de complicación para su salud (problemas en la

rodilla de la extremidad sana, diabetes, hipertensión, entre otras). Por ello, se desarrolló SCM-IoT un modelo para crear aplicaciones en el contexto del IoT basado en una arquitectura lógicamente centralizada y orientado a datos (data-centred) similar al modelo de las bases de datos distribuidas, en donde los datos se encuentran localizados en diferentes espacios de almacenamiento distribuidos lógica o geográficamente, pero interconectados por una infraestructura de red que asegura la recuperación e intercambio de información entre las aplicaciones y los espacios de almacenamiento. El desarrollo de SCM-IoT refleja la integración y coordinación de servicios en el contexto del IoT, particularmente enfocado en la domótica que tiene una gran vinculación y colaboración con el dominio del cuidado de la salud. Además, dentro de los aspectos considerados en el desarrollo del modelo SCM-IoT se encuentran:

- Determinar en todo momento la ubicación del paciente en alguna de las habitaciones de la casa.
- En aquellos casos cuando el paciente posea limitaciones físicas severas, proporcionar asistencia para el traslado de la persona a la habitación en donde regularmente se realiza alguna actividad de rehabilitación, incluyendo el encendido y apagado de luces o la apertura y el cerrado de puertas y cortinas.
- Regular el uso de la iluminación, ventilación y calefacción artificiales de acuerdo con las condiciones ambientales del entorno con el propósito de brindar el mayor confort al paciente, buscando reducir al mismo tiempo el consumo innecesario de energía en el hogar.
- Evaluar la posibilidad del surgimiento de alguna emergencia como un incendio mediante el análisis de los datos reportados por los sensores asociados a cada habitación en toda la casa.
- Realizar solicitudes de servicio a los bomberos e informar de la habitación en la que se inició el incidente en caso de incendio.
- En una situación de emergencia, determinar si la casa se encuentra habitada, en cuyo caso, solicitar el servicio de emergencia médica e informar de aquellas habitaciones en las que se detectó presencia de algún residente antes del incendio para la rápida localización de las personas por los rescatistas.

El desarrollo del modelo SCM-IoT consistió esencialmente en definir el modelo de datos de las notas, definir el modelo de datos para el libro, definir las condiciones de actualización

para los productores de notas, definir las condiciones de notificación para los subscritores de contenido en las notas y definir las extensiones propuestas en BPEL.

Definición del modelo de datos de las notas

El modelo de datos para una nota consiste en identificar los aspectos más relevantes que la conforman y, en su diseño más simple pero eficaz, agruparlos en una estructura plana. Los aspectos de información relevantes para la aplicación se designan como campos. Los campos seleccionados que describen el estado de la habitación son:

- **id**, identificador de la habitación.
- **nombre**, nombre de la habitación.
- **presencia**, indica si algún residente está presente en la habitación (SI o NO).
- **Temperatura**, indica el rango de temperaturas en el que se encuentra la temperatura detectada por el sensor en la habitación (BAJA, NORMAL, ALTA, MUYALTA).
- **humo**, indica si el sensor correspondiente ha detectado humo en la habitación (SI o NO).
- **gastoxico**, indica si el sensor correspondiente ha detectado gas tóxico como dióxido de carbono en la habitación (SI o NO).
- **calefactor**, indica si el calefactor de la habitación está en funcionamiento (SI o NO).
- **ventilador**, indica si el aire acondicionado de la habitación está en funcionamiento (SI o NO).
- Alarma, indica si la alarma sonora o luminosa o ambas de la habitación está activada (SI o NO).

Una nota que describe los aspectos más relevantes de la información del estado de la habitación se representa como un elemento XML, el cual contiene un grupo fijo de subelementos, cada uno representando a su vez a un campo de la nota. Los nombres de la nota y de los campos son los que se indicaron anteriormente. Un ejemplo de una nota es el siguiente elemento XML:

```
<habitacion id="SAL">
  <nombre>SALA</nombre>
  <presencia>SI</presencia>
  <Temperatura>BAJA</Temperatura>
  <humo>NO</humo>
  <gastoxico>NO</gastoxico>
  <calefactor>SI</calefactor>
  <ventilador>NO</ventilador>
  <Alarma>NO</Alarma>
</habitacion>
```

La interpretación de esta nota de tipo `habitacion` es que existe una habitación en la casa con nombre de habitación “SALA” e identificador “SAL”, la cual se encuentra ocupada por un residente, con una temperatura “BAJA” en la habitación por lo que el sistema “SI” ha activado el funcionamiento del calefactor pero “NO” el del ventilador (aire acondicionado) y que “NO” se ha detectado ni humo ni gas tóxico en la habitación por lo que “NO” se encuentra la alarma de incendio activada.

La nota se visualiza como el renglón de una tabla en donde cada valor del campo está situado bajo el nombre correspondiente de una columna, tal y como se muestra en la Tabla 3.13.

id	nombre	presencia	Temperatura	humo	gas	calefactor	ventilador	Alarma
SAL	SALA	SI	BAJA	NO	NO	SI	NO	NO

Tabla 3.13 habitacion

La definición formal XSD de la nota de tipo `habitacion` es la que se muestra a continuación:

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:complexType name="Habitacion">
    <xs:sequence>
      <xs:element name="nombre" type="xs:string" />
      <xs:element name="presencia" type="xs:string" />
      <xs:element name="Temperatura" type="xs:string" />
      <xs:element name="humo" type="xs:string" />
      <xs:element name="gastoxico" type="xs:string" />
      <xs:element name="calefactor" type="xs:string" />
      <xs:element name="ventilador" type="xs:string" />
      <xs:element name="Alarma" type="xs:string" />
    </xs:sequence>
    <xs:attribute name="id" type="xs:string" use="required" />
  </xs:complexType>
</xs:schema>
```

En esta definición se observa que se introduce la definición del tipo `Habitacion` como un grupo de los campos antes descritos. Una vez definida la estructura de una nota para la información relevante de cada habitación, un libro de notas de este tipo se define como una colección de dichas notas.

Definición del modelo de datos del libro de notas

Un libro de notas `habitaciones` consiste en un grupo fijo de notas de tipo `habitacion` como el que se ha definido anteriormente. El libro registra la información conocida del estado de cada habitación obtenida de la última actualización reportada por los sensores y por las decisiones tomadas para poner en funcionamiento o no a los actuadores. Sin embargo, el libro tiene que situarse al lado de otros libros, por ejemplo, libros de sensores y actuadores, formando así una colección y definiendo el contexto apropiado para el tratamiento de toda la información relevante para la aplicación IoT.

El elemento `casa` es el nombre de la colección de libros y establece el contexto en el que se sitúa el libro de habitaciones como se muestra a continuación:

```
<casa>
  <habitaciones>
    <habitacion id="PAS">
      <nombre>PASILLO</nombre>
      <presencia>NO</presencia>
      <Temperatura>BAJA</Temperatura>
      <humo>NO</humo>
      <gastoxico>NO</gastoxico>
      <calefactor>NO</calefactor>
      <ventilador>NO</ventilador>
      <Alarma>NO</Alarma>
    </habitacion>
    <habitacion id="SAL">
      <nombre>SALA</nombre>
      <presencia>SI</presencia>
      <Temperatura>BAJA</Temperatura>
      <humo>NO</humo>
      <gastoxico>NO</gastoxico>
      <calefactor>SI</calefactor>
      <ventilador>NO</ventilador>
      <Alarma>NO</Alarma>
    </habitacion>
    <habitacion id="COM">
      <nombre>COMEDOR</nombre>
      <presencia>NO</presencia>
      <Temperatura>BAJA</Temperatura>
      <humo>NO</humo>
```

```
      <gastoxico>NO</gastoxico>
      <calefactor>NO</calefactor>
      <ventilador>NO</ventilador>
      <Alarma>NO</Alarma>
    </habitacion>
    <habitacion id="COC">
      <nombre>COCINA</nombre>
      <presencia>NO</presencia>
      <Temperatura>BAJA</Temperatura>
      <humo>NO</humo>
      <gastoxico>NO</gastoxico>
      <calefactor>NO</calefactor>
      <ventilador>NO</ventilador>
      <Alarma>NO</Alarma>
    </habitacion>
    <habitacion id="REC">
      <nombre>RECAMARA</nombre>
      <presencia>NO</presencia>
      <Temperatura>BAJA</Temperatura>
      <humo>NO</humo>
      <gastoxico>NO</gastoxico>
      <calefactor>NO</calefactor>
      <ventilador>NO</ventilador>
      <Alarma>NO</Alarma>
    </habitacion>
    <habitacion id="EST">
      <nombre>ESTUDIO</nombre>
      <presencia>NO</presencia>
      <Temperatura>BAJA</Temperatura>
      <humo>NO</humo>
      <gastoxico>NO</gastoxico>
      <calefactor>NO</calefactor>
      <ventilador>NO</ventilador>
      <Alarma>NO</Alarma>
    </habitacion>
    <habitacion id="BAN">
      <nombre>BANIO</nombre>
      <presencia>NO</presencia>
      <Temperatura>BAJA</Temperatura>
      <humo>NO</humo>
      <gastoxico>NO</gastoxico>
      <calefactor>NO</calefactor>
      <ventilador>NO</ventilador>
      <Alarma>NO</Alarma>
    </habitacion>
  </habitaciones>
  <sensores>...</sensores>
  <actuadores>...</actuadores>
</casa>
```

Es relevante notar que al final del elemento `casa`, aparecen los elementos `sensores` y `actuadores` que corresponden a los libros que contienen información sobre los sensores y actuadores usados en la infraestructura de domótica para la casa.

Un libro de notas en la colección, como aquel de las habitaciones, se visualiza como una tabla estructurada por campos (columnas) y por un conjunto fijo de hojas (renglones) en

donde se registra el estado de cada habitación. La Tabla 3.14 describe el estado de cada habitación de la casa, en donde cada renglón representa el estado de una habitación y cada columna el valor de un campo.

id	nombre	presencia	Temperatura	humo	gas	calefactor	ventilador	Alarma
PAS	PASILLO	NO	BAJA	NO	NO	NO	NO	NO
SAL	SALA	SI	BAJA	NO	NO	SI	NO	NO
COM	COMEDOR	NO	BAJA	NO	NO	NO	NO	NO
COC	COCINA	NO	BAJA	NO	NO	NO	NO	NO
REC	RECAMARA	NO	BAJA	NO	NO	NO	NO	NO
EST	ESTUDIO	NO	BAJA	NO	NO	NO	NO	NO

Tabla 3.14 casa/habitaciones

La definición XSD de la colección `casa` de libros de notas es la siguiente:

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:complexType name="Habitacion">
    <xs:sequence>
      <xs:element name="nombre" type="xs:string" />
      <xs:element name="presencia" type="xs:string" />
      <xs:element name="Temperatura" type="xs:string" />
      <xs:element name="humo" type="xs:string" />
      <xs:element name="gastoxico" type="xs:string" />
      <xs:element name="calefactor" type="xs:string" />
      <xs:element name="ventilador" type="xs:string" />
      <xs:element name="Alarma" type="xs:string" />
    </xs:sequence>
    <xs:attribute name="id" type="xs:string" use="required" />
  </xs:complexType>

  <xs:element name="casa">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="habitaciones">
          <xs:complexType>
            <xs:sequence>
              <xs:element maxOccurs="unbounded" name="habitacion" type="Habitacion">
              </xs:element>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

El contenido del libro de notas en donde se registran los eventos reportados por los sensores, así como las acciones requeridas a los actuadores se representa igualmente en un documento XML.

Por otra parte, una vez definido el libro principal de notas es necesario definir los libros secundarios que ambos procesos productores y subscriptores usan para acceder al libro principal. Las notas que se encuentran en estos libros secundarios contienen la información mínima esencial sobre la ubicación de la instalación de los dispositivos de domótica. En consecuencia, la ubicación determina la habitación de la que los sensores detectan información ambiental o en la que los actuadores modifican las condiciones del ambiente local. Una nota proveniente de un sensor que incluye esencialmente los campos de identificador y de lectura. El campo de lectura contiene la información que se origina en el medio ambiente, correspondiendo a la más recientemente obtenida por el sensor. Un extracto del libro de notas para los sensores se muestra a continuación:

```
<casa>
  <habitaciones>...</habitaciones>
  <sensores>
    <presencia>
      <sensor id="PRE0">
        <lectura>NO</lectura>
      </sensor>
      ...
    <Temperatura>
      <sensor id="TEM0">
        <lectura>BAJA</lectura>
      </sensor>
      ...
    </Temperatura>
    <humo>
      <sensor id="HUM0">
        <lectura>NO</lectura>
      </sensor>
      ...
    </humo>
    <gastoxico>
      <sensor id="GAS0">
        <lectura>NO</lectura>
      </sensor>
      ...
    </gastoxico>
  </sensores>
  <actuadores>...</actuadores>
</casa>
```

Los sensores se agruparon por su tipo en sensores de presencia, temperatura, humo y gas tóxico bajo elementos XML correspondientes del mismo nombre. Los demás tipos de sen-

sores tienen una estructura de nota similar con la excepción del sensor de temperatura cuyos valores reportados en el campo lectura toma los valores indicados antes (‘BAJA’, ‘NORMAL’, ‘ALTA’, ‘MUYALTA’).

El libro de notas para los actuadores tiene una estructura similar a la de los sensores, agrupándose bajo el elemento XML `actuadores`, los cuales a su vez se agrupan de acuerdo con su tipo en calefactores, ventiladores (aires acondicionados) y alarmas. La información esencial que contiene cada nota del libro de actuadores es su identificador y su estado de funcionamiento. Un extracto del libro de notas para los actuadores se muestra a continuación:

```
casa>
  <habitaciones>...</habitaciones>
  <sensores>...</sensores>
  <actuadores>
    <calefactores>
      <actuador id="CAL0">
        <estado>NO</estado>
      </actuador>
      ...
    </calefactores>
    <ventiladores>
      <actuador id="CLI0">
        <estado>NO</estado>
      </actuador>
      ...
    </ventiladores>
    <Alarmas>
      <actuador id="ALA0">
        <estado>NO</estado>
      </actuador>
      ...
    </Alarmas>
  </actuadores>
</casa>
```

A diferencia de los sensores, el campo estado es de escritura porque indica la orden dada al actuador para que modifique de inmediato su funcionamiento, suponiendo que está encendido y en buenas condiciones de operación.

Definición de las condiciones de actualización para los sensores

La definición de las condiciones para realizar una actualización proveniente de alguno de los sensores y consiste en verificar que:

- El identificador de la habitación es aquel que corresponde con la ubicación del sensor
- La lectura más recientemente detectada por el sensor es distinta de la ya registrada en el libro.

Estas condiciones se describen formalmente como expresiones de XPath 1.0 de la siguiente manera:

```
$habitacion/@id=$ubicacion[@dispositivo=$sensor/@id]/@habitacion
and
$sensor/lectura != $habitacion/presencia
```

En donde la variable **`$habitacion`** contiene una copia de la información registrada en el libro de notas y en consecuencia la última conocida, mientras que la variable **`$sensor`** contiene la información más reciente obtenida directamente del sensor.
Estas condiciones son similares para todos los demás tipos de sensores por lo que se aplican a todos ellos. Además, debido a que estas condiciones son bastante usuales, se generan automáticamente, simplificando con ello la programación.

Definición de las condiciones de notificación para los actuadores

Las condiciones de notificación para los actuadores se clasifican en dos partes, confort y seguridad.

Condiciones de confort y ahorro de energía

Dentro de las condiciones de confort que el sistema permite garantizar está la de asegurar que la temperatura de una habitación sea agradable para quienes la ocupan, es decir, que la temperatura se encuentre en un rango considerado como normal, el cual varía entre 15°C y 25°C, siendo estos límites sujetos a una posible configuración de acuerdo a las preferencias de los habitantes de la casa. Sin embargo, también se consideran las condiciones de ahorro de energía que establecen que solamente se consuma energía cuando la habitación esté ocupada. En consecuencia, las condiciones a cumplirse conjuntamente son que:

- La temperatura actualizada más recientemente sea baja (condición de confort).
- La habitación esté ocupada por al menos un residente (condición de ahorro de energía).
- El calefactor o ventilador (aire acondicionado) de la habitación se encuentre apagado (condición de reducción de tráfico en la red y en consecuencia de ahorro de emergía).

Las condiciones anteriores se describen como expresiones lógicas en el modelo SCM-IoT de la siguiente manera:

```
$actualizacion/Temperatura = 'BAJA' and
$actualizacion/presencia = 'SI' and
$actualizacion/calefactor = 'NO'
$actualizacion/ventilador = 'NO'
```

En donde **`$actualizacion`** designa a la variable que representa a la última actualización del libro de notas y que, por lo tanto, contiene la más reciente información disponible sobre la temperatura de la habitación, la presencia de algún residente en ella y el estado del calefactor o ventilador (aire acondicionado) instalado en una habitación.

id	nombre	presencia	Temperatura	humo	gas	calefactor	ventilador	Alarma
PAS	PASILLO	NO	BAJA	NO	NO	NO	NO	NO
SAL	SALA	SI	BAJA	NO	NO	SI	NO	NO
COM	COMEDOR	SI	BAJA	NO	NO	SI	NO	NO
COC	COCINA	NO	MUYALTA	SI	SI	NO	NO	SI
REC	RECAMARA	SI	BAJA	NO	NO	SI	SI	NO
EST	ESTUDIO	NO	BAJA	NO	NO	NO	SI	NO

Tabla 3.15 Estado de habitaciones en condiciones de incendio

En la Tabla 3.15 se presenta el estado de las habitaciones en una situación de incendio con la alarma activada y cuando tres habitaciones están ocupadas en el momento en el que se genera el incendio.

Condiciones de seguridad

Las condiciones de seguridad se refieren a aquellas relacionadas con brindar las acciones necesarias para enfrentar contingencias tales como los incendios. Para lograr este fin, se

describen con precisión las condiciones que determinan un incendio y acondicionar las habitaciones con el equipo de detección necesario que proporcione la información requerida oportunamente. Las condiciones que conjuntamente se cumplen para determinar la existencia de un incendio son:

- Las temperaturas más recientes obtenidas por el sensor son MUY ALTAS, es decir, muy por arriba de los límites usuales de las temperaturas altas en climas cálidos o desérticos.
- La presencia de altas concentraciones de humo que es una señal ampliamente conocida de incendio.
- La presencia de gases tóxicos como monóxido de carbono que además de señalar un posible incendio son peligrosos por sí mismos.

Las anteriores condiciones se establecen de la siguiente forma:

```
$habitacion/Temperatura = 'MUYALTA' and
$habitacion/humo = 'SI' and
$habitacion/gastoxico = 'SI' and
$Incendio = 'NO'
```

En donde la variable **`$habitacion`** contiene una copia de los registros asentados en el libro principal y en consecuencia corresponden a las lecturas más recientes reportadas por los sensores.

Descripción de las extensiones

Para incorporar los componentes del modelo SCM-IoT al modelo de composición de servicios de BPEL se incorporaron nuevas construcciones sintácticas que representen a estos componentes. Las nuevas construcciones que proporciona el modelo SCM-IoT corresponden a nuevos elementos de marcado XML. El modelo de composición de servicios de BPEL usa elementos de marcado XML para representar las actividades de BPEL. Entre las actividades más básicas se encuentran aquellas para demandar y atender servicios Web. En consecuencia, para extender al lenguaje BPEL, las nuevas construcciones se definen como nuevas actividades obtenidas por la composición de las ya existentes. Para introducir en el lenguaje BPEL una nueva actividad es necesario definir:

- La estructura sintáctica de la nueva actividad por composición de los elementos de marcado que representan a las actividades de BPEL.
- El comportamiento asociado a la nueva actividad se obtiene por la composición del comportamiento de las actividades indicadas en su definición sintáctica.
- Una interpretación abstracta bajo su representación en un modelo formal como las redes de Petri, en particular, las redes de Petri coloreadas o CPN.

La representación en un modelo formal de las extensiones permite además de contar con un marco teórico que ofrezca un medio para formular y demostrar (o refutar) que el sistema cuenta con propiedades que garantizan buen comportamiento, tales como progreso, acotamiento e invarianza, entre otras.

Para distinguir los elementos XML de aquellos de la extensión se utilizaron nombres calificados en todo momento para mantener una clara distinción. Los nombres de las actividades BPEL utilizan nombres calificados con prefijo **`bpel:`** de su espacio de nombres, mientras que los nuevos elementos del modelo SCM-IoT tienen nombres calificados con el prefijo **`iota`**: para su espacio de nombres. Los elementos del modelo SCM-IoT que extienden a BPEL son los siguientes:

- `iota:each page="…">…</iota:each>` para seleccionar páginas bajo una condición.
- `<iota:update notebook="…">…</iota:update>` para solicitar una actualización.
- `<iota:updated notebook="…">…</iota:updated>` para recibir la notificación de una actualización.

3.4.4 Implementación de la formalización del mecanismo de composición de servicios en el IoT

Las extensiones de BPEL con el modelo SCM-IoT se describieron con redes de Petri Coloreadas o CPN con el propósito de validar que tiene un significado definido con precisión sin ambigüedades. Además, esta representación permite usar las herramientas de verificación automatizada que tienen las CPN. Para la definición de cada operación, la CPN que la representa corresponde con una red abstracta la cual se caracteriza por tener un patrón es-

tructural bien definido pero cuyos elementos se instancian en cada invocación. Los elementos que se instancian son el tipo de cada lugar, las anotaciones sobre los lugares y sus arcos, así como las condiciones de habilitación y disparo de las transiciones.

En la Figura 3.29 se presenta una CPN que muestra la representación de instancias de las tres operaciones propuestas con el fin de experimentar el efecto que cada operación produce sobre el medio compartido.

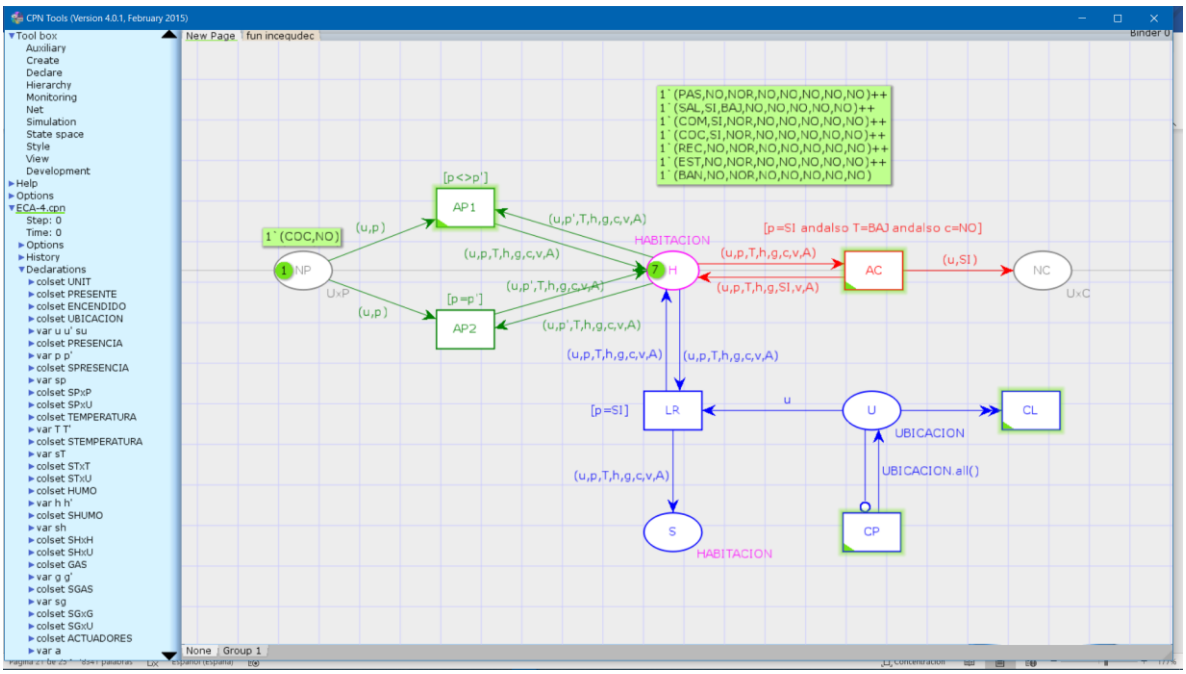


Figura 3.29 Marcado inicial de la red de prueba de las operaciones each, update y updated

La CPN de color azul describe el comportamiento de la operación de selección **`each`**, en tanto que las CPN de colores verde y rojo describen respectivamente el comportamiento de las operaciones **`update`** y **`updated`**. Estas operaciones comparten el lugar H que corresponde al libro de registro de habitaciones de la casa.

Selección

La selección de marcas se realizó usando como criterio el indicado, es decir, el anotado en la condición de la transición LR que impone restricciones sobre el contenido de cada marca del lugar H. El criterio de selección consistió en que en cada habitación se encuentre presente algún residente **`[p=SI]`**, de acuerdo con los datos asentados en el libro.

La operación cuenta con dos controles dados por las transiciones CP y CL que se disparan para iniciar y terminar la selección. El disparo de la transición CP causa que se depositen en el lugar U marcas con los nombres de todas las habitaciones en donde se encuentra algún residente tal como lo indica la anotación **`UBICACION.all()`**, dando con ello inicio al proceso de selección ya que el lugar U es habilitante de la transición LR. La selección de cada marca de H se produce con cada disparo de la transición LR, la cual permanece habilitada mientras existan marcas en H que cumplan la condición de selección. Cuando no quedan más marcas seleccionables, es que el lugar U tal vez no este vacío. Para usar nuevamente esta operación, todas las marcas sobrantes de U se eliminan disparando la transición CL. Al final de la ejecución de la operación, el lugar S tiene las marcas de H que cumplen el criterio de selección sin modificar el contenido de H. Para garantizar la indivisibilidad de la operación, se disparan solamente las transiciones CP, LR y CL en ese orden, siendo necesario disparar LR tantas veces como permanezca habilitada.

En la Figura 3.30 se muestra el marcado de la red después de los disparos sucesivos de las transiciones CP, LR y LR, en ese preciso número y orden.

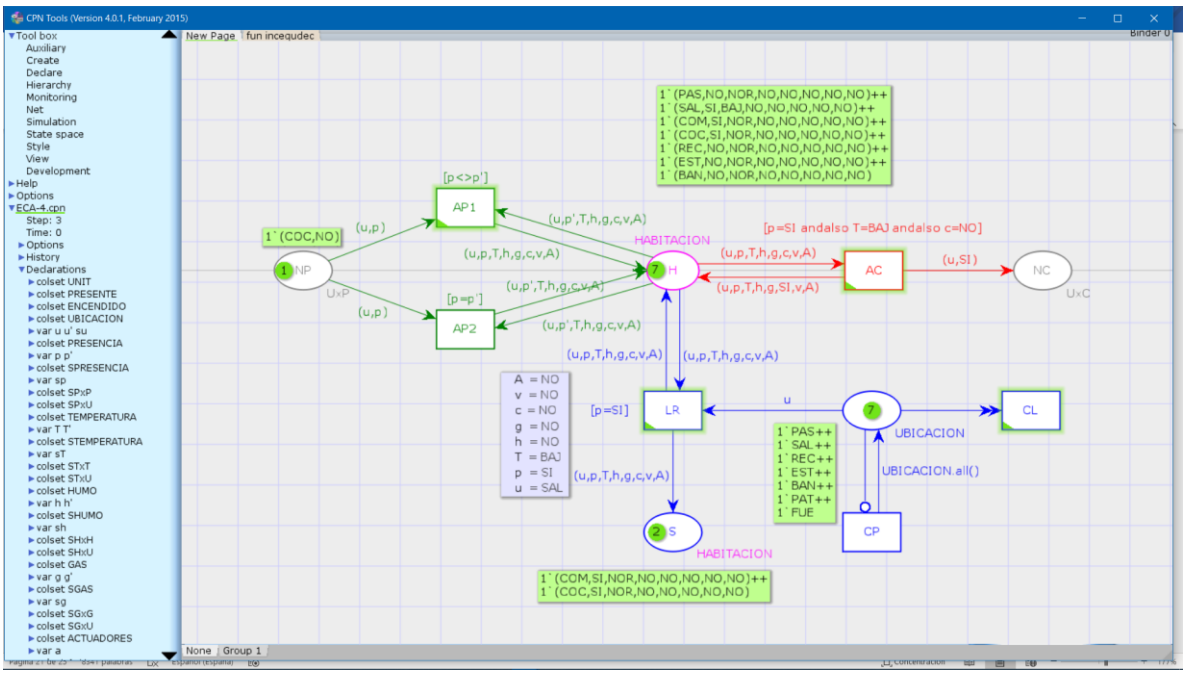


Figura 3.30 Marcado de la red después de los disparos de las transiciones CP, LR y LR

En la Figura 3.30 se observa que aún queda un elemento por seleccionar por lo que la transición LR permanece habilitada. Así mismo, se observa también la vinculación de variables que habilita a LR. Después de dispararse LR, el lugar S contiene las marcas correspondien-

tes a las habitaciones SALa, COMedor y COCina como se espera. Además, se visualiza que las anotaciones de los arcos de entrada y de salida de LR con el lugar H son idénticas, con lo cual se asegura que el contenido de H permanezca inalterado.

Las propiedades que tienen esta representación de la operación de selección son que siempre termina en un número de pasos que no excede al número de marcas en U, es decir, el número de habitaciones. Esto es debido a que con cada disparo de la transición LR se elimina una marca de U lo que causa que eventualmente se agoten. Además, esta representación mantiene inalterado el contenido de marcas del medio H.

Actualización

La modificación del contenido del libro se consigue mediante la invocación de la operación **`update`**. El patrón estructural de la red que representa a esta operación consiste en dos transiciones AP1 y AP2, teniendo ambas como lugares de entrada NP y H. Las condiciones de habilitación anotadas en AP1 y AP2 son mutuamente exclusivas porque determinan cuándo la petición de actualización, depositada en el lugar NP se realiza o no, dependiendo de que el valor p de la solicitud en NP sea distinto al registrado en p' en H, como se muestra en la Figura 3.31.

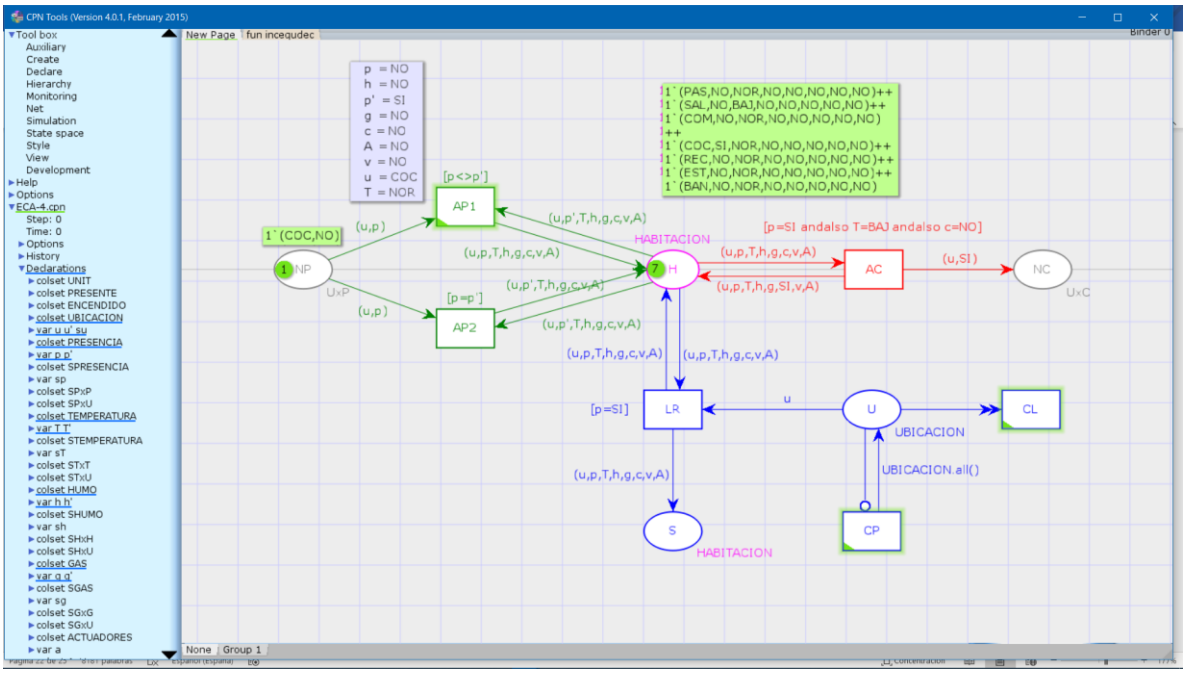


Figura 3.31 CPN de actualización de las transiciones AP1 y AP2

Adicionalmente, en la Figura 3.31 se muestra que la transición AP1 está habilitada mientras la AP2 no. También se muestra el listado de vinculaciones de valores con las variables que permiten la habilitación. Esto debido a que la petición (COC,NO) de la actualización es aceptable ya que el valor p del campo presencia, proveniente de la instancia de (u,p) dada por u=COC y p=NO, es distinto del valor p' proveniente de la instancia (u,p',T,h,g,c,v,A), dada por u=COC, p'=SI, T=NOR, h=NO, g=NO, c=NO, v=NO, A=NO. Lo anterior se interpreta diciendo que el sensor de presencia de la COCina reporta que el residente ya no está presente en esa habitación. En consecuencia, la transición AP1 elimina esta instancia de (u,p',T,h,g,c,v,A) del lugar H e inserta a cambio la instancia de (u,p,T,h,g,c,v,A), actualizando el anterior valor de p' con el nuevo valor p. La instancia tiene valor (COC,NO,NOR,NO,NO,NO,NO,NO) con lo cual queda asentado en el libro que no hay residente alguno detectado en la COCina.

En el caso de que la petición de actualización se proporcione por la marca (COC,SI), la transición AP2 se habilita mientras que la AP1 no, con lo cual el contenido de H no presenta ningún cambio. Se percibe que las anotaciones de los arcos de entrada y de salida son idénticas en AP2, pero distintas en aquellas que corresponden a AP1. La diferencia de las anotaciones en AP1 se encuentran en el campo de presencia, en donde el valor previo está dado en p' mientras que el valor actual está dado en p. Estas anotaciones sobre los arcos producen el efecto de actualizar el contenido de H después de disparar AP1 y de dejar su contenido inalterado cuando se dispara AP2. En cualquier caso, la petición de actualización de NP se descarta. Además de su terminación, esta representación tiene la propiedad de preservar tanto el número de marcas en H como la correspondencia y unicidad entre marcas y habitaciones.

Notificación de actualización

La notificación de que el contenido del medio fue modificado se consigue mediante el uso de la operación **`updated`**. Existen dos versiones para el patrón estructural de esta operación que consiste en una o más transiciones, dependiendo en el énfasis en reducir la complejidad de la red al reducir el número de elementos. Sin embargo, para cualquiera que sea la representación elegida se tiene que garantizar que al final de su ejecución se preserve tanto el

número de marcas como la correspondencia que guardan con el estado de cada habitación. En la Figura 3.32 se presenta una versión de la operación **updated** con una transición.

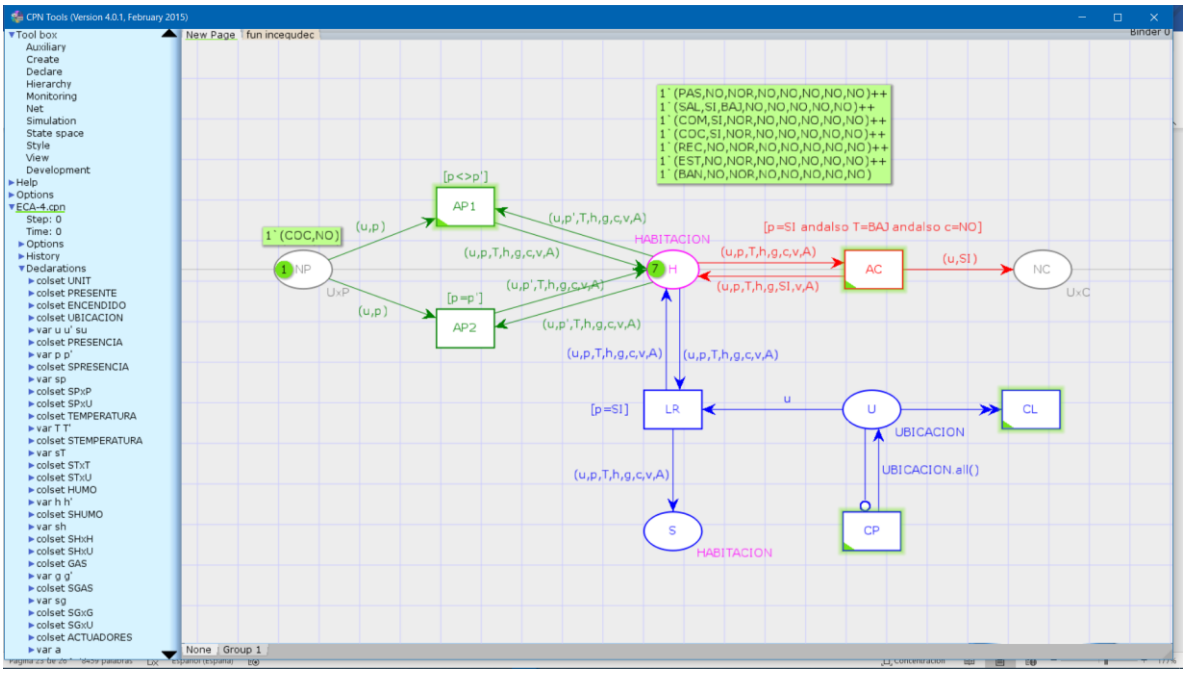


Figura 3.32 Operación updated con una transición

En esta representación, la condición de notificación [p=SI andalso T=BAJ andalso c=NO] corresponde con la condición de habilitación de la transición AC. La habilitación y posterior disparo de esta transición origina que se produzcan dos acciones: 1) depositar la instancia de (u,SI) con u=SAL en el lugar NC, y 2) localizar la instancia de (u,p,T,h,g,c,v,A) bajo la substitución u=SAL, p=SI, T=BAJ, h=NO, g=NO, c=NO, v=NO, A=NO en H para eliminarla e insertar en su lugar la instancia de (u,p,T,h,g,SI,v,A) bajo la misma substitución. A pesar de que esta representación tiene las ventajas de ser simple y de preservar las propiedades del contenido del medio, tiene también la desventaja de que no es pura ya que se combina con una operación de actualización.

En la Figura 3.33 se presenta la segunda representación en la cual se distinguen perfectamente las operaciones de notificación y la actualización. En esta representación, la condición de notificación [p=SI andalso T=BAJ andalso c=NO] queda anotada en la transición AC1. A diferencia de la representación anterior, la habilitación y posterior disparo de AC1 causa que se elimine del lugar H la instancia de la marca (u,p,T,h,g,c,v,A) que corresponde a la substitución u=SAL, p=SI, T=BAJ, h=NO, g=NO, c=NO, v=NO, A=NO. Esta instan-

cia se mueve hacia el lugar AX1 para que después se modifique y procese por la transición AC3.

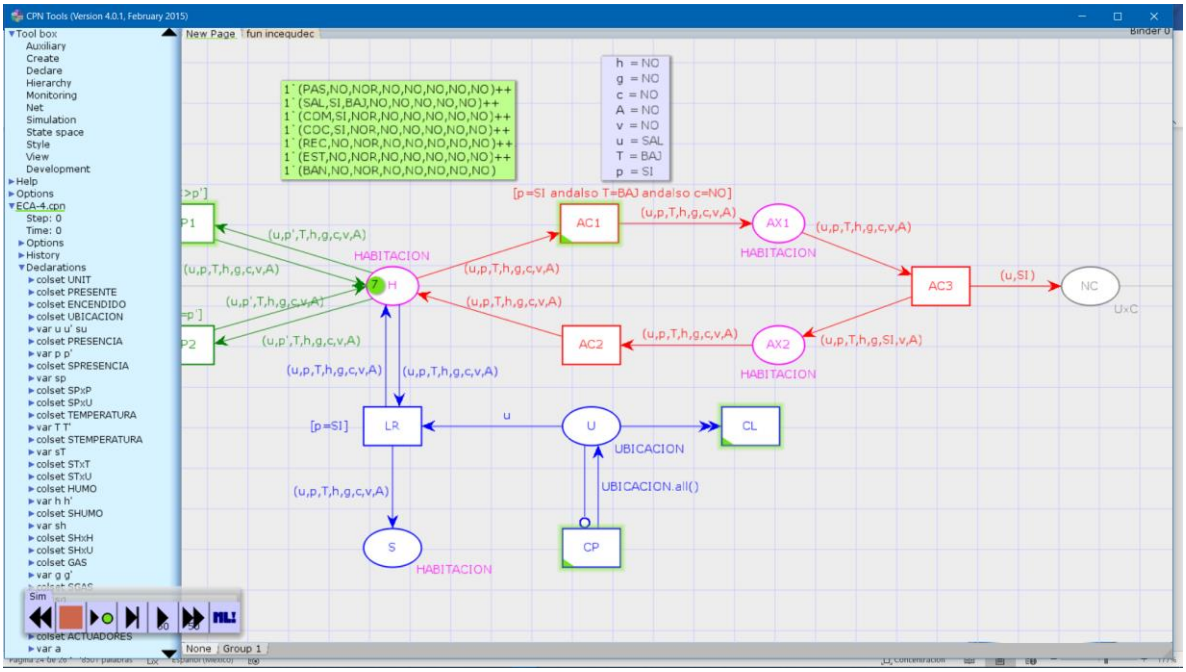


Figura 3.33 Operaciones de notificación y actualización

Adicionalmente, en la Figura 3.3 también se visualiza que AC3 modifica el campo c del calefactor para asignarle el valor SI y depositar la marca en el lugar AX2. Finalmente, la transición AC2 inserta esta marca en H, recuperando de este modo las propiedades que tiene el medio. Además de su complejidad y tamaño, el grave inconveniente de esta representación es que para asegurar que a su término el medio conserva sus propiedades, la sucesión de disparos de las transiciones AC1, AC3 y AC2 tiene que ser indivisible y en ese orden. Desafortunadamente, los mecanismos con los que cuentan las CPN, como ejecución priorizada de transiciones, no garantizan la indivisibilidad de esa secuencia de disparos. Pero si permite introducir elementos adicionales en esta representación, pero eso incrementa considerablemente también su complejidad y tamaño.

3.5 Etapa de Pruebas

En esta etapa se presentan las pruebas y la validación del mecanismo de composición de servicios en el contexto del IoT, realizadas particularmente a través de la validación de las extensiones **`each, update y updated`** propuestas en el modelo SCM-IoT.

3.5.1 Pruebas y validación del mecanismo de composición de servicios en un escenario del IoT

La validación de las extensiones (**`each, update y updated`**) se demostró mediante una red de Petri que usa las tres operaciones propuestas como extensión al modelo de BPEL y de su lenguaje de marcado. Los pasos de validación consistieron en:

- Siguiendo el diseño de la arquitectura por capas propuesto, el sistema se dividió en la capa de sensores, la capa de administración del medio y la capa de actuadores.
- La capa de sensores consiste en una colección de procesos cliente relativamente independientes, los cuales son encargados de la producción de contenido en el modelo centralizado de datos de SCM-IoT. El contenido se compone de la información reportada por cada sensor la cual es enviada a la capa de administración del medio en forma de peticiones de actualización de páginas de libros que conforman el medio.
- La capa de administración del medio recibe las peticiones de actualización provenientes de la capa de sensores. En caso de ser relevantes, las actualizaciones se aceptan para quedar asentadas en el campo de una página de un libro. Cada actualización desencadena la revisión de todas las condiciones de notificación de contenido y, en su caso, de llevar a cabo la notificación misma al subscriptor interesado correspondiente.
- La capa de actuadores consiste en una colección de procesos independientes los cuales son los que finalmente se encargan de realizar las acciones necesarias para garantizar el cumplimiento de las condiciones de confort y de seguridad que ofrece el sistema de domótica. Los actuadores se activan mediante una notificación que le indica que el sistema alcanzó las condiciones definidas como subscritor de servicio.
- La capa de administración del medio está dividida a su vez en dos subcapas: una subcapa para la atención de peticiones de productores, situados en la capa de sensores, y otra subcapa para la atención de notificaciones de subscriptores, situados en la capa de actuadores.

En la Figura 3.34 se muestra una vista parcial de la CPN construida sistemáticamente al emplear los patrones estructurales que representan las extensiones descritas para construir las dos subcapas que forman la capa de administración del medio.

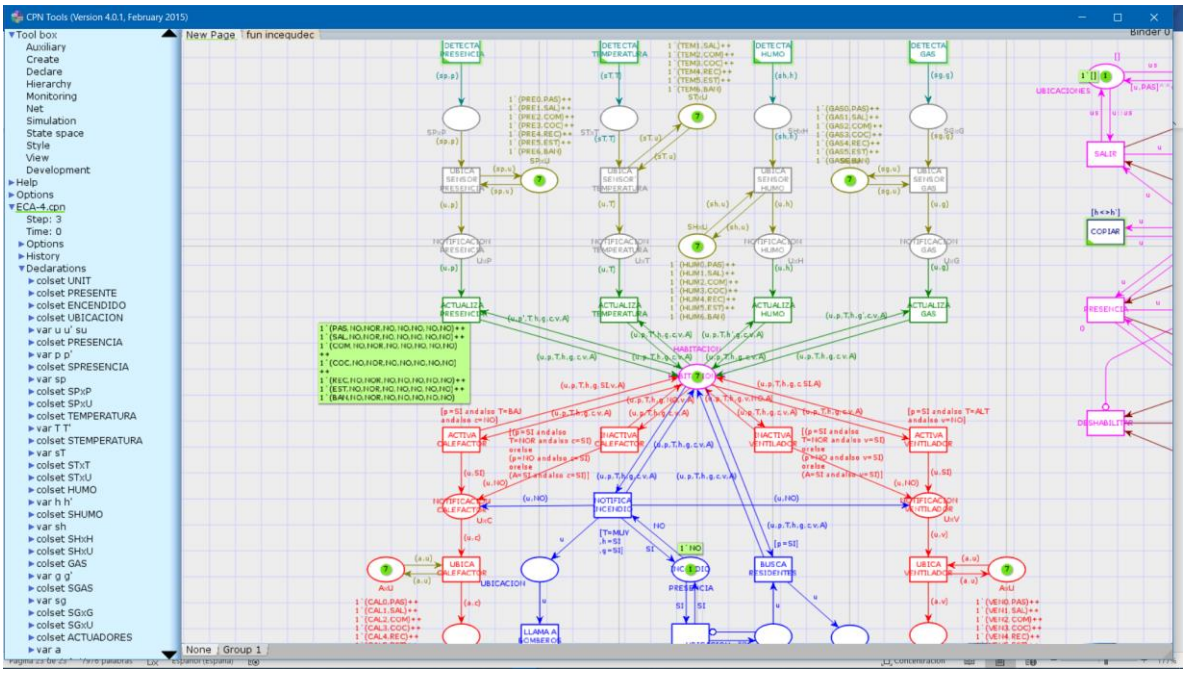


Figura 3.34 CPN construida sistemáticamente al emplear los patrones estructurales

Adicionalmente, en la Figura 3.34 se aprecia que se utilizaron colores para distinguir claramente en donde surge la instancia de los patrones estructurales de cada extensión. En color azul aparecen las instancias de la operación de selección, mientras que las instancias de las operaciones de actualización y notificación aparecen en colores verde y rojo, respectivamente. Así, por ejemplo, la transición ACTUALIZAPRESENCIA en color verde, representa una instancia de una operación de actualización, la cual analiza la relevancia de las peticiones emitidas en la capa de sensores. Además se señala que el patrón estructural de la operación de actualización no fue usado por completo por razones de claridad, ya que la red completa es demasiado compleja para presentarse en su totalidad. No obstante, el comportamiento final, aunque si bien es el mismo, es menos eficiente porque admite cualquier petición de actualización sea relevante o no. Por otra parte, la transición ACTIVACALEFACTOR en color rojo, representa una instancia del patrón estructural que corresponde a la operación de notificación en su forma corta. La condición de notificación corresponde a una condición de confort porque solicita la activación del calefactor por baja temperatura en la habitación, pero siempre y cuando esté ocupada. Finalmente, la transición

BUSCARESIDENTES corresponde a una instancia de una operación de selección de todas las habitaciones ocupadas por al menos con la presencia de un residente. A continuación, se describe el comportamiento de ambas subcapas, presentando los fragmentos de código BPEL que incluyen las extensiones.

Subcapa de atención a sensores

La atención a sensores, como productores de contenido, se concentra en un subproceso conocido como Recepcionista, ya que recibe las peticiones de actualización enviadas por estos. La Recepcionista recibe las peticiones mediante una composición `<bpel:pick><bpel:onMessage>` que permite seleccionar solamente un mensaje de entre diversos mensajes que llegan simultáneamente.

Cada elemento `<bpel:onMessage>` contiene las instrucciones BPEL para procesar cada petición de servicio recibida. Sin embargo, todos servicios a los que se refieren estas peticiones son de actualización del medio, en algún campo de alguna página del libro de habitaciones. En consecuencia, el código es muy similar a todos ellos por lo que será suficiente mostrar el que corresponde a la detección de presencia, como el que se muestra a continuación:

```
<bpel:scope name="Recepcionist">
  <bpel:sequence>
    <bpel:pick>
      <!-- RECIBE PETICION DE ACTUALIZACION -->
     <bpel:onMessage  operation="detectaPresencia" partnerLink="SensoresPL" portType="SensoresPT"
      variable="sensorPresencia">
        <bpel:scope>
          <bpel:sequence>
            <!-- DETERMINA UBICACIÓN DEL SENSOR A PARTIR DE SU IDENTIFICADOR -->
            <bpel:assign>
              <bpel:from>
                $ubicaciones[@dispositivo=$sensorPresencia/@id]/@habitacion
              </bpel:from>
              <bpel:to>$sensorPresencia/@id</bpel:to>
            </bpel:assign>
            <!-- SOLICITA ACTUALIZACION DE LA PAGINA DE LA HABITACION -->
            <iota:update notebook="habitaciones">
              <iota:with note="sensorPresencia"/>
              <iota:former page="habitacionAntes"/>
              <iota:latter page="habitacion"/>
              <iota:when>
                $sensorPresencia/@lectura != $habitacion/@presencia
              </iota:when>
            </iota:update notebook="habitaciones">
            <bpel:sequence>
              <!-- COMIENZA CODIGO BPEL GENERADO POR update -->
              <!-- CORRESPONDENCIA -->
              <bpel:if>
```

```
                <bpel:condition>
                  $sensorPresencia/@id != $habitacion/@id
                </bpel:condition>
                <bpel:assign>
                  <bpel:from>
                  $habitaciones/habitacion[@id=$sensorPresencia/@id]
                  </bpel:from>
                  <bpel:to>$habitacion</bpel:to>
                </bpel:assign>
              </bpel:if>
              <!-- RELEVANCIA -->
              <bpel:if>
                <bpel:condition>
                  $sensorPresencia/lectura != $habitacion/presencia
                </bpel:condition>
                <bpel:sequence>
                  <!-- PERMANENCIA -->
                  <bpel:assign>
                    <bpel:from>$habitacion</bpel:from>
                    <bpel:to>$habitacionAntes</bpel:to>
                  </bpel:assign>
                  <!-- ACTUALIZACION -->
                  <bpel:assign>
                    <bpel:from>$sensorPresencia/presencia</bpel:from>
                    <bpel:to>$habitacion</bpel:to>
                  </bpel:assign>
                  <bpel:assign>
                    <!-- CONSISTENCIA -->
                    <bpel:from>$habitacion</bpel:from>
                    <bpel:to>
                      $habitaciones/habitacion[@id=$sensorPresencia/@id]
                    </bpel:to>
                  </bpel:assign>
                </bpel:sequence>
              </bpel:if>
            </bpel:sequence>
          <!-- TERMINA CODIGO GENERADO POR update -->
        </bpel:sequence>
        </bpel:scope>
      </bpel:onMessage>

      <bpel:onMessage  operation="detectaTemperatura" partnerLink="SensoresPL" port-
Type="SensoresPT"
      variable="sensorTemperatura">
      . . .
      </bpel:onMessage>
    </bpel:pick>

    <bpel:sources>
      <bpel:source link="ReceptionistToDirector"/>
    </bpel:sources>
  </bpel:sequence>
</bpel:scope>
```

El patrón de interacción que se observa determina el flujo de datos que van de los productores de contenido hacia el medio compartido donde se almacenan en páginas de libros. En el listado, se observan también las etapas por las que atraviesa el proceso de actualización en esta implementación: correspondencia, relevancia, permanencia, actualización y consistencia. Así mismo se muestra su implementación en BPEL.

La Figura 3.35 muestra la correspondencia de este código con la CPN que lo representa cuando se usa la operación de actualización que es directa.

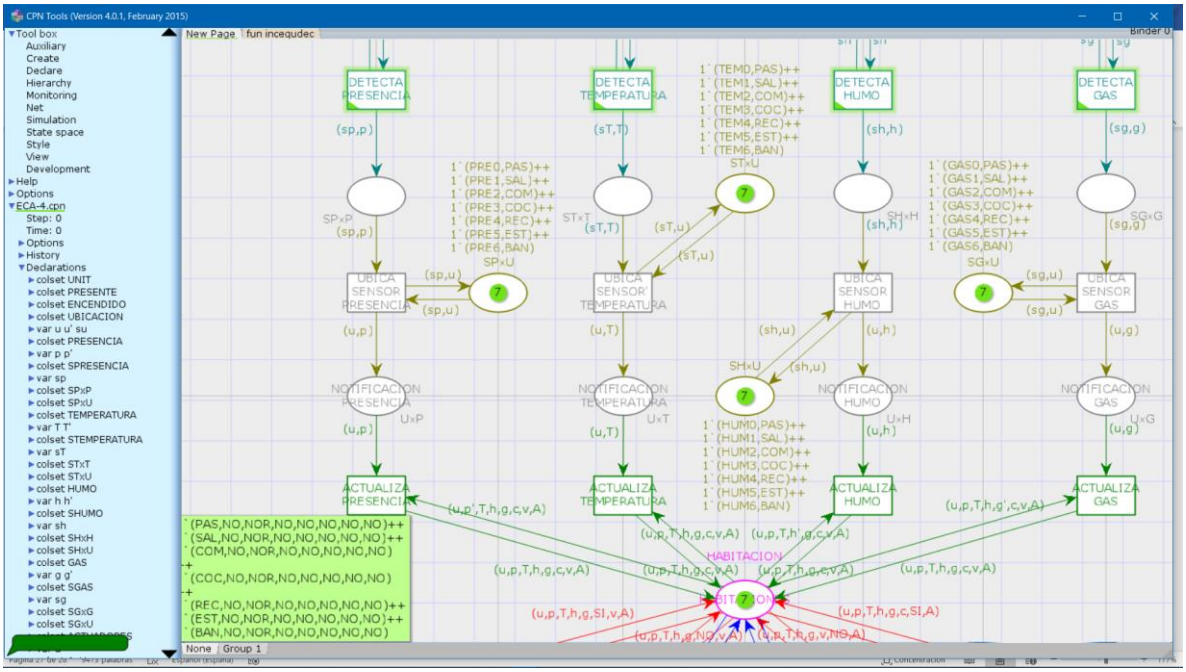


Figura 3.35 CPN de la operación de actualización directa

En la Figura 3.35, los elementos de la red que definen el comportamiento de la Recepcionista se encuentran ubicados por arriba del lugar HABITACIONES. La transición DETECTAPRESENCIA corresponde con la composición `<bpel:pick>` `<bpel:onMessage>` la cual recibe una petición que obedece a la marca genérica (sp,p) formado por el identificador sp del sensor de presencia y el valor p obtenido por el sensor. La transición UBICASENSORPRESENCIA corresponde con la obtención de la habitación donde se encuentra el sensor determinada mediante consulta al libro de ubicación de dispositivos. El libro de dispositivos se representa en la CPN como el lugar de entrada y salida de la transición UBICASENSORPRESENCIA con arcos etiquetados con la marca genérica (sp,u). La habitación o donde se ubica el sensor aparece en el par (u,p), el cual es finalmente la petición de actualización que se usa como nota en la invocación de la actualización representada por la transición ACTUALIZAPRESENCIA.

La actualización de la presencia de un residente en una habitación permite que se modifique el contenido de la página que almacena información del estado de la habitación. Dicha actualización provoca la revisión de las condiciones de notificación realizada en la subcapa de atención a actuadores.

Subcapa de atención a actuadores

La atención a los actuadores, como subscriptores de contenido, se concentra en un subproceso conocido como Director. El Director analiza las modificaciones hechas sobre las páginas de los libros originadas por las actualizaciones enviadas por los sensores.
El Director proporciona esta atención en un ciclo que inicia después de que ocurre alguna modificación en las páginas y que termina cuando las modificaciones hechas por los subscriptores en respuesta finalizan. Durante este ciclo de revisión y notificación por el director y de actualización por el subscriptor, el libro permanece cerrado a cualquier intento de actualización de los productores. De esta manera, el patrón de interacción de todos los participantes es un ciclo que atraviesa por las siguientes etapas:

- Envío de solicitud de actualización por algún productor.
- Recepción de la petición por la Recepcionista.
- Evaluación de las condiciones de notificación por el Director y, en su caso, notificación a subscriptores.
- Actuación de los subscriptores pero orquestada por el Director.
- Conclusión de las actividades de los subscriptores.
- Reinicio del ciclo.

Es relevante resaltar el hecho de que no se atiende ninguna otra petición de actualización hasta que no se haya concluido la orquestación de actividades por el Director, lo cual solo ocurre cuando el contenido del libro alcanza un estado estable en donde ninguna condición de notificación se cumple.
En el siguiente fragmento de código en BPEL, se muestran las acciones que realiza el Director para notificar al calefactor, como subscriptor, si se ha cumplido la condición de actualización:

```
 $habitacion/Temperatura = 'BAJA' and
$habitacion/presencia = 'SI' and
$habitacion/calefactor = 'NO'
```

Estas condiciones de confort son de gran interés para el servicio de calefacción. El Director realiza acciones similares para notificar sobre otras condiciones de confort, de ahorro de

energía o de emergencia a sus respectivos subscriptores. Tal y como se muestra en el siguiente código:

```
  <bpel:scope name="Director">
    <!- El Director reanuda actividades después de que la Recepcionista se lo indique ->
    <bpel:targets>
      <bpel:target link="ReceptionistToDirector"/>
    </bpel:targets>
    <bpel:variables>
      <bpel:variable name="NotebookChanged" type="xs:boolean">TRUE</bpel:variable>
    </bpel:variables>
    <bpel:sequence>
      <!-- Repetir actividades mientras exista alguna actualizacion pendiente -->
      <bpel:while>
        <bpel:condition>$NotebookChanged</bpel:condition>
        <bpel:sequence>
          <!-- Determinar si ha ocurrido alguna actualizacion -->
          <iota:updated notebook="habitaciones">
            <iota:latter page="habitacion"/>
            <iota:notice note="NotebookChanged"/>
          </iota:updated>
          <bpel:if>
            <!-- Condiciones de confort por temperatura baja -->
            <bpel:condition>
              $habitacion/Temperatura = 'BAJA' and
              $habitacion/presencia = 'SI' and
              $habitacion/calefactor = 'NO'
            </bpel:condition>
            <bpel:sequence>
              <!-- Prepara orden de activado de calefactor en habitacion -->
              <bpel:asssign>
                <bpel:from>
                  <bpel:literal>SI</bpel:literal>
                </bpel:from>
                <!-- Actualización directa (sin usar update) -->
                <bpel:to>$habitacion/calefactor</bpel:to>
                <bpel:to>$calefactor/estado</bpel:to>
              </bpel:asssign>
              <!-- Obten dirección del calefactor en habitacion -->
              <bpel:assign>
                <bpel:from>
                  $ubicaciones[@habitacion=$habitacion/@id]/@dispositivo
                </bpel:from>
                <!-- Asigna dirección en nota para envio -->
                <bpel:to>$calefactor/@id</bpel:to>
              </bpel:assign>
              <!-- Invocacion del servicio al calefactor -->
              <bpel:invoke operation="NotificaCalefactor" partnerLink="ActuadoresPL"
portType="ActuadoresPT" variable="calefactor"/>
            </bpel:sequence>
            <!-- CONTINUA LISTA DE CONDICIONES DE NOTIFICACION -->
            . . .
            <bpel:else>
              <!-- NINGUNA DE LAS CONDICIONES SE VERIFICA, TERMINAR CICLO -->
              <bpel:assign>
                <bpel:from>
                  <bpel:literal>FALSE</bpel:literal>
                </bpel:from>
                <bpel:to>$NotebookChanged</bpel:to>
              </bpel:assign>
```

```
            </bpel:else>
          </bpel:if>
        </bpel:sequence>
      </bpel:while>
      <bpel:if>
      </bpel:if>
    </bpel:sequence>
  </bpel:scope>
```

El comportamiento del Director es mucho más complejo que el de la Recepcionista. En este sentido, solamente se muestra el comportamiento del Director al detectar una condición de notificación. Aunque el comportamiento para atender otras condiciones de notificación tiene en lo general la misma estructura, el comportamiento del Director en su conjunto conlleva el problema de cómo lograr su adecuada composición. La revisión de la lista de condiciones mediante la prueba consecutiva de condiciones alternativas de la construcción `<bpel:if>` no siempre es adecuada. La razón es que no es justa o imparcial en la revisión de condiciones, favoreciendo siempre aquellas que aparecen primero en el texto, lo que impide la elección de condiciones que están en conflicto con otras que aparecen antes. Para resolver este problema, las condiciones de notificación para la integración y coordinación de servicios en el IoT se definieron para ser mutuamente exclusivas de modo que no interfieran entre sí. Una vez conocido el comportamiento del Director y de los subscriptores, como se observa en la Figura 3.36 se valida la correspondencia entre el fragmento anterior de código BPEL y la CPN.

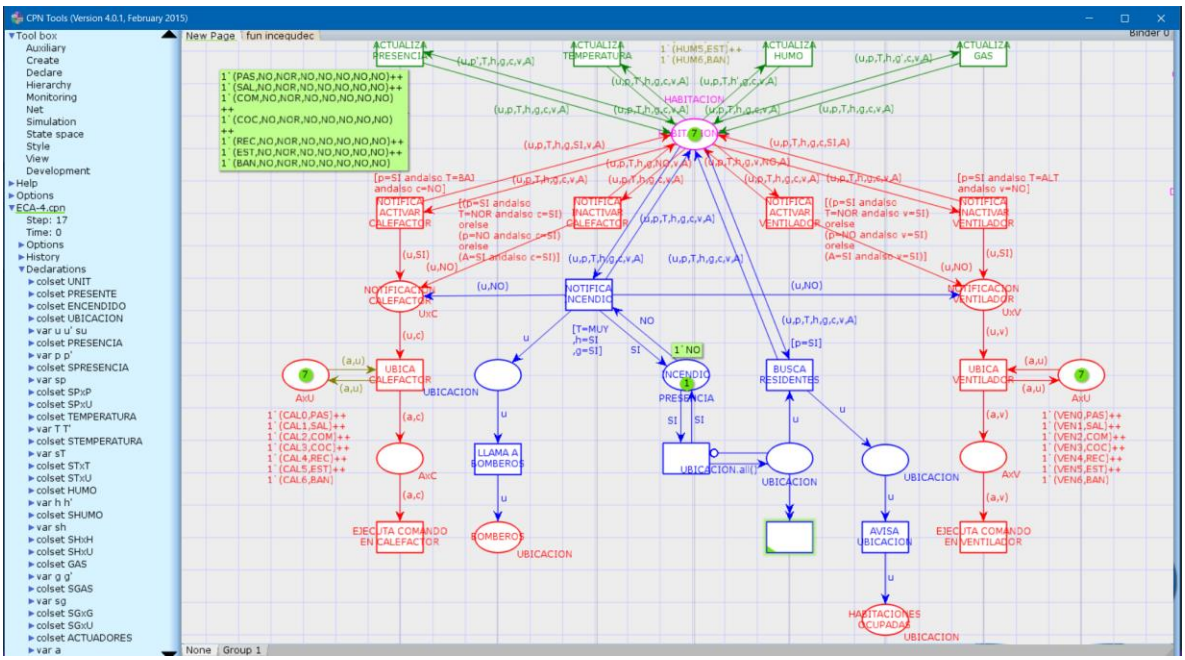


Figura 3.36 Correspondencia entre BPEL y la CPN

Adicionalmente, como se visualiza en la Figura 3.36 los elementos de la red que define el comportamiento del Director se ubican debajo del lugar HABITACION. Los elementos de la CPN que definen el comportamiento del Director del código BPEL mostrado, se sitúan en la parte izquierda de la Figura 3.36. Se observa también que la dirección del flujo de datos determina el orden de actuación de los participantes: productor, recepcionista, director y, finalmente, subscriptor, en ese orden.

Por otra parte, la transición NOTIFICAACTIVARCALEFACTOR representa a la operación de notificación `<iota:updated notebook="habitaciones">…</iota:updated>` al Director. En consecuencia, la transición tiene anotada la condición para notificar la activación del calefactor siempre que se cumpla la condición de confort de temperatura baja. Esta condición corresponde a la condición dada por la construcción condicional:

```
<bpel:if><bpel:condition>$habitacion/Temperatura = 'BAJA' and …
```

Además, si se verifica la condición, entonces se actualiza la página del libro sin recurrir a la operación **update** y se prepara el mensaje correspondiente para notificar al calefactor. Si no se alcanzan otras condiciones de la lista, entonces el Director termina su participación. Es importante observar que no es necesario pedir que la Recepcionista reanude su actividad, ya que esto ocurre solamente cuando arriban nuevas peticiones de actualización.

Capítulo 4
Resultados

El contenido del presente capítulo se enfoca en describir los resultados obtenidos de la implementación de los casos de estudio propuestos en esta tesis doctoral.

4.1 Resultados de la composición de servicios en el cuidado de la salud en el IoT

El IoT y el aprendizaje automático permiten a la PISIoT obtener y analizar variables biomédicas y otras variables recopiladas por los dispositivos inteligentes en la red de monitoreo con el propósito de contribuir al control del sobrepeso y la obesidad. En el caso de estudio de salud, los datos recopilados por los dispositivos inteligentes se analizaron para obtener información sobre los hábitos alimenticios de los pacientes, determinar el estado de los pacientes de acuerdo con su IMC e identificar variables críticas y posibles recomendaciones que ayudan a los pacientes a controlar o perder peso y mejorar su salud.

En este sentido, dentro de los resultados del caso de estudio, los datos obtenidos de las personas mayores se analizaron para identificar variables críticas de acuerdo con los valores de las variables biomédicas (frecuencia cardíaca, sueño, calorías quemadas, peso y minutos de actividad física) y otras variables (pasos, pisos, calorías consumidas, distancia recorrida, agua consumida y ejercicio) recolectadas por el dispositivo portátil y la báscula inteligente. Además, la información se analizó para obtener una idea de los patrones de alimentación y la actividad física de las personas mayores. Por esta razón, se establecieron tres períodos para la monitorización del paciente.

En el primer período (Agosto-Septiembre de 2018), solo se utilizó la plataforma de los proveedores de los dispositivos inteligentes (dispositivo portátil y báscula inteligente) para comprender el proceso de recopilación de datos y las variables recopiladas por cada dispositivo, así como para identificar la ingesta de alimentos, la cantidad de agua ingerida por las personas mayores y los valores generados para cada variable. Un análisis exploratorio de los datos recopilados identificó algunas desventajas o limitaciones en los dispositivos y las plataformas de sus proveedores. Por ejemplo, cuando la batería del dispositivo portátil está

vacía, la recolección de datos se pierde durante la carga de energía y debido a la inactividad del dispositivo; esto se informa como tiempo de sueño. Lo mismo sucede cuando el paciente se baña y si el dispositivo no es resistente al agua, el paciente se lo quita y se pierden datos. Además, las plataformas de los proveedores de los dispositivos solo muestran los datos obtenidos y no generan ningún tipo de recomendación médica. Al comienzo de este primer período, el 100% de las personas mayores tuvieron un alto peso y un IMC entre 30 y 35, y desafortunadamente, al final del período, ninguna de las personas mayores presentó una disminución de peso. La Figura 4.1 muestra el promedio semanal de cada variable para cada uno de los 40 pacientes adultos mayores monitoreados en la última semana del primer período.

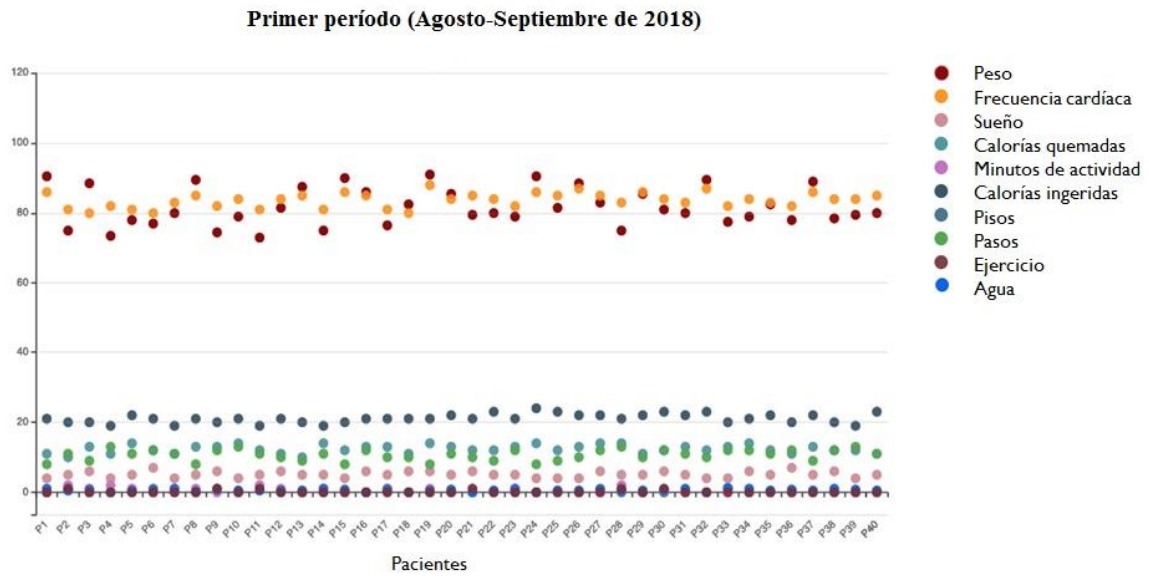


Figura 4.1 Adultos mayores monitoreados en el primer período.

En el segundo período (Octubre de 2018 a Enero de 2019), se introdujo la PISIoT para controlar el peso de las personas mayores. En primera instancia, la PISIoT solicitó, a través de los servicios REST desarrollados, los datos recopilados por los dispositivos inteligentes a cada proveedor durante el primer período de cada paciente (Agosto-Septiembre de 2018). Con base en esta información, la PISIoT obtuvo el IMC de los adultos mayores, que estuvo entre 30 y 35, clasificándolos en el nivel de obesidad 1, y posteriormente comenzó el segundo período de monitoreo. La Figura 4.2 inciso a) muestra las variables biomédicas y otras variables monitorizadas por los dispositivos inteligentes en la PISIoT durante una semana para un adulto mayor con obesidad de 76 años de edad, con una estatura de 1.66 metros, un peso de 91 kilogramos y un IMC de 33.1, la monitorización del paciente reveló

valores altos en el consumo de calorías y en la frecuencia cardíaca promedio diaria en reposo. Además, se identificaron valores bajos para la actividad física, pasos diarios, pisos, calorías quemadas, sueño y consumo de agua. La PISIoT proporcionó las recomendaciones de acuerdo con las variables críticas identificadas cada día. Cuando las recomendaciones diarias hechas para ciertas variables no se cumplieron durante tres días, la PISIoT generó una recomendación médica mayor. La Figura 4.2 inciso b) muestra las recomendaciones relacionadas con el peso para el adulto mayor con obesidad de 76 años.

Figura 4.2 Segundo período de monitorización: (a) variables; (b) recomendaciones de peso.

Adicionalmente, en el caso del adulto mayor con obesidad de 76 años, la PISIoT recomendó solicitar un servicio de análisis clínicos y un servicio de nutriólogo. Además, la PISIoT sugirió automáticamente los servicios médicos basados en el IoT para permitir a los ancianos seleccionar el servicio que más se adapte según su disponibilidad de tiempo, la ubicación, el costo y la confianza. Así mismo, la PISIoT notificó al médico familiar del paciente. Los resultados del análisis clínico fueron altos debido a una dieta y movilidad deficiente. Por esta razón, el adulto mayor recibió un nuevo plan de alimentación en la primera semana del segundo período de monitorización, con el propósito de reducir su peso, IMC y reducir la probabilidad de sufrir un infarto de miocardio. Posteriormente, siguiendo el nuevo plan nutricional y las recomendaciones de la PISIoT, se continuó con el proceso de monitorización. Al final del segundo período, el adulto mayor de 76 años logró perder 5 kilogramos, logrando un peso de 86 kilogramos y un IMC de 31.2.

Por otro lado, al final del segundo período, se identificó que el 100% de las personas mayores lograron perder peso (de 1 a 5 kilogramos) y por consiguiente bajaron su IMC. La Figura 4.3 presenta el promedio semanal de cada variable para cada uno de los 40 adultos mayores en la última semana del segundo período.

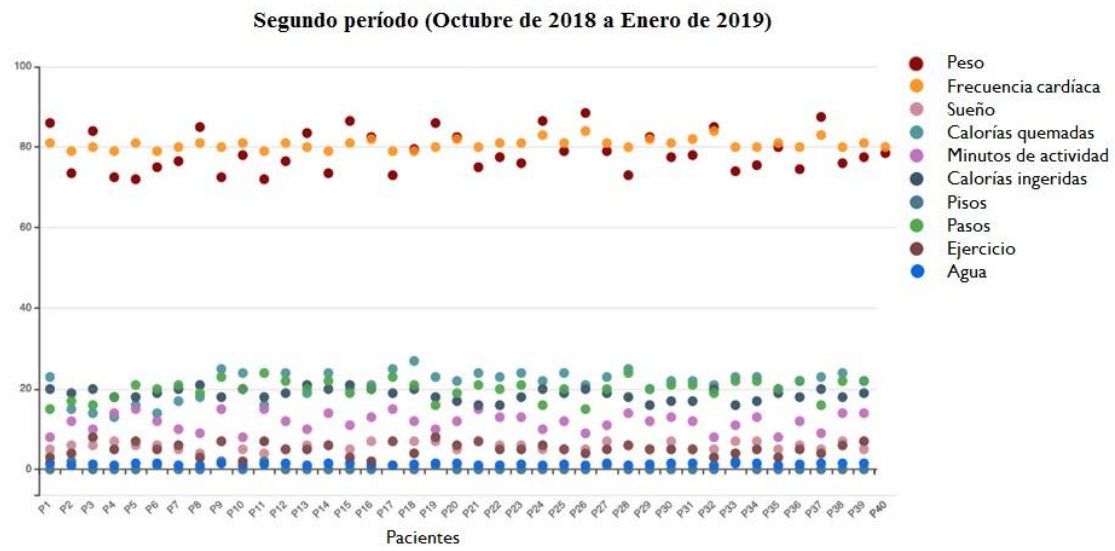


Figura 4.3 Adultos mayores monitorizados en el segundo período

Además, se descubrió que la pérdida de peso se logró en gran medida gracias la monitorización en tiempo real y a las recomendaciones de los profesionales de la salud que fueron generadas por la PISIoT, una vez que se identificaron las variables de riesgo de aumento de peso. Además, después del análisis de datos, se encontró que los pacientes cumplieron algunas o la mayoría de las recomendaciones emitidas por la PISIoT, lo que posteriormente se reflejó en un cambio positivo en las variables relacionadas con las recomendaciones.

En el tercer período (Febrero-Mayo de 2019), aunque el peso y el IMC de los adultos mayores disminuyeron, se mantuvieron en el nivel de obesidad 1. La Figura 4.4 inciso a) muestra las variables biomédicas y otras variables monitorizadas en este período para un adulto mayor con obesidad de 76 años (ahora con un nuevo peso de 86 kilogramos), en el que se observó que hubo una disminución en la ingesta de calorías y una mejora en el promedio en reposo de la frecuencia cardíaca. Además, se identificó un aumento en los pasos diarios, las calorías quemadas, los minutos de actividad y el consumo de agua. Sin embargo, también se observó que el adulto mayor dormía menos de ocho horas. Nuevamente, la PISIoT proporcionó las recomendaciones correspondientes basadas en el árbol de reglas de recomendación para la obesidad 1.

Por otro lado, ante el incumplimiento de ciertas recomendaciones diarias para ciertas variables en un período de tres días, la PISIoT generó nuevas recomendaciones. Las recomendaciones relacionadas con el sueño para el adulto mayor con obesidad de 76 años se presentan en la Figura 4.4 inciso b). En donde, la PISIoT recomendó al paciente la solicitud del servicio de nutriólogo para elaborar un nuevo plan de alimentación.

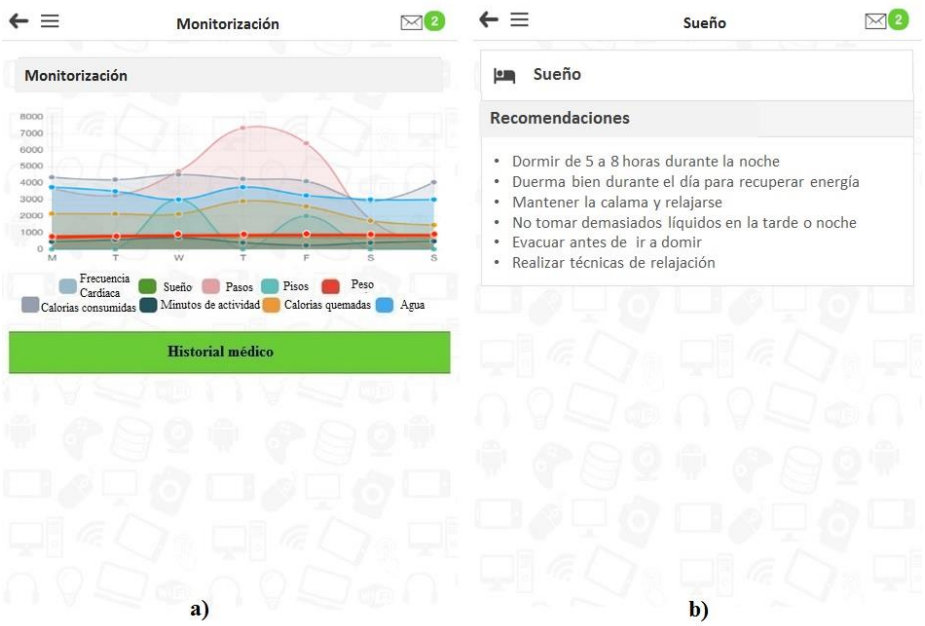


Figura 4.4 Monitorización del tercer período: (a) variables; (b) recomendaciones de sueño.

En la primera semana del tercer período de monitorización, el adulto mayor con obesidad recibió su nuevo plan de alimentación con el objetivo de continuar perdiendo peso, reduciendo su IMC y la probabilidad de un infarto de miocardio, mejorando así su salud. Por lo tanto, la monitorización continuó haciendo uso de la PISIoT, y al final del tercer período, el adulto mayor perdió otros 7 kilogramos, logrando un nuevo peso de 79 kilogramos y un IMC de 28.7. Por tal motivo, la PISIoT lo reclasificó ahora con sobrepeso.

Adicionalmente, la PISIoT recomendó solicitar el servicio de análisis clínico y de nutrición para el adulto mayor de 76 años de edad, para confirmar un buen estado de salud para su nuevo peso y para elaborar un nuevo plan nutricional para mantener este nuevo peso y evitar cambios drásticos si el paciente pierde mucho peso en un periodo corto de tiempo, lo cual no es recomendable debido al riesgo de descompensación u otras complicaciones. Finalmente, al final del tercer período, se identificó que, afortunadamente, el 40% de los adultos mayores monitorizados lograron una pérdida de peso (de 1 a 7 kilogramos) y un IMC más bajo. En consecuencia, la PISIoT los reclasificó; es decir, pasaron de obesidad 1 a so-

brepeso. La pérdida de peso con la utilización de la PISIoT redujo considerablemente la probabilidad de infarto de miocardio para los pacientes, mejoró su salud y aumentó su calidad de vida. Estos resultados se lograron gracias a que la PISIoT utiliza una combinación de dispositivos portátiles, dispositivos inteligentes, el aprendizaje automático y el IoT para prevenir, tratar y controlar el sobrepeso, la obesidad y las enfermedades o problemas de salud asociados a estas enfermedades crónico degenerativas.

Por otra parte, al utilizar la PISIoT en el caso de estudio de la composición de servicios en salud en el contexto del IoT, fue posible identificar una correlación entre las variables biomédicas (frecuencia cardíaca, sueño, calorías quemadas, peso y minutos de actividad física) y las otras variables (pasos, pisos, calorías consumidas, distancia recorrida, consumo de agua y ejercicio) detectadas por los dispositivos inteligentes (dispositivo portátil y báscula inteligente). Esta correlación se presenta en la Tabla 4.1, donde, además, las variables se colocaron en relación con la correlación más alta identificada para cada variable biomédica. Algunas correlaciones se describen a continuación:

- El aumento de peso en los pacientes se debió a un aumento en el consumo de calorías, pocos o ningún minuto de actividad física, una reducción en la cantidad de pasos y pisos, poco ejercicio, poco o nulo consumo de agua y una disminución en las calorías quemadas, y todo lo anterior influye para que se produzca un aumento en la frecuencia cardíaca del paciente.
- La frecuencia cardíaca alta en los pacientes se correlaciona con otras variables, como el aumento de sueño y, por lo tanto, en el peso debido a una disminución en minutos de actividad física, ejercicio, pasos, pisos y calorías quemadas.
- Dormir poco se correlaciona con un aumento en la frecuencia cardíaca y, en consecuencia, un mayor consumo de calorías. Asimismo, hay una disminución de las calorías quemadas y, en consecuencia, un posible aumento de peso.
- La falta de calorías quemadas se correlaciona con una baja actividad física, pasos y ejercicio, y hay un aumento en la frecuencia cardíaca, el sueño y posiblemente en el peso.
- El aumento de la actividad física contribuye a un aumento en el número de pasos, pisos y ejercicio, y es posible una mejora en la frecuencia cardíaca. Además, hay un aumento en las calorías quemadas, y si se mantiene la misma ingesta de calorías, el peso disminuye.

Variables biomédicas	Peso	Frecuencia cardíaca	Sueño	Calo rias quema-das	Minutos de ac-tividad física
Correlación de variables	Calorías consumidas	Sueño	Frecuencia cardíaca	Minutos de actividad físi-ca	Pasos
	Minutos de actividad física	Minutos de actividad física	Calorías quemadas	Pasos	Pisos
	Pasos	Ejercicio	Calorías consumidas	Pisos	Ejercicio
	Pisos	Pasos	Peso	Ejercicio	Frecuencia cardíaca
	Ejercicio	Pisos		Frecuencia cardíaca	Calorías quemadas
	Agua	Calorías consumidas		Peso	Calorías consumidas
	Frecuencia cardíaca	Peso		Sueño	Peso
	Calorías quemadas				

Tabla 4.1 Correlación entre variables biomédicas y otras variables

Por otra parte, se observó que la pérdida de peso de los adultos mayores fue en gran medida a su disciplina y disposición para seguir los planes de alimentación propuestos y cumplir con las recomendaciones emitidas por la PISIoT durante los períodos de monitorización, y sobre todo por la honestidad al informar puntualmente los alimentos y bebidas consumidos. En este sentido, se encontró que el apoyo de un familiar es vital porque los miembros de este grupo de edad tienen dificultades para usar las nuevas tecnologías, especialmente aplicaciones de este tipo. En algunos otros casos, los adultos mayores olvidaban registrar los alimentos que consumían en el transcurso del día. Por esta razón, el familiar fue el encargado de registrar los alimentos y bebidas, ayudando así a los adultos mayores brindándoles una mayor motivación para lograr su objetivo de pérdida de peso y reducir la probabilidad de sufrir un infarto de miocardio u otras enfermedades asociadas con el sobrepeso y la obesidad.

4.2 Resultados de la composición de servicios en la domótica

HEMS-IoT ofreció a los residentes de las casas inteligentes una expectativa más significativa de consumo de energía en los hogares inteligentes, ya que recopiló datos más profundos a través de su capa de servicio utilizando el aprendizaje automático y las tecnologías de big data.

Por otra parte, se realizó un análisis de datos diarios, semanales y mensuales para identificar tendencias en el consumo de energía. Debido a la falta de datos, se utilizó la técnica de agregación promedio para calcular el promedio de la valoración original y transferir este valor a una frecuencia menor. Según los expertos, esta técnica presentó resultados notables cuando se aplica en redes de sensores (Considine et al., 2004) (Rajagopalan y Varshney, 2006) (Fasolo et al. 2007). Así mismo, se analizaron las recomendaciones de HEMS-IoT y se organizaron los datos de consumo de energía estableciendo un orden continuo y se asignó 0 como el valor de consumo diario de energía a los dispositivos domóticos que no se usan a diario, como la plancha.

Con respecto a la iluminación y la temperatura de la habitación, se encontró que del 12.5% al 22.5% de la energía total es consumida por los sistemas de aire acondicionado. Además, los resultados revelan que durante los primeros meses del segundo período de monitorización, los residentes de las casas todavía prestaron poca atención a la cantidad de energía que estaban usando; es decir, se encontró evidencia de uso excesivo de los aires acondicionados y luces encendidas en habitaciones vacías. También se identificaron hogares inteligentes donde los niveles de consumo de energía fueron más altos cuando los residentes adultos estaban ausentes. Este fenómeno fue particularmente visible en hogares con niños. Al final, estos hogares se clasificaron en la categoría de eficiencia de consumo de energía baja como resultado del análisis de conglomerados. También se identificaron dos hogares inteligentes donde el consumo de energía fue desproporcionado con respecto al número de habitantes. Este fenómeno se explica por el uso de los aires acondicionados.

Para identificar los patrones de consumo de energía, se realizó un análisis diario prestando especial atención a la época del año: primavera, verano, otoño e invierno. En general, se identificó que el consumo diario de energía era mayor en la noche y en la tarde que en la mañana. Asimismo, se descubrió que algunas horas son indicadores importantes del consumo diario de energía en las casas inteligentes. Los resultados también indicaron que los niveles de uso de energía varían significativamente según las estaciones. Sin lugar a dudas, la demanda de energía eléctrica aumentó durante el invierno (uso de calefacción) y disminuyó durante el verano (uso de aires acondicionados). Finalmente, los resultados confirman que durante las vacaciones escolares de verano (Julio - Agosto), el consumo de energía en las casas inteligentes con el segundo diseño aumentó significativamente debido a las consolas de videojuegos.

El análisis del consumo de energía en las casas inteligentes también permitió proponer una serie de recomendaciones de ahorro de energía que ayudaron a los residentes a reducir el

consumo de energía y reducir los pagos de energía. En general, es muy importante prestar mucha atención al uso de las luces de las habitaciones y los sistemas de aire acondicionado. Según el Ministerio de Energía de México (SENER), ambos causan una gran cantidad de desperdicio de electricidad. También se realizó una simulación de solicitudes de servicio en el IoT, ya que esta funcionalidad del sistema no se aprovechó completamente durante el caso de estudio debido a que los proveedores de los servicios no cuentan con la infraestructura necesaria. Los resultados obtenidos de la simulación fueron satisfactorios, lo que confirma que HEMS-IoT está bien preparado para invocar los servicios en el IoT tan pronto como los proveedores adopten este nuevo paradigma.

Por otro lado, durante la ejecución del caso de estudio, se reunieron datos de consumo de energía antes de que los residentes hicieran uso de la aplicación HEMS-IoT, pero también mientras la utilizaban. Como resultado, se realizó una comparación del antes y después para identificar las diferencias entre los dos períodos de monitorización (mediados de Enero a mediados de Septiembre de 2018 y mediados de Enero a mediados de Septiembre de 2019). Dado que en México los recibos de energía se emiten cada dos meses, el análisis sigue la misma tendencia. Según los resultados, representados visualmente en la Figura 4.5, HEMS-IoT disminuyó el consumo de energía de 42 kWh a 90 kWh cada dos meses. Se atribuyen estos resultados al hecho de que los residentes de las casas inteligentes siguieron las recomendaciones de ahorro de energía emitidas por HEMS-IoT. Además, se observó que los resultados satisfactorios también se debieron a los siguientes aspectos:

- Interés de los residentes de las casas inteligentes por reducir el consumo de energía y disciplina para cambiar los hábitos de consumo de energía.
- Uso adecuado y óptimo de los dispositivos domóticos, particularmente aquellos con mayor demanda de energía, como los aires acondicionados (por ejemplo, limitando la temperatura ambiente de 20 a 25 ° C y configurando el apagado automático del dispositivo para habitaciones vacías).
- Aceptación de las recomendaciones HEMS-IoT para el ahorro de energía. El consumo de energía depende en gran medida del comportamiento de los residentes. En este sentido, el rendimiento del sistema depende de si los residentes siguen las recomendaciones de HEMS-IoT. Por ejemplo, los resultados revelan cambios de comportamiento con respecto al uso del aire acondicionado. Inicialmente, estos se usaron por períodos de tiempo más largos, incluso en habitaciones vacías; sin embargo, durante el segundo período de monitorización, se identificó evidencia de que

los residentes limitaron las temperaturas del aire acondicionado de 18 a 24 ° C y configuraron la aplicación para apagar automáticamente el aire acondicionado una vez que la habitación estaba vacía.

- Los residentes de las casas inteligentes que aceptaron más recomendaciones de HEMS-IoT y cambiaron sus hábitos de automatización del hogar lograron reducir mejor su consumo de energía.

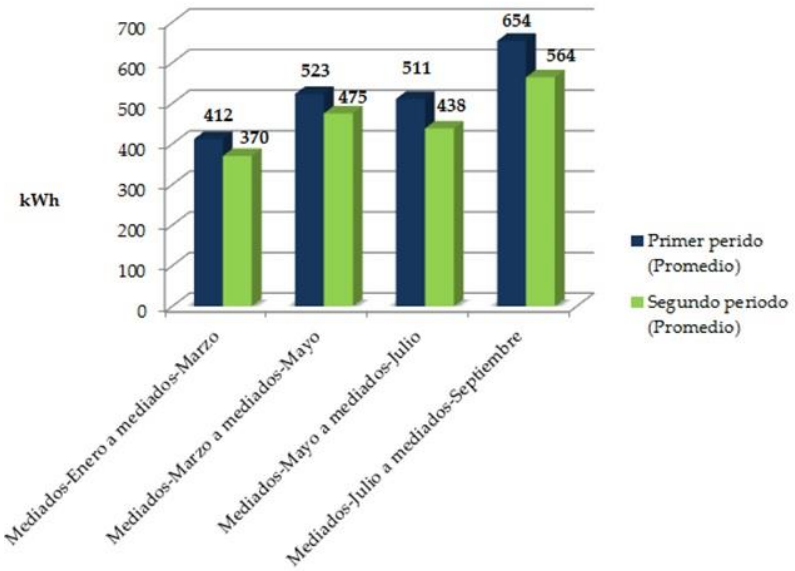


Figura 4.5 Comparación de consumo de energía.

Por otra parte, es muy importante que las aplicaciones para dispositivos móviles logren sus objetivos con respecto a la satisfacción del usuario final, que depende en gran medida de aspectos de calidad como el diseño, el servicio, la asistencia, la utilidad y la usabilidad. Por otro lado, para evaluar los sistemas de recomendación, las métricas de precisión y exactitud se utilizan ampliamente en la literatura. Sin embargo, varios investigadores encontraron que el uso de sistemas de recomendación que brindan asistencia personalizada para encontrar información relevante genera una alta percepción de satisfacción (Konstan y Riedl, 2012) (McNee et al., 2006). Ante este contexto, es necesario investigar los sistemas de recomendación desde una perspectiva centrada en el usuario (Knijnenburg y Willemsen, 2015). Por lo tanto, el sistema HEMS-IoT se evaluó a través del Marco de evaluación centrado en el usuario para sistemas de recomendación (Knijnenburg et al., 2012) que se basa en una teoría sobre el comportamiento humano que se utiliza en situaciones como el uso de un sistema específico.

El objetivo general de esta última evaluación fue medir la percepción del usuario de HEMS-IoT con respecto a la calidad de las recomendaciones generadas por el sistema, el nivel de satisfacción, la efectividad y la intención de uso. La Figura 4.6 representa el modelo teórico utilizado en este experimento. Específicamente, las suposiciones apuntan a explicar la calidad de recomendación percibida por el usuario sobre el método de recomendación de ahorro de energía, así como también cómo la calidad percibida influye en la efectividad del sistema y la satisfacción percibida por el usuario. Además, este experimento consideró la intención de uso del usuario (Lee y Choi, 2017), con el objetivo de determinar si la efectividad del sistema y la satisfacción percibida del usuario tienen una influencia positiva en los usuarios para continuar usando el sistema HEMS-IoT.

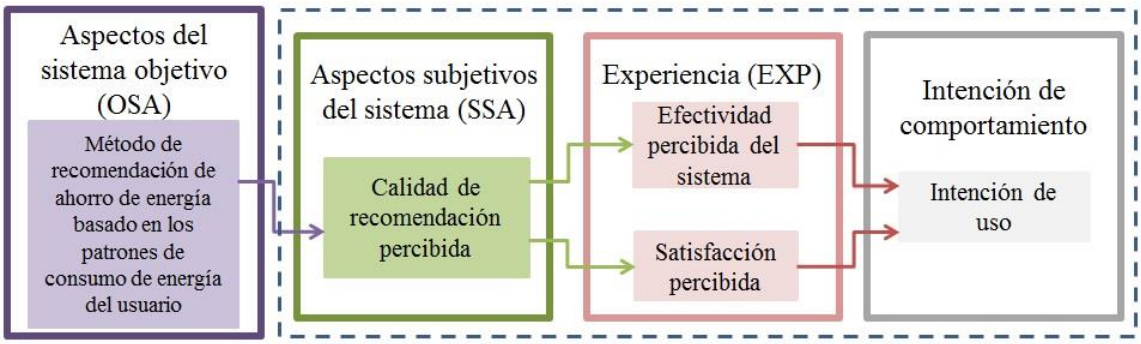


Figura 4.6 Modelo teórico centrado en el usuario

A las personas involucradas en este caso de estudio (21 mujeres y 14 hombres entre 18 y 59 años) se les hicieron preguntas para medir su experiencia y percepciones usando HEMS-IoT. El cuestionario utilizado en este trabajo fue adaptado de instrumentos estandarizados de recolección de datos enfocados en medir la interacción entre el usuario y el sistema.

Específicamente, el cuestionario utilizado en (Knijnenburg et al., 2012) fue ligeramente adaptado para identificar la percepción del usuario con respecto a la satisfacción percibida, la efectividad percibida del sistema y la calidad de las recomendaciones de ahorro de energía. Mientras tanto, el cuestionario propuesto en (Lee y Choi, 2017) se utilizó para medir la calidad de recomendación percibida, y los cuestionarios presentados en (Pu et al., 2011) y (Herbjørn y Breivik, 2005) se utilizaron para identificar la intención de uso del usuario. La Tabla 4.2 describe el cuestionario final, que se basa en una escala Likert de cinco puntos ("Totalmente en desacuerdo" a "Totalmente de acuerdo") que ayuda a cuantificar la satisfacción del usuario. La Tabla 4.3 describe los resultados obtenidos para la calidad de recomendación percibida (PRQ) de los usuarios, la satisfacción percibida (PS), la efectividad

percibida del sistema (PSE), así como su intención de uso (ITU). Como se observa, los resultados promedio fueron 3.81 para PRQ, 3.45 para PS, 2.98 para PSE y 4.07 para ITU.

Percepción de la calidad de recomendación	**Eficacia percibida del sistema**
1. Me gustan las recomendaciones de ahorro de energía proporcionadas por el sistema.	11. Es útil el sistema.
2. Las recomendaciones de ahorro de energía se ajustan a mis preferencias de confort.	12. El sistema me hace más consciente sobre el consumo de energía en casa.
3. Las recomendaciones de ahorro de energía proporcionadas por el sistema fueron bien elegidas	13. Tomo mejores decisiones de ahorro de energía con el sistema
4. Las recomendaciones de ahorro de energía fueron relevantes.	14. Se tiene mejor ahorro de energía sin la ayuda del sistema
5. El sistema proporcionó diversas recomendaciones de ahorro de energía ineficientes	15. Se disminuye el costo por el consumo de energía utilizando el sistema
Percepción de la satisfacción	**Intención de uso**
6. Me gustan las recomendaciones de ahorro de energía que he aceptado.	16. Utilizaré nuevamente este sistema para el ahorro de energía
7. Estoy a gusto con las recomendaciones de ahorro de energía aceptadas.	17. Utilizaré con más frecuencia este sistema para el ahorro de energía
8. Me siento contento de tener un consumo de energía más eficiente.	18. Le platicaré a mis amigos o conocidos sobre este sistema
9. Recomendaría algunas de las recomendaciones de ahorro de energía que he aceptado a amigos o familiares.	19. Es muy probable que utilice este sistema para el ahorro de energía en casa
10. Las recomendaciones de ahorro de energía aceptadas se ajustan a mis preferencias de confort.	20. Es muy probable que recomiende a mis familiares utilizar este sistema

Tabla 4.2 Cuestionario

Para identificar la importancia de todos los aspectos presentados en la Figura 4.6, los resultados obtenidos se analizaron estadísticamente utilizando el valor T (estadístico), P (importancia) y la correlación (efecto de tamaño).

Usuario	PRQ	PS	PSE	ITU
U1	3.8	3.2	3.2	3.8
U2	3.8	3.2	2.6	4.4
U3	4.2	3.4	3.2	3.8
U4	3.6	3.6	3.2	3.8
U5	3.4	3.2	3.2	3.8
U6	3.8	3.2	3.2	4.2
U7	3.6	3.4	2.8	3.8
U8	4.2	3.2	3.2	4.4
U9	3.4	3.4	2.6	3.8
U10	3.6	3.2	3.2	4.2
U11	4.2	3.2	2.8	4.2
U12	3.6	3.4	2.8	3.8
U13	3.8	3.4	2.8	3.8
U14	3.8	3.2	2.8	4.2
U15	3.6	3.4	3.2	4.2
U16	3.6	3.2	2.8	4.2
U17	3.6	3.4	2.6	4.2
U18	3.8	3.2	3.2	4.2
U19	4.2	4.2	3.2	3.8
U20	3.8	3.6	2.8	4.2
U21	3.8	3.6	3.4	3.6
U22	3.8	3.4	3.2	4.2
U23	3.8	3.8	2.8	4.2
U24	3.8	3.4	3.2	3.8
U25	3.6	3.4	2.8	3.8
U26	4.2	3.8	2.8	3.8
U27	4.2	3.6	2.6	4.4
U28	3.8	3.6	3.2	4.2
U29	3.8	3.6	2.6	4.2
U30	3.8	3.6	2.6	4.2
U31	3.8	3.8	3.2	4.4
U32	3.8	3.4	3.2	4.2
U33	3.8	3.6	3.2	4.2
U34	4.2	3.8	3.2	4.2
U35	3.8	3.2	2.8	4.2
Promedio	3.81	3.45	2.98	4.07

Tabla 4.3 Resultados de la evaluación centrada en el usuario.

La Figura 4.7 representa los resultados obtenidos de este análisis. Además, se realizó la correlación de Pearson para determinar la relación entre la calidad de recomendación percibida, la efectividad percibida del sistema y la satisfacción percibida. Como se observa en la Figura 4.7, la calidad de recomendación percibida influye positivamente en la percepción de la efectividad del sistema ($r = 0.675$, $p < 0.5$) y la satisfacción percibida ($r = 0.891$, $p < 0.01$). Además, la efectividad percibida del sistema ($r = 0.669$, $p < 0.05$) y la satisfacción percibida ($r = 0.943$, $p < 0.01$) tienen una influencia importante en la intención de uso del usuario.

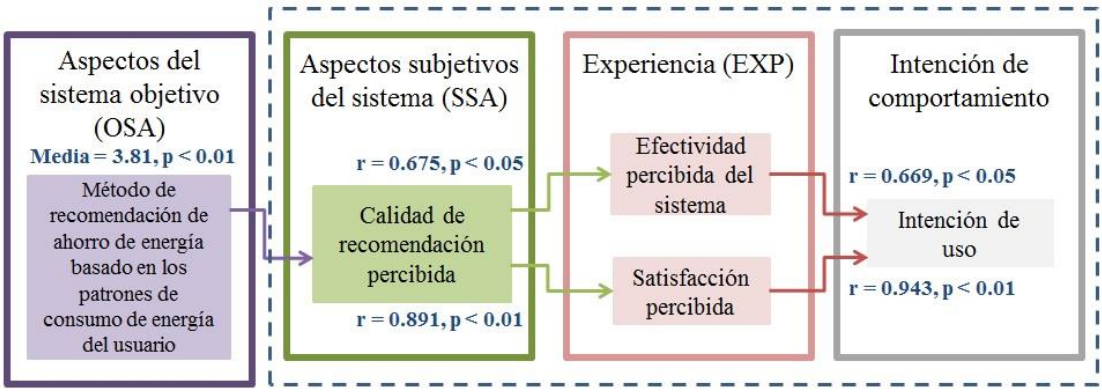


Figura 4.7 Resultados experimentales.

Finalmente, se identificó que la satisfacción percibida obtuvo valores de correlación más altos que los obtenidos por la efectividad percibida del sistema; por lo tanto, se concluye que la satisfacción percibida tiene un efecto más fuerte en la intención del usuario de continuar utilizando el sistema HEMS-IoT.

4.3 Resultados de la integración y coordinación de servicios en el IoT

Los resultados obtenidos de la implementación de las extensiones **each, update** y **updated** que integran el modelo SCM-IoT para la integración y coordinación de servicios en el IoT se presentan en tres apartados: selecciones, actualizaciones y notificaciones por actualización.

Selecciones

La selección es un servicio de notificación diseñado para los subscriptores de contenido interesados en reconocer la existencia en el libro de hojas cuyo contenido cumple condiciones preestablecidas. En el lenguaje de marcado del modelo SCM-IoT, la selección de hojas en el libro se denota por el elemento `<iota:each page="…">…</iota:each>` junto con algunos elementos adicionales los cuales se describen a continuación:

- El libro origen de las páginas cuyo nombre está dado por el atributo **notebook** del elemento `<iota:from notebook="…">`

- La condición que cumplen todas las páginas para incluirse en la selección está indicada por el elemento `<iota:when>…</iota:when>`

La selección es una operación indivisible sobre sus páginas, es decir, una vez seleccionada una página, esta no pertenece a otra selección ni tampoco es parte de alguna otra notificación. Estas restricciones garantizan la consistencia e integridad del contenido del libro.
Una selección en el lenguaje de marcado se definió para localizar residentes en cada habitación de una casa pero que aún no fueron avisados de alguna emergencia. El siguiente fragmento de código muestra el resultado de cómo se expresó este tipo de notificación en el lenguaje de marcado:

```
<iota:each page="habitacion">
  <iota:from notebook="habitaciones"/>
  <iota:when>
    $habita/presencia = 'SI' and
    $habita/Alarma = 'NO'
  </iota:when>
</iota:each>
```

En este resultado, la selección de páginas proviene del libro `habitaciones` siempre que el contenido cumpla con dos condiciones: 1) Si el campo presencia tiene el valor 'SI' y 2) Si el campo Alarma tiene el valor 'NO'. Una vez seleccionadas todas las habitaciones ocupadas por al menos un residente, se actualiza el campo Alarma con el valor 'SI' para establecer la notificación correspondiente. Dicha actualización evita notificar reiteradamente de la emergencia por esta notificación.
La actualización de las páginas de un libro es una operación fundamental que asegura la consistencia de la información observada del entorno físico del sistema y aquella que el modelo del sistema tiene de dicho entorno.

Actualizaciones

Una actualización es uno de los servicios que se ofrecen para administrar el contenido del libro. En particular, este servicio está dirigido a los productores de contenido. El propósito de una actualización es modificar el contenido de una de las páginas del libro de acuerdo con lo establecido en la nota del servicio.

En el lenguaje de marcado de SCM-IoT, la operación de actualización se ingresó con el elemento `<iota:update notebook="...">`, el cual indica el nombre del libro en donde se solicita realizar la actualización. Además del nombre del libro, la actualización requiere de los siguientes elementos adicionales:

- La ***nota***, representada por el elemento `<iota:with note="…"/>`, indica el nombre de la variable que contiene la información necesaria para establecer la actualización como el identificador de la página y el nuevo valor con el que se modifica uno de sus campos.
- La ***copia válida***, representada por el elemento `<iota:latter page="…"/>`, indica el nombre de la variable que contiene la última actualización disponible conocida por el sistema.
- La ***copia del original***, indicada por elemento `<iota:former page="…"/>`, establece el nombre de la variable en donde se encuentra una copia del contenido previo de la página, anterior a la actualización.
- La ***condición***, indicada por el elemento `<iota:when>…</iota:when>`, establece el criterio para realizar o no la actualización, generalmente que el valor del campo indicado en la nota sea distinto del asentado en la página del libro.

Por otra parte, se enfatiza que el hecho de que el contenido de la copia válida es idéntico al original excepto en el campo dado en la nota. La validez de una copia asegura que su contenido corresponde con aquel del entorno físico externo en el que se encuentra inmerso el sistema de domótica. En consecuencia, la validez de una copia se mantiene vigente hasta que ingresa alguna petición relevante de actualización para ella.

Tan pronto como una copia ha perdido su validez, su contenido se deposita en la próxima copia del original para su futura referencia. La condición asegura que solamente cambios significativos se realicen en el libro lo cual disminuye el tráfico de solicitudes en la red, con lo cual a su vez se disminuye el número de notificaciones de actualización a los subscriptores.

Por consiguiente, el siguiente fragmento de código muestra el resultado de una actualización para el reporte de datos obtenido por el sensor de presencia:

```
<iota:update notebook="habitaciones">
  <iota:with note="sensorPresencia"/>
  <iota:former page="habitacionAntes"/>
  <iota:latter page="habitacion"/>
  <iota:when>
    $sensorPresencia/lectura != $habitacionAntes/presencia
  </iota:when>
</iota:update>
```

En este resultado, se solicitó realizar la actualización en el libro "habitaciones" de acuerdo con lo establecido en la nota "sensorPresencia", sobre el contenido original de la página copiada en la variable "habitacionAntes" y cuya actualización se depositó en la variable "habitacion". Sin embargo, la actualización solo se lleva a cabo si se cumple la condición "$sensorPresencia/lectura != $habitacionAntes/presencia", es decir, cuando la lectura reportada por el sensor es distinta a la asentada en el libro para la página solicitada. La actualización es un proceso que transita por varias etapas tal y como se muestra en la siguiente instancia del código BPEL que le corresponde:

```
<bpel:sequence>
  <!-- CORRESPONDENCIA -->
  <bpel:if>
    <bpel:condition>
      $sensorPresencia/@id != $habitacion/@id
    </bpel:condition>
    <bpel:assign>
      <bpel:from>$habitaciones/habitacion[@id=$sensorPresencia/@id]</bpel:from>
      <bpel:to>$habitacion</bpel:to>
    </bpel:assign>
  </bpel:if>
  <!-- RELEVANCIA -->
  <bpel:if>
    <bpel:condition>
      $sensorPresencia/lectura != $habitacion/presencia
    </bpel:condition>
    <bpel:sequence>
      <!—- TRANSICION -->
      <bpel:assign>
        <bpel:from>$habitacion</bpel:from>
        <bpel:to>$habitacionAntes</bpel:to>
      </bpel:assign>
      <!-- ACTUALIZACION -->
      <bpel:assign>
        <bpel:from>$sensorPresencia/presencia</bpel:from>
        <bpel:to>$habitacion</bpel:to>
      </bpel:assign>
      <bpel:assign>
        <!-- CONSISTENCIA -->
        <bpel:from>$habitacion</bpel:from>
        <bpel:to>$habitaciones/habitacion[@id=$sensorPresencia/@id]</bpel:to>
      </bpel:assign>
    </bpel:sequence>
  </bpel:if>
</bpel:sequence>
```

Recibida la petición de actualización en la nota, las etapas del proceso de actualización fueron las siguientes:

- **Correspondencia:** se verifica si la copia disponible, indicada por el elemento ***latter***, es válida, es decir, que corresponde con la página solicitada en la nota; si la copia no es válida, se obtiene una copia válida del libro.
- **Relevancia:** se verifica si el valor de la modificación dado en la nota es igual al registrado en la copia válida, la operación concluye sin modificar el contenido del libro; en otro caso, el proceso continúa.
- **Transición:** se obtiene la copia del contenido original obtenida de la copia válida para futura referencia y se deposita en la variable indicada por el elemento ***former***.
- **Actualización:** se actualiza la copia válida asignando en el campo del tópico el valor dado en la nota.
- **Consistencia:** se asegura la consistencia entre la copia válida y el libro, copiando su contenido en la página correspondiente.

Por lado, el tratamiento adecuado de las actualizaciones, en escenarios caracterizados por un gran número de sensores dedicados a monitorear sistemas que tienen un gran dinamismo, requiere extender el número de páginas actualizadas para la notificación oportuna de sus subscriptores. Lo anterior se logra mediante un sistema de colas de páginas válidas en donde se incorporan todas las páginas que se actualizan y requieren ser atendidas por los subscriptores interesados. Puesto que, en el transcurso de su atención es probable que suceda que las páginas se actualicen nuevamente, para el mantenimiento de las colas de páginas es importante con estrategias para el adecuado tratamiento, como la detección y extracción de alguna página inválida y la posterior reinserción en la cola como página válida. Sin embargo, para el dominio de aplicaciones de la domótica, aún para casas grandes en donde los sensores instalados son relativamente pocos y las condiciones del medio ambiente son muy estables, la estrategia elegida en el modelo SCM-IoT para registrar la última página válida fue suficiente.

Además, para demostrar el resultado del efecto de la operación de actualización, se muestran las modificaciones que se producen en el libro en el momento en que la casa se encuentra inicialmente en el estado mostrado en la Tabla 4.4.

id	nombre	presencia	Temperatura	humo	gas	calefactor	ventilador	Alarma
PAS	PASILLO	NO	BAJA	NO	NO	NO	NO	NO
SAL	SALA	SI	BAJA	NO	NO	SI	NO	NO
COM	COMEDOR	NO	BAJA	NO	NO	NO	NO	NO
COC	COCINA	NO	BAJA	NO	NO	NO	NO	NO
REC	RECAMARA	NO	BAJA	NO	NO	NO	NO	NO
EST	ESTUDIO	NO	BAJA	NO	NO	NO	NO	NO

Tabla 4.4 Estado del libro de habitaciones

En estas condiciones, si el residente ubicado en la sala decide salir de ella. Este cambio de estado es detectado por el sensor de presencia instalado en esa habitación, reportando el valor ‘NO’ en un mensaje dirigido al administrador del libro. En la Tabla 4.5 se presenta el contenido de este mensaje.

id	tipo	lectura
SAL	PRESENCIA	NO

Tabla 4.5 Actualización de datos del sensor

Adicionalmente, para la petición, se cumple que **`$sensor/@id='SAL' and $sensor/lectura='NO'`**. La petición establece entonces que la nota de interés es aquella cuyo id es 'SAL', es decir la sala.

Además, para la copia válida de la página, los datos de la Tabla 4.4 indican que se tienen los requisitos de que **`$habitacion/@id='SAL' and $habitacion/presencia='SI'`**. Como se observa, las condiciones de actualización se cumplen por lo que se define el valor de actualización con lo cual se realiza la actualización. Generalmente una actualización desencadena una notificación dirigida al calefactor, de acuerdo a las condiciones de confort establecidas para el sistema.

Notificaciones por actualización

Una notificación es otro de los servicios que se ofrecen para administrar el contenido del libro. En particular, este servicio está dirigido a los subscriptores de contenido. El propósito de una notificación es avisar al subscriptor que el contenido de una de las páginas del libro cumple con sus criterios de selección y actuación. La actuación del subscriptor asegura que

se cumplan con las condiciones de confort o seguridad que esperan los residentes de la casa.

En el lenguaje de marcado de SCM-IoT, la operación de notificación se introduce con el elemento `<iota:updated notebook="...">`, el cual indica el nombre del libro en donde ocurre una actualización. Además del nombre del libro, la notificación requiere de los siguientes elementos adicionales:

- La ***copia válida***, representada por el elemento `<iota:latter page="…"/>`, indica el nombre de la variable que contiene la última actualización disponible conocida por el sistema.
- La ***copia del original***, indicada por el elemento `<iota:former page="…"/>`, establece el nombre de la variable en donde se encuentra una copia del contenido previo de la página, anterior a la actualización.

Mediante estos dos elementos, el subscriptor determina la información útil sobre la naturaleza del cambio. Sin embargo, para establecer las condiciones tanto de confort como de seguridad se comprueba que es suficiente con utilizar solamente la copia válida. El siguiente fragmento de código muestra el resultado de una solicitud de actualización:

```
<iota:updated notebook="habitaciones">
  <iota:latter page="habitacion"/>
</iota:updated>
```

En comparación con el proceso de actualización, el proceso de notificación es simple siempre que el sistema tenga muy poco dinamismo como ocurre en este caso con la domótica de casas habitación. El proceso se reduce a vincular el inicio de la ejecución del subscriptor justo después de que el productor de contenido genera una actualización que es de interés para el subscriptor. En este caso se asegura la vigencia de la copia válida por lo que el subscriptor proporciona una respuesta oportuna a condiciones emergentes.

Para mostrar el resultado del efecto de una notificación desencadenada por una actualización del libro, la Tabla 4.6 muestra el contenido en el momento donde el sensor de la sala reportó que el residente que se encontraba ahí dejo de detectarse, asumiendo que entonces abandonó esa habitación. Además, en la Tabla 4.6 se visualiza el estado del libro de habitaciones cuando el residente no se encuentra en la sala, pero no apagó el calefactor siendo la temperatura aún baja.

id	nombre	presencia	Temperatura	humo	Gas	calefactor	ventilador	Alarma
PAS	PASILLO	NO	BAJA	NO	NO	NO	NO	NO
SAL	SALA	NO	BAJA	NO	NO	SI	NO	NO
COM	COMEDOR	NO	BAJA	NO	NO	NO	NO	NO
COC	COCINA	NO	BAJA	NO	NO	NO	NO	NO
REC	RECAMARA	NO	BAJA	NO	NO	NO	NO	NO
EST	ESTUDIO	NO	BAJA	NO	NO	NO	NO	NO

Tabla 4.6 Presencia de un residente en la sala

De acuerdo con las condiciones de ahorro de energía, en caso de que el residente no haya desactivado la calefacción, el sistema lo apaga después de un plazo precautorio previendo que pronto regrese a la sala. El administrador del libro es el encargado de vincular a productores de contenido con sus subscriptores mediante notificaciones apropiadas. En este sentido, en la Tabla 4.7 se presenta el resultado generado después de entregar una notificación similar a la descrita al subscriptor que controla al calefactor, en donde se genera la orden de desactivar al calefactor correspondiente, modificando con ello al libro en el campo calefactor.

id	nombre	presencia	Temperatura	humo	Gas	calefactor	ventilador	Alarma
PAS	PASILLO	NO	BAJA	NO	NO	NO	NO	NO
SAL	SALA	NO	BAJA	NO	NO	NO	NO	NO
COM	COMEDOR	NO	BAJA	NO	NO	NO	NO	NO
COC	COCINA	NO	BAJA	NO	NO	NO	NO	NO
REC	RECAMARA	NO	BAJA	NO	NO	NO	NO	NO
EST	ESTUDIO	NO	BAJA	NO	NO	NO	NO	NO

Tabla 4.7 Apagar calefactor

Por tal motivo, el comportamiento de productores de contenido y sus subscriptores se codifica en el lenguaje de marcado de reglas del modelo SCM-IoT. Además, es importante señalar que el modelo SCM-IoT consideró elementos que caracterizan a las arquitecturas guiadas por eventos, así como la utilización de las operaciones de selección, actualización y notificación.

Capítulo 5
Conclusiones

El contenido del presente capítulo se enfoca en describir las conclusiones al finalizar el desarrollo e implementación del mecanismo de composición de servicios bajo en el enfoque del IoT que se elaboró en esta tesis doctoral y plantear el trabajo a futuro del proyecto de investigación.

5.1 Conclusiones

El IoT es un paradigma que se incursionó rápidamente en escenarios de la tecnología inalámbrica moderna. La idea básica es la presencia en el entorno del mundo real de diversos objetos interconectados, tales como los RFID, sensores, actuadores, teléfonos inteligentes, entre otros; los cuales a través de esquemas de direccionamiento únicos, interactúan entre sí, cooperando y colaborando con otros objetos vecinos para lograr un objetivo común. Estos dispositivos inteligentes se caracterizan por su capacidad de detección, procesamiento y por la creación de redes.

Por otra parte, en los objetos inteligentes existe una problemática al realizar la SC, la cual es uno de los principios básicos de la SOC, donde los servicios se combinan para atender las solicitudes de los usuarios. Dos aspectos importantes de la SC son la orquestación y la coreografía de servicios. La SC se estudia ampliamente en el contexto de los servicios Web y procesos de negocios, donde se desarrollan y utilizan una serie de estándares para implementaciones en el mundo real. Sin embargo, las características actuales de los sistemas en el IoT, así como la problemática de la restricción de recursos, hacen que las técnicas desarrolladas para la SC Web sean inadecuadas cuando se aplican en el contexto del IoT. Además, actualmente los dispositivos heterogéneos que se conectan en el IoT presentan problemas de conectividad, interoperabilidad y de integración. Adicionalmente, el soporte en tiempo real para los protocolos basados en la Web ha facilitado la llegada de nuevas aplicaciones en el IoT. Del mismo modo, la SC reutiliza varios servicios de componentes existentes al unirse a ellos de una manera creativa; por ello al ser aplicados en el IoT, optimizan el desarrollo de las aplicaciones. Del mismo modo, la SC aplicada en el IoT permite combinar servicios de múltiples objetos inteligentes para satisfacer las solicitudes de los usuarios en

una amplia gama de dominios de aplicaciones y se utiliza para crear aplicaciones innovadoras de una manera más eficiente. Ante este contexto, se requieren mecanismos avanzados que permitan la combinación de un conjunto de servicios atómicos en forma de servicios compuestos, en donde estos servicios compuestos se utilicen a través de la orquestación y coreografía de servicios en diferentes escenarios para satisfacer las necesidades de los usuarios.

Por otro parte, el cuerpo humano proporciona constantemente información sobre el estado de salud. Esta información se obtiene a través de sistemas o dispositivos que miden, capturan o detectan valores y variables en puntos específicos del cuerpo de manera invasiva o no invasiva. Por otra parte, el personal de salud utiliza los valores de las variables biomédicas para tomar decisiones sobre diagnósticos y tratamientos a fin de mejorar la salud de los pacientes. En todo el mundo, la calidad de vida, particularmente entre los ancianos, se ve afectada considerablemente por el sobrepeso y la obesidad. Ante este contexto, el IoT permite interconectar, detectar, identificar y procesar datos entre objetos o servicios para cumplir un objetivo común. Las principales ventajas del IoT en la atención médica son el monitoreo, análisis, diagnóstico y control de afecciones como el sobrepeso y la obesidad, y la generación de recomendaciones para prevenirlas. Sin embargo, los objetos utilizados en el IoT tienen recursos limitados, por lo que es necesario considerar otras alternativas para el análisis de datos, como el aprendizaje automático. El aprendizaje automático es un subconjunto de inteligencia artificial que consiste en estudiar los algoritmos y modelos estadísticos utilizados en los sistemas informáticos para lograr objetivos específicos de manera efectiva, basados en patrones e inferencias.

En la actualidad, existen varios desafíos en el sector de la salud que brindan áreas de oportunidad para que el IoT y el aprendizaje automático proporcionen soluciones o alternativas que contribuyan a mejorar la atención médica y la calidad de vida de las personas. En el caso de estudio de la composición de servicios en salud bajo el enfoque del IoT, se presentó a la PISIoT: una plataforma de salud inteligente basada en IoT y aprendizaje automático para la prevención, detección, tratamiento y monitoreo del sobrepeso y la obesidad. La API Weka y el algoritmo de aprendizaje automático J48 se usaron para identificar variables críticas, mientras que Apache Mahout y RuleML se usaron para generar recomendaciones médicas. Las principales limitaciones de la PISIoT son que, por el momento, se enfoca solo en la detección, prevención, tratamiento y monitoreo del sobrepeso y la obesidad. Del mismo modo, utiliza un tipo de dispositivo portátil y una báscula inteligente, y solo utiliza el algoritmo de aprendizaje automático J48.

Por otra parte, la eficiencia energética se ha convertido en un área clave de investigación, porque el consumo de energía aumenta exponencialmente a medida que pasan los años, particularmente en el sector residencial. Si se combina con el paradigma del IoT, los sistemas de automatización del hogar son alternativas prometedoras de ahorro de energía. El IoT es capaz de recopilar, distribuir y analizar datos con éxito para convertirlos en conocimiento e información; sin embargo, los dispositivos en el IoT para casas inteligentes tienen recursos limitados. Para superar esta limitación, es importante considerar otras alternativas de manejo de datos, como el aprendizaje automático y el big data, para recopilar, administrar y analizar grandes volúmenes de datos. Las tecnologías de análisis de big data se utilizan para obtener y analizar grandes cantidades de datos, mientras que el sistema necesita algoritmos de aprendizaje automático y modelos estadísticos basados en patrones e inferencias para cumplir sus objetivos. Además, el aprendizaje automático ofrece alternativas a los problemas basados en el aprendizaje e identifica los antecedentes y las características de dichos problemas para aprender de ellos y aumentar el funcionamiento del sistema.

Adicionalmente, los desafíos actuales en el sector residencial relacionados con el consumo de energía son áreas de oportunidad para el paradigma del IoT, el aprendizaje automático y las tecnologías de big data. Las técnicas de aprendizaje automático y las tecnologías de big data son importantes para el sistema HEMS-IoT, ya que son utilizadas para analizar y clasificar la eficiencia del consumo de energía, identificar patrones de comportamiento del usuario y garantizar la comodidad en el hogar. Se utilizó el algoritmo de aprendizaje automático J48 y Weka API para aprender los patrones de consumo de energía y los patrones de comportamiento del usuario. Además, RuleML y Apache Mahout se usaron para crear recomendaciones de ahorro de energía basadas en las preferencias del usuario para preservar la comodidad y seguridad de la casa inteligente. En conclusión, tres factores fueron fundamentales para lograr la reducción del consumo de energía en el caso de estudio de la composición de servicios en la domótica: 1) el compromiso de los habitantes de hogares inteligentes para cambiar sus hábitos de consumo de energía, 2) el seguimiento de las recomendaciones de ahorro de energía del sistema y 3) el hecho que el sistema permite a los usuarios modificar los parámetros operativos de los dispositivos domóticos. Desde esta perspectiva, se espera que los resultados obtenidos en este trabajo sirvan de motivación para que más usuarios confíen en HEMS-IoT cuando busquen una alternativa de administración inteligente del hogar que permita optimizar el consumo de energía y, por lo tanto, ahorrar. El sistema HEMS-IoT tiene cinco limitaciones principales.

1. La aplicación para dispositivos móviles solo funciona en el sistema operativo Android, aunque se conoce que el mercado de tabletas está dominado por iOs.
2. El sistema solo es compatible con algunos tipos de sensores de automatización del hogar.
3. Solo se utilizaron tecnologías de big data y el algoritmo de aprendizaje automático J48.
4. El sistema no genera recomendaciones personalizadas de ahorro de energía debido a algunas limitaciones surgidas durante el proceso de investigación.
5. No se recopilan datos sobre el consumo de energía de dispositivos domóticos antes de la implementación de HEMS-IoT.

Por otra parte, las Redes de Petri son una herramienta de modelado gráfica y matemática para describir y estudiar las relaciones entre partes de un sistema que se caracterizan por ser concurrentes, asíncronos, distribuidos, paralelos, no deterministas y/o estocástico. Además, se utilizan *tokens* o marcas para simular las actividades dinámicas y concurrentes de un sistema. Las CPN son una red Petri clásica extendida y son una gran herramienta para el modelado y visualización de comportamientos dentro un sistema, pues proporcionan: descripciones jerárquicas, simulaciones interactivas y una representación gráfica intuitiva y atractiva. Por tal motivo, el modelo SCM-IoT fue descrito formalmente utilizando las CPN debido a que estas constituyen un método formal con fundamentos bien establecidos que permite modelar sistemas que exhiben un elevado grado de concurrencia como aquellos que caracterizan a los sistemas IoT.

Además, SCM-IoT es un modelo para el desarrollo de aplicaciones en el contexto del IoT que tiene una arquitectura orientada a datos lógicamente centralizada y permite extender o simplificar el desarrollo de las aplicaciones que son propias de la inteligencia artificial o del aprendizaje automatizado. Dichas aplicaciones involucran la colaboración de agentes inteligentes que intercambian información a través de un repositorio. Entre las ventajas que el modelo SCM-IoT proporciona se mencionan las siguientes:

- La simplicidad y flexibilidad del mecanismo de integración de nuevos participantes.
- El desacoplamiento entre las actividades de los participantes lo que les confiere un alto grado de independencia y autonomía.

- La mayor facilidad con la que se demuestra que el sistema cumple con sus requerimientos cuando las interacciones se describen utilizando un lenguaje definido con base en reglas.
- La organización de los tipos de sensores y actuadores en tópicos permite simplificar el diseño, especializando el tipo de interacción descrito en las reglas.

Sin embargo, además de las ventajas presentadas anteriormente el modelo SCM-IoT presenta las siguientes desventajas:

- Es necesario que las reglas de interacción de los participantes con el medio estén bien definidas de manera que sean mutuamente excluyentes en los casos en que se aplican, ya que de otro modo se observan distintos resultados bajo las mismas condiciones.
- La aplicación de las reglas conducen al medio a un estado estable en donde no se observen cambios posteriores, ya que de otra manera causan la aplicación sin terminación de al menos una regla.
- El orden de aplicación de las reglas es irrelevante, es decir, de que el orden de aplicación de las reglas no afecta el resultado producido sobre el medio porque de otro modo se tiene dependencias en su orden de ejecución que da lugar a resultados no deseados.

Aunque el modelo SCM-IoT es similar al modelo de bases de datos activas existen diferencias importantes que se describen a continuación en forma de ventajas:

- El tipo de eventos de la mayoría de las bases de datos disponibles se enfocan a eventos que ocurren a nivel de administración de la base de datos y no a nivel de la aplicación. En consecuencia, el programador construye una capa de abstracción que contiene los conceptos de una determinada aplicación sobre la capa provista de las bases de datos.
- La gran mayoría de las tablas que se utilizan en las aplicaciones de la domótica son pequeñas por lo que es muy notoria la desproporción con el gran tamaño, complejidad y costo de los administradores de bases de datos que se requiere para administrar dichas tablas.
- La tecnología utilizada se construye sobre el estándar XML de la W3C.

- Las herramientas disponibles como XPath y XSLT proporcionan una enorme flexibilidad para modelar datos complejos que son difíciles de definir en el modelo estándar de las bases de datos relacionales.

Además, una desventaja del modelo SCM-IoT con respecto al modelo de bases de datos activas es que la implementación de las bases de datos requiere un alto grado de ineficiencia porque se realiza la sustitución del documento XML entero cuando se ha actualizado solo un campo de una nota.

Finalmente, se concluye que se logró cumplir con el objetivo general y comprobar la hipótesis de esta tesis doctoral al desarrollar un lenguaje de composición de servicios bajo el enfoque del IoT que permite realizar la integración y coordinación de servicios ofrecidos e invocados en tiempo real por dispositivos inteligentes utilizando la comunicación M2M y P2M a través del envío y recepción de datos o eventos. Así mismo, se logró cumplir con los objetivos específicos, ya que se analizaron plataformas de servicios del IoT lo que permitió identificar las mejores ventajas y características, aspectos implementados en PISIoT y HEMS-IoT. También se logró identificar y analizar el proceso de comunicación de diversos dispositivos *wearables* y sensores para seleccionar los más adecuados de acuerdo a la particularidad de cada caso de estudio. Adicionalmente, se diseñó y determinó la orquestación y coreografía de servicios con base en la expresividad de los lenguajes composicionales BPEL y WSCDL, pero proponiendo tres extensiones de selección, actualización y notificación. Por otra parte, se definieron y desarrollaron tres casos de estudios: uno en el dominio del cuidado de la salud, otro en la domótica y uno más enfocado en la integración y coordinación de servicios en el contexto del IoT. Además, por medio de las CPN se realizó el modelado de los casos estudio y se logró validar la funcionalidad del mecanismo de composición de servicios bajo el enfoque del IoT.

5.2 Trabajo a futuro

Como parte del trabajo a futuro, se tiene considerado realizar la monitorización de otras enfermedades degenerativas crónicas y condiciones asociadas con el sobrepeso y la obesidad, como presión arterial alta, diabetes, enfermedades cardiovasculares y cáncer (del endometrio, senos, ovarios, próstata, hígado, vesícula biliar, riñones y colon). Del mismo modo, se pretende utilizar otros dispositivos portátiles e inteligentes para recopilar diversas

variables biomédicas relacionadas con enfermedades crónico degenerativas. Además, se tiene considerado utilizar otros algoritmos de aprendizaje automático para evaluar e identificar los que presenten el mejor funcionamiento.

Por otra parte, se pretende implementar HEMS-IoT a mayor escala y entre hogares inteligentes utilizando una mayor cantidad de dispositivos domóticos. Además, también se considera incorporar funcionalidades basadas en la ubicación en HEMS-IoT confiando en el GPS de los dispositivos móviles. En este sentido, se pretende que HEMS-IoT sea capaz de estimar la hora de llegada de los residentes a su hogar para aumentar la comodidad realizando acciones como reproducir música o encender el aire acondicionado con anticipación según el estado de ánimo del usuario. Finalmente, se tiene considerado incluir paneles solares para minimizar el consumo de energía eléctrica e implementar estrategias de seguridad más avanzadas, como *blockchain* y ciberseguridad.

Además, se pretende incrementar la expresividad del lenguaje de composición de servicios en el IoT, ya que el modelo SCM-IoT fue diseñado para integrar elementos de otros mecanimos o especificaciones, o incorporar nuevos elementos que permitan tener un lenguaje con mayor expresividad. Por consiguiente, se tiene considerado implementar en el modelo SCM-IoT las reglas ECA o Evento-Condición-Acción, las cuales son utilizadas en entornos dinámicos y están diseñadas para los sistemas que necesitan una respuesta automática a determinadas condiciones o eventos. Así mismo, por la capacidad de integración y aplicación al medio del modelo SCM-IoT, se tiene considerado extrapolarlo a otros dominios de aplicación como el industrial y el de transportación y logística, ya que en estos dominios se presentan igualmente problemáticas de composición de servicios aunque con otras características particulares que se tienen que considerar. Finalmente, se pretende incorporar nuevos dispositivos inteligentes que cuenten con las nuevas características técnicas que se van incrementando de acuerdo a los nuevos retos y desafíos por atender en el desarrollo de soluciones en el contexto del IoT.

Anexos

A continuación se presentan un anexo que complementa el mecanismo de composición de servicios bajo el enfoque del IoT.

Anexo A. Modelo SCM-IoT

```
<?xml version="1.0" encoding="utf-8"?>

<iota:rules
 name="Domotica"
 xmlns:iota="http://iota.org/iota"
 xmlns:bpel="http://docs.oasis-open.org/wsbpel/2.0">

 <!-- Libros, Libros de Notas y Paginas -->

 <iota:notebooks collection="casa">
  <iota:notebook name="habitaciones" type="Habitaciones">
   <iota:page name="habitacionAntes" path="habitaciones/habitacion" index="@id" type="Habitacion"/>
   <iota:page name="habitacion" path="habitaciones/habitacion" index="@id" type="Habitacion"/>
  </iota:notebook>
  <iota:notebook name="sensores" type="Sensores">
   <iota:page name="sensorPresencia" path="sensores/sensor" index="@id" type="Sensor"/>
   <iota:page name="sensorTemperatura" path="sensores/sensor" index="@id" type="Sensor"/>
   <iota:page name="sensorHumo" path="sensores/sensor" index="@id" type="Sensor"/>
   <iota:page name="sensorGasToxico" path="sensores/sensor" index="@id" type="Sensor"/>
  </iota:notebook>
  <iota:notebook name="actuadores" type="Actuadores">
   <iota:page name="actuador" path="actuadores/actuador" index="@id" type="Actuador"/>
  </iota:notebook>
  <iota:book name="ubicaciones" type="UbicacionesDispositivos">
   <iota:page name="ubicacion" path="ubicaciones/ubicacion" index="@id" type="UbicacionDispositivo"/>
  </iota:book>
 </iota:notebooks>

 <!-- Variables globales -->

 <bpel:variables>
  <bpel:variable name="Incendio" type="string">NO</bpel:variable>
 </bpel:variables>

 <!-- Sensores -->

 <iota:rule name="DetectaPresencia">
  <iota:on>
   <iota:requested operation="detectaPresencia" partnerLink="SensoresPL" portType="SensoresPT"
    variable="sensorPresencia"/>
  </iota:on>
  <iota:do>
   <bpel:sequence>
    <bpel:assign>
     <bpel:from>$ubicaciones[@dispositivo=$sensorPresencia/@id]/@habitacion</bpel:from>
     <bpel:to>sensorPresencia/@id</bpel:to>
    </bpel:assign>
    <iota:update notebook="habitaciones">
     <iota:with note="sensorPresencia"/>
     <iota:former page="habitacionAntes"/>
     <iota:latter page="habitacion"/>
```

```
        <iota:when>
          $sensorPresencia/lectura != $habitacionAntes/presencia
        </iota:when>
      </iota:update>
    </bpel:sequence>
  </iota:do>
</iota:rule>

<iota:rule name="DetectaTemperatura">
  <iota:on>
    <iota:requested operation="detectaTemperatura" partnerLink="SensoresPL" portType="SensoresPT"
      variable="sensorTemperatura"/>
  </iota:on>
  <iota:do>
    <bpel:sequence>
      <bpel:assign>
        <bpel:from>$ubicaciones[@dispositivo=$sensorTemperatura/@id]/@habitacion</bpel:from>
        <bpel:to>$sensorTemperatura/@id</bpel:to>
      </bpel:assign>
      <bpel:if>
        <bpel:condition>
          number($sensorTemperatura/lectura) &lt; number($TEMPERATURA-NORMAL)
        </bpel:condition>
        <bpel:assign>
          <bpel:from>
            <bpel:literal>BAJA</bpel:literal>
          </bpel:from>
          <bpel:to>$sensorTemperatura/lectura</bpel:to>
        </bpel:assign>
        <elseif>
          <bpel:condition>
            number($sensorTemperatura/lectura) &lt; number($TEMPERATURA-ALTA)
          </bpel:condition>
          <bpel:assign>
            <bpel:from>
              <bpel:literal>NORMAL</bpel:literal>
            </bpel:from>
            <bpel:to>$sensorTemperatura/lectura</bpel:to>
          </bpel:assign>
        </elseif>
        <elseif>
          <bpel:condition>
            number($sensorTemperatura/lectura) &lt; number($TEMPERATURA-MUYALTA)
          </bpel:condition>
          <bpel:assign>
            <bpel:from>
              <bpel:literal>ALTA</bpel:literal>
            </bpel:from>
            <bpel:to>$sensorTemperatura/lectura</bpel:to>
          </bpel:assign>
        </elseif>
        <bpel:else>
          <bpel:assign>
            <bpel:from>
              <bpel:literal>MUYALTA</bpel:literal>
            </bpel:from>
            <bpel:to>$sensorTemperatura/lectura</bpel:to>
          </bpel:assign>
        </bpel:else>
      </bpel:if>
      <iota:update notebook="habitaciones">
        <iota:with note="sensorTemperatura"/>
        <iota:former page="habitacionAntes"/>
        <iota:latter page="habitacion"/>
        <iota:when>
          $sensorTemperatura/lectura != $habitacionAntes/Temperatura
        </iota:when>
```

```
      </iota:update>
    </bpel:sequence>
  </iota:do>
</iota:rule>

<iota:rule name="DetectaHumo">
  <iota:on>
    <iota:requested operation="detectaHumo" partnerLink="SensoresPL" portType="SensoresPT"
      variable="sensorHumo"/>
  </iota:on>
  <iota:do>
    <bpel:sequence>
      <bpel:assign>
        <bpel:from>$ubicaciones[@dispositivo=$sensorHumo/@id]/@habitacion</bpel:from>
        <bpel:to>sensorHumo/@id</bpel:to>
      </bpel:assign>
      <iota:update notebook="habitaciones">
        <iota:with note="sensorHumo"/>
        <iota:former page="habitacionAntes"/>
        <iota:latter page="habitacion"/>
        <iota:when>
          $sensorHumo/lectura != $habitacionAntes/humo
        </iota:when>
      </iota:update>
    </bpel:sequence>
  </iota:do>
</iota:rule>

<iota:rule name="DetectaGasToxico">
  <iota:on>
    <iota:requested operation="detectaGasToxico" partnerLink="SensoresPL" portType="SensoresPT"
      variable="sensorGasToxico"/>
  </iota:on>
  <iota:do>
    <bpel:sequence>
      <bpel:assign>
        <bpel:from>$ubicaciones[@dispositivo=$sensorGasToxico/@id]/@habitacion</bpel:from>
        <bpel:to>$sensorGasToxico/@id</bpel:to>
      </bpel:assign>
      <iota:update notebook="habitaciones">
        <iota:with note="sensorGasToxico"/>
        <iota:former page="habitacionAntes"/>
        <iota:latter page="habitacion"/>
        <iota:when>
          $sensorGasToxico/lectura != $habitacionAntes/gastoxico
        </iota:when>
      </iota:update>
    </bpel:sequence>
  </iota:do>
</iota:rule>

<!-- Actuadores -->

<iota:rule name="ActivaCalefactor">
  <iota:on>
    <iota:updated notebook="habitaciones">
      <iota:latter page="habitacion"/>
    </iota:updated>
  </iota:on>
  <iota:if>
    $habitacion/Temperatura = 'BAJA' and
    $habitacion/presencia = 'SI' and
    $habitacion/calefactor = 'NO'
  </iota:if>
  <iota:do>
    <bpel:sequence>
      <bpel:asssign>
```

```
        <bpel:from>
          <bpel:literal>SI</bpel:literal>
        </bpel:from>
        <bpel:to>$habitacion/calefactor</bpel:to>
        <bpel:to>$calefactor/estado</bpel:to>
      </bpel:asssign>
      <bpel:assign>
        <bpel:from>$ubicaciones[@habitacion=$habitacion/@id]/@dispositivo</bpel:from>
        <bpel:to>$calefactor/@id</bpel:to>
      </bpel:assign>
      <bpel:invoke operation="NotificaCalefactor" partnerLink="ActuadoresPL" portType="ActuadoresPT"
        variable="calefactor"/>
    </bpel:sequence>
  </iota:do>
</iota:rule>

<iota:rule name="InactivaCalefactor">
  <iota:on>
    <iota:updated notebook="habitaciones">
      <iota:latter page="habitacion"/>
    </iota:updated>
  </iota:on>
  <iota:if>
    $habitacion/Temperatura = 'NORMAL' and
    $habitacion/presencia = 'SI' and
    $habitacion/calefactor = 'SI'
    or
    $habitacion/presencia = 'NO' and
    $habitacion/calefactor = 'SI'
    or
    $habitacion/Alarma = 'SI' and
    $habitacion/calefactor = 'SI'
  </iota:if>
  <iota:do>
    <bpel:sequence>
      <bpel:asssign>
        <bpel:from>
          <bpel:literal>NO</bpel:literal>
        </bpel:from>
        <bpel:to>$habitacion/calefactor</bpel:to>
        <bpel:to>$calefactor/estado</bpel:to>
      </bpel:asssign>
      <bpel:assign>
        <bpel:from>$ubicaciones[@habitacion=$habitacion/@id]/@dispositivo</bpel:from>
        <bpel:to>$calefactor/@id</bpel:to>
      </bpel:assign>
      <bpel:invoke partnerLink="ActuadoresPL" portType="ActuadoresPT" operation="NotificaCalefactor"
        variable="calefactor"/>
    </bpel:sequence>
  </iota:do>
</iota:rule>

<iota:rule name="ActivaVentilador">
  <iota:on>
    <iota:updated notebook="habitaciones">
      <iota:latter page="habitacion"/>
    </iota:updated>
  </iota:on>
  <iota:if>
    $habitacion/Temperatura = 'ALTA' and
    $habitacion/presencia = 'SI' and
    $habitacion/ventilador = 'NO'
  </iota:if>
  <iota:do>
    <bpel:sequence>
      <bpel:asssign>
        <bpel:from>
```

```
        <bpel:literal>SI</bpel:literal>
       </bpel:from>
       <bpel:to>$habitacion/ventilador</bpel:to>
      </bpel:asssign>
      <bpel:assign>
       <bpel:from>$ubicaciones[@habitacion=$habitacion/@id]/@dispositivo</bpel:from>
       <bpel:to>$ventilador/@id</bpel:to>
      </bpel:assign>
      <bpel:invoke operation="NotificaVentilador" partnerLink="ActuadoresPL" portType="ActuadoresPT"
       variable="ventilador"/>
    </bpel:sequence>
  </iota:do>
</iota:rule>

<iota:rule name="InactivaVentilador">
  <iota:on>
    <iota:updated notebook="habitaciones">
      <iota:latter page="habitacion"/>
    </iota:updated>
  </iota:on>
  <iota:if>
    $habitacion/Temperatura = 'NORMAL' and
    $habitacion/presencia = 'SI' and
    $habitacion/ventilador = 'SI'
    or
    $habitacion/presencia = 'NO' and
    $habitacion/ventilador = 'SI'
    or
    $habitacion/Alarma = 'SI' and
    $habitacion/ventilador = 'SI'
  </iota:if>
  <iota:do>
    <bpel:sequence>
      <bpel:asssign>
        <bpel:from>
          <bpel:literal>SI</bpel:literal>
        </bpel:from>
        <bpel:to>$habitacion/ventilador</bpel:to>
      </bpel:asssign>
      <bpel:assign>
        <bpel:from>$ubicaciones[@habitacion=$habitacion/@id]/@dispositivo</bpel:from>
        <bpel:to>$ventilador/@id</bpel:to>
      </bpel:assign>
      <bpel:invoke operation="NotificaVentilador" partnerLink="ActuadoresPL" portType="ActuadoresPT"
        variable="ventilador"/>
    </bpel:sequence>
  </iota:do>
</iota:rule>

<iota:rule name="NotificaIncendio">
  <iota:on>
    <iota:updated notebook="habitaciones">
      <iota:latter page="habitacion"/>
    </iota:updated>
  </iota:on>
  <iota:if>
    $Incendio = 'NO' and
    $habitacion/Temperatura = 'MUYALTA' and
    $habitacion/humo = 'SI' and
    $habitacion/gastoxico = 'SI'
  </iota:if>
  <iota:do>
    <bpel:sequence>
      <bpel:asssign>
        <bpel:from>
          <bpel:literal>SI</bpel:literal>
        </bpel:from>
```

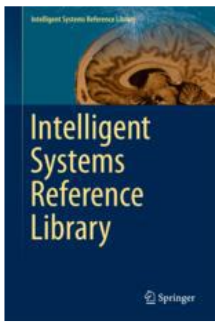
```
        <bpel:to>$Incendio</bpel:to>
      </bpel:asssign>
      <bpel:assign>
        <bpel:from>$ubicaciones[@habitacion=$habitacion/@id]/@dispositivo</bpel:from>
        <bpel:to>$ubicacion/@id</bpel:to>
      </bpel:assign>
      <bpel:invoke operation="NotificaBomberos" partnerLink="EmergenciaPL" portType="EmergenciaPT"
       variable="ubicacion"/>
    </bpel:sequence>
  </iota:do>
 </iota:rule>

 <iota:rule name="BuscaResidentes">
  <bpel:variables>
    <bpel:variable name="habita" type="Habitacion"/>
  </bpel:variables>
  <iota:on>
    <iota:each page="habita">
      <iota:from notebook="habitaciones"/>
      <iota:when>
        $habita/presencia = 'SI' and
        $habita/Alarma = 'NO'
      </iota:when>
    </iota:each>
  </iota:on>
  <iota:if>
   $Incendio = 'SI'
  </iota:if>
  <iota:do>
    <bpel:sequence>
      <bpel:asssign>
        <bpel:from>
          <bpel:literal>SI</bpel:literal>
        </bpel:from>
        <bpel:to>$habita/Alarma</bpel:to>
      </bpel:asssign>
      <bpel:invoke operation="NotificaRescate" partnerLink="EmergenciaPL" portType="EmergenciaPT"
       variable="habita"/>
    </bpel:sequence>
  </iota:do>
 </iota:rule>

</iota:rules>
```

Productos académicos

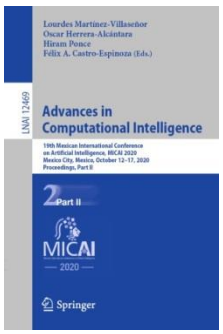
Revistas indizadas - Journal Citation Reports

applied sciences	Como primer autor Machorro-Cano, I.; Alor-Hernández, G.; Paredes-Valverde, M.A.; Ramos-Deonati, U.; Sánchez-Cervantes, J.L.; Rodríguez-Mazahua, L. PISIoT: A Machine Learning and IoT-Based Smart Health Platform for Overweight and Obesity Control. Appl. Sci. 2019, 9, 3037. Doi: https://doi.org/10.3390/app9153037. IF: 2.217. Estatus: Publicado
energies	Como primer autor Machorro-Cano, I.; Alor-Hernández, G.; Paredes-Valverde, M.A.; Rodríguez-Mazahua, L..; Sánchez-Cervantes, J.L.; Olmedo-Aguirre, J.O.. HEMS-IoT: A Big Data and Machine Learning-Based Smart Home System for Energy Saving. Energies 2020, (ISSN 1996-1073), Doi: https://doi.org/10.3390/en13051097 IF: 2.717. Estatus: Publicado

Capítulos de libro

	Como primer autor Machorro-Cano I., Alor-Hernández G., Cruz-Ramos N.A., Sánchez-Ramírez C., Segura-Ozuna M.G. (2018) A Brief Review of IoT Platforms and Applications in Industry. In: García-Alcaraz J., Alor-Hernández G., Maldonado-Macías A., Sánchez-Ramírez C. (eds) New Perspectives on Applied Industrial Tools and Techniques. Management and Industrial Engineering. Springer, Cham. Chapter 16. pp 293-324. Print ISBN 978-3-319-56870-6, Online ISBN 978-3-319-56871-3. DOI: https://doi.org/10.1007/978-3-319-56871-3_15 Estatus: Publicado.
	Como primer autor Machorro-Cano I., Alor-Hernández G., Olmedo-Aguirre J.O., Rodríguez-Mazahua L., Segura-Ozuna M.G. (2020) IoT Services Orchestration and Choreography in the Healthcare Domain. In: García-Alcaraz J., Sánchez-Ramírez C., Avelar-Sosa L., Alor-Hernández G. (eds) Techniques, Tools and Methodologies Applied to Global Supply Chain Ecosystems. Intelligent Systems Reference Library, vol 166. Springer, Cham. Chapter 19, pp 293-324. Print ISBN 978-3-030-26487-1, Online ISBN 978-3-030-26488-8. DOI: https://doi.org/10.1007/978-3-030-26488-8_19 Estatus: Publicado.

Revistas en otros índices

	Como primer autor Machorro-Cano I., Ramos-Deonati U., Alor-Hernández G., Sánchez-Cervantes J.L, Sánchez-Ramírez C., Rodríguez-Mazahua L. and Segura-Ozuna M.G. (2017) An IoT-Based Architecture to Develop a Healthcare Smart Platform. In: Valencia-García R., Lagos-Ortiz K., Alcaraz-Mármol G., Del Cioppo J., Vera-Lucio N., Bucaram-Leverone M. (eds) Technologies and Innovation. CITI 2017. Communications in Computer and Information Science, vol 749. Springer, Cham. pp 133-145. Print ISBN 978-3-319-67282-3, Online ISBN 978-3-319-67283-0. DOI: https://doi.org/10.1007/978-3-319-67283-0_10 Estatus: Publicado.
	Como primer autor Machorro-Cano Isaac, Alor-Hernández Giner, Olmedo-Aguirre José Oscar, Rodríguez-Mazahua Lisbeth and Segura-Ozuna Mónica Guadalupe. Design of a Language for IoT Service Composition. 4th International Workshop on Intelligent Decision Support Systems for Industry (WIDSSI 2018). Research in Computer Science 148(4), 2019, pp. 39-46, ISSN 1870-4069. Estatus: Publicado.
	Como primer autor Machorro-Cano Isaac, Paredes-Valverde Mario Andrés, Alor-Hernandez Giner, Salas-Zárate1 María del Pilar, Segura-Ozuna Mónica Guadalupe, and Sánchez-Cervantes José Luis. PESSHIoT: Smart Platform for Monitoring and Controlling Smart Home Devices and Sensors. R. Valencia-García et al. (Eds.): CITI 2019, CCIS 1124, pp. 1–14, 2019. Doi: https://doi.org/10.1007/978-3-030-34989-9_11 Estatus: Publicado.
	En colaboración Reyes-Campos J., Alor-Hernández G., Machorro-Cano I., Sánchez-Cervantes J.L., Muñoz-Contreras H., Olmedo-Aguirre J.O. (2020) Energy Saving by Using Internet of Things Paradigm and Machine Learning. In: Martínez-Villaseñor L., Herrera-Alcántara O., Ponce H., Castro-Espinoza F.A. (eds) Advances in Computational Intelligence. MICAI 2020. Lecture Notes in Computer Science, vol 12469. Springer, Cham. Doi: https://doi.org/10.1007/978-3-030-60887-3_38 Estatus: Publicado.
	En colaboración Reyes-Campos J., Alor-Hernández G., Machorro-Cano I., Sánchez-Cervantes J.L., Muñoz-Contreras H., Olmedo-Aguirre J.O. (2020) IntelihOgarT: A smart platform to contribute comfort in Intelligent Home Environments by using Internet of Things paradigm and Machine Learning (EN). Communications in Computer and Information Science (2020), Springer. Doi: https://doi.org/10.1007/978-3-030-62015-8_11 Estatus: Publicado.

Artículos en congreso

COMIA 2018	Artículo en colaboración Uriel Ramos Deonati, Giner Alor Hernandez, Isaac Machorro Cano, José Luis Sánchez Cervantes y Lisbeth Rodríguez Mazahua (2018). Diseño de una aplicación Web para el seguimiento y Control del Sobrepeso u Obesidad a través del paradigma del IoT. Congreso Mexicano de Inteligencia Artificial COMIA 2018. Research in Computer Science, 147(8), 2018, pp. 25-38, ISSN 1870-4069. Estatus: Publicado.

Proyectos de investigación

TECNM TECNOLÓGICO NACIONAL DE MÉXICO	Participación como colaborador Desarrollo de una plataforma inteligente para la prevención, monitorización y tratamiento del sobrepeso u obesidad bajo el enfoque del Internet de las Cosas (IoT). Convocatoria 2018 de apoyo a la investigación científica y tecnológica de los programas educativos de los institutos tecnológicos - TecNM. Estatus: Finalizado

Registros de derechos de autor

INDAUTOR Instituto Nacional del Derecho de Autor	Participación como colaborador Uriel Ramos Deonati, Giner Alor Hernandez, Isaac Machorro Cano, José Luis Sánchez Cervantes. PISIoT – Plataforma Inteligente para la Prevención, Monitorización y Tratamiento del Sobrepeso u Obesidad bajo el enfoque del Internet de las Cosas (IoT). Número de registro: 03-2018-121711430100-01. Estatus: Registrado.
INDAUTOR Instituto Nacional del Derecho de Autor	Participación como colaborador Jesús Miguel Echevarría Díaz, José Luis Sánchez Cervantes, Luis Omar Colombo Mendoza, Ignacio López Martínez, Isaac Machorro Cano. SENTINEL: Sistema de Tele-cuidado de Adultos Mayores basado en Analítica Big data para el Internet de las Cosas. Estatus: Registro en proceso

	Participación como colaborador Josimar Reyes Campos, Giner Alor Hernández, Isaac Machorro Cano, José Luis Sánchez Cervantes y Lisbeth Rodríguez Mazahua. Módulo de Control Domótico Automático para el incremento y personalización del confort en una casa inteligente a través del paradigma del internet de las cosas. Estatus: Registro en proceso
	Participación como colaborador Isaac Machorro Cano, Giner Alor Hernández, José Luis Sánchez Cervantes, Lisbeth Rodríguez Mazahua y José Oscar Olmedo Aguirre. SCM-IOT: Mecanismo de Composición de Servicios en el contexto del Internet de las Cosas. Estatus: Registro en proceso

Convenios de colaboración

	Convenio ITO – UNPA Se realizó un convenio de colaboración con la Universidad del Papaloapan (UNPA), en donde se trabajó con el personal del área de la salud que labora dentro de la institución educativa y en algunos casos también laboran en instituciones públicas de salud de primer y segundo nivel. Número de convenio: GTV/CO-E/017/04-18. Estatus: Finalizado
	Convenio ITO – CSAC Se cuenta con un convenio de colaboración con el Centro de Salud Arroyo Choapam (Institución pública de salud de primer nivel), en donde se está trabajando con el personal del área de la salud que labora dentro de la institución. Número de convenio: GTV/CO-E/018/06-19. Estatus: Finalizado

Referencias

Abdalla, A.M.M.; Dress, S. y Zaki, N. (2011). Detection of Masses in Digital Mammogram Using Second Order Statistics and Artificial Neural Network. *Int. J. Comput. Sci. Inf. Technol*. 3, 176–186.

Afzaal H y Zafar N. A. (2017). Modeling of IoT-based border protection system. *First International Conference on Latest trends in Electrical Engineering and Computing Technologies (INTELLECT)*. 1-6. doi: 10.1109/INTELLECT.2017.8277639

Aggarwal CC, Ashish N and Sheth A (2013) The Internet of Things. A Survey from the Data-Centric Perspective. *Managing and Mining Sensor Data* 383-428.

Alirezaie, M.; Renoux, J.; Köckemann, U.; Kristoffersson, A.; Karlsson, L.; Blomqvist, E.; Tsiftes, N.; Voigt, T. y Loutfi, A. (2017) An Ontology-based Context-aware System for Smart Homes E-care@home. *Sensors*, 17, 1586.

AllJoyn. Recuperado el 27 de Marzo de 2017, de https://allseenalliance.org/

Alur, R. (2003). Timed Automata. International Conference on Computer Aided Verification, Springer, 8 – 22. doi: https://doi.org/10.1007/3-540-48683-6_3

Arrayent. Recuperado el 18 de Junio de 2017, de https://www.arrayent.com/

Arkessa. Available online: Recuperado el 22 de Julio de 2017, de http://www.arkessa.com/

Arm Mbed. Recuperado el 22 de Julio de 2017, de https://www.mbed.com/en/

Atzori, L., Iera, A. y Morabito, G. (2010). The Internet of Things: A survey. *Computer Networks*. Vol. 54, Issue 15, 2787-2805. doi: https://doi.org/10.1016/j.comnet.2010.05.010

Atzori, L., Iera, A., Morabito, G. y Nitti, M. (2012). The Social Internet of Things (SIoT) - When Social Networks meet the Internet of Things: Concept, Architecture and Network Characterization. *Computer Networks*. 56, 3594-3608.

ARIS. ARIS COMMUNITY. Recuperado el 22 de Diciembre de 2018, de https://www.ariscommunity.com/users/josephe-blondaut/2017-04-19-welcome-aris-10-welcome-future

Axeda. Recuperado el 28 de Marzo de 2017, de http://www.ptc.com/

Ayla Networks. Recuperado el 28 de Marzo de 2017, de https://www.aylanetworks.com/

Baeten, J.C.M. (2005). Abrief history of process algebra. *Theoretical Computer Science*, 131 – 146. doi: 10.1016/j.tcs.2004.07.036

Baker T., Asim M., Tawfik H., Aldawsari B. y Buyya R. (2017). An energy-aware service composition algorithm for multiple cloud-based IoT applications. *Journal of Network and Computer Applications*, 96–108. doi: http://dx.doi.org/10.1016/j.jnca.2017.03.008

Bandyopadhyay, D. y Sen, J. (2011). Internet of Things - Applications and Challenges in Technology and Standardization. *Wireless Personal Communications*. Vol. 58, 49–69. doi: https://doi.org/10.1007/s11277-011-0288-5

Belkeziz, R. y Jarir, Z. (2017). IoT coordination: designing a context-driven architecture. In: *International Conference on Signal-Image Technology and Internet-Based Systems (SITIS)*, pp. 388–395. doi: https://doi.org/10.1109/sitis.2017.70

Bergesio L., Bernardos A.M. y Casar J.R. (2017). An Object-Oriented Model for Object Orchestration in Smart Environments. *Procedia Computer Science*, 440–447. doi: https://doi.org/10.1016/j.procs.2017.05.415

Blanc, S., Bayo-Monton, J.L., Campelo, J.C. y Fernandez-Llatas, C. (2016). Process choreography in wireless sensor and actuator networks: a proposal to achieve Network Virtualization. *Int. J. Actor-Netw. Theor. Tech. Innov*. 1–11.

Boa, F. y Chen, I. R. (2012). Trust Management for the Internet of Things and Its Application to Service Composition. *World of Wireless, Mobile and Multimedia Networks (WoWMoM), IEEE International*. 1-6.

BoschSI. Recuperado el 02 de Agosto de 2017, de https://www.bosch-si.com

Botta, A., De Donato, W., Persico, V. y Pescapé, A. (2015). Integration of Cloud Computing and Internet of Things: a Survey. *Journal of Future Generation Computer Systems*. Vol. 56, 684-700. doi: https://doi.org/10.1016/j.future.2015.09.021

BUGswarm. Recuperado el 28 de Marzo de 2017, de http://developer.bugswarm.net/

Cacciagrano, D.R y Culmone, R. (2020) IRON: Reliable domain specific language for programming IoT devices. *Internet of Things*, 1-10. doi: https://doi.org/10.1016/j.iot.2018.09.006

Cano, J., Rutten, E., Delaval, G., Benazzouz, Y. y Gurgen, L. (2014). ECA rules for IoT environment: a case study in safe design. In: *IEEE Eighth International Conference on Self-Adaptive and Self-Organizing Systems Workshops*, 116–121. London. doi: https://doi.org/10.1109/sasow.2014.32

Carriots. Recuperado el 27 de Marzo de 2017, de https://www.carriots.com/

Cassar, G., Barbaghi, P., Wang, W., De, S. y Moessner K. (2013). Composition of Services in Pervasive Environments: A Divide and Conquer Approach. *2013 IEEE Symposium on Computers and Communications (ISCC)*. 226 – 232.

Chakray. Programming Languages: Types and Features. Recuperado el 10 de Julio de 2020, de https://www.chakray.com/programming-languages-types-and-features/

Chen, I.R., Guo, J. y Bao, F. (2015). Trust Management for Service Composition in SOA-based IoT Systems. *IEEE Wireless Communications and Networking Conference (WCNC)*. 3444-3449.

Chen I.R., Guo J. y Bao F. (2016) Trust Management for SOA-Based IoT and Its Application to Service Composition. *IEEE Transactions on Services Computing*, 482 – 495. doi: 10.1109/TSC.2014.2365797

Chen, M., Ma, Y., Song, J., Lai, C.F. y Hu, B. (2016). Smart clothing: connecting human with clouds and big data for sustainable health monitoring. *Mob. Netw. Appl*. 21, 825–845. doi: https://doi.org/10.1007/s11036-016-0745-1

Chen, L. y Englund, C. (2017). Choreographing services for smart cities: smart traffic demonstration. *IEEE 85th Vehicular Technology Conference (VTC Spring)*. 1-5. doi: 10.1109/VTCSpring.2017.8108625

Chen, X.; Ching, W.-K.; Aoki-Kinoshita, K.F. y Furuta, K. (2010). Support Vector Machine Methods for the Prediction of Cancer Growth. *In Proceedings of the 2010 Third International Joint Conference on Computational Science and Optimization*, Huangshan, China, 28–31 May; pp. 229–232, doi:10.1109/CSO.2010.70.

Cherrier, S., Ghamri-Doudane, Y., Lohier, S. y Roussel, G. (2016). D-LITe: building Internet of Things choreographies. *Softw. Eng.* 2, 235, 1 - 35.

Comptona, M., Barnaghi, P.; Bermudez, L.; García-Castro, R.; Corcho, O.; Cox, S., Graybeal, J.; Hauswirth, M.; Henson, C.; Herzog, A. y et al. (2012) The SSN ontology of the W3C semantic sensor network incubator group. *J. Web Semant,* 17, 25–32.

Considine, J.; Li, F.; Kollios, G. y Byers, J. (2004) Approximate aggregation techniques for sensor databases. *In Proceedings of the 20th International Conference on Data Engineering*, Boston, MA, USA, 30 March–2 April; Volume 20, pp. 449–460.

Contiki. Recuperado el 28 de Marzo de 2017, de http://www.contiki-os.org/

CPN Tools. A tool for editing, simulating, and analyzing Colored Petri nets. Recuperado el 30 de Marzo de 2020, de http://cpntools.org/

Cubo, J., Nieto, A. y Pimentel E. (2014). A Cloud-Based Internet of Things Platform for Ambient Assisted Living. *Sensors*. 14, 14070-14105.

Data-flair. 4 Major IoT Protocols – MQTT, CoAP, AMQP, DDS. Recuperado el 02 de Diciembre de 2018, de https://data-flair.training/blogs/iot-protocols/

Dar, K., Taherkordi, A., Baraki, H., Eliassen, F. y Geihs, K. (2015). A resource oriented integration architecture for the Internet of Things: A business process perspective. *Pervasive and Mobile Computing*. 20, 145-159.

Dar, K., Taherkordi, A., Rouvoy, R. y Eliassen, F. (2011). Adaptable Service Composition for Very-Large-Scale Internet of Things Systems. *Pervasive and Mobile Computing*, 40, 145-159.

Data Flair. IoT Contiki OS – Top 5 Communication Components in Contiki. Recuperado el 15 de Diciembre de 2018, de https://data-flair.training/blogs/iot-contiki-os/

Daud, N.; Noor, N.L.M.; Aljunid, S.A.; Noordin, N. y Teng, N.I.M.F. (2018). Predictive Analytics: The Application of J48 Algorithm on Grocery Data to Predict Obesity. *In Proceedings of the 2018 IEEE Conference on Big Data and Analytics (ICBDA)*, Langkawi Island, Malaysia, 21–22 November; pp. 1–6, doi:10.1109/ICBDAA.2018.8629623.

Dechsupa, C., Vatanawood, W. y Thongtak, A. (2016). Formal Verification of Web Service Orchestration Using Colored Petri Net. *Proceedings of the International MultiConference of Engineers and Computer Scientists (IMECS)*, Vol. 1, 1-6.

DeviceHive. Recuperado el 20 de Marzo de 2017, de http://devicehive.com/

De-La-Hoz-Correa, E.; Mendoza Palechor, F.; De-La-Hoz-Manotas, A.; Morales Ortega, R. y Sánchez Hernández, A.B. (2019) Obesity Level Estimation Software based on Decision Trees. *J. Comput. Sci.* 15, 1–11, doi:10.3844/jcssp.2019.67.77.

Dijkman RM, Sprenkels B, Peeters T y Janssen A (2015). Business models for the Internet of Things. *International Journal of Information Management*, 672–678. doi: https://doi.org/10.1016/j.ijinfomgt.2015.07.008

Dugan, T.M.; Mukhopadhyay, S.; Carroll, A. y Downs, S. (2015). Machine Learning Techniques for Prediction of Early Childhood Obesity. *Appl. Clin. Inform.* 6, 506–520.

Duhart, C., Sauvage, P. y Bertelle, C. (2016). A Resource Oriented Framework for Service Choreography over Wireless Sensor and Actor Networks. *International Journal of Wireless Information Networks*. 173–186. doi: 10.1007/s10776-016-0316-1

D'Hyver, C. y Gutiérrez Robledo, L.M. (2014). Geriatrics, 3rd ed.; Modern Manual; Publisher: City, Mexico; pp. 2–13.

Echelon. Recuperado el 22 de Julio de 2017, de http://www.echelon.com/

Eclipse.org. Papyrus Documentation. Recuperado el 07 de Noviembre 2018, de, https://www.eclipse.org/papyrus/documentation.html

Eclipse IoT. IoT Developer Survey 2020 Results. Recuperado el 03 de Noviembre 2020, de, https://outreach.eclipse.foundation/eclipse-iot-developer-survey-2020

Eterovic T., Kaljic E., Donko D, Salihbegovic A. Y Ribic S.(2015). An Internet of Things Visual Domain Specific Modeling Language based on UML. *Information, Communication and Automation Technologies (ICAT), 2015 XXV International Conference*.1-5. doi: 10.1109/ICAT.2015.7340537

Etherios. Recuperado el 28 de Marzo de 2017, de http://www.etherios.com/

EVRYTHNG. Recuperado el 28 de Marzo de 2017, de https://evrythng.com/

Exosite. Recuperado el 41 de Julio de 2017, de https://exosite.com/

Fageeri, S.O.; Ahmed, S.M.M.; Almubarak, S.A. y Mu'Azu, A.A. (2017). Eye refractive error classification using machine learning techniques. *In Proceedings of the 2017 International Conference on Communication, Control, Computing and Electronics Engineering (ICCCCEE)*, Khartoum, Sudan, 16–18 January; pp. 1–6, doi:10.1109/ICCCCEE.2017.7867660.

Fernández-López, M., Gómez-Pérez, A. y Juristo, N. (1997). Methontology: from ontological art towards ontological engineering. *In: Proceedings of the Ontological Engineering AAAI-97 Spring Symposium Series*, March 1997, pp. 24–26. Stanford University, EEUU.

Fasolo, E.; Rossi, M.; Widmer, J. y Zorzi, M. (2007) In-network aggregation techniques for wireless sensor networks: A survey. *IEEE Wirel. Commun.* 14, 70–87.

Gama, K., Touseau, L. y Donsez, D. (2012). Combining heterogeneous service technologies for building an Internet of Things middleware. *Computer Communications*. 35, 405-417.

Giang, N.K., Blackstock, M., Lea, R. y Leung, V.C.M. (2015). Developing IoT applications in the fog: a distributed dataflow approach. In: *5th International Conference on the Internet of Things (IoT)*, pp. 1–8. doi: https://doi.org/10.1109/iot.2015.7356560

Giang, N.K., Lea, R., Blackstock,M. y Leung, V.C.M. (2016). On building smart city IoT applications: a coordination-based perspective. In: *SmartCities 16 Proceedings of the 2nd International Workshop on Smart*, pp. 1–7. doi: https://doi.org/10.1145/3009912.3009919

Gierej S (2017) The Framework of Business Model in the Context of Industrial Internet of Things. *Procedia Engineering*, 206 – 212. doi: 10.1016/j.proeng.2017.03.166

Glova k, Sabol T and Vajda V (2014) Business Models for the Internet of Things Environment. *Procedia Economics and Finance*, 1122-1129. doi: https://doi.org/10.1016/S2212-5671(14)00566-8

Gluhak A, Krco S, Nati M et al (2014) A Survey on Facilities for Experimental Internet of Things Research. *IEEE Communications Magazine* 49, 58-67.

Goudos, S.K.; Dallas, P.I.; Chatziefthymiou, S. y Kyriazakos, S. (2017). A Survey of IoT Key Enabling and Future Technologies: 5G, Mobile IoT, Sematic Web and Applications. *Wirel. Pers. Commun*, 97, 1645–1675.

GroveStreams. Recuperado el 28 de Marzo de 2017, de https://www.grovestreams.com/

Gubbi, J., Buyya, R., Marusic, S. y Palaniswami, M. (2013). Internet of Things (IoT): A Vision, Architectural Elements, and Future Directions. *Future Generation Computer Systems*. 29, 1645-1660.

Han, J.; Kamber, M. y Pei, J. (2012). Classification: Basic Concepts, Chapter 8, Data Mining: Concepts and Techniques, 3rd ed.; *Elsevier*: Amsterdam, The Netherlands, pp. 327–391, ISBN 978-0-12-381479-1.

Han, S. N., Khan, I., Lee, G. M., Crespi, N. y Glitho, R. H. (2016). Service composition for IP smart object using realtime Web protocols: Concept and research challenges. *Computer Standards & Interfaces*. 43, 79-90. doi. http://dx.doi.org/10.1016/j.csi.2015.08.006

HarvestGeek. Recuperado el 27 de Marzo de 2017, de https://www.kickstarter.com/projects/2077260917/harvestgeek-brains-for-your-garden

Hashemi, A.; Pilevar, A.H.; Rafeh, R.; Moallem, P.; Karimizadeh, A. y Yazdchi, M. (2013). Mass Detection in Lung CT Images Using Region Growing Segmentation and Decision Making Based on Fuzzy Inference System and Artificial Neural Network. *Int. J. Image Graph. Signal Process*. 5, 16–24.

Herbjørn, N. y Breivik, E. (2005) The influence of media on advertising effectiveness a comparison of Internet, posters and radio. *Int. J. Mark. Res*., 47, 383–405.

Hiremath, S.; Yang, G. y Mankodiya, K. (2014). Wearable Internet of Things: Concept, Architectural Components and Promises for Person-Centered Healthcare. *In Proceedings of the 4th International Conference on Wireless Mobile Communication and Healthcare—Transforming Healthcare Through Innovations in Mobile and Wireless Technologies (MOBIHEALTH)*, Athens, Greece, 3–5 November; pp. 304–307.

Hui, T.K.L.; Sherratt, R.S. y Díaz Sánchez, D. (2016) Major requirements for building Smart Homes in Smart Cities based on Internet of Things technologies. *Future Gener. Comput. Syst.* 76, 358–369.

Ian Skerrett, Eclipse - Profile of an IoT Developer: Results of the IoT Developer Survey. Recuperado el 2 de Diciembre de 2018, de https://ianskerrett.wordpress.com/2016/04/14/profile-of-an-iot-developer-results-of-the-iot-developer-survey/

IBM®. IBM Knowledge Center. Recuperado el 22 de Diciembre de 2018, de https://www.ibm.com/support/knowledgecenter/SSB23S_1.1.0.15/gtps6/s6wsadv.html

InfoBright. Recuperado el 27 de Marzo de 2017, de https://infobright.com/

InterDigital. Recuperado el 02 de Agosto de 2017, de http://www.interdigital.com/iot/

Intersog. 12 POPULAR PROGRAMMING LANGUAGES FOR IOT DEVELOPMENT IN 2017. Recuperado el 29 de Noviembre de 2018, de https://iot.intersog.com/blog/12-popular-programming-languages-for-iot-development-in-2017/

Iobridge. Recuperado el 28 de Marzo de 2017, de https://iobridge.com/

IOTA. Recuperado el 02 de Agosto de 2017, de https://iota.org/ (accessed on 07 August 2017)

IoT Agenda. RIOT operating system. Recuperado el 15 de Diciembre de 2018, de https://internetofthingsagenda.techtarget.com/definition/RIOT-operating-system

IoT Agenda. XMPP: IoT protocol winner, or second place to MQTT?. Recuperado el 2 de Diciembre de 2018, de https://internetofthingsagenda.techtarget.com/feature/XMPP-IoT-protocol-winner-or-second-place-to-MQTT

IoT for all. Top 3 Programming Languages for IoT Development In 2018. Recuperado el 29 de Noviembre de 2018, de https://www.iotforall.com/2018-top-3-programming-languages-iot-development/

Isazadeh, A.; Pedrycz, W. y Mahan, F. (2014). ECA rule learning in dynamic environments. *Expert Systems with Applications*, volumen 41, 7847-7857. doi: https://doi.org/10.1016/j.eswa.2014.06.028

JasperIoT. Recuperado el 29 de Julio de 2017, de https://www.jasper.com/

Ju J, Kim MS and Ahn JH (2016). Prototyping Business Models for IoT Service. *Procedia Computer Science*, 882 – 890. doi: https://doi.org/10.1016/j.procs.2016.07.106

Juric, M.B. A Hands-on Introduction to BPEL. Recuperado el 19 de Junio de 2020, de https://www.oracle.com/technical-resources/articles/matjaz-bpel.html

Kaa Project. Recuperado el 20 de Marzo de 2017, de http://www.kaaproject.org/

Knijnenburg, B.P. y Willemsen, M.C. (2015) Evaluating Recommender Systems with User Experiments. *In Recommender Systems Handbook*; Springer: Boston, MA, USA; 309–352.

Knijnenburg, B.P.; Willemsen, M.C.; Gantner, Z.; Soncu, H. y Newell, C. (2012) Explaining the user experience of recommender systems. User Model. *User Adapt. Interact*. 2012, 441–504.

Kawsar, F., Sundramoorthy, V., Kortuem, G. et al. (2010) Smart Objects as Building Blocks for the Internet of Things. *IEEE Internet Computing*, vol. 14, issue 1, 44 – 51. doi: 10.1109/MIC.2009.143

Konstan, J.A. y Riedl, J. (2012) Recommender systems: From algorithms to user experience. User Model. *User Adapt. Interact*., 22, 101–123.

Kureshi, N.; Abidi, S.S.R. y Blouin, C. (2016). A Predictive Model for Personalized Therapeutic Interventions in Non-small Cell Lung Cancer. *IEEE J. Biomed. Health Inform.* 20, 424–431.

Lee, N., Lee, H., Lee, H. y Ryu, W. (2015) Implementation of smart home service over web of object architecture. *In: 2015 International Conference on Information and Communication Technology Convergence (ICTC)*, pp. 1215–1219. https://doi.org/10.1109/ictc.2015.

7354778

Lee, S. y Choi, J. (2017) Enhancing user experience with conversational agent for movie recommendation: Effects of self-disclosure and reciprocity. *J. Hum. Comput. Stud.*, 103, 95–105.

Li, L., Li, S. y Shao, S. (2014). QoS-Aware Scheduling of Services-Oriented Internet of Things. *IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS*. 10, 1497 - 1505.

Li, S., Xu, L. D. y Zhao S. (2015). The internet of things: a survey. *Information Systems Frontiers*. 17, 243-259.

Linkafy. Recuperado el 29 de Julio de 2017, de https://www.linkafy.com/

Liu J., Chen, Y., Chen, X., Ding, J., Choedhury, K. R., Hu, Q. y Wang, S. (2013). A Cooperative Evolution for QoS-driven IoT Service Composition. *Automatika*. 54, 438 – 447.

Li S, Xu LD and Zhao S (2015) The Internet of Things: A Survey. *Information Systems Frontiers* 17, 243-259.

Liu, L., Liu, X. y Li, X. (2012). Cloud-Based Service Composition Architecture for Internet of Things. *Communications in Computer and Information Science*. 312, 559-564.

Lua. Recuperado el 29 de Noviembre de 2018, de https://www.lua.org/

Lyazidi; A. y Mouline, S. (2015). ONDAR : An Ontology for Home Automation. *In Proceedings of the 2015 15th International Conference on Intelligent Systems Design and Applications (ISDA)*, Marrakesh, Morocco, 14–16 December; pp. 260–265.

Machorro-Cano, I.; Alor-Hernández, G.; Cruz-Ramos, N.A.; Sánchez-Ramírez, C. y Segura-Ozuna, M.G. (2018) A Brief Review of IoT Platforms and Applications in Industry; *New Perspe*; Springer, Cham, Switzerland, pp. 293–324.

Macker, J.P. y Taylor, I. (2017). Orchestration and analysis of decentralized workflows within heterogeneous networking infrastructures. *Future Generation Computer Systems*. 75, 388–401. doi: http://dx.doi.org/10.1016/j.future.2017.01.007

Mardonova, M. y Choi, Y. (2018). Review of Wearable Device Technology and Its Applications to the Mining Industry. *Energies*, 11, 547.

Marvell. Recuperado el 22 de Julio de 2017, de https://www.marvell.com/

Matsui, K. (2018) An information provision system to promote energy conservation and maintain indoor comfort in smart homes using sensed data by IoT sensors. *Future Gener. Comput. Syst.*, 82, 388–394.

McNee, S.M.; Riedl, J. y Konstan, J.A. (2006) Being accurate is not enough: How accuracy metrics have hurt recommender systems. *In Proceedings of the CHI'06 Extended Abstracts on Human Factors in Computing Systems*, Montreal, QC, Canada, 22–27 April; pp. 1097–1101.

Medina J. L. Y Villar E. (2017). Software Engineering and Real-Time Group, 1-4. Doi: https://www.istr.unican.es/publications/jmp-ev-2017.pdf.

Medium corporation. 12 Popular Programming Languages For IoT Development In 2017. Recuperado el 29 de Noviembre de 2018, de https://medium.com/@Intersog/12-popular-programming-languages-for-iot-development-in-2017-b8bf6ab5aef3

Microsoft LoT. Recuperado el 27 de Marzo de 2017, de http://www.lab-of-things.com/

Mineraud J., Mazhelis O., Su X. and Tarkoma S. (2016). A gap analysis of Internet-of-Things platforms. Computer Communications, 89, 5-16, doi: https://doi.org/10.1016/j.comcom.2016.03.015

Miorandi, D., Sicari, S., De Pellegrini, F. y Chlamtac, I. (2012). Internet of things: Vision, applications and research challenges. *Ad Hoc Networks*. 10, 1497-1516.

mnubo. Recuperado el 28 de Marzo de 2017, de http://mnubo.com/

Montali, M. y Plebani, P. (2017). IoT-based compliance checking of multi-party business processes modeledwith commitments. *Eur. Conf. Serv. Oriented Cloud Comput*. 10465, 179–195. doi: https://doi.org/10.1007/978-3-319-67262-5_14

Mozilla. Rust language. Recuperado el 29 de Noviembre de 2018, de https://research.mozilla.org/rust/

NanoService. Recuperado el 27 de Marzo de 2017, de http://www.conor.vc/sensinode-releases-nanoservice-platform-for-the-internet-of-things/#.WNB7mWf9nIU

NewAer. Recuperado el 27 de Marzo de 2017, de https://newaer.com/

Nimbits. Recuperado el 22 de Marzo de 2017, de https://github.com/bsautner/com.nimbits

Ogata S., Nakagawa H., Aoki Y., Kobayashi K. y Fukushima Y. (2019) A Tool to Edit and Verify IoT System Architecture Model. *JSPS KAKENHI Grant Number JP15K00097 y JP17KT0043, y la Fundación para el Avance de las Telecomunicaciones*. 1-5. doi: http://ceur-ws.org/Vol-2019/demos_3.pdf

Ojo, K., González, Y., Cano, E.E. y Rovetto, C.A. (2018). Modelado del funcionamiento de un dispositivo para el control de la asistencia estudiantil mediante Redes de Petri Coloreadas. *Memorias De Congresos UTP*, 1(1), 13-20. Recuperado a partir de https://revistas.utp.ac.pa/index.php/memoutp/article/view/1836

OpenRemote. Recuperado el 22 de Marzo de 2017, de http://www.openremote.com/

Open.sen.se. Recuperado el 22 de Marzo de 2017, de http://open.sen.se/

Oracle IoT. Recuperado el 20 de Marzo de 2017, de https://www.oracle.com/solutions/internet-of-things/

Ouaddah, A.; Mousannif, H., Elkalam, A.A. y Ouahman, A.A. (2017). Access control in the Internet of Things: Big challenges and new opportunities. *Comput. Netw*. 112, 237–262.

Packtpub. HTTP basics. Recuperado el 2 de Diciembre de 2018, de https://www.packtpub.com/mapt/book/application_development/9781783553532/2/ch02lvl1sec17/http-basics

Pahl, C., Ioini, N.E., Helmer, S. y Lee, B. (2018). An Architecture Pattern for Trusted Orchestration in IoT Edge Clouds. *2018 Third International Conference on Fog and Mobile Edge Computing (FMEC)*. 1-8.

Pang, Z., Zheng, L., Tian, J., Walter, S. K., Dubroba, E. y Chen, Q. (2015). Design of a terminal solution for integration of in-home health care devices and services towards the Internet-of-Things. *Enterprise Information Systems*. 9, 86-116.

Paraimpu. Recuperado el 22 de Marzo de 2017, de http://blog.paraimpu.com/

Pattanapairoj, S., Silsirivanit, A., Muisuk, K., Seubwai, W., Cha'On, U., Vaeteewoottacharn, K., Sawanyawisuth, K., Chetchotsak, D. y Wongkham, S. (2015). Improve discrimination power of serum markers for diagnosis of cholangiocarcinoma using data mining-based approach. *Clin. Biochem*. 48, 668–673.

Perera, C., Zaslavsky, A., Christen, P. y Georgakopoulos, D. (2013) Context Aware Computing for The Internet of Things: A Survey. *IEEE Communications Surveys & Tutorials* vol. 16, issue 1, 414 - 454. doi: 10.1109/SURV.2013.042313.00197

Picodotdev. Características de los lenguajes de programación. Recuperado el 15 de Julio de 2020, de https://picodotdev.github.io/blog-bitix/2015/10/caracteristicas-de-los-lenguajes-de-programacion/

Pisching, M. A., Junquiera, F., Santos Filho, D. J. y Miyagi, E. (2015). Service Composition in the Cloud-Based Manufacturing Focused on the Industry 4.0. *International Federation for Information Processing*. 450, 65-72.

Postscapes. IoT Standards and Protocols. Recuperado el 2 de Diciembre de 2018, de https://www.postscapes.com/internet-of-things-protocols/

Prince K., Barrett M. and Oborn E. (2014) Dialogical strategies for orchestrating strategic innovation networks: The case of the Internet of Things. *Information and Organization*, 106–127. doi: https://doi.org/10.1016/j.infoandorg.2014.05.001

Pu, P.; Chen, L. y Hu, R. (2011) A user-centric evaluation framework for recommender systems. *In Proceedings of the RecSys' 11 Fifth ACM Conference on Recommender Systems*, Chicago, IL, USA, 23–27 October; Volume 157, pp. 157–164.

Qubit Labs. 10+ Top IoT Programming Languages and Tools. Recuperado el 29 de Noviembre de 2018, de https://qubit-labs.com/top-iot-programming-languages-tools/

Qu, C., Liu, F., Tao, M. y Deng, D. (2016). An OWL-S based specification model of dynamic entity services for Internet of Things. *Journal of Ambient Intelligence and Humanized Computing*. 7, 73-82.

Quora. Recuperado el 2 de Diciembre de 2018, de https://www.quora.com/Which-are-the-communication-protocol-used-for-IOT

R. Recuperado el 29 de Noviembre de 2018, de https://www.r-project.org/

Rajagopalan, R. y Varshney, P.K. (2006) Data aggregation techniques in sensor networks: A survey. *Electr. Eng. Comput. Sci*. 6, 48–63.

Rapti E, Karageorgos A and Gerogiannis VC (2015). Decentralised Service Composition using Potential Fields in Internet of Things Applications. *Procedia Computer Science*, 700 – 706. doi: https://doi.org/10.1016/j.procs.2015.05.079

Ren, M., Ren, L. y Jain, H. (2018). Manufacturing service composition model based on synergy effect: Asocial network analysis approach. Applied Soft Computing. 70, 288–300. doi: https://doi.org/10.1016/j.asoc.2018.05.039

Risteska Stojkoska, B.L. y Trivodaliev, K.V. (2017) A review of Internet of Things for smart home: Challenges and solutions. *J. Clean. Prod.* 140, 1454–1464.

Robles D. A., Escamilla P. J. y Tryfonas T. (2017). IoTsec: UML Extension for Internet of Things Systems Security Modelling. *International Conference on Mechatronics, Electronics and Automotive Engineering (ICMEAE)*. 151-156. doi: 10.1109/ICMEAE.2017.20

Rodríguez Valenzuela, S., Holgado Terriza, J. A., Gutiérrez Guerrero J. M., y Muros Cobos, J. L. (2014). Distributed Service-Based Approach for Sensor Data Fusion in IoT Environments. *Sensors*. 14, 19200-19228.

Rodríguez Valenzuela, S., Holgado Terriza, J. A., Muros Cobos, J. L. y Gutiérrez Guerrero J. M. (2012). Data Fusion Mechanism based on a Service Composition Model for the Internet of Things. *Jornadas Sarteco*. 61, 1-6.

Said O and Masud M (2013) Towards Internet of Things: Survey and Future Vision. *International Journal of Computer Networks (IJCN)* 5, 1-17.

Salle, A. D., Gallo, F. y Perucci, A. (2016). Dependable Composition of Software and Services in the Internet of Things: A Biological Approach. *Software Engineering and Formal Methods - Lecture Notes in Computer Science*. 9509, 312-323.

SAP. Recuperado el 28 de Marzo de 2017, de https://www.sap.com/

Sathyadevi, G. (2011). Application of CART algorithm in hepatitis disease diagnosis. *In Proceedings of the 2011 International Conference on Recent Trends in Information Technology (ICRTIT)* Tamil Nadu, India, 3–5 June; pp. 1283–1287.

Seeger J., Deshmukh R. A., Sarafov, V. y Bröring A. (2018). Dynamic IoT Choreographies. Managing Discovery, Distribution, Failure and Reconfiguration. *IEEE Pervasive Computing's, IoT communications*, vol. 18, issue 1, 19 - 27. doi: 10.1109/MPRV.2019.2907003

SensorCloud. Recuperado el 25 de Marzo de 2017, de http://www.sensorcloud.com/

Sequans. Recuperado el 07 de Agosto de 2017, de http://www.sequans.com

Sezer, O.B.; Can, S.Z. y Dogdu, E. (2015) Development of a Smart Home Ontology and The Implementation of A Semantic Sensor Network Simulator: An Internet of Things Approach. *In Proceedings of the 2015 International Conference on Collaboration Technologies and Systems (CTS)*, Atlanta, GA, USA, 1–5 June; pp. 12–18.

Shehu, U., Safdar, G. A. y Epiphaniou, G. (2015). Network-aware Composition for Internet of Thing Services. *Transactions on Networks and Communications*. 3, 45-58.

Shoaip, N.; Elmogy, M.M.; Riad, A.M.; Zaghloul, H.; Badria, F.A.; Giannoccaro, I.; Hassanien, A.E. y Gaber, T. (2017). Early-Stage Ovarian Cancer Diagnosis Using Fuzzy Rough Sets with SVM Classification. *In Handbook of Research on Machine Learning Innovation and Trends*; IGI Global: Hershey, PA, USA; pp. 43–60.

Sharaff, A. y Rath, S.K. (2020). Formalization of UML Class Diagram Using Colored Petri Nets. *2020 First International Conference on Power, Control and Computing Technologies (ICPC2T)*, IEEE, 311 – 315. doi: 10.1109/ICPC2T48082.2020.9071490

Showkat Ara, S., Ush Shamaszaman, Z. y Chong, I. (2014). Web-of-Objects Based User-Centric Semantic Service Composition Methodology in the Internet of Things. *International Journal of Distributed Sensor Networks*. 1-11.

Sine Wave. Recuperado el 28 de Marzo de 2017, de http://www.sine-wave.com/

Sofia 2. Recuperado el 20 de Marzo de 2017, de http://sofia2.com/

Sood, S.K. y Mahajan, I. (2018). A Fog-Based Healthcare Framework for Chikungunya. *IEEE Internet Things J.* 5, 794–801.

Sosa-Reyna, C.M., Tello-Leal, E. y Lara-Alabazares, D. (2018). Methodology for the Model-Driven Development of Service Oriented IoT Applications. *Journal of Systems Architecture*, vol. 90, 15-22. doi: https://doi.org/10.1016/j.sysarc.2018.08.008

Stavropoulos, T. G., Vrakas, D. y Vlahavas, I. (2013). A survey of service composition in ambient intelligence environments. *Artificial Intelligence Review*, 40, 247-270.

Stelmach, S. (2013). Service Composition Scenarios in the Internet of Things Paradigm. *IFIP International Federation for Information Processing 2013*. 394, 53-60.

Subbalakshmi, G.; Ramesh, K.; Rao, C. (2011). Decision Support in Hearth Disease Prediction System using Naive Bayes. *Indian J. Comput. Sci. Eng*., 2, 170–176.

Suca, C.; Córdova, A.; Condori, A.; Cayra, J. y Sulla, J. (2016) Comparison of Classification Algorithms for Prediction of Cases of Childhood Obesity. 1–9.

Sulistyo, S. (2013). Software Development Methods in the Internet of Things. *IFIP International Federation for Information Processing*. 50-59.

Sun, M., Zhou, Z., Wang, J., Du, C. y Gaaloul, W. (2019) Energy-Efficient IoT Service Composition for Concurrent Timed Applications. *Future Gener. Comput. Syst*., 100, 1017–1030. doi: https://doi.org/10.1016/j.future.2019.05.070

Sundmaeker, H., Guillemin, P., Fries, P. y Woelfflé, S. (2010). Vision and Challengues for Realising the Internet of Things. *CERP-IoT – Cluster of European Research Projects on the Internet of Things*. 11-24.

Swiatek, P. (2015). ComSS – Platform for Composition and Execution of Streams Processing Services. *Intelligent Information and Database Systems - Lecture Notes in Computer Science*. 9012, 494-505.

Tartar, A.; Kilic, N. y Akan, A. (2013). A new method for pulmonary nodule detection using decision trees. *In Proceedings of the 2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, Osaka, Japan, 3–7 July; pp. 7355–7359, doi:10.1109/EMBC.2013.6611257.

Techahead. Top 6 Programming Languages for IoT Projects. Recuperado el 29 de Noviembre de 2018, de https://www.techaheadcorp.com/blog/top-6-programming-languages-for-iot-projects/

Techopedia. The Top 10 Coding Languages for IoT Projects. Recuperado el 29 de Noviembre de 2018, de https://www.techopedia.com/the-top-10-coding-languages-for-iot-projects/2/32549

Tektonidis, D., Karagioannidis, C., Kouroupetroglou, C. y Koumpis, A. (2014). Intuitive User Interfaces to Help Boost Adoption of Internet-of-Things and Internet-of-Content Services for All. *Inter-cooperative Collective Intelligence: Techniques and Applications, Studies in Computational Intelligence*. 495, 93-110.

ThingSpeak. Recuperado el 25 de Marzo de 2017, de https://thingspeak.com/

ThingWorx. Recuperado el 28 de Marzo de 2017, de https://www.thingworx.com/

Thramboulidis K. y Christoulakis F. (2016). UML4IoT—A UML-based approach to exploit IoT in cyber-physical manufacturing systems. *Computers in Industry*. 82. 259-272. doi: http://dx.doi.org/10.1016/j.compind.2016.05.010

Toptal. Buggy C# Code: The 10 Most Common Mistakes in C# Programming. Recuperado el 29 de Noviembre de 2018, de https://www.toptal.com/c-sharp/top-10-mistakes-that-c-sharp-programmers-make

Trappey, A.J.; Trappey, C.V.; Govindarajan, U.H.; Chuang, A.C. y Sun, J.J. (2017). A review of essential standards and patent landscapes for the Internet of Things: A key enabler for Industry 4.0. *Adv. Eng. Inform.*, 33, 208–229.

Urbieta A, González-Beltrán A, Mokhtar SB, Hossain MA y Capra L (2017) Adaptive and context-aware service composition for IoT-based smartcities. *Future Generation Computer Systems*, 262–274. doi: http://dx.doi.org/10.1016/j.future.2016.12.038

Valiente-Rocha, P.A.y Lozano-Tello, A. (2010) Ontology-Based Expert System for Home Automation Controlling Ontology-Based Expert System for Home Automation Controlling. *In Proceedings of the 23rd International Conference on Industrial Engineering and Other Applications of Applied Intelligent Systems, IEA/AIE 2010*, Cordoba, Spain, 1–4 June; Volume 6096, pp. 661–670.

Vidyasankar k (2016). A Transaction Model for Executions of Compositions of Internet of Things Services. *Procedia Computer Science*, 195 – 202. doi: https://doi.org/10.1016/j.procs.2016.04.116

Wang, Z., Li, W. y Dong, H. (2017). Review on open source operating systems for internet of things. IOP Conf. *Journal of Physics*, 887, 1-6. doi :10.1088/1742-6596/887/1/012044

Webofthings. What's in HTTP/2 for the Internet of Things? ½. Recuperado el 2 de Diciembre de 2018, de https://webofthings.org/2015/10/25/http2-for-internet-of-things-1/

Welbourne, E., Battle, L., Cole, G., et al. (2009). Building the Internet of Things Using RFID. *IEEE Internet Computing*, vol. 13, issue: 3, 48-55. doi: 10.1109/MIC.2009.52

Wen, Z., Yang, R., Garraghan, P., Lin, T., Xu, J. y Rovatsos, M. (2017). Fog Orchestration for Internet of Things Services. *IEEE Internet Computing*, 16 – 24. doi: 10.1109/MIC.2017.36

Westergaard, M. y Kristensen, L.M. (2009). The Access/CPN Framework: A Tool for Interacting with the CPN Tools Simulator. In: Franceschinis G., Wolf K. (eds) Applications and Theory of Petri Nets. PETRI NETS 2009. *Lecture Notes in Computer Science*, vol 5606. Springer, Berlin, Heidelberg. doi: https://doi.org/10.1007/978-3-642-02424-5_19

Whatis.techtarget. Advanced Message Queuing Protocol (AMQP). Recuperado el 2 de Diciembre de 2018, de https://whatis.techtarget.com/definition/Advanced-Message-Queuing-Protocol-AMQP

Whitmore, A., Agarwal, A. y Xu, L.D. (2015) The Internet of Things – A Survey of Topics and Trends. *Information Systems Frontiers* 17, 261-274.

Wind River. Recuperado el 29 de Julio de 2017, de https://www.windriver.com/

(WHO-1) World Health Organization. Obesity and Overweight. Recuperado el 10 de Mayo de 2019, de https://www.who.int/en/news-room/fact-sheets/detail/obesity-and-overweight

(WHO-2) World Health Organization. Global Strategy on Diet, Physical Activity and Health. Recuperado el 19 de Mayo de 2019, de https://www.who.int/dietphysicalactivity/childhood/en/

Wongpatikaseree, K.; Ikeda, M.; Buranarach, M.; Supnithi, T.; Lim, A.O. y Tan, Y. (2012) Activity Recognition using Context-Aware Infrastructure Ontology in Smart Home Domain. *In Proceedings of the 2012 Seventh International Conference on Knowledge, Information and Creativity Support Systems*, Victoria, Australia, 8–10 November; pp. 50–57.

Wortmann, F. y Flüchter, K. (2015). Internet of things. *Bus Inf Syst Eng*. 57, 221-224, doi: 10.1007/s12599-015-0383-3

Wovyn. Recuperado el 02 de Agosto de 2017, de http://www.wovyn.com/

W3C. Recuperado el 19 de Junio de 2020, de https://www.w3.org/TR/ws-cdl-10-primer/

Xively. Recuperado el 28 de Marzo de 2017, de https://www.xively.com/

XOBXOB(2017). Recuperado el 25 de Marzo de 2017, de http://xobxob.com/

Xu, L.D., He, W. y Li, S. (2014). Internet of Things in Industries: A Survey. *IEEE Transactions on Industrial Informatics*, 10, 2233-2243.

Yaler. Recuperado el 28 de Marzo de 2017, de https://yaler.net/

Yang, Z. y Li, D. (2014). IoT information service composition driven by user requirement. *IEEE 17th International Conference on Computational Science and Engineering*. 509-513. doi: 10.1109/CSE.2014.280

Yang, Y.; Chen, H.; Wang, D.; Luo, W.; Zhu, B. y Zhang, Z. (2014). Diagnosis of pancreatic carcinoma based on combined measurement of multiple serum tumor markers using artificial neural network analysis. *Chin. Med. J.*, 127, 1891–1896.

Yu, J., Bang, H. C., Lee, H. y Lee, Y. S. (2016). Adaptive Internet of Things and Web of Things convergence platform for Internet of reality services. *The Journal of Supercomputing*. 72, 84-102.

Zatar. Recuperado el 30 de Marzo de 2017, de http://www.zatar.com/

Zhou, J.y Reniers, G. (2017). A Petri-net based simulation analysis approach for cascading effect of vapor cloud explosions. *Journal of loss prevention in the process industries.* ISSN 0950-4230 - 48, pp 118-125, doi: https://doi.org/10.1016/J.JLP.2017.04.017

Zhou, J., Leppänen, T., Harjula, E., Yu, C, Jin, H. y Tianruo Yang, L. (2013). CloudThings: a Common Architecture for Integrating the Internet of Things with Cloud Computing. *IEEE 17th International Conference on Computer Supported Cooperative Work in Design*. 651-657.

2lemetry. Recuperado el 27 de Marzo de 2017, de https://aws.amazon.com/es/iot/