

Metodología Basada en Modelos en Tiempo de Ejecución para la Construcción y Operación de Sistemas Autoconscientes de Internet de las Cosas

Lenin Erazo-Garzón¹

¹LIDI – Universidad del Azuay
Cuenca, Ecuador

lerazo@uazuay.edu.ec

Abstract. *Nowadays, one of the technological challenges is the development of software for IoT systems, since they operate in highly changing scenarios, being complex with traditional Software Engineering methodologies to identify all the system requirements in the development stage. An alternative is to increase their autonomy, providing them with self-awareness capabilities with the support of runtime models, to transfer several of the functionalities that are programmed during development to runtime. This doctoral work proposes to develop a methodology based on runtime models for the construction and operation of self-aware IoT systems.*

Resumen. *Hoy en día, uno de los desafíos tecnológicos es el desarrollo de software para sistemas de IoT, ya que operan en escenarios altamente cambiantes, siendo complejo con las metodologías de Ingeniería de Software tradicionales identificar todos los requisitos del sistema en la etapa de desarrollo. Una alternativa es incrementar su autonomía, dotándoles de capacidades de autoconsciencia con el apoyo de los modelos en tiempo de ejecución, a fin de trasladar varias de las funcionalidades que se programan durante el desarrollo al tiempo de ejecución. Este trabajo doctoral propone desarrollar una metodología basada en modelos en tiempo de ejecución para la construcción y operación de sistemas autoconscientes de IoT.*

1. Introducción

El Internet de las Cosas (IoT) es un paradigma tecnológico que se usa para referirse a una infraestructura compuesta por un conjunto de objetos (cosas) inteligentes, ubicuos e interconectados mediante las tecnologías de información para procesar datos del mundo físico y virtual; y, actuar con la mínima intervención humana [Madakam et al. 2015]. El IoT promueve el desarrollo de innovadoras aplicaciones en diversos dominios (p. ej., hogar, ciudad inteligente, industria), agregando una nueva dimensión a la vida de las personas, basada en la interacción permanente con los objetos inteligentes, en cualquier lugar y momento, lo cual impacta favorablemente en el desarrollo de la sociedad.

Los sistemas de IoT operan en entornos altamente cambiantes que deben ser controlados oportunamente para garantizar su correcto y continuo funcionamiento [Miorandi et al. 2012]. Debido a la incertidumbre intrínseca asociada a los cambios, resulta complejo identificar todos los requisitos de un sistema de IoT durante la etapa de desarrollo. Además, las metodologías y herramientas de Ingeniería de Software

tradicionales han demostrado ser poco productivas para afrontar el dinamismo y la escalabilidad de los sistemas de IoT, siendo los desarrolladores los que gestionan la evolución del sistema en el tiempo de diseño [Sosa-Reyna et al. 2018]. Esta limitación se agudiza aún más, ya que estos enfoques metodológicos y herramientas se centran en los detalles de implementación (programación a bajo nivel) en lugar del dominio del problema, volviéndose complejo para los desarrolladores por la heterogeneidad de los dispositivos de IoT (sensores, actuadores) que deben ser integrados [Chen et al. 2015].

Un enfoque apropiado es incorporar la autoconsciencia computacional a los sistemas de IoT con el apoyo de los modelos en tiempo de ejecución, a fin de aumentar su capacidad autónoma y resiliente frente a las situaciones imprevistas [Esterle and Grosu 2016]. Un sistema de IoT es considerado autoconsciente, si es capaz de obtener y mantener el conocimiento sobre sí mismo y el entorno (p. ej., estructura, estado, evolución), sin control externo [Kounev et al. 2017]. Dicho conocimiento puede ser capturado directamente de los datos crudos recolectados desde los sensores físicos o virtuales (observación inmediata – perceptual) o extraído de modelos conceptuales internos, construidos a partir de los datos históricos [Lewis et al. 2017].

Los modelos en tiempo de ejecución (*models@run.time*) constituyen una auto-representación causalmente conectada de un sistema observado. En este sentido, la información recopilada desde el sistema de IoT puede utilizarse para retroalimentar los modelos en tiempo de ejecución. Luego, estos modelos pueden analizarse para conocer el estado y la posible evolución del sistema; y, en caso de ser necesario recomendar acciones correctivas. Por lo tanto, el uso de los *models@run.time* en los sistemas de IoT contribuye a trasladar varias de las funcionalidades que tradicionalmente se programan durante el desarrollo al tiempo de ejecución, lo cual eleva el nivel de abstracción y automatización del proceso de desarrollo, así como también mejora la gestión de la complejidad, la incertidumbre y el cambio [Bencomo et al. 2019].

La autoconsciencia es un campo emergente dentro de la Ingeniería de Software con importantes desafíos y oportunidades de investigación [Elhabbash et al. 2019]. A su vez, según [Bencomo et al. 2019] uno de los desafíos actuales de los *models@run.time* es la autoconsciencia computacional. Particularmente, los trabajos de investigación sobre autoconsciencia en IoT son escasos y se orientan principalmente a nivel conceptual [Gurgen et al. 2013; Tammemäe et al. 2017; Esterle et al. 2020].

Este artículo presenta una propuesta de tesis doctoral cuyo propósito es desarrollar una metodología y su infraestructura de soporte basada en modelos en tiempo de ejecución para la construcción y operación de sistemas autoconscientes de IoT. En la actualidad, la solución propuesta se encuentra en la etapa de evaluación experimental y refinamiento, con la finalidad de garantizar un método formal, ordenado, eficaz, eficiente y repetible para el desarrollo de sistemas autoconscientes de IoT.

La estructura de este documento es la siguiente: Sección 2 presenta el estado del arte sobre la autoconsciencia en IoT. Sección 3 plantea los objetivos y preguntas de investigación de la tesis doctoral. Sección 4 explica la metodología de investigación empleada. Secciones 5 y 6 describen la solución propuesta y los resultados preliminares de su evaluación. Finalmente, la Sección 7 presenta las conclusiones y el trabajo futuro.

2. Trabajos Relacionados

Según la revisión sistemática sobre autoconsciencia en Ingeniería de Software

elaborada por [Elhabbash et al. 2019], los avances dentro del dominio de IoT son poco significativos. Recién en los últimos años ha crecido el interés de los investigadores sobre la autoconsciencia en IoT, siendo en su mayoría trabajos conceptuales, enfocados en identificar los beneficios, desafíos y recomendaciones en torno a la construcción de sistemas autoconscientes de IoT. En [Tammemäe et al. 2017] se formulan varios estudios de caso para demostrar que incorporar la autoconsciencia a nivel de los nodos edge y fog de una infraestructura de IoT es beneficioso para mejorar el rendimiento de los recursos durante el procesamiento de los datos. A su vez, [Gurgen et al. 2013] describen una arquitectura general para un sistema ciber-físico autoconsciente, basada en el modelo MAPE-K y un middleware orientado a servicios distribuidos, junto con los desafíos y recomendaciones para su implementación en las infraestructuras de una ciudad inteligente. Por su parte, [Götzinger et al. 2020] proponen el framework RoSA, el cual está basado en un modelo jerárquico de agentes que proporciona facilidades para implementar y evaluar aplicaciones autoconscientes para sistemas ciber-físicos.

Con respecto a la autoconsciencia colectiva, [Esterle et al. 2018] presentan un artículo de posición sobre un enfoque de diseño de aplicaciones de IoT, basado en una arquitectura autoconsciente que utiliza modelos de interacción colectiva entre los nodos de IoT para mejorar el rendimiento de la red global. En otro trabajo, [Esterle et al. 2020] amplían la idea de la autoconsciencia de sistemas individuales y proponen una hoja de ruta hacia una autoconsciencia en red, a fin de aumentar la capacidad de los sistemas ciber-físicos para aprender sobre el entorno y las interacciones con otros sistemas. En un ámbito general, [Lewis et al. 2017] presentan un framework conceptual que define los niveles, aspectos y el dominio de un sistema autoconsciencia.

3. Objetivos y Preguntas de Investigación

Este trabajo de investigación doctoral propone como objetivo general: desarrollar una metodología y su infraestructura de soporte basada en modelos en tiempo de ejecución para la construcción y operación de sistemas autoconscientes de IoT. A su vez, el objetivo general se descompone en los siguientes objetivos específicos: i) analizar el estado del arte sobre la autoconsciencia computacional y el uso de los `models@run.time` en IoT y otros dominios relacionados; ii) definir un proceso metodológico formal, sistémico y repetible que integre las actividades, guías, recursos, artefactos y herramientas necesarias para la construcción y operación de sistemas autoconscientes de IoT; iii) diseñar la arquitectura de la infraestructura que dará soporte al método propuesto; iv) construir los componentes de la infraestructura de soporte, incluyendo metamodelos, lenguajes específicos de dominio (DSLs), herramientas de configuración y otras herramientas a nivel de middleware para mantener la relación causal entre los modelos y el sistema de IoT observado, así como para ejecutar las capacidades de autoconsciencia especificadas en los modelos; e, v) instanciar y evaluar la metodología y su infraestructura para conocer la percepción de los desarrolladores sobre su efectividad, eficiencia, facilidad de uso, utilidad e intención de uso futuro.

Para alcanzar los objetivos se definen las siguientes preguntas de investigación: PI-1. ¿Cuáles son los problemas y desafíos sobre el uso de los `models@run.time` para incorporar la autoconsciencia a los sistemas de IoT? PI-2. ¿Qué actividades, guías, recursos, artefactos y herramientas deben incorporarse en la metodología? PI-3. ¿Cómo organizar las actividades y recursos de la metodología? PI-4. ¿Cómo diseñar la arquitectura de la infraestructura de soporte? PI-5. ¿Cómo construir los artefactos y

herramientas de la infraestructura de soporte? PI-6. ¿Cómo evaluar y demostrar que la metodología y su infraestructura es considerada por los desarrolladores como efectiva, eficiente, fácil de usar, útil, y que tienen la intención de utilizarla en el futuro?

4. Metodología de Investigación

La metodología empleada para el desarrollo de la tesis doctoral está basada en el Modelo de Investigación y Transferencia de Tecnología propuesto por [Gorschek et al. 2006]. Este modelo se fundamenta en un proceso que inicia con la formulación del problema y la revisión del estado del arte. Luego se definen las soluciones candidatas, las cuales a través de un ciclo evolutivo son objeto de evaluaciones empíricas y una mejora continua hacia la búsqueda de una solución adecuada y realista. Con respecto a la evaluación empírica de la solución se utilizan cuasi-experimentos basados en el Modelo de Evaluación de Métodos (MEM) [Moody 2001].

5. Solución Propuesta

5.1. Metodología Basada en Modelos en Tiempo de Ejecución

La metodología propuesta integra un conjunto de actividades, guías, recursos y herramientas que indican cómo construir un sistema autoconsciente de IoT. En la Figura 1, se presenta un diagrama de proceso de alto nivel basado en la notación SPEM 2.0, incluyendo las seis actividades principales que conforman la metodología, así como los artefactos de entrada y salida que se utilizan durante el proceso. Las dos primeras actividades requieren de la intervención de los desarrolladores, mientras que las cuatro restantes se ejecutan automáticamente. A continuación, se describen estas actividades:

- 1. Diseño de la arquitectura de monitoreo del sistema de IoT.** Tiene como alcance el modelado visual de la arquitectura multi-capa de un sistema de IoT, la cual incluya: i) las entidades de IoT (físicas o digitales) y sus propiedades a ser monitoreadas; ii) las entidades digitales (recursos de hardware / software) que soportan los procesos de recolección, transporte, procesamiento y almacenamiento de datos, a nivel de las capas edge, fog y cloud; y, iii) los flujos de datos entre las entidades digitales, basados en protocolos de comunicación síncronos o asíncronos.
- 2. Configuración de las capacidades de autoconsciencia del sistema de IoT.** Construye el modelo de autoconsciencia, el cual incorpora especificaciones sobre: i) el dominio (sujetos y objetos) de autoconsciencia; ii) los aspectos a capturar sobre los objetos de autoconsciencia (p. ej., estado, causalidad, evolución); iii) los procesos pre-reflexivos¹ y reflexivos² de autoconsciencia [Lewis et al. 2017], con sus respectivos métodos de recolección, cálculo y modelos de análisis que determinan la forma como un sistema de IoT ejecuta sus capacidades de autoconsciencia; iv) las métricas (directas, indirectas e indicadores) para evaluar los aspectos de autoconsciencia; v) los recursos de implementación (fórmulas, funciones o servicios web) que son utilizados por los métodos de autoconsciencia para calcular las métricas; y, vi) las recomendaciones prescriptivas asociadas a los

¹ Proceso capaz de obtener conocimiento subjetivo directamente de los datos crudos recolectados desde los sensores (observación perceptual), permitiendo al sistema de IoT responder a los eventos de manera inmediata.

² Proceso capaz de construir modelos conceptuales internos que capturan conocimiento a partir de los datos históricos recopilados por los sensores.

criterios de decisión y umbrales de los indicadores.

3. **Sincronización entre los modelos de arquitectura - autoconsciencia y el sistema de IoT observado (mantiene la relación causal).** A partir de las especificaciones de los modelos y su evolución, esta actividad implementa o reconfigura automáticamente: i) los recursos de software del sistema de IoT (APIs, aplicaciones, servicios, brokers, base de datos) que soportan el monitoreo; y, ii) los recursos de implementación que operacionalizan los métodos de autoconsciencia, a fin de calcular las métricas y evaluar los aspectos de autoconsciencia.
4. **Monitoreo.** Ejecuta los recursos de software creados en la actividad anterior con el propósito de recolectar, transportar, procesar y almacenar los datos crudos obtenidos desde los sensores.
5. **Aprendizaje.** Utiliza los datos generados por la actividad de monitoreo y las especificaciones de los modelos para producir métricas directas e indirectas que valoren los aspectos de autoconsciencia. Las métricas directas son obtenidas mediante los métodos de recolección y representan los datos crudos recolectados directamente desde los sensores u otras fuentes. Mientras que las métricas indirectas se derivan de otras métricas directas e indirectas por medio de la construcción, refinamiento y aplicación de modelos conceptuales (p. ej., modelos de regresión, de aprendizaje automático) especificados en los métodos de cálculo.
6. **Razonamiento.** Utiliza las especificaciones del modelo de autoconsciencia para aplicar modelos de análisis que combinen y comparen los valores obtenidos para ciertas métricas con los criterios de decisión (umbrales), a fin de obtener nuevos indicadores cuantitativos y cualitativos que determinen el estado o la posible evolución futura de un objeto de autoconsciencia; y, en caso de ser necesario recomendar las acciones prescriptivas incluidas en el modelo de autoconsciencia.

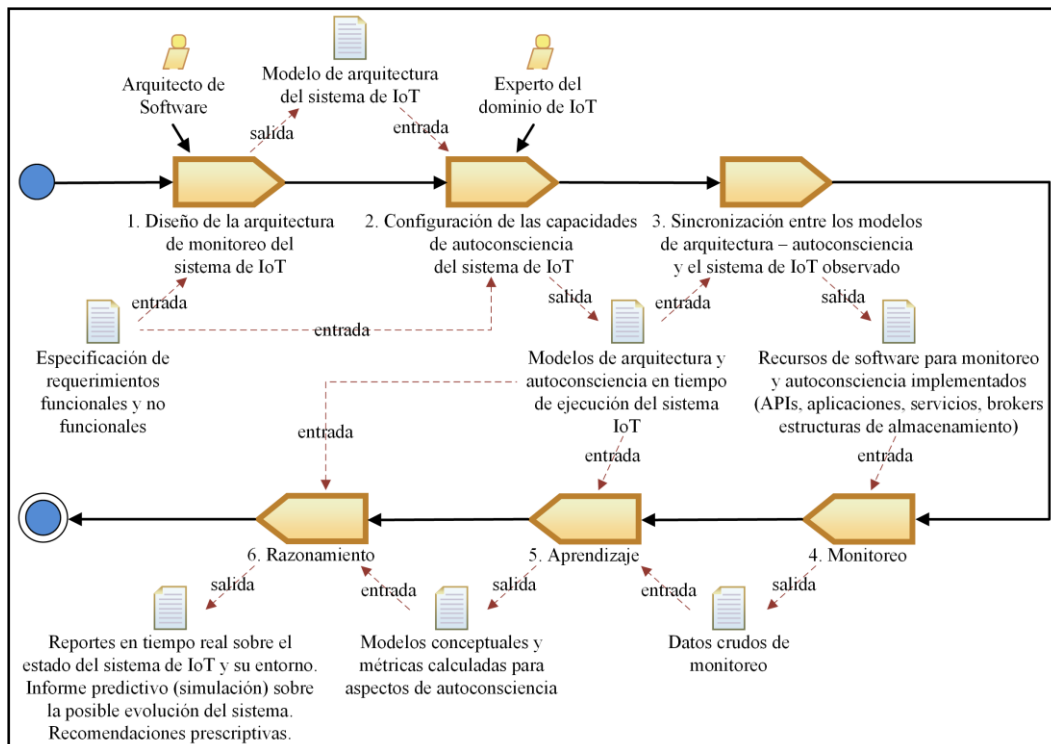


Figura 1. Proceso metodológico de alto nivel basado en *models@run.time*.

5.2. Infraestructura de Soporte

La Figura 2 presenta la arquitectura de la infraestructura de soporte, cuyos principales componentes se describen brevemente a continuación:

- **Monitor-IoT.** Lenguaje específico de dominio (DSL) para modelar arquitecturas multi-capa que soporten los procesos de monitoreo de un sistema de IoT. Monitor-IoT incluye un diseñador gráfico desarrollado en Obeo Designer y un metamodelo alineado con la arquitectura de referencia para IoT ISO/IEC 30141:2018.
- **Configurador de capacidades de autoconsciencia de sistemas de IoT.** Aplicación web desarrollada en Node.js para mantener y generar el modelo de autoconsciencia en tiempo de ejecución. El configurador utiliza un metamodelo construido a partir del framework conceptual de autoconsciencia de [Lewis et al. 2017] y la ontología de medición de software de [García et al. 2004].
- **Middleware de la infraestructura de soporte.** Integrado por los motores de sincronización y autoconsciencia, los cuales han sido construidos en Node.js. El *motor de sincronización* detecta automáticamente cualquier cambio en los modelos y ejecuta tareas de implementación y reconfiguración de los recursos del sistema de IoT. Mientras que el *motor de autoconsciencia* crea demonios (daemon) que encapsulan y ejecutan los procesos pre-reflexivos y reflexivos en segundo plano, según lo establecido en el modelo de autoconsciencia en tiempo de ejecución.

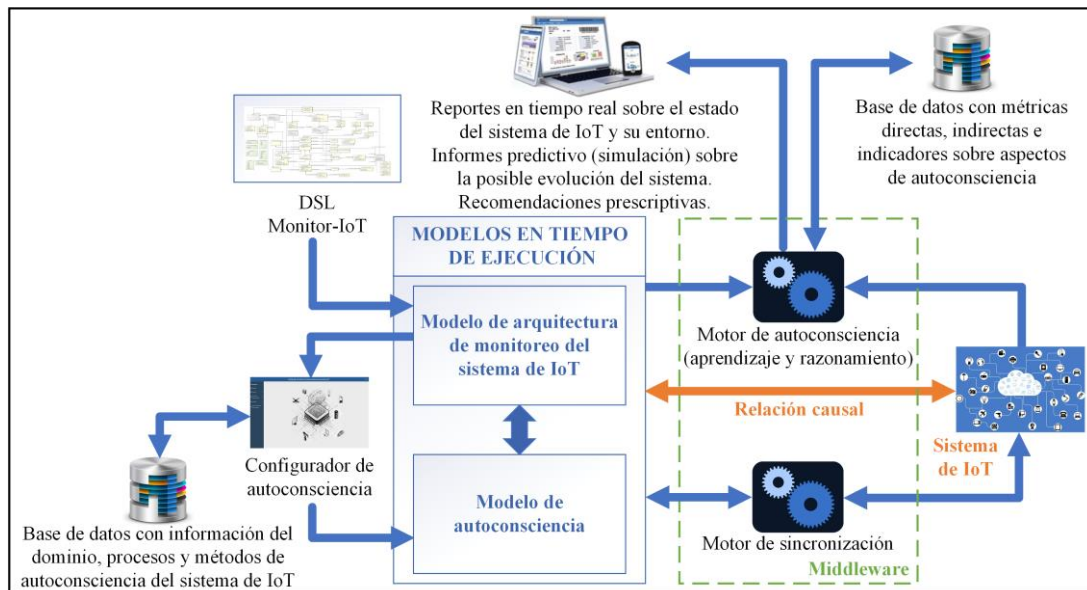


Figura 2. Arquitectura de la infraestructura basada en models@run.time.

6. Resultados Preliminares de la Evaluación de la Solución Propuesta

De acuerdo con el paradigma de Goal-Question Metric (GQM), en la Tabla 1 se define el objetivo de evaluación de la solución propuesta. Al momento, se han evaluado las actividades 1 y 3 de la metodología mediante la aplicación de cuasi-experimentos basados en el MEM. Los experimentos contemplaron el uso de Monitor-IoT para el diseño de la arquitectura de un sistema de gestión de emergencias para adultos mayores que viven solos en su hogar, así como el despliegue del motor de sincronización para la implementación y reconfiguración automática de los recursos de software especificados

en el modelo de arquitectura (boletines de los experimentos están disponibles en: <http://t.ly/vsPr>). En los experimentos se utilizaron formularios de trabajo para recolectar métricas sobre el rendimiento de los participantes y se realizaron encuestas para conocer su percepción sobre las herramientas utilizadas. Para analizar los datos recogidos, se utilizó estadística descriptiva y pruebas de hipótesis. La Tabla 2 presenta un resumen de los resultados obtenidos en los experimentos para las variables de rendimiento (efectividad y eficiencia) y de percepción (facilidad de uso, utilidad e intención de uso). Los valores promedio de efectividad demuestran que los participantes fueron capaces de cumplir en gran medida con las tareas experimentales. A su vez, los tiempos promedio invertidos en el cumplimiento de las tareas experimentales son mínimos en comparación al tiempo que se necesitaría para implementar los recursos de software del escenario experimental con el apoyo de métodos y herramientas de Ingeniería de Software tradicionales. Finalmente, las variables de percepción tienen como común denominador que su promedio es mayor al valor neutro (3) de la escala de Likert, lo que demuestra que los participantes perciben a los componentes evaluados como fáciles de usar, útiles, y que tienen la intención de utilizarlos a futuro.

Tabla 1. Definición de Goal-Question-Metric para la evaluación de la solución

<i>Evaluar:</i>	La metodología y su infraestructura de soporte basada en <code>models@run.time</code> para la construcción y operación de sistemas autoconscientes de IoT.
<i>Con el propósito de:</i>	Analizar y determinar el rendimiento (efectividad y eficiencia) de los usuarios en el uso de la metodología y su infraestructura soporte, así como su percepción con respecto a la facilidad de uso, utilidad e intención de uso futuro de la solución.
<i>Desde el punto de vista del:</i>	Investigador.
<i>En el contexto de:</i>	Un grupo de estudiantes de los últimos semestres de la carrera de Ingeniería de Sistemas de la Universidad del Azuay y Universidad de Cuenca.

Tabla 2. Estadísticos descriptivos de las variables de rendimiento y percepción

Variable	Actividad 1		Actividad 3	
	Media	Desviación estándar	Media	Desviación estándar
<i>Efectividad:</i> Grado de cumplimiento de los objetivos del método.	86,98%	15,69%	94,53%	8,36%
<i>Eficiencia (minutos):</i> Tiempo empleado para aplicar el método.	86,78	16,75	22,09	6,74
<i>Facilidad de uso:</i> Grado en que una persona cree que utilizando un método requerirá menos esfuerzo.	4.43	0.51	4,14	0,71
<i>Utilidad percibida:</i> Grado en que una persona cree que utilizando un método mejorará su desempeño.	4.55	0.49	4,53	0,51
<i>Intención de uso:</i> Medida en que una persona tiene la intención de utilizar un método.	4.56	0.55	4,44	0,76

7. Conclusiones y Trabajo Futuro

La revisión de la literatura evidencia que dotar de autoconsciencia a los sistemas de IoT con el apoyo de los `models@run.time` representa una brecha de investigación en la actualidad. De ahí que, como parte de los resultados alcanzados en esta tesis doctoral en curso, se propone desde una perspectiva general una metodología e infraestructura de soporte basada en `models@run.time` para la construcción y operación de sistemas autoconscientes de IoT. Los componentes de la solución metodológica evaluados hasta el momento, han alcanzado indicadores favorables de efectividad, eficiencia y usabilidad. Como trabajo futuro se tiene previsto difundir a través de publicaciones cada una de las actividades y componentes de la metodología y su infraestructura de soporte, junto con los resultados de las evaluaciones que demuestran su utilidad. A su vez, se tiene planificado continuar con las evaluaciones de las restantes actividades de la metodología y con las tareas de mejora continua de la solución.

References

- Bencomo, N., Götz, S. and Song, H. (2019). Models@run.time: A Guided Tour of the State of the Art and Research Challenges. *Software & Systems Modeling*, 18(5):3049-3082.
- Chen, X. et al. (2015). Runtime Model Based Approach to IoT Application Development. *Frontiers of Computer Science*, 9(4):540-553.
- Elhabbash, A., Salama, M., Bahsoon, R. and Tino, P. (2019). Self-Awareness in Software Engineering: A Systematic Literature Review. *ACM Transactions on Autonomous and Adaptive Systems*, 14(2):1-42.
- Esterle, L. and Brown, J. (2020). I Think Therefore You Are: Models for Interaction in Collectives of Self-Aware Cyber-Physical Systems. *ACM Transactions on Cyber-Physical Systems*, 4(4):1-25.
- Esterle, L. and Grosu, R. (2016). Cyber-physical Systems: Challenge of the 21st Century. *e & i Elektrotechnik und Informationstechnik*, 133(7):299-303.
- Esterle, L. and Rinner, B. (2018). An Architecture for Self-Aware IoT Applications. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6588-6592.
- García, F. et al. (2004). Una Ontología de la Medición del Software. Universidad de Castilla-La Mancha.
- Gorschek, T., Garre, P., Larsson, S. and Wohlin, C. (2006). A Model for Technology Transfer in Practice. *IEEE Software*, 23(6):88-95.
- Gotzinger, M. et al. (2020). RoSA: A Framework for Modeling Self-Awareness in Cyber-Physical Systems. *IEEE Access*, 8:141373-141394.
- Gurgen, L., Gunalp, O., Benazzouz, Y. and Gallissot, M. (2013). Self-Aware Cyber-Physical Systems and Applications in Smart Buildings and Cities. In *2013 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 1149-1154.
- Kounev, S. et al. (2017). The Notion of Self-Aware Computing. In *Self-Aware Computing Systems*, pages 3-16. Springer.
- Lewis, P. et al. (2017). Towards a Framework for the Levels and Aspects of Self-Aware Computing Systems. In *Self-Aware Computing Systems*, pages 51-85. Springer.
- Madakam, S., Ramaswamy, R. and Tripathi, S. (2015). Internet of Things (IoT): A literature review. *Journal of Computer and Communications*, 3(5):164-173.
- Miorandi, D., Sicari, S., De Pellegrini, F. and Chlamtac, I. (2012). Internet of Things: Vision, Applications and Research Challenges. *Ad Hoc Networks*, 10(7):1497-1516.
- Moody, D. (2001). *A Practical Method for Representing Large Entity Relationship Models*. PhD Thesis. University of Melbourne, Australia.
- Sosa-Reyna, C., Tello-Leal, E. and Lara-Alabazares, D. (2018). Methodology for the Model-Driven Development of Service Oriented IoT Applications. *Journal of Systems Architecture*, 90:15-22.
- Tammemäe, K. et al. (2017). Self-Aware Fog Computing in Private and Secure Spheres. In *Fog Computing in the Internet of Things*, pages 71-99. Springer.