World Journal of Information Systems 1(2) 1-15
DOI: https://doi.org/10.17013/wjis.v1i2.9

Article

Model-Driven Design and Simulation of IoT Systems Featuring Mobile Nodes

José A. Barriga * [1], Pedro J. Clemente [1], Miguel A. Pérez-Toledano [1] and Arturo Barriga [1]

[1] Quercus Software Engineering Group. http://quercusseg.unex.es. Department of Computer and Telematic Systems Engineering. Universidad de Extremadura.

***** Correspondence: jose@unex.es

(Article history: Received March 05, 2024; Received in revised form April 30, 2024; Accepted May 20, 2024; Available online June 15, 2024)

Abstract: The Internet of Things (IoT) has extended its reach into numerous domains, including smart cities, agriculture, and industrial systems, where the mobility of devices plays an increasingly critical role. However, mobility brings significant challenges such as packet loss, increased latency, and higher energy consumption, all of which are exacerbated by the lack of a standardized mobility management framework for key IoT protocols like MQTT and CoAP. To address these issues, this paper presents a Model-Driven Development (MDD) methodology aimed at simplifying and streamlining the design of IoT systems. Central to this approach is the use of pre-deployment simulations to identify and address potential design issues. The proposed methodology integrates a Domain-Specific Language (DSL) into the Simulate IoT platform, enabling the design, code generation, and simulation of IoT systems with a focus on mobility. This framework supports both mobile and stationary devices while incorporating fog and cloud infrastructure. To illustrate its practical application, a case study is presented involving a drone-based forest fire detection system. This example demonstrates how MDD and simulation can effectively reduce the complexity of mobility management in IoT systems while improving development efficiency.

Keywords: IoT systems; IoT simulation; Model-driven development; Mobile nodes; IoT mobility.

1. Introduction

The IoT is utilized across a variety of sectors, including smart cities, residential areas, agriculture, industrial operations, and intelligent buildings, highlighting its wide-ranging applications and the unique requirements of each domain [1]. A critical feature of IoT

© 2024 ITMA. All rights reserved.

ecosystems is device mobility [2], essential for scenarios involving mobile elements like personal transportation devices (e.g., bicycles, scooters) or GPS-equipped livestock. Similarly, manufacturing and industrial environments frequently rely on mobile IoT devices to support dynamic operations within factories [3].

Managing mobility in IoT systems introduces significant challenges. Device movement can disrupt connectivity, leading to packet loss [4], increased delays, or jitter [5], particularly when devices move beyond gateway ranges or transition between gateways, introducing network variability. Additionally, energy consumption becomes a critical concern in mobile IoT systems due to limited battery capacity and the significant energy demands associated with mobility [6]. Furthermore, popular IoT communication protocols, such as MQTT and CoAP, lack inherent mobility management features, further complicating the management of these systems.

To address these challenges, robust strategies and thorough testing are essential. This paper introduces Simulate IoT-Mobile [7], an enhanced version of the Simulate IoT framework, developed using a MDD [8] approach. This framework facilitates the design, simulation, and automated code generation of IoT systems, with a specific focus on mobility testing. MDD provides high-level abstractions that simplify system modeling and validation, reducing the reliance on specific coding or frameworks. By leveraging this domain-focused approach, Simulate IoT-Mobile improves the efficiency and effectiveness of the development and testing process.

Simulate IoT-Mobile builds upon the capabilities of Simulate IoT by incorporating features designed specifically for simulating the behavior of mobile nodes in IoT systems. These enhancements provide developers with tools to evaluate the dynamic aspects of mobility, offering a more comprehensive solution for IoT mobility challenges.

The primary contributions of this work are as follows:

1. **Validation of MDD in Complex IoT Environments:** Demonstrates the effectiveness of MDD in addressing the complexities of IoT systems, particularly with regard to mobility challenges.

2. **Enhanced Model-to-Text (M2T) Transformations:** Introduces transformations to simulate mobile device dynamics, including logic for managing gateway

© 2024 ITMA. All rights reserved.

connections and disconnections, battery consumption, and communication delays or jitter.

3. **Support for IoT Architectures with Mobile Devices:** Extends the framework to include architectural components required to support mobile devices.

4. **Case Study Validation:** Applies the Simulate IoT-Mobile framework to a drone-based forest fire detection system, illustrating its practical application and validating its capabilities in real-world scenarios.

These contributions highlight the significance of incorporating mobility considerations into MDD methodologies, providing an effective framework for managing and simulating mobile devices within IoT ecosystems.

The manuscript is organized as follows: Section 2 reviews related works to provide context and highlight the research gap. Section 3 introduces an overview of Simulate IoT-Mobile, explaining its key features and enhancements. Section 4 describes the technical details of extending Simulate IoT to incorporate mobility-related functionalities, including support for mobile nodes and IoT architectures. Section 5 presents a practical use case involving a drone-based forest fire detection system, demonstrating the application and validation of Simulate IoT-Mobile. Finally, Section 6 concludes the manuscript by summarizing the contributions and outlining potential directions for future research.

2. Related Works

Addressing device mobility in IoT environments has been approached from multiple angles, including enhancing communication protocols to incorporate mobile-specific features such as battery optimization, Quality of Service (QoS), and latency reduction, as well as developing high-level frameworks to manage mobility more efficiently.

For instance, mobility-enabling protocols based on 6LoWPAN technology (e.g., MIPv6, HMIPv6, ZoroMSN, LoWMob) focus on optimizing QoS, resource management, security, and routing. However, existing research highlights a gap in comprehensive solutions that fully address the diverse demands of Wireless Sensor Networks (WSNs) utilizing 6LoWPAN technology. This shortfall has led to the continuous development of

© 2024 ITMA. All rights reserved.

new communication protocols tailored to the resource constraints of IoT and WSN environments, with varying degrees of success. While these efforts emphasize the continued relevance of 6LoWPAN, the wide-ranging requirements of IoT systems underscore the need for further advancements.

Innovative approaches, such as those comparing classical and bio-inspired mobility protocols in WSNs [10], show potential for improving network performance metrics like congestion and latency. Yet, practical real-world implementations of these protocols remain a significant research challenge. Meanwhile, widely used IoT communication protocols, including MQTT, CoAP, and DDS, are recognized for their strong performance but lack native mechanisms for effectively managing mobile nodes. Solutions like intermediate buffering have been proposed to address issues such as data loss during disconnections or broker migrations in MQTT environments [11]. However, these solutions have not been tested extensively on a large scale and offer only partial resolutions to mobility challenges.

In response to these limitations, this paper introduces SimulateIoT-Mobile, a novel framework designed to support the development of IoT systems with mobile nodes, particularly those using the MQTT protocol. This tool enables developers to model, validate, generate, and simulate IoT systems incorporating mobile devices. By leveraging simulation results, developers can refine their systems, uncover potential design flaws, and address inefficiencies, thereby providing a robust foundation for mobility-focused IoT development.

3. Simulate IoT-Mobile Overview

Simulate IoT-Mobile builds upon the foundation of Simulate IoT [12] by introducing advanced capabilities specifically designed to integrate mobile devices into IoT simulations. This extension leverages an MDD approach, applying metamodeling techniques to enhance the abstraction level in IoT system design and simulation. By relying on high-level models, the framework guides the software development process, enabling the automatic generation of application code through M2T transformations. This methodology not only minimizes errors but also promotes technological independence, streamlining the development process.

© 2024 ITMA. All rights reserved.

The integration of mobile device capabilities into Simulate IoT required modifications across three key metamodeling layers. These include extending the Metamodel (Abstract Syntax) to define the structural elements for mobile devices, enhancing the Graphical Concrete Syntax to allow users to visually design mobile device models, and augmenting the M2T transformations to facilitate the automatic generation of application code from these models.

As the original Simulate IoT framework lacked native support for mobile devices, this extension necessitated the development of a comprehensive MQTT mobility model. This model enables the simulation of mobile device behavior, incorporating aspects such as movement patterns, routing logic, and battery consumption. These functionalities are essential for accurately modeling IoT environments with mobile components. Simulate IoT-Mobile extends the Edge/Mist, Fog, and Cloud layers of the original framework to accommodate mobile connectivity, simulate data publication to multiple brokers, and track battery usage. Additional features have also been incorporated to enhance MQTT protocol support for mobile device operations.

The enhancements introduced in Simulate IoT-Mobile are described in detail in the following section, covering the expansions made to the Metamodel, the Concrete Syntax, and the M2T transformations. Furthermore, a case study is provided to illustrate the framework's new simulation capabilities, demonstrating its contributions and advancements beyond the original Simulate IoT platform.

4. Simulate IoT-Mobile: Metamodel, Concrete Syntax and Model-to-Text Transformations

*4.1 Simulate*IoT-Mobile Metamodel

A metamodel provides an abstract representation of key concepts and their relationships within a specific domain, serving as a foundational framework for modeling aspects of reality. As Selic [8] describes, metamodels enable the derivation of concrete models that support the generation of application code, either partially or fully. This process enhances adaptability to different technologies while reducing the likelihood of errors in system development.

© 2024 ITMA. All rights reserved.

The original SimulateIoT metamodel, as detailed in [12], defines the essential elements and their interconnections within the IoT domain. It includes components such as sensors, actuators, edge nodes, fog nodes, cloud nodes, databases, complex-event processing services, data definitions, topics, and message brokers. While comprehensive, it lacks the capabilities required to model IoT systems with mobile nodes, a significant limitation in addressing the challenges of mobility in IoT environments.

To overcome this shortfall, the SimulateIoT-Mobile metamodel was introduced [7]. This extension builds upon the foundational SimulateIoT metamodel by adding constructs specifically designed to represent the dynamics of IoT systems incorporating mobile nodes. Although this metamodel has been previously discussed in [7], a detailed description is included here for the sake of completeness.

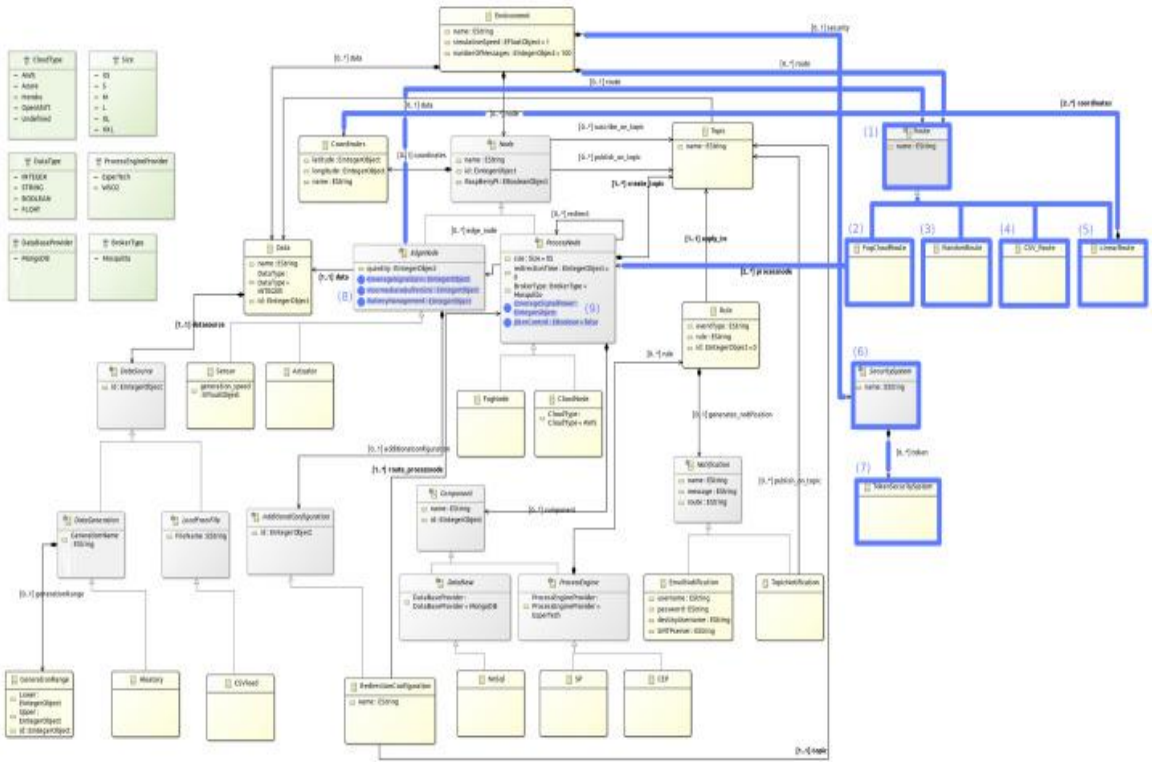


Figure 1. Simulate IoT-Mobile metamodel. The elements highlighted in blue are those included as part of this work.

© 2024 ITMA. All rights reserved.

An excerpt of the SimulateIoT-Mobile metamodel, emphasizing elements critical to modeling mobile IoT devices, is shown in Figure 1. These extensions are highlighted in blue for clarity. The new features include the necessary classes and relationships to represent mobility-related entities and services. Key additions are:

1. **Route modeling:** Defines paths that mobile devices can traverse (entities 1, 2, 3, 4, and 5).

2. **Token-based security system:** Introduced to manage security aspects related to IoT mobility.
3. **Signal properties:** Includes attributes for modeling signal gain and signal power for mobile nodes (entity 8) and gateways (entity 9).
4. **Battery management module:** Simulates energy consumption of mobile devices (entity 9).
5. **Intermediate buffering:** Reduces packet loss during mobility scenarios.
6. **Jitter control:** Simulates and mitigates data jitter.

These enhancements significantly improve the expressiveness of the metamodel, enabling more accurate simulations of IoT systems with mobile nodes while addressing critical aspects such as connectivity, security, and energy management.

4.2 SimulateIoT-Mobile Concrete Syntax

To improve the modeling of IoT environments, a Graphical Concrete Syntax (graphical editor) has been developed using the Eugenia tool. This enhancement enables intuitive model design and visualization directly aligned with the SimulateIoT metamodel. Built on the Eclipse Graphical Modeling Framework (GMF) and the Eclipse Modeling Framework (EMF), the graphical editor provides a robust platform for creating and managing models.

These models adhere to the EMF standards and are validated using the Object Constraint Language (OCL), ensuring compliance with the metamodel's specifications. This rigorous validation process guarantees consistency and accuracy throughout the modeling workflow. An example of a model created with this graphical editor is shown in Figure 3, demonstrating its practical application in designing IoT systems. This

© 2024 ITMA. All rights reserved.

visualization highlights the editor's role in simplifying and streamlining the process of constructing complex IoT models.

4.3 SimulateIoT Design and Implementation phase. Model-to-Text Transformations

After successfully defining and validating models conforming to the SimulateIoT-Mobile metamodel, a M2T transformation process, powered by Acceleo, automates the generation of a comprehensive set of software artifacts. This process produces an integrated software infrastructure that includes MQTT messaging brokers (leveraging the MQTT protocol), the necessary device infrastructure, an architecture tailored to support IoT device mobility, databases, a graphical analysis platform, a stream processing engine, and additional components. All of these elements are packaged as Docker containers for seamless deployment.

In the deployment phase, these generated artifacts are systematically deployed and orchestrated to create the operational simulation environment. This phase involves configuring and integrating the various components to ensure proper functionality and interoperability, resulting in a fully operational and cohesive system.

Figure 2 illustrates the deployment architecture of a generic SimulateIoT-Mobile IoT environment. The diagram highlights the strategic arrangement and interconnection of the generated components, providing a clear depiction of the system’s structure and the interaction dynamics among its elements. This visual representation emphasizes the organized approach to deployment, ensuring a well-coordinated simulation environment.

© 2024 ITMA. All rights reserved.

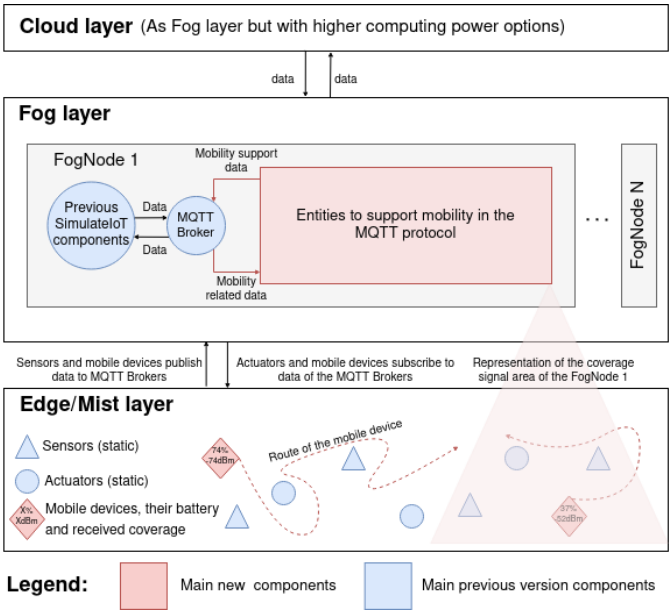


Figure 2. Example of deploy diagram. The main components are identified by numbers in the figure.

5. Case Study. A Drone-based Forest Fire Detection System

This section demonstrates the modeling and simulation of a drone-based forest fire detection IoT environment using the tools provided by Simulate IoT-Mobile.

The system employs drones for environmental monitoring, specifically aimed at detecting fires. These drones collect data on smoke particle concentrations and their locations. Once a drone's storage capacity is reached, the collected data is offloaded to the nearest gateway. The data is then processed by a Complex Event Processing (CEP) engine, which triggers an alert to notify the fire warden if smoke levels exceed a threshold of 100. This setup exemplifies an automated, efficient approach to early fire detection and response, leveraging advanced IoT technologies to enhance environmental surveillance. In this scenario, drones complete their monitoring and data collection tasks in approximately ten minutes. For longer missions, it is recommended to equip drones with internal CEP engines to ensure uninterrupted, real-time data analysis, given the critical nature of their tasks.

© 2024 ITMA. All rights reserved.

To model this IoT system, the following components and configurations are considered:

1. **Forest Division:** The forest is divided into five distinct areas: four based on cardinal points (north, south, east, west) and a central area where the fire warden's base is located.

2. **Fog Nodes:** Each area is equipped with a Fog node, functioning as a gateway for drones to offload their collected data.

3. **Drones:** Two drones are deployed in each area, flying and collecting environmental data, which they publish to the corresponding Fog node.

4. **CEP Engines:** Each Fog node hosts a CEP engine designed to analyze the offloaded data for smoke particles. If smoke is detected, an event is generated to signal a potential fire to the fire warden.

5. **Fire Warden System:** The central fire warden's base is equipped with an actuator—a fire alarm—designed to notify the firekeepers in the event of a fire.

This use case illustrates how Simulate IoT-Mobile enables the design and simulation of complex IoT environments, emphasizing the importance of integrating mobility and edge-based analytics for real-time, mission-critical applications like forest fire detection.

5.1 Case Study. Model Definition

Figure 3 provides an excerpt of the IoT model, focusing on the architectures of the West Fog Node and the Firekeeper Ward Fog Node for clarity. The architectures for the North, East, and South Fog Nodes are identical to that of the West Fog Node and are therefore omitted from the figure.

The West Fog Node and the Firekeeper Ward Fog Node share a similar structure, with one key difference: the Firekeeper Ward Fog Node includes a Fire Alarm Topic and a Fire Alarm Actuator to handle fire alerts. To describe this IoT model, the shared architecture is addressed first.

In the depicted system architecture, each Fog node is equipped with specialized drones for environmental smoke data collection. The West Fog Node operates with West Drone

© 2024 ITMA. All rights reserved.

1 and West Drone 2, while the Firekeeper Ward Fog Node manages Firekeeper Area Drone 1 and Firekeeper Area Drone 2. These drones gather smoke data from their assigned areas and publish the information to specific Topics for processing. For instance, the drones associated with the West Fog Node publish their data to the West/Drone Data Topic on the West Fog Node, while the Firekeeper Ward Fog Node drones publish to the Drone Data Topic on the Firekeeper Ward Fog Node.

The drone-generated data is synthetically created based on predefined Data Generation Strategy elements, as shown in Figure 3. The paths followed by each drone are outlined in CSV Route elements, which contain the geographic coordinates of the routes for each drone.

The collected data is subsequently analyzed by CEP engines deployed on each Fog node. When a CEP engine detects smoke patterns in the data, it sends an alert to the Fire Alarm Topic hosted on the Firekeeper Ward Fog Node. This alert is then captured by the Fire Alarm Actuator, which subscribes to the Fire Alarm Topic. Upon receiving an alert, the actuator triggers a protocol to notify firekeepers, specifying the location of the detected fire to enable a rapid and precise response.

This IoT system demonstrates an effective and automated approach to forest fire detection and management, leveraging IoT technologies to provide swift and informed responses to potential fire threats. By incorporating features such as real-time data processing, targeted alerts, and drone-based surveillance, this system enhances the efficiency of forest fire monitoring and mitigation efforts.

© 2024 ITMA. All rights reserved.

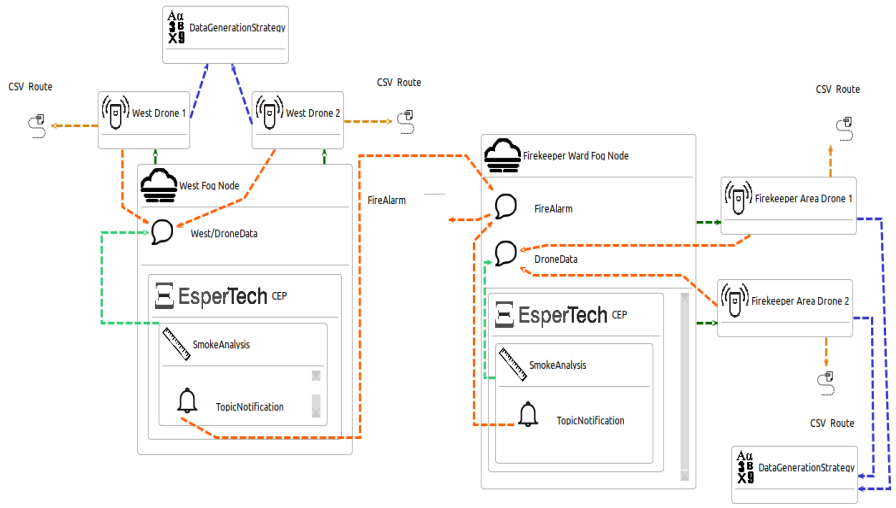


Figure 3. Case study. A drone-based forest fire detection IoT system model conforms to the SimulateIoT-Mobile metamodel.

5.2 Case Study. Code Generation and Deployment

After the model has been defined and validated, the M2T transformation process is executed with the following objectives:

1. **Generate Device Behavior Code:** Java and Python code is generated to represent the operational logic of each device. This includes key functionalities such as data generation, data publication, route following, and the simulation of battery consumption, ensuring the software accurately reflects the modeled behaviors.

2. **Create Component Configuration Code:** Configuration scripts are produced to set up various system components, such as message brokers. These scripts define the topics specified in the model and generate configuration files or deployment scripts for stream processors included in the model.

3. **Generate Docker Configuration Files:** For each Fog Node and Edge Node (such as drones and the fire alarm in this use case), a Docker container configuration file is generated to encapsulate the necessary software environment.

© 2024 ITMA. All rights reserved.

4. **Generate Deployment Scripts:** A deployment script is created to automate the simulation execution. This script handles actions such as compiling generated components, packaging them into Docker containers, and orchestrating their deployment. Using Kubernetes, these containers are distributed across the network, enabling scalable and flexible simulation deployment.

By executing the simulation based on this transformed model, a thorough analysis of the IoT environment can be conducted before its real-world implementation. This approach enables prior validation of the system's functionality and performance within a controlled and replicable environment, mitigating risks and identifying potential issues early in the development cycle.

6. Conclusions

This study presents a Model-Driven Development (MDD) approach specifically designed for the Internet of Things (IoT) domain, with a focus on simulations that integrate mobile nodes. By leveraging established metamodeling techniques, the approach facilitates the creation and validation of models that adhere to the Simulate IoT-Mobile metamodel. These validated models are then transformed into executable code through an automated M2T transformation process, enabling the deployment of the defined IoT simulations.

The tools and methodology introduced empower users to design, evaluate, and refine various IoT system architectures and policies. This iterative process allows for the development of optimized IoT systems with mobile nodes, culminating in the deployment and comprehensive analysis of the modeled systems. These analyses provide valuable insights into the system’s performance and areas for potential improvement.

By offering a structured, MDD-based framework, this approach significantly contributes to the IoT field, supporting the systematic design, evaluation, and enhancement of complex IoT systems that incorporate mobile node functionality.

Acknowledgments: This work was supported by the Government of Extremadura, Council for Economy, Science, and Digital Agenda under grant GR21133 and project IB20058, as well as by the European Regional Development Fund (ERDF). Additional funding was provided by Project TED2021-129194B-I00, funded by

© 2024 ITMA. All rights reserved.

MICIU/AEI/10.13039/501100011033 and "ERDF A way of making Europe" by the European Union NextGenerationEU/PRTR. Support was also provided by the Cátedra Telefónica at the Universidad de Extremadura (Red de Cátedras Telefónica).



Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

[1] Eugene Siow, Thanassis Tiropanis, and Wendy Hall. "Analytics for the Internet of Things: A survey." *ACM Computing Surveys (CSUR)*, vol. 51, no. 4, 2018, p. 74.

[2] Safwan M. Ghaleb, Shamala Subramaniam, Zuriati Ahmed Zukarnain, and Abdullah Muhammed. "Mobility management for IoT: a survey." *EURASIP Journal on Wireless Communications and Networking*, vol. 2016, no. 1, 2016, pp. 1–25.

[3] Francisco Almada-Lobo. "The Industry 4.0 revolution and the future of Manufacturing Execution Systems (MES)." *Journal of Innovation Management*, vol. 3, no. 4, 2015, pp. 16–21.

[4] Muneer Bani Yassein, Shadi Aljawarneh, and Walaa Al-Sarayrah. "Mobility management of Internet of Things: Protocols, challenges and open issues." *2017 International Conference on Engineering & MIS (ICEMIS)*, IEEE, 2017, pp. 1–8.

[5] Jorge E. Luzuriaga, Juan Carlos Cano, Carlos Calafate, Pietro Manzoni, Miguel Perez, and Pablo Boronat. "Handling mobility in IoT applications using the

© 2024 ITMA. All rights reserved.

MQTT protocol." *2015 Internet Technologies and Applications (ITA)*, IEEE, 2015, pp. 245–250.

[6] Xilong Liu and Nirwan Ansari. "Toward green IoT: Energy solutions and key challenges." *IEEE Communications Magazine*, vol. 57, no. 3, 2019, pp. 104–110.

[7] José A. Barriga, Pedro J. Clemente, Miguel A. Pérez-Toledano, Elena Jurado-Málaga, and Juan Hernández. "Design, code generation and simulation of IoT environments with mobility devices by using model-driven development: SimulateIoT-Mobile." *Pervasive and Mobile Computing*, vol. 89, 2023, p. 101751. ISSN: 1574-1192. DOI: https://doi.org/10.1016/j.pmcj.2023.101751.

[8] Bran Selic. "The pragmatics of model-driven development." *IEEE Software*, vol. 20, no. 5, 2003, pp. 19–25.

[9] Maha Bouaziz and Abderrezak Rachedi. "A survey on mobility management protocols in Wireless Sensor Networks based on 6LoWPAN technology." *Computer Communications*, vol. 74, 2016, pp. 3–15.

[10] Ranida Hamidouche, Zibouda Aliouat, Abdelhak Mourad Gueroui, Ado Adamou Abba Ari, and Lemia Louail. "Classical and bio-inspired mobility in sensor networks for IoT applications." *Journal of Network and Computer Applications*, vol. 121, 2018, pp. 70–88.

[11] J. E. Luzuriaga, J. C. Cano, C. Calafate, P. Manzoni, M. Perez, and P. Boronat. "Handling mobility in IoT applications using the MQTT protocol." *2015 Internet Technologies and Applications (ITA)*, IEEE, 2015, pp. 245–250.

[12] José A. Barriga, Pedro J. Clemente, Encarna Sosa-Sánchez, and Álvaro E. Prieto. "SimulateIoT: Domain Specific Language to Design, Code Generation and Execute IoT Simulation Environments." *IEEE Access*, vol. 9, 2021, pp. 92531–92552. DOI: 10.1109/ACCESS.2021.3092528

© 2024 ITMA. All rights reserved.