# Investigation of Modeling Techniques in Wireless Sensor Network Design

**Raheleh Ilaghi Hosseini**
**Department of Computer Engineering, Kerman Branch, Islamic Azad University, Kerman,Iran**

**Reyhaneh Ilaghi Hosseini**
**Department of Computer Engineering, Kerman Branch, Islamic Azad University, Kerman,Iran**

**Mokhtar Mohammadi Ghanatghestani**
**Department of Computer Engineering, Kerman Branch, Islamic Azad University, Kerman,Iran**

## Abstract
Wireless sensor networks (WSNs) have experienced significant growth in recent years and hold great potential in various fields such as health, environment, and military. Despite their remarkable capabilities, the successful development of WSN remains a challenging endeavour. In current real-world WSN deployments, numerous programming approaches have been suggested, with a focus on addressing low-level system issues. To simplify WSN design and abstract from technical low-level details, high-level approaches have been recognized and several solutions have been proposed. Particularly, the model-driven engineering (MDE) approach is emerging as a promising solution. This paper presents an investigation of low level and high level methodologies for sensor network development. The primary objective of this research is to explore the feasibility and application of high-level-based approaches to streamline WSN design. The study concentrates on a set of criteria to highlight the shortcomings of relevant approaches, and finally, presents future directions to address the limitations of existing solutions.

**Keywords**: Wireless sensor network, Model driven engineering, low level abstraction ,high level abstraction

**Introduction**

WSNs are now an essential component of various applications, including environmental monitoring [1], military surveillance [2], and medicine [3]. They enable effective communication, reliable inspection, and the execution of various tasks. These networks consist of numerous sensor nodes that are densely distributed and communicate wirelessly to transmit and receive environmental data. Each sensor node is equipped with sensors, a radio transceiver, a processor, and a power supply. Developing WSNs is a complex task, with various requirements to meet, such as power consumption. Many current studies are dedicated to exploring WSNs. Some authors have discussed the concept, generations, routing, architecture, and storage management of WSNs. Other researchers [4-7] provide an overview of routing protocols and existing middleware for sensor networks. Additionally, authors [8-10] have reviewed sensor localization techniques and presented new schemes for IoT (Internet of Things) infrastructure. Some works have focused on modeling techniques for WSNs, describing node and network behavior [11]. However, there is a lack of research on programming approaches and modeling techniques used in WSN development. Our research focuses on exploring the feasibility and application of high-level-based approaches to simplify the development of WSN systems and enhance their maintainability and portability. Currently, there is a growing interest in high-level abstraction design using model-driven engineering (MDE) methodology, particularly through the use of standard mechanisms such as the Modeling and Analysis of Real Time and Embedded Systems (MARTE) profile and design patterns. Representing WSN systems at higher abstraction levels reduces complexity, increases reusability and flexibility of models, automates processes, and improves model quality. Additionally, it allows for early error detection before network deployment. This paper assesses various research works on WSN development, starting with low-level-based approaches and then discussing the need for appropriate high-level design methods. The classification of WSN design approaches is illustrated in Figure 1, and a set of comparison criteria related to design environment, power supply design, reconfiguration scenario, and performance evaluation is considered.

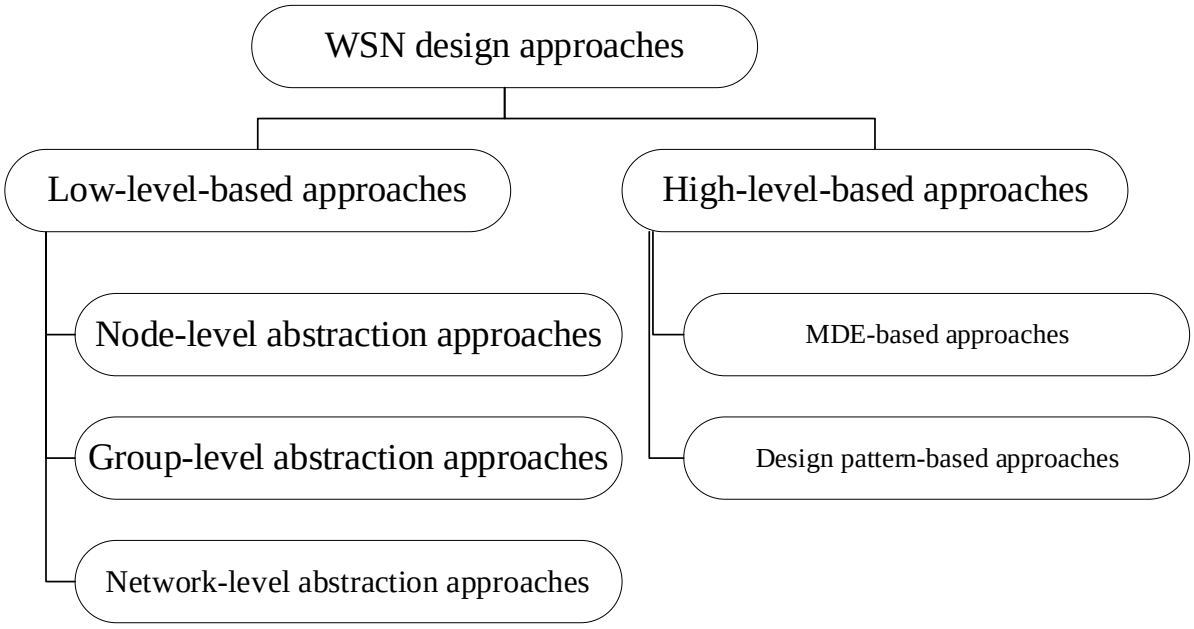


**Figure (1) WSN design approaches**

**Comparison Principles**

We concentrate on a set of principles for comparing low level and high level methods as shown in Figure 2. These criteria are categorized into four groups namely design environment, power supply design, reconfiguration scenario, and non-functional property (NFP) verification. Our study emphasizes modeling constraints and requirements typically related to energy efficiency and reconfiguration, which are of interest for our future research.

**Design environment:** This group encompasses the design abstraction level. We aim to determine whether the approach is MDE-based or developed at a lower abstraction level. This group establishes modeling standards and techniques. Our focus is on the utilization of the UML/MARTE standard, as well as the definition and application of WSN design patterns.

**Power supply design:** This group centers on modeling the WSN power supply section. It is a crucial criterion to consider as power resources in WSN applications and a primary concern [12,13]. This group covers the support of a typical power supply section using a local battery, as well as the modeling of an energy harvesting power supply unit. Energy harvesting is a promising solution to meet network energy requirements.

**Reconfiguration scenario:** This group concentrates on the level and structure of reconfiguration engines. It includes two reconfiguration scenarios and the MAPE (Monitor, Analyzer, Planner, and Executor) loop modules. Two reconfiguration scenarios are commonly adopted in WSN applications. The first scenario is node-level-based reconfiguration, involving hardware and software reconfiguration [14]. The second scenario is network-level-based reconfiguration, which involves modifying the network topology. The general structure of the reconfiguration engine is based on the MAPE loop [15].

**NFP verification:** This group focuses on supporting temporal verification and performance evaluation. It includes works that utilize transformations from system models to simulation or analysis tool models in order to perform verification.

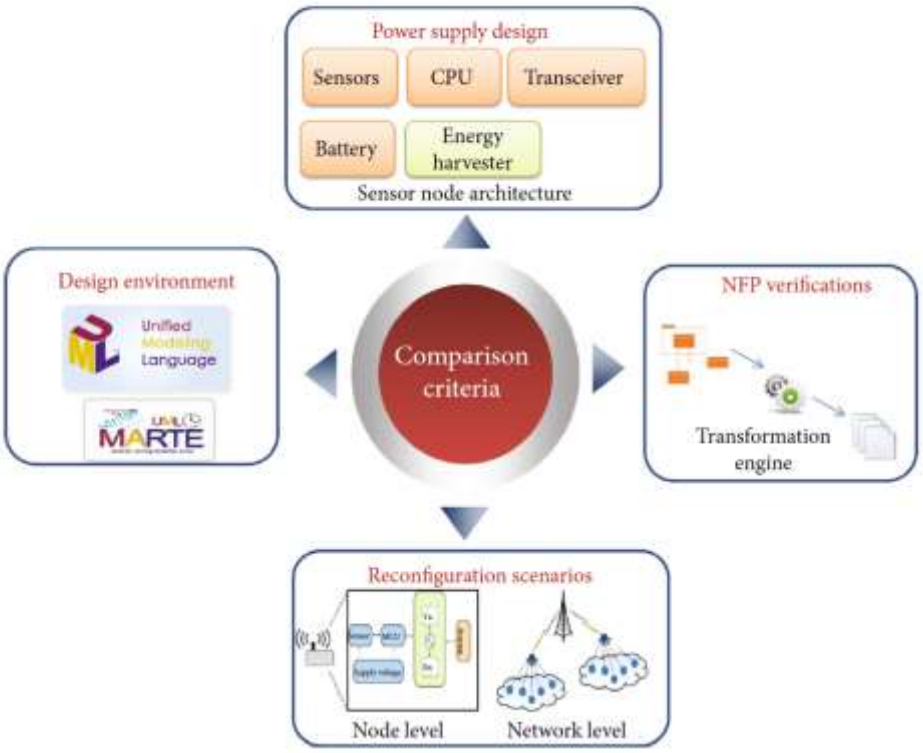


**Figure (2) Comparison criteria**

**Low-Level-Based Approaches for WSN Development**

Numerous research studies have been studied aimed at the development and design of WSN systems and their associated requirements. The work can be categorized into node-level, network-level, and group-level abstraction approaches.

**1. Node-Level Abstraction Approaches**

Node-level abstraction involves the decomposition of the overall application into a series of localized behaviors for each node, where specific code is executed on every individual node. This methodology frequently employs virtual machine approaches and embedded operating systems (OSs). Programming based on operating systems emphasizes the abstraction of hardware, thereby enabling flexible management of the hardware resources utilized in WSN applications. TinyOS [16], which is implemented using the nesC programming language [17], and Contiki [18] are prominent operating systems utilized for node-level abstraction in WSNs. While both operating systems provide to

the requirements of WSNs, they exhibit distinct characteristics. TinyOS is particularly advantageous in scenarios with limited resources and low power applications, whereas Contiki excels in situations where flexibility is paramount. The advancement of operating systems for wireless sensor devices is regarded as a crucial component of IoT systems [19]. Several operating systems have been proposed in this context, including RIOT [20], Mbed OS [21], and FreeRTOS [22], all of which provide preventive priority-based task scheduling, comprehensive real-time support, and facilitate the development of IoT applications. Mate [23] is a recognized virtual machine within the realm of WSNs, focusing on the necessity for innovative programming examples to address the constraints of WSNs, such as limited energy resources and bandwidth. Although a virtual machine approach allows for dynamic reprogrammability, it is hindered by the overhead introduced by the instructions. Various simulation environments are employed for WSN research [24, 25], with the NS-3 simulator being the most frequently referenced. NS-3 [26, 27], developed in C++, also offers an optional scripting interface in Python [28]. In addition to its scalability, NS-3 provides a robust platform for WSN simulation.

## 2. Group-Level Abstraction Approaches

Each node linked to a group-level programming entity is regarded as a neighbor node for other nodes within the same network. When nodes are organized based on their physical proximity—such as geographical distance or the number of communication hops—the resulting collection is termed a neighborhood-based group. Examples of such neighborhood programming abstractions for WSNs include Hood and Abstract Region. These algorithms facilitate local data processing within a neighborhood context. Hood aids in the design of distributed algorithms by utilizing the neighborhood abstraction, employing data sharing to enhance scalability and collaboration. It incorporates a caching technique to conserve energy, minimize communication failures among nodes, and utilizes mirroring for time synchronization. Abstract Region offers interfaces for identifying neighboring nodes, facilitating data sharing, and enabling data reduction within local neighborhoods. Both power consumption and scalability are enhanced through data sharing. In the case of Hood, the caching technique is instrumental in mitigating network failures. Additionally, it adapts to varying network requirements and conditions, accommodating different levels of energy and bandwidth utilization while achieving the desired accuracy in shared operations. When a group is formed based on logical attributes—such as node type or sensor input—it is referred to as a logical group. An example of a logical-based group abstraction is EnviroTrack, an application specifically designed for target tracking, where nodes detecting the same event are clustered together. Similar to Hood and Abstract Regions, EnviroTrack offers data sharing and aggregation capabilities to meet the demands of WSNs. However, it excels in more dynamic scenarios, providing superior support. The SPIDEY language was proposed as a logical-based group, where nodes are grouped based on shared properties. ZigBee[29] technology is widely used for mesh network topology, providing secure, flexible, scalable, and reliable communication. Network-level abstraction approaches treat the entire sensor network as a single system, with TinyDB [7] viewing it as a database system and focusing on energy efficiency.

## 3. Network-Level Abstraction Approaches

In the network-level abstraction, the entire sensor network is considered as a unified system that describes the overall behavior. TinyDB [7] treats the entire network as a database system, enabling users to issue queries in a declarative SQL-like language. To enhance energy efficiency, TinyDB focuses on determining when, where, and how to sample and transmit data. Additionally, TinyDB optimizes the routing tree for query dissemination and result collection. However, there are limitations to database abstraction, such as the restriction to only one accessible table at a time, which is not ideal for heterogeneous sensors. Regiment [11] is a functional language specifically designed for macroprogramming sensor networks, allowing the direct utilization of a program state to represent the discovery of each individual node. Another example of programming abstraction is Kairos [30], which enables macroprogramming and utilizes a caching technique to minimize power consumption and communication.

As a concequence, From the low-level approaches, it is evident that the approaches provide substantial support for various WSN attributes, including operating systems, power sources, communication capabilities, and issues related to reconfiguration. Currently, WSN design is primarily conducted at the implementation level, which contributes to the increased complexity of these systems due to their dependence on specific platforms. By elevating the level of abstraction, designers can better manage the growing complexity.

## High-Level-Based Approaches for WSN Development

Modeling techniques and languages are utilized in WSN design to operate at higher abstraction levels, streamline analysis steps, and address issues prior to deployment. The primary reason for employing high-level-based approaches is to create new applications with less effort compared to traditional methods. This section outlines existing component-based approaches for modeling WSN and also discusses WSN approaches based on the MDE methodology. The utilization of modeling techniques and languages in WSN design allows for higher abstraction

levels, simplifies analysis steps, and resolves issues before deployment. The main reason for using high-level-based approaches is to create new applications with less effort than traditional methods.

## 1. MDE-Based Approaches for WSN

The MDE model, based on exploiting representations to address the complexity of embedded systems, has significantly contributed to the development life cycle of embedded systems in various fields such as WSN design and development, energy supply designs for WSN, and self-adaptive system development [31-34]. This approach increases the abstraction level of development, reduces errors by enabling system analysis at an early design stage, and simplifies code generation. Additionally, the MDE is based on several basics and concepts, including the model, the metamodel, the model transformation concept, and the UML extension mechanisms. Researchers have proven the effectiveness of the MDE paradigm in reducing the design complexity of WSN applications through its principles of abstraction, separation of concerns, reuse, and automation. The MDE-based framework methodology proposed in [34] focuses on energy consumption analysis and provides three modeling approaches for WSN design.
Extended the MARTE profile by a set of stereotypes in order to allow the specification of complementary information for WSN features, such as the Sensor stereotype extending «HwSensor» to map sensors and their characteristics and the Synchronize stereotype extending «Synchronization Resource » used to model synchronization between sensor nodes. In addition, it supports the Communication stereotype extending «SaCommStep» to transmit/receive messages. To represent nodes, the WiSeN profile uses the Node stereotype extending «ResourceUsage». It uses the MsgPackage stereotype extending «MessageComResource» to give information about the structure of message. The WiSeN profile contains other information that does not exist in the MARTE standard. It uses the Gateway stereotype to assist in the communication with the external system. The WiSeN profile can be refined and extended to address the power supply section where the limited energy is the most important constraint in WSNs. In addition, this profile can provide new extension of MARTE to support the reconfigurable aspect of the WSN system since MARTE contains concepts related to reconfiguration. Authors proposed in [30] a highlevel methodology based on the MARTE profile for designing the power section for a WSN node. Four main elements are defined in the design: energy scavenging device, energy accumulating device, consumption of the node, and recharging energy. This methodology provides an extension of the «HW_PowerSupply» with «HW_ Harvesting» to describe the harvester, and the «HW_Harvesting » is also extended with «HW_PV» to describe the harvesting done by a solar panel.

To address the increasing complexity of WSN systems, numerous studies have suggested the use of high-level methodologies to meet the strict WSN constraints and streamline the development process. Many of these approaches utilized high-level modeling concepts to define various WSN fundamentals. Some authors employed UML and MARTE annotations, while others introduced new UML/MARTE extensions to specifically address power consumption and reconfiguration concerns in WSN. Additionally, some methodologies proposed a complete development cycle that begins with modeling and progresses to validation through a series of model transformation rules. Consequently, it can be inferred that the MDE methodology is well-suited to handle WSN complexity. However, existing studies still have limitations, such as a lack of adaptation processes for designing and validating WSN models. Furthermore, most adaptive works focus on node-level-based reconfiguration scenarios, addressing hardware and software reprogramming, but not architectural reconfiguration in high-level modeling. Additionally, the modeling phase does not consider MAPE loop modules, and explicit support for power requirement modeling is absent. As a result, there is a lack of high-level abstraction modeling for energy harvesting modules, and NFP verification and validation steps are not effectively carried out. Despite these limitations, the use of high-level modeling languages and methodologies assists designers in managing the growing complexity of WSN systems. Nevertheless, there is still a significant need for generic and reusable models to aid designers in easily modeling their systems. In this context, design patterns [14] offer a promising solution by promoting generic models for addressing recurring problems. The following provides some basic definitions of design patterns and recalls existing pattern-based work. Based on the findings of the prior investigation, there is a scarcity of research on design patterns in the field of WSN. Existing methodologies tend to concentrate on sensor node elements or network structure, with only a small number addressing reconfiguration scenarios. However, the existing studies fail to address crucial WSN needs like power efficiency and real-time limitations. Furthermore, they lack clear provisions for architectural reconfiguration in WSN and provide only minimal assistance in modeling WSN systems. Additionally, a majority of the research does not delve into high-level modeling languages and industry standards.

### Design Patterns for Wireless Sensor Network Development

Design patterns are widely recognized as a viable strategy for software design. A design pattern serves to encapsulate the application flow and design components from a higher level of abstraction, ensuring the reusability of the design. Specifically, a design pattern represents a general and reusable solution to a recurring issue in software design. Its effectiveness in modeling and representing complex systems has been well established.

Furthermore, it promotes the reuse of software models and enhances software quality. Each design pattern is characterized by fundamental elements that adhere to the pattern template outlined in the referenced work. The pattern name acts as an identifier for the pattern, its purpose, and its solution. The intent of the pattern articulates the objective behind its creation, while the pattern problem defines the context in which the pattern is applicable. Finally, the pattern structure is depicted through graphical representations, utilizing class and sequence diagrams. As the development of WSNsprogresses and the programming challenges associated with sensor nodes become more pronounced, software designers have increasingly focused on articulating design patterns for WSN development. In this regard, a design pattern specifically for a sensor node was introduced in the referenced study. This proposed pattern delineated the architecture of a sensor node, encompassing sensors, a power source, communication channels, and memory. The authors illustrated both the static and dynamic aspects of the proposed pattern through UML class diagrams and UML sequence diagrams, respectively.

Table 1 presents a summary of how the suggested method fulfills the design criteria outlined in the eintire review. The utilization of advanced techniques and reusable models in developing WSN offers a promising solution to reduce the increasing complexity of diverse systems like IoT systems. The application of MDE and models has been demonstrated as a beneficial solution through its principles of abstraction, separation of concerns, reuse, and automation. In MDE, models are the fundamental concept and are viewed as an abstraction of the system being developed. Apart from abstraction, automation is carried out in terms of model manipulation and refinement via model transformations. Furthermore, MDE addresses the challenge of managing heterogeneity in software and hardware by utilizing modeling languages, particularly domain-specific ones. Models created using these languages are designed to be more user-friendly than traditional code artifacts, thereby enhancing reusability. An intriguing perspective is to extend the proposed EARN-MDE process for designing intricate and critical reconfigurable applications.

**Table 1: Evaluation of the proposed MDE-based approach**

| | Design invironment | Power supply design | Reconfiguration scenario | NFP verification |
|---|---|---|---|---|
| Proposed approach | New UML/MARTE extensions | It supports battery | It proposes new network-level based reconfiguration strategy based on the MAPE loop | Automatic generated analysis scripts |
| | Mape loop dsign patterns | It proposes new energy harvesting source design | --- | Simulation and real deployment steps |

**Summary and Conclusion**

This paper presents a comprehensive study on programming methodologies and modeling techniques pertinent to the development of WSNs. We have categorized the existing works into two distinct groups based on the level of abstraction in their design. The first group focuses on low-level techniques specifically related to WSN programming models. Conversely, the second group encompasses high-level approaches, which include MDE methodology, particularly those utilizing UML and MARTE standards along with pattern-based concepts. Our findings indicate that the development of WSNs can be explored across various levels of abstraction.Two low-level development techniques have been introduced to address either node behavior or network architecture . While these approaches have demonstrated their efficacy in WSN development, there remains a notable absence of standardized mechanisms that adequately address the complexities inherent in WSNs. Consequently, elevating the abstraction level emerges as a viable solution to mitigate the limitations associated with low-level approaches. Indeed, Model-Driven Engineering, particularly the MARTE profile, has garnered significant attention in the realm of WSN development. The UML/MARTE profile facilitates the modeling of both hardware and software components within the WSN system.

In this context, our future work involves proposing an MDE-based method for developing an energy-aware reconfigurable WSN. The framework we suggest utilizes MDE concepts, the UML/MARTE profile, and design patterns to facilitate high-level specification and automatic analysis of WSN. Our initial research focused on energy sources in WSN, highlighting the lack of explicit support for power requirement modeling in existing studies. To address this gap, we introduced a well-structured approach for energy harvesting specification based on the MARTE profile. Another aspect of our research delved into reconfiguration scenarios in WSN. We observed that current works concentrate on low-level specifications, particularly node-level-based reconfiguration scenarios involving hardware and software. However, explicit support for network-level-based reconfiguration is lacking. To address

these issues, we devised an MDE-based process named EARN (Energy-Aware Reconfigurable Node) - MDE-based process to support architectural reconfiguration in WSN applications. This process enables the automatic generation of a high-level reconfigurable WSN model within an energy harvesting environment. It involves annotating the system model with reconfiguration semantics, followed by the automatic generation and integration of pattern instances into the initial system model based on a set of instantiation and integration rules.

**References**

[1] Suryadevara,. Mukhopadhyay, Kelly, and Gill, 2015, WSN-based smart sensors and actuator for power management in intelligent buildings, IEEE/ASME Transactions on Mechatronics, 20(2), 564–571.
[2] DurisicTafaDimic, and Milutinovic, 2012, A survey of military applications of wireless sensor networks, Mediterranean Conference on Embedded Computing (MECO), Bar, Montenegro , June, 2012.
[3] Furtado and Trobec, 2011, Applications of wireless sensors in medicine, Proceedings of the 34th International Convention MIPRO, Opatija, Croatia, May , 2011.
[4] Akyildiz, Y. Sankarasubramaniam, Cayirci, Wireless sensor networks: a survey, 38( 4), 393–422.
[5] Mottola Picco, Logical, neighborhoods: a programming abstraction for wireless sensor networks, in Distributed Computing in Sensor Systems, 2006. Lecture Notes in Computer Science, 40(26), 150–168.
[6] Jiang , Manivannan, Routing protocols for sensor networks,2004, First IEEE Consumer Communications and Networking Conference, Las Vegas NV, USA, January 2004.
[7] Madden, Franklin, Hellerstein, and Hong, 2010, An acquisitional query processing system for sensor networks, ACM Transactions on Database Systems, 30(1), 122–173.
[8] Molla and Ahamed, A survey of middleware for sensor network and challenges, 2006 International Conference on Parallel Processing Workshops (ICPPW'06), Columbus, OH, USA, August 2006.
[9] R. C. Shit, S. Sharma, D. Puthal, and A. Y. Zomaya, 2018, Location of things (lot): a review and taxonomy of sensors localization in IoTinfrastructure, IEEE Communications Surveys & Tutorials, 20( 3).
[10] Shit, Sharma, Puthal et al., 2019,Ubiquitous localization (UbiLoc): a survey and taxonomy on device free localization for smart world," IEEE Communications Surveys & Tutorials, 21( 4), 3532–3564.
[11] Newton, Morrisett, Welsh, 2017, The regiment macroprogramming system, 6th International Symposium on Information Processing in Sensor Networks, Cambridge, MA, USA, April 2007.
[12] Shaikh and Zeadally, 2016, Energy harvesting in wireless sensor networks: a comprehensive review, Renewable and Sustainable Energy Reviews, 151041–1054.
[13] Basagni, Naderi, Petrioli, and Spenza, 2013, Wireless sensor networks with energy harvesting, in Mobile Ad Hoc Networking, Cutting Edge Directions, S. Basagni, M. Conti, S. Giordano, and I. Stojmenovic, Eds., John Wiley& Sons, 2013.
[14] Autonomic Computing, An architectural blueprint for autonomic computing, IBM White Paper, 31, 2006.
[15] Rajasekaran, Jeyabharath, andVeena, 2014n, hardware-software reconfigurable techniques for wireless sensor network,Research Journal of Applied Sciences, Engineering and Technology, 8(17), 1855–1862.
[16] Levis, Madden, Polastre et al., TinyOS: an operating system for sensor networks,2005, in Ambient Intelligence, Springer, Berlin, Heidelberg, 2005.
[17] Gay, Levis, von Behren, Welsh, Brewer, and Culler, The nesC language, ACM SIGPLAN Notices,49(4), 41–51, 2014.
[18] Dunkels, Gronvall, and Voigt, Contiki - a lightweight and flexible operating system for tiny networked sensors, in 29th Annual IEEE International Conference on Local Computer Networks, Tampa, FL, USA, November 2004.
[19] Rodriguez-Zurrunero, Utrilla, Rozas, and Araujo,2019, Process management in Iot operating systems: crossinfluence between processing and communication tasks in end-devices, Sensors, 19(4), 805.
[20] Baccelli, Hahm, GÃijnes, WÃďhlisch, and Schmidt,2013, RIOT OS: towards an OS for the Internet of things, IEEE Conference on Computer Communications Workshops Turin, Italy, April 2013.
[21] Arm Mbed, "Mbed OS," https://www.mbed.com/en/platform/mbed-os/.
[22] TheFreeRTOSreferencemanual,ttps://www.freertos.org/Documentation/FreeRTOS_Reference_Manual_V9.pdf.
[23] Levis and Culler, Maté: a tiny virtual machine for sensor networks, 2002, Proceedings of the 10th international conference on Architectural support for programming languages and operating systems, pp. 85–95, San Jose, CA,USA, October .
[24] Živković, Nikolić, Protić, and Popović, A survey and classification of wireless sensor networks simulators based on the domain of use, Adhoc and Sensor Wireless Networks, 20(3), 2014.
[25] Toor and Jain, 2017, A survey on wireless network simulators, Bulletin of Electrical Engineering and Informatics,6(1) 62–69.
[26] Ns-3 overview, August 2010, http://www.nsnam.org.
[27] Lacage, 2009, Experimentation with ns-3, Trilogy Summer School, 2009.

[28] Saabith, Fareez, and Vinothraj, 2019, Python current trend applications-an overview, International Journal of Advance Engineering and Research Development, 6(10),2019.

[29] Omojokun,A survey of Zigbee wireless sensor network technology: topology, applications and challenges, International Journal of Computer Applications, 130(9), 47–55, 2015.

[30] R. Gummadi, O. Gnawali, and R. Govindan, Macro-programming wireless sensor networks using Kairos, in Distributed Computing in Sensor Systems,2005, Lecture Notes in Computer Science,35(60), Springer, Berlin,Heidelberg, 2005.

[31] Hac, Wireless Sensor Network Designs, John Wiley & Sons Ltd, 2003.

[32] Doddapaneni, Ever, Gemikonakli, Malavolta, Mostarda, and Muccini, A model-driven engineering framework for architecting and analysing wireless sensor networks, Third International Workshop on SoftwareEngineering for Sensor Network Applications , Zurich, Switzerland, June 2012.

[33] Boonma, Somchit, and Natwichai, A model-driven engineering platform for wireless sensor networks," in 2013 Eighth International Conference on P2P, Parallel, Grid, Cloud and Internet Computing, 671–676, Compiegne, France, October 2013.

[34] Argyris, Mura, and Prevostini, Using MARTE for designing power supply section of WSNs, in M-BED Proceeding of the 1st Workshop on Model Based Engineering for Embedded Systems Design, Germany, 2010.