WISE: A Semantic and Interoperable Web of Things Architecture for Smart Environments

Published Version:
Bedogni, L., Manfredini, S., Poggi, F., Rossi, D. (2022). WISE: A Semantic and Interoperable Web of Things Architecture for Smart Environments. New York : IEEE [10.1109/WF-IoT54382.2022.10152138].

Availability:
This version is available at: https://hdl.handle.net/11585/935993 since: 2023-07-24

Published:

DOI: http://doi.org/10.1109/WF-IoT54382.2022.10152138

(Article begins on next page)

WISE: A Semantic and Interoperable Web of Things Architecture for Smart Environments

Luca Bedogni*, Sebastiano Manfredini*, Francesco Poggi†, Davide Rossi‡
*Department of Physics, Informatics and Mathematics, University of Modena and Reggio Emilia, Modena, Italy
†Institute of Cognitive Science and Technologies, National Research Council, Rome, Italy
‡Department of Computer Science and Engineering, University of Bologna, Bologna, Italy
Email: {sebastiano.manfredini,luca.bedogni}@unimore.it, francesco.poggi@cnr.it, daviderossi@unibo.it

***Abstract*—The rapid proliferation of Internet of Things devices has led to a number of different standards and technologies which offer novel and exciting services. One of the key aspect of the Internet of Things is its ubiquitness, as devices may spontaneously form networks and leave them possibly in short time frames. This is the case of Smart Environments such as Smart Homes, in which users carry a set of devices like wearables and mobile applications to monitor their behavior and provide contextual services. However, the interoperability and seamless interaction of different devices is yet to be fully realized. In this paper we propose WISE, a framework that leverages the Web of Thing architecture and Semantic technologies to overcome technical and conceptual interoperability difficulties and enables the creation of cooperative Smart Environments that self-adapt on the basis of users' preferences. The use of Semantic technologies enables to understand which devices can provide the needed affordances to meet the user preferences, while the WoT architecture is leveraged to access devices in a standardized manner. We also propose a reference implementation based on off-the-shelf devices which demonstrate the feasibility of WISE.**

***Index Terms*—IoT, WoT, Semantic technologies**

I. Introduction

The Internet of Things (IoT) pervades many aspects of daily lives, as devices are able to contextualize the user which is carrying them, enabling context-aware services and a more personalized user experience. IoT devices differ in size, communication technology used, purpose and services offered, which has led to a proliferation of different standards, network protocols and alike, which inherently gave birth to the so-called Islands of Things [18], considered as environments in which devices share data and can communicate, but from which it is difficult to obtain data which can be used also by other networks, or in which devices built by different vendors can interoperate. This heterogeneity eventually gave birth to the Web of Things (WoT) [13], which is a standardization effort aiming to solve the interoperability problem in terms of devices and services re-usability. Among the different characteristics of WoT, it is possible for the devices to expose their Thing Description, which describes in a standardized way what the device is, what services it can offer, how to interact with it, and what are its capabilities. Things Descriptions can be shared in various ways [6], such as via the use of a standardized directory service. The WoT standardization makes it easier to discover devices in a network and obtain their description thanks to a common representation of devices capabilities and methodologies to access and leverage them.

Nevertheless, true interoperability between heterogeneous devices is yet to be achieved, since in many cases it is not still possible to automatically assess the type of data a device can offer, as it may be written with different terms and the relation between any two data offered by separate devices may not be evident from a machine perspective. Hence WoT handles the interoperability problem at a low level, but from a high level perspective this interoperability is yet to be achieved.

In this paper we adopt Semantic technologies to proceed towards filling this gap, by proposing a Web of Things Semantic Infrastructure for Smart Environments (WISE), a system in which WoT capabilities are derived from a Semantic vocabulary, and in which users can define personal preferences about their `Smart Environments`. One of the main novelty introduced by WoT's Thing Description is in fact the ability to add semantic annotations to the thing's elements as well as extending it with new elements from various semantic vocabularies thanks to the adoption of a representation based on JSON-LD, which is a lightweight Linked Data format based on the popular JSON notation. WISE leverages this semantic annotation ability and the ability of a TD directory to support semantic look-ups to implement cyber-physical systems that not only ease the interoperability of heterogeneous things but also exploits this heterogeneity to provide innovative services.

To demonstrate the feasibility of WISE, after discussing an architectural representation of it we also provide a candidate implementation made on ESP-32 microcontrollers and leveraging a Raspberry Pi 4 as Controller of the system.

The rest of this paper is organized as follows: Section II presents the basic ideas behind WISE, introduces a motivating scenario and lists the research questions we face; Section III discusses related work from literature; Section IV presents WISE and the different building blocks from which it is created; Section V presents our prototype implementation and highlights results obtained from experiments performed on it; Section VI concludes our paper, discusses the results and presents future work on this topic.

II. Motivation and research questions

The overarching idea governing WISE is that of creating a cyber-physical system that dynamically extends by including users and their devices when they become part of the system's environment. Users, in fact, can selectively decide to share two kinds of elements with the system: sensor readings from their personal devices and their environment preferences. The system uses its own sensors and the users' sensors to drive environment adaptation mechanisms that try to meet the preferences shared by the users. In this perspective users are enticed into becoming part of the system knowing that this will result in a better personal experience. This approach offers mutual advantages to all involved parties: the system can improve its awareness thanks to the readings coming from users' sensors, which include both environment measures (e.g. noise level) and personal measures (e.g. heart rate), whereas the users know that by sharing these information with the system they enable adaptations driven by their personal preferences in any Smart Environment they visit.

To better understand the possibilities opened by such an approach here we introduce a brief motivating scenario. The scenario is limited to a very specific domain (smart buildings) and one adaptation axis (ambient temperature) but the solution we propose, as the reader can easily imagine, can be applied to different contexts. A user enters a smart building governed by an WISE-based system. The user has a smartphone, with the usual array of sensors, and a smartwatch that detects her/his heart rate. An app running on its smartphone detects that she/he is entering a new, not-yet-configured WISE domain, this produces a notification asking the user if she/he is willing to share the readings of the phone sensors and those of its connected devices (as is the case for the smartwatch). The app also asks the user which of its recorded personal preferences (about illumination level, temperature, humidity, background music and so forth) she/he wants to share with the system. The user decides to share its temperature preference, the readings from the smartwatch's heart rate sensor as well as the readings from its Bluetooth adapter that are used to detect beacons placed in various rooms of the building [5], allowing the system to precisely locate the user. From that moment, until the user exits the building, whenever the user enters a room, the system (by operating on the adapters controlling the HVAC system) adapts the temperature of the room to align it with the user's preference. Since the temperature preference of the user is expressed in terms of activity level (e.g. 18 °C on high activity level, 20 °C on low activity level) the system also makes use of the reading from the user's heart rate sensor integrated in her/his smartwatch to drive the adaptation. When multiple users with clashing preferences are present in the same room a conflict mitigation approach is used (as detailed in subsection IV-B).

The analysis of these kind of scenarios lead us to the definition of a set of research questions on which we focus in the present work:

- how to design a `Smart Environment` that is able to integrate and leverage user contributed affordances within a homogeneous systems;
- how can the system discover and interoperate with the novel capabilities offered by devices dynamically entering and leaving the network?
- how can a Smart Environment automatically reconfigure itself, leveraging its own affordances to suit the user needs?

In the following we answer to these research question by providing a general definition of WISE, which can be easily extended to fit different use cases, and which serves as an architecture which enables the seamless interaction of WoT devices and extends the network possibilities thanks to the capabilities offered by such devices. We then provide a prototype implementation of it, which leverages off-the-shelf devices and focuses on a `Smart Home` scenario, demonstrating the applicability of the proposed approach in real world scenarios.

III. Related work

The Internet of Things and more recently the Web of Things have gained attention due to the possibility to interconnect heterogeneous devices and build complex services, through communication and data analysis [19]. In general IoT and WoT networks have been used in a plethora of scenarios, ranging from the Smart City [27] to e-Health [1], from Industry 4.0 [7] to Autonomous Driving [16].

More centered about the work we have done in this paper is the use of semantic technologies to connect IoT and WoT devices [10]. This enables not only the possibility for devices to communicate, but also to understand and automatically provide data according to the other devices needs. This also requires WoT devices to be discoverable, as novel devices entering on a network must know which other devices are available in the network [26]. Moreover, WoT devices also need to be built on ontologies which represent the data and the services they can offer, so that devices can cooperate and share information automatically and without the need for human intervention [22] [21] [2]. Among these different ontologies [14], certainly SOSA/SSN is one of the most popular, as it describes many different services typically needed in WoT scenarios [25] [11].

IoT and WoT devices are used in Smart Environments due to their ability to describe the physical world and interact with it. While the WoT specification provides a standardized layer through which devices can physically interact, the semantic layer serves as a common representation and naming of data and services, to enable the interaction not only at a low level but also at the application layer.

An interesting work is presented in [4], where the authors integrate and test different ontologies and WoT models. A similar approach is presented in [20] and [9], where ontology-based models are used to support the dynamic adaptation of software systems. In [12] the authors propose a system to enable cross-domain interaction, between devices pertaining to different producers and different capabilities, and [23] leverages Semantic Web technologies to capture the objectives of a system composed of sensors and actuators. [8] presents a mechanism based on ontologies to enable interoperation between multiple IoT semantics-based systems.

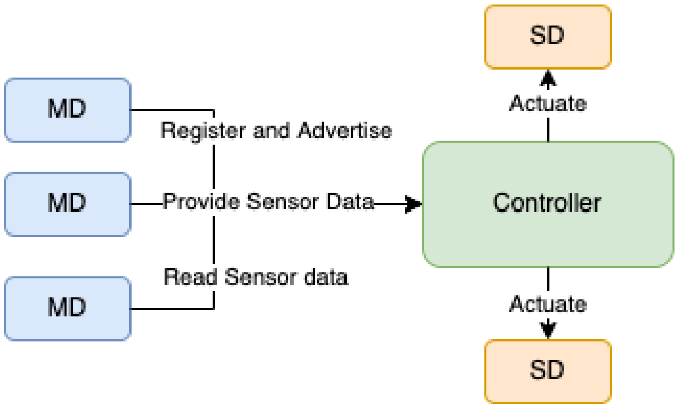

Fig. 1. Overview of the proposed architecture of WISE. Mobile Devices (MD) register themselves on the Controller and can then provide and read sensor data. Static Devices (SD) are already in the network, hence already registered on the Controller and can the receive commands to perform specific actions.

Closer to the work presented in this paper is [3], where the authors use a semantic reasoner to interpret user preferences in Smart Environments. However, they do not use a semantic representation of things, but rather let users describe their plans with natural language and then translate such plans into specific actions on the things.

The work presented in [15] address the problem of configuration of Smart Environments, which is done through a visual programming interface. Although the usage of visual programming interfaces, particularly for service composition, is a well studied topic [17], the configuration of the for the user has to be done manually and could not be transferred to other smart environments.

In [24] the author propose a Semantic content centric framework for IoT devices, which enables the automatic and seamless discovery of devices in a network.

IV. Proposed Framework

In this section we describe WISE from an architectural point of view, and how the building blocks of it communicate with each other.

In Figure 1 we show the main components of WISE. We now detail how each of them is composed, and how it interacts with the other devices.

The `Controller` is the main building block of WISE, and it is composed of a Thing Directory, an MQTT broker, and a custom made engine which is responsible of handling all the messages and performing the operations between the devices.

The `Mobile devices` are the user devices, which can be of different kinds such as wearables, mobile applications, or custom made devices. The `Static devices` are the devices which control the Smart Environment, such as switches, smart light bulbs, thermostats and alike.

The overarching idea of WISE is that `Mobile Devices`, personal devices of users, carry a set of preferences which are defined by the user itself. These can include the preferred temperature, ambient light, humidity, personal needs and alike. In general, these are preferences which can be general (i.e. they are valid in any Smart Environment) or can be set for a specific Smart Environment. Any `Static Devices` or `Mobile Devices` can contribute to the knowledge of the `Smart Environment` by providing sensor data. This is done since all `Static Devices` and `Mobile Devices` in WISE are implemented as WoT devices, exposing their capabilities through which they can be queried. Whenever a `Mobile Devices` enters a SE, it needs to query the `Controller`, registering itself in the Thing Directory and advertising its preferences, which are stored in the engine. A similar flow is also performed by the `Static Devices`, which however can also perform real actions on the `Smart Environment`. Nevertheless, they too register in the Thing Directory, but do not advertise any preference. Clearly `Static Devices` will be already available in a network, hence they register to the `Controller` only when they are run for the first time, while `Mobile Devices` need to re-register every time they re-join the network after departing it.

The `Controller` is also the device in the system which gathers all the information from the `Static Devices` and `Mobile Devices` in an `Smart Environment`, which can happen through any supported protocol such as CoAP, MQTT, WebSockets or anything which is available in the `Controller` itself. Upon any value change observed by the `Controller`, it looks for all the preferences obtained by the devices, and checks whether they are satisfied or not. In case it needs to perform an action to fulfill a preference, it can act on one or more static devices, according to their advertised services.

No static binding is performed by the `Controller`, which instead leverages WebThings[1], a semantic vocabulary through which it assess which `Static Devices` can be queried to satisfy the `Mobile Devices` preference and through which of its advertised capabilities. WebThings is a platform originally developed by the Mozilla Foundation for monitoring and controlling devices over the web. It is an open source implementation of the emerging WoT standards at the W3C.

The WebThings vocabulary defines a set of common `Properties` of interest about the environment, such as `TemperatureProperty`, `HumidityProperty` and `BrightnessProperty`. When users join the system they advertise their preferences in terms of desired values for the aforementioned properties. For instance, a user can set a target `TemperatureProperty` of 19 degrees and a `BrightnessProperty` of 30% (i.e. dark), and not setting the `HumidityProperty`.

WISE also uses the SOSA/SSN ontology [11] to represent the main information about sensors and actuators in the system. In particular, sensors and actuators are defined as instances of the `sosa:Sensor` and `sosa:Actuator` classes, respectively. This is needed so that the system knows which devices can provide data and which can instead offer actions to be performed. Clearly, a hardware device may have sensors

[1] https://webthings.io/schemas/

and actuators, and in this case it is an instance of `wot:Thing` meaning that it is a Thing in the WoT architecture, which is also instance of both `sosa:Sensor` and `sosa:Actuator`.

In order to check whether users' preferences are satisfied, the `Controller` reads sensor data (e.g. the values of the `TemperatureProperty` and `BrightnessProperty`) from any `sosa:Sensor` close to the user (e.g. in the same room). If one (or more) of the current values does not satisfy the preferences, the `Controller` identifies a `sosa:Actuator` (e.g. the thermostat) able to act on the property (`TemperatureProperty`) and activates it by triggering the appropriate action.

Data can be retrieved by the `Controller` in two different ways: the data could be pushed by the `Mobile Devices` directly to the `Controller`, or the `Controller` could ask the `Mobile Devices` by querying it. The choice depends on the kind of data and how the network is configured. Clearly there may be devices which are available in the network configured to periodically push values to the `Controller`, either when values change or at a constant frequency. For sporadic `Mobile Devices` entering the network instead, it would be the `Controller` which queries the device about a specific value whenever it needs to update it, since the `Mobile Devices` may not know how the network is configured.

A. *Multiple rooms*

A single `Smart Environment` may be composed of different rooms, in which preferences need to be set differently. A user sets her/his own preferences regardless of the area, but the `Controller` need to act according to the devices in an area. For instance, if a user moves from area A to area A', its preferences need to be applied to area A' from that point onward. Therefore, the `Controller` also keeps track of the location of the device. We note that indoor localization is nowadays available with multiple technologies such as magnetometers, dead reckoning, beacons, barometers, RF signals and more, and it is up to the `Smart Environment` to decide which suits the best [19].

Regardless of the localization technology used, we assume that the `Controller` knows in which room each device is, and can therefore aim to meet the preferences of each device in each area. To do so, the `Controller` monitors for changing values in each area defined, to foresee potential actions to perform to fulfill the preference of the users. Since all the devices, both the `Static Devices` and `Mobile Devices` are implemented as Things in a WoT network, the `Controller` can simply query the Thing endpoint to retrieve the appropriate values. Clearly, the `Controller` is aware of all the data it can retrieve since the `Devices` registered in the `Thing Description Directory`, which can be queried in order to assess which devices are available and which capabilities they offer. We also note that this mechanism enables the `Mobile devices` to query for data available in other `Mobile devices`, since the `Controller` exposes what is available to the whole network.

B. *Clashing preferences*

It may happen that two or more preferences collide between each other, for instance if a user reports a desired temperature of "at least 20 degrees" and another one reports "no more than 19 degrees". Of course it is impossible to satisfy them both at the same time, hence the engine takes the decision which penalizes less the user preferences. More formally, given a measurement M for an area A and two associated preferences of user j denoted as $P^M_{j,A}$ and user k denoted as $P^M_{k,A}$, we define Δ^M_A as:

$$\Delta^M_A = \sum_{i \in A} |P^M_{i,A} - \sigma|, \tag{1}$$

the goal is then to find a target setting σ which minimizes Δ^M_A. In other words, we look for a setting which is the closest one to satisfying the participant devices in an area, hence minimizing the error between the target setting and all user preferences.

C. *Devices leaving the `Smart Environment`*

In a dynamic environment, devices can enter and leave the `Smart Environment` at any time and without directly notifying the `Controller` of their departure. Therefore upon registering themselves, `Static Devices` and `Mobile Devices` are periodically queried by the `Controller` to check if they are still in the network. When devices are not reachable or do not reply within a given time frame, they are considered as having left the environment and their description is removed from the system, as well as their preferences.

V. Implementation and Evaluation

In this Section we provide a reference implementation of WISE, deployed in a prototype smart home. We have implemented the `Controller` in a Raspberry PI 4. The `Static devices` are implemented in ESP-32 microcontrollers, which enable a WiFi connection to the network while also allowing to interact with the actuators. The same applies for the `Mobile devices`, which represent two wearable devices which may enter and exit the network dynamically.

In the `Controller` we have implemented the logic of the engine in a Python script, which also offers a Web based interface used to interact with WISE. We have also implemented mDNS on the `Controller`, to allow for an easy discovery of it by the clients. Finally, the `Controller` also offers a `Thing Description Directory` (TDD) to register and search the devices within the system. We used an implementation of the W3C TDD which is freely available on GitHub [2]. We have also set the t parameter to 60 seconds.

The `Mobile devices` and `Static devices` have been developed using the well known WebThingArduino library, adapted to work also with the ESP-32 microcontrollers. We also offer easy configuration for the user on the wearable device, by offering a Web based interface directly on the device itself through which it is possible to set any wanted preference. We note that Things in a WoT network need to provide an HTTP interface as part of the WoT standard,

[2]https://github.com/linksmart/thing-directory

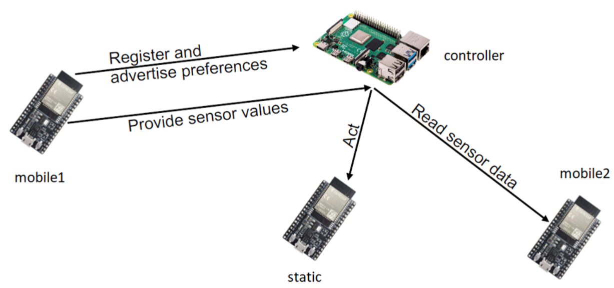

Fig. 2. Communication example between the Mobile and Static Devices through the Controller in our candidate implementation with ESP-32 microcontrollers and a Raspberry Pi 4 which acts as the Controller.

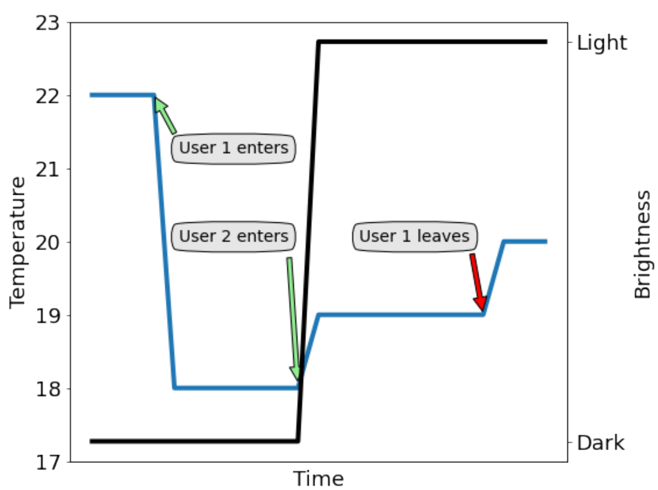

Fig. 3. Example run of our system in which two devices dynamically enter and leave the network. Green arrows are user joining the system, while red arrows refer to users departing from it.

hence adding a Web-based interface does not involve adding any additional running service on the device itself. Figure 2 provides the overview of our developed system.

In Figure 3 we show an example run of our system in which there are two devices, named as ”User 1” and ”User 2”, which enter and leave the network. Specifically, ”User 1” has set her preferred temperature to 18 degrees, and has not specified any preference regarding the brightness of the Smart Environment. ”User 2” has instead set a reference temperature of 20 degrees, and a *bright* environment. The system starts from its normal state in which no specific action has been taken, and as soon as ”User 1” enters the network, the system reconfigures itself and reduces the temperature to 18 degrees. Upon the entrance of ”User 2”, the system now detects two clashing preferences, as explained in Section IV-B, and sets the temperature to 19 degrees, which is the target temperature between the preferences of the two users. It also switches on the light of the room, as ”User 2” is the only one specifying it. On the next even, which is ”User 1” leaving the system, the system reconfigures again itself by setting the target temperature to 20 degrees, which is the preference of ”User 2”, since there are no more clashing preferences.

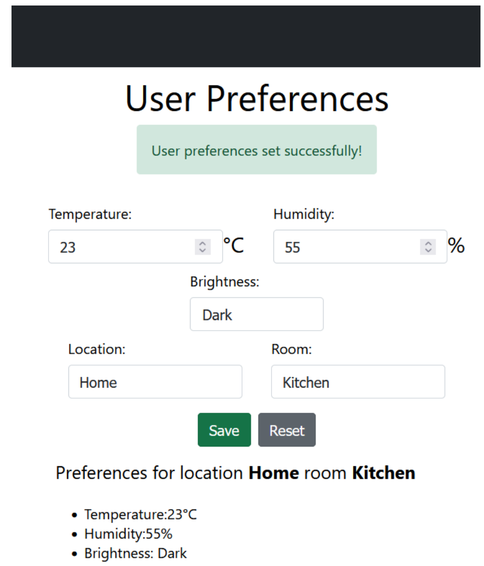

Fig. 4. The wearable preference system implemented in the ESP-32 `Mobile devices` for the `Kitchen` room in the `Home Smart Environment`.

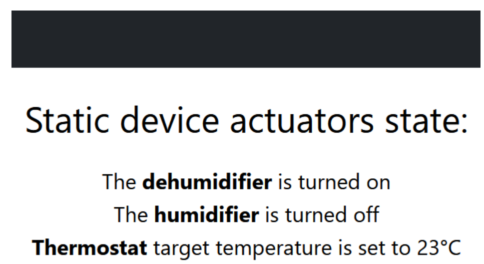

Fig. 5. The `Static device` web based interface. It is evident the status of the different actions it can take, and what is the reference temperature it needs to achieve.

In Figure 4 we show the Web based interface which is visible on the wearable device, and which allows the user to set her own preferences. In our implementation, we allowed the user to set their preferred `Temperature`, `Humidity` and `Brightness`, but it is straightforward to note that more variables can be easily enabled.

In Figure 5 we show instead the Web based interface of the `Static device`, where it is possible to see the status of the actuators currently available on the `Smart Environment`. We also highlight that there are two statuses related to the actions the `Static devices` can perform, and also a target temperature set to fulfill the preferences of the `Mobile device`, as instructed by the `Coordinator`. Finally, we also note that no action is taken to fulfill the preference concerning the `Brightness`, since no device which can act on it is present in our `Smart Environment`, hence the `Controller` cannot take any action in such direction.

VI. Conclusion

In this paper we have presented WISE, a Web of Things Semantic Infrastructure for Smart Environments, which is able to automatically reconfigure Smart Environments for users based on their preferences. We have presented it from an architectural point of view in Section IV, and provided a candidate implementation in Section V. WISE leverages semantic technologies to match preferences issued by users and affordances available in a Web of Things network, so that users joining and leaving any Smart Environment are able to have their personal preferences met by the system. Through the use of SOSA/SSN preferences and affordances can be matched, so that the system knows which `sosa:Actuator` can handle the preference and act accordingly to meet them.

Our candidate implementation has shown the feasibility of our approach, using off-the-shelf technologies and hardware, and also provided a sample execution of WISE showing how the system reacts when devices join and leave the system.

Future works on this topic include the extension of the semantic infrastructure to a wider set of `sosa:Sensor` and `sosa:Actuator`, to widen the applicability of WISE to more Smart Environments. We will also enrich WISE by providing an interface based on SPARQL through which users can query WISE and get history data as well as precise information about past events. Finally, we will also introduce localization capabilities, assumed as available in the present work, and build a larger testbed to evaluate WISE.

References

[1] Moses A. Agana, O. Ajah Ofem, and Bassey Igbo Ele. A Framework for a Fuzzy Smart Home IoT e-Health Support System. In Advances in Information and Communication, pages 432–447. Springer, 2020.

[2] Francesco Antoniazzi and Fabio Viola. Building the semantic web of things through a dynamic ontology. IEEE Internet of Things Journal, 6(6):10560–10579, 2019.

[3] Vincenzo Catania, Gaetano Carmelo La Delfa, Salvatore Monteleone, Davide Patti, Daniela Ventura, and Giuseppe La Torre. Goose: goal oriented orchestration for smart environments. International Journal of Ad Hoc and Ubiquitous Computing, 32(3):159–170, 2019.

[4] Victor Charpenay, Sebastian Käbisch, and Harald Kosch. Semantic data integration on the web of things. In Proceedings of the 8th International Conference on the Internet of Things, IOT '18, New York, NY, USA, 2018. Association for Computing Machinery.

[5] Dongyao Chen, Kang G. Shin, Yurong Jiang, and Kyu-Han Kim. Locating and tracking BLE beacons with smartphones. In Proceedings of the 13th International Conference on emerging Networking EXperiments and Technologies (CoNEXT), pages 263–275. ACM, 2017.

[6] Andrea Cimmino, Michael McCool, Farshid Tavakolizadeh, and Kunihiko Toumura. Web of Things (WoT) Discovery, W3C Working Draft 2 June 2021. 2021.

[7] Tharaka de Vass, Himanshu Shee, and Shah Miah. Iot in supply chain management: Opportunities and challenges for businesses in early industry 4.0 context. Operations and Supply Chain Management: An International Journal, 14(2):148–161, 2021.

[8] Ester Giallonardo, Francesco Poggi, Davide Rossi, and Eugenio Zimeo. Making Smart Buildings and Personal Systems Cooperate via Knowledge Base Overlays. In Proceedings of the 6th EAI International Conference on Smart Objects and Technologies for Social Good, page 181 – 186, 2020.

[9] Ester Giallonardo, Francesco Poggi, Davide Rossi, and Eugenio Zimeo. Semantics-driven programming of self-adaptive reactive systems. International Journal of Software Engineering and Knowledge Engineering, 30(6):805 – 834, 2020.

[10] Amelie Gyrard, Soumya Kanti Datta, and Christian Bonnet. A survey and analysis of ontology-based software tools for semantic interoperability in iot and wot landscapes. In 2018 IEEE 4th World Forum on Internet of Things (WF-IoT), pages 86–91, 2018.

[11] Armin Haller, Krzysztof Janowicz, Simon JD Cox, Maxime Lefrançois, Kerry Taylor, Danh Le Phuoc, Joshua Lieberman, Raúl García-Castro, Rob Atkinson, and Claus Stadler. The modular SSN ontology: A joint W3C and OGC standard specifying the semantics of sensors, observations, sampling, and actuation. Semantic Web, 10(1):9–32, 2019.

[12] George Hatzivasilis, Ioannis Askoxylakis, George Alexandris, Darko Anicic, Arne Bröring, Vivek Kulkarni, Konstantinos Fysarakis, and George Spanoudakis. The interoperability of things: Interoperable solutions as an enabler for iot and web 3.0. In 2018 IEEE 23rd International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), pages 1–7, 2018.

[13] Matthias Kovatsch, Ryuichi Matsukura, Michael Lagally, Toru Kawaguchi, Kunihiko Toumura, and Kazuo Kajimoto. Web of Things (WoT) Architecture, W3C Recommendation 9 April 2020. 2020.

[14] Wenbin Li, Giuseppe Tropea, Ahmed Abid, Andrea Detti, and Franck Le Gall. Review of standard ontologies for the web of things. In 2019 Global IoT Summit (GIoTS), pages 1–6, 2019.

[15] Simon Mayer, Ruben Verborgh, Matthias Kovatsch, and Friedemann Mattern. Smart configuration of smart environments. IEEE Transactions on Automation Science and Engineering, 13(3):1247–1255, 2016.

[16] Dimitar Minovski, Christer Åhlund, and Karan Mitra. Modeling quality of iot experience in autonomous vehicles. IEEE Internet of Things Journal, 7(5):3833–3849, 2020.

[17] Federico Montori, Vincenzo Armandi, and Luca Bedogni. A hierarchical architectural model for iot end-user service composition. In 2022 IEEE 19th Annual Consumer Communications Networking Conference (CCNC), pages 411–416, 2022.

[18] Federico Montori, Luca Bedogni, and Luciano Bononi. On the integration of heterogeneous data sources for the collaborative internet of things. In 2016 IEEE 2nd International Forum on Research and Technologies for Society and Industry Leveraging a better tomorrow (RTSI), pages 1–6, 2016.

[19] Federico Montori, Luca Bedogni, Marco Di Felice, and Luciano Bononi. Machine-to-machine wireless communication technologies for the Internet of Things: Taxonomy, comparison and open issues. Pervasive and Mobile Computing, 50:56–81, 2018.

[20] Francesco Poggi, Davide Rossi, and Paolo Ciancarini. Integrating semantic run-time models for adaptive software systems. Journal of Web Engineering, 18(1):1 – 41, 2019.

[21] Francesco Poggi, Davide Rossi, Paolo Ciancarini, and Luca Bompani. An application of semantic technologies to self adaptations. In 2016 IEEE 2nd International Forum on Research and Technologies for Society and Industry Leveraging a Better Tomorrow (RTSI), pages 1 – 6, 2016.

[22] Remy Rojas, Lionel Médini, and Amélie Cordier. Toward constrained semantic wot. In Proceedings of the Seventh International Workshop on the Web of Things, WoT '16, page 31–37, New York, NY, USA, 2016. Association for Computing Machinery.

[23] Davide Rossi, Francesco Poggi, and Paolo Ciancarini. Dynamic high-level requirements in self-adaptive systems. In Proceedings of the ACM Symposium on Applied Computing, page 128 – 137, 2018.

[24] Michele Ruta, Floriano Scioscia, Eugenio Di Sciascio, Domenico Rotondi, and Salvatore Piccione. Semantic-based knowledge dissemination and extraction in smart environments. In 2013 27th International Conference on Advanced Information Networking and Applications Workshops, pages 1289–1294, 2013.

[25] Samya Sagar, Maxime Lefrançois, Issam Rebaï, Maha Khemaja, Serge Garlatti, Jamel Feki, and Lionel Médini. Modeling Smart Sensors on top of SOSA/SSN and WoT TD with the Semantic Smart Sensor Network (S3N) modular Ontology. In ISWC 2018 : 17th Internal Semantic Web Conference, pages 163 – 177, 2018.

[26] Fernando Serena, María Poveda-Villalón, and Raúl García-Castro. Semantic Discovery in the Web of Things. In Irene Garrigós and Manuel Wimmer, editors, Current Trends in Web Engineering, pages 19–31, Cham, 2018. Springer International Publishing.

[27] Saurabh Singh, Pradip Kumar Sharma, Byungun Yoon, Mohammad Shojafar, Gi Hwan Cho, and In-Ho Ra. Convergence of blockchain and artificial intelligence in IoT network for the sustainable smart city. Sustainable Cities and Society, 63:102364, 2020.