PROCEDURAL METADATA: STRUCTURED GUIDE FOR DATA INTEROPERABILITY IN SUPPORT OF WEB OF THINGS

Nakyoung Kim, Hyeontaek Oh, Jun Kyun Choi
Korea Advanced Institute of Science and Technology

Abstract *– This paper presents the concept of a new metadata type, defined as procedural metadata, which enhances automation and interoperability of the web of things. As data-driven days have arrived, data has become ubiquitous, and it is considered as a new valuable asset for the fourth industrial revolution. Accordingly, systems and services are expected to intelligently make decisions and perform tasks by interacting with environments through sensing and actuation. This requires accumulated knowledge of procedures to be described in human and machine-readable ways. Procedural metadata is proposed as the means to provide the common descriptions on composable procedures of not only individual devices but also smart systems as a whole, based on existing data models and ontologies. As a type of metadata, procedural metadata does not affect the original formats, models, vocabularies, ontologies, etc. of data, devices, and systems. However, it rather provides additional information on common descriptions of logic and workflows to make decisions and perform tasks. This paper describes the proposed concept with examples, principles, and discussions. For a proof of concept it is implemented in Matlab based on a scenario analysis in a smart-home environment. Then, its potential and feasibility to be applied to the web environment are given.*

Keywords – *Internet of things (IoT), metadata, procedural metadata, semantic web of things (SWoT), web of things (WoT).*

1. INTRODUCTION

As the Internet of things (IoT) era has arrived, data is now ubiquitous and considered as a new valuable asset for the emerging fourth industrial revolution. However, data is currently overwhelming in not only its volume but also its diversity, especially due to the wide-varying nature of IoT [1]. Generated from heterogeneous sources, manifold data is collected, stored, processed, transported, shared, and used across different platforms and application domains. Due to the heterogeneity, it is extremely hard to achieve interoperability, and the potential value of data is getting suffocated by its own volume and diversity. Accordingly, the needs for efforts to homogenize the data ecosystem and empower semantic linkages are rising.

Through the concepts of web of things (WoT) and semantic web of things (SWoT), enhanced interoperability in IoT ecosystems can be fulfilled on the web, an information space consisting of data resources [2]. The web has been developed for data semantics in order to embrace heterogeneous data, platforms, and application domains. Accordingly, the web provides technologies that support the interoperability of IoT. WoT is based on the idea that connected devices are accessible via the web to enable interoperability across heterogeneous IoT platforms and application domains [3], [4]. Furthermore, SWoT brings semantic web and WoT together to associate semantically annotated information to web-enabled IoT [5], [6].

To support interoperability and common understandings of data on the web, many web data formats, models, vocabularies, ontologies, and other technologies have been introduced. Not only the traditional information and communication technology (ICT)-related industries but also other non-ICT industries, such as energy, transportation, health, geography, etc., are developing their own ones. Consequently, more and more heterogeneous data concepts, formats, models, and ontologies from many consortia/fora and organizations are investigated without a consensus of the future data eco-society. Therefore, interoperability remains as the urgent issue in the process of moving towards a future data-driven eco-society [7].

Moreover, today's advanced systems and services require intelligent decision making and task performance which involve bidirectional interactions of sensing and actuating. In order to empower devices to intelligently engage in smart systems, devices and data need additional descriptions of not only their various aspects but also their roles within smart systems, specifically in the areas of decision making and performing tasks. Many approaches have targeted providing common descriptions of devices and data. However, not many works have tackled the problem of how to provide a common understanding and description of logic and workflows of aggregated devices in smart systems. Self-management and autonomic capabilities are considered to be the driver in the development of solutions to the scalability and heterogeneity problems of IoT [1]. Merely providing the descriptions of properties, relations, and functionalities of devices and data is not enough for intelligent smart systems and services. Accordingly, common understandings of not only data and devices but also cooperative decision logic and action workflows need to be established for automation and interoperability. Therefore, this paper utilizes metadata as means to provide the common descriptions of composable procedures of not only individual devices but also smart systems as a whole in order to support interoperability and the automation of procedures.

Metadata has been defined and used to complement data and other resources with additional information. There is metadata for different purposes and characteristics; these are described in Table 2 and Table 3 [8]. However, these conventional types of metadata face some shortcomings as more autonomic systems are desired. Recent services and applications are expected to perform intelligent decision making and actions, which require information on know-how or accumulated knowledge melted into data, devices, and systems. Correspondingly, the decision making and task-performing procedures should be described in machine- and human-readable ways.

Accordingly, the concept of a new metadata type, so-called procedural metadata is presented in this paper. The goal of procedural metadata is to provide the common descriptions on composable procedures of devices and systems based on existing data models and ontologies. Procedural metadata is a script type of data that describes event, condition, and action-based procedures of smart systems. It describes composable logic, functions, and workflows among data, devices and systems so that the they can interoperably and automatically engage together to intelligently make decisions and perform tasks. Operated according to procedure metadata, data, devices, and systems can be fully or partially composed to deploy interoperable, autonomic systems.

The remaining paper is organized as follows: Section 2 highlights related work, followed by a description of the proposed concept of procedural metadata with a discussion on the principle, feasibility, and security aspects of the concept in Section 3. In Section 4, scenario analysis and implementations for a proof of concept are presented. Finally, a conclusion is in Section 5.

Table 1 – Conventional types of metadata

Metadata	Description
Descriptive metadata	It includes basic information for finding or understanding a resource, such as title, author, subjects, keywords, publishers.
Administrative metadata - Technical - Preservation - Rights	It includes information to help manage the data resource. It provides information about not only how the data can be opened, read, used, etc. but also how it should be managed for future use and rights of the data .
Structural metadata	It includes information on how the components of an object are organised or structured and describes the relationships of parts of resources to one another .
Markup languages	It provides structural or semantic features within the data content by integrating metadata and flags (tags/indexes).

Table 2 – Primary uses and example properties of metadata

Metadata	Primary Uses	Example Properties
Descriptive metadata	- Discovery - Display - Interoperability	- Subject - Genre - Publication data
Technical metadata	- Interoperability -Digital object management - Preservation	- File type, size - Creation date/time - Compression scheme

Metadata	Primary Uses	Example Properties
Preservation metadata	- Interoperability - Digital object management - Preservation	- Checksum - Preservation event
Rights metadata	- Interoperability - Digital object management	- Copyright status - License terms - Rights holder
Structural metadata	- Navigation	- Sequence - Place in hierarchy
Markup languages	- Navigation - Interoperability	- Paragraph - Heading - List

2. RELATED WORK

Many approaches in projects and standardization activities target interoperable, semantic, automated, and dynamic IoT ecosystems. Accordingly, works on data models, ontologies, platforms, frameworks, and software have been continuously done. Developed by World Wide Web Consortium (W3C), a semantic sensor network (SSN) [9] is one of the most significant and widespread models to describe sensors and IoT-related concepts. W3C SSN ontology provides the concepts describing connected devices in great detail across a wide range of applications. Even though there exists a number of other semantic sensor network ontologies, many of them utilize W3C SSN as their basis or underlying model. Hence, W3C SSN currently plays a key role in IoT, WoT and SWoT [10]. IoT-Lite [11] is a lightweight semantic model which is an instantiation of W3C SSN ontology. Due to its detailed description, W3C SSN can be considered to be heavy, and IoT-Lite is developed to minimize concepts and relationships of W3C SNN while providing answers to most end-user queries. There exist ongoing European projects that leverage the W3C SSN ontology as their starting points for achieving interoperability. For example, Federated Interoperable Semantic Testbeds and Applications (FIESTA) [12] and Inter-IoT [13] utilize W3C SSN ontology as their basis and extend it in their own ways. Currently, W3C is making progress on establishing the Thing Description (TD) repository for common representation for a WoT description. TD provides the semantic metadata and functional description of devices in events, properties, and actions [14], [15]. Along with TD, W3C is also working on an application programming interface (API) specification, WoT Scripting API [14]. This provides a uniform way to write applications for WoT and allow scripts to run on devices for discovery, provisioning, and the control of them [16].

In literature, approaches to develop languages, frameworks, software, etc. are also in progress. To support representations of things, a descriptive language for devices, IoT Device Description Language (IoT-DLL), is presented in [17]. Although it aims for automation and interoperability of smart devices, it is compatible only when the devices are described in this particular language. In [18], a Semantic IoT Framework is proposed, which facilitates realizing the changes in data or metadata, analyzing data, and firing domain-specific rules or policies. The core part of the proposed framework is composed of the modules that perform updating the data changes, activating the actions, and processing data. However, the interoperability issue still has not been solved since these procedures occur on the platform level so that it is not applicable to data-level processing; more importantly it is out of the proposed framework. In [19], a tool called SysFlow Workflow Engine is presented, which receives a systems modeling language workflow in an XML metadata interchangeable format to parse and execute workflows. This work provides a way to automate the workflows by utilizing metadata, but the proposed tool is yet restricted to some specific domains, and its shortcoming comes when the users do not have access or their preference in the proposed tool for their interested application domain.

The aforementioned approaches are focused on semantically providing common descriptions on data and devices in order to support automation and interoperability in existing WoT environments or their own developed platforms, frameworks, and infrastructures. However, to truly enable automatic and interoperable interactions among devices, devices and systems need to be empowered by additional information on the basic procedures of how they are allowed to interact with others. Common understandings of not only data and devices but also cooperative decision logic and action workflows need to be established for automation and interoperability. For common descriptions on logic, functions, and workflows among devices in systems and services, the concept of a new metadata type, so-called procedural metadata is presented in the following section.

3. PROPOSED PROCEDURAL METADATA FOR SEMANTIC WEB OF THINGS

In WoT, devices and data should be properly managed and engaged by relations and procedures. However, the data models and ontologies with conventional metadata have shortcomings to convey common descriptions on procedures among data elements, devices, and systems. Accordingly, this paper presents a concept of a new type of metadata, so-called procedural metadata. The goal of procedural metadata is to provide the common descriptions on composable procedures of not only individual devices but also smart systems as a whole based on existing data models and ontologies. With procedural metadata, a device or a system can make decisions and perform tasks automatically by cooperating with heterogeneous devices and systems. Since procedural metadata helps different types of data, devices, and systems to collaborate as long as they support micro-formatted metatag structures, it facilitates a high-level of interoperability in WoT.

3.1 Concept of procedural metadata

In this paper, smart systems indicate IoT environments consisting of one or more connected devices. Applicable data models and ontologies are not restricted to one type of solution, thus they are not specified in this paper. It is assumed that the common descriptions on the data and device level are accomplished based on previous and ongoing works, for example, W3C's TD and SSN. Instead, procedural metadata aims to achieve a higher level of interoperability, with common descriptions on procedures. Heterogeneous types of data, devices, and systems can engage interoperably and support automation by composing a set of procedures described in granular components of data, devices, and systems.

Procedural metadata is a script type of data that describes event, condition, and action-based procedures of smart systems. It describes composable logic, functions, and workflows among data, devices, and systems, thus they can interoperably and automatically engage together to make decisions and perform tasks. To empower semantics, WoT data is often micro-formated and structured in parsable forms so that the data can be expressed and managed with their relationships and properties in granular scales as described in Fig. 1. The procedural metadata utilizes the microdata and provides the composable knowledge to make decisions and perform tasks by expressing the procedural relations among data elements, devices, and systems in granular scales. As procedural metadata is designed to work based on tag data rather than the entire raw data, it has advantages to processing time and complexity. Consequently, procedural metadata facilitates data, devices, and systems to be fully or partially composed efficiently. As described in Fig. 2, a set of procedural metadata can be composed as a part of data, devices, and systems for complex decision making and performing tasks.

Moreover, as a form of metadata, procedural metadata can be not only internally inherent in but also externally accompanied with data and devices. Therefore, the concept of procedural metadata can be easily integrated on the top of the previous and ongoing work without much overhead. For an example, the concept of procedural metadata can be integrated with W3C's TD as shown in Fig. 3. In this case, procedural metadata can be externally stored and managed in a repository just like TD, and the procedural metadata can be utilized by the WoT Scripting API as needed.

3.2 Principle of procedural metadata

In order to not only support interoperability but also manage procedural metadata efficiently and accessibly, there should be basic principles in generation, organization, and utilization of procedural metadata. Some underlying principles are provided as follows:

- Procedural metadata is not a mandatory part of data, devices, and systems, but it is optional.
- Procedural metadata should not affect data's original formats, models, vocabularies, ontologies, etc.
- The inputs and outputs of each procedural metadata should be clarified so that a set of procedural metadata can be composed through output-input connections.
- The inputs and outputs of procedural metadata need to be a set of IoT resource or data and a sort of action needed to be taken, including actuating, sensing, data managing, etc.
- As an executable script-type data, procedural metadata needs to be described based on sets of conditions and actions of IoT re-

sources, vocabularies, relationships, and controls both when it is managed internally and externally.

- If inserted in data internally, procedural metadata needs to be marked with proper tags so that it is efficiently searchable and parsable.
- Internal embedment in data or a device is recommended when the procedure is specific to a certain data, device, system, or application domain.
- For generic usage, it is recommended to manage procedural metadata externally and in universal vocabularies.
- For the procedural metadata externally managed in a repository, the modification made after the first creation and its version should be specified.
- The maintenance policies should be articulated, especially the manner in which access, change, usage, etc. of the procedural metadata are allowed.

3.3 Feasibility and security

Procedural metadata can be both internally and externally managed as a part of the data itself or accompanying data. Therefore, the memory consumption and time complexity overheads can be managed flexibly according to requirements and specifications. As described in the upper part of Fig. 4, the procedural metadata can be directly attached to the devices and data published by the devices.

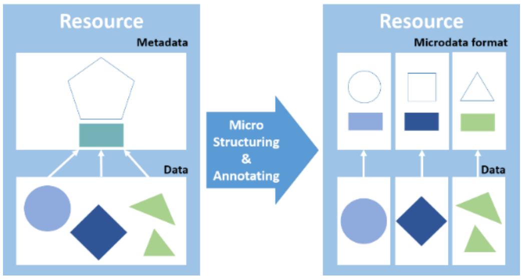


Fig. 1 – Concept of microdata structuring and annotation

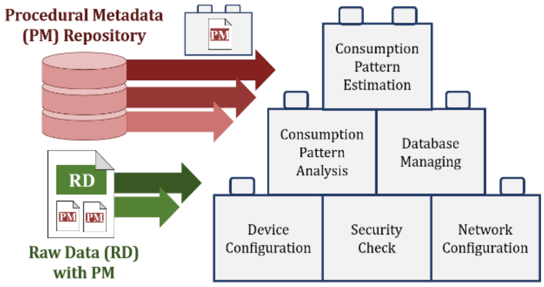


Fig. 2 – Example of procedural metadata composing

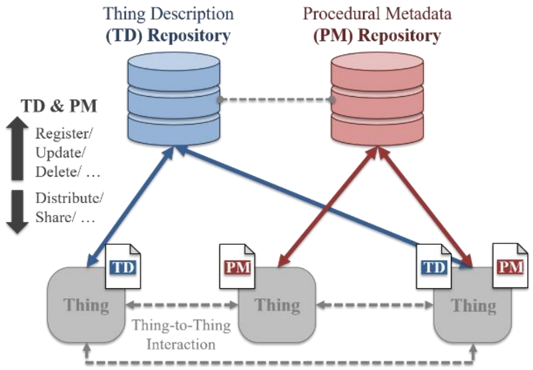
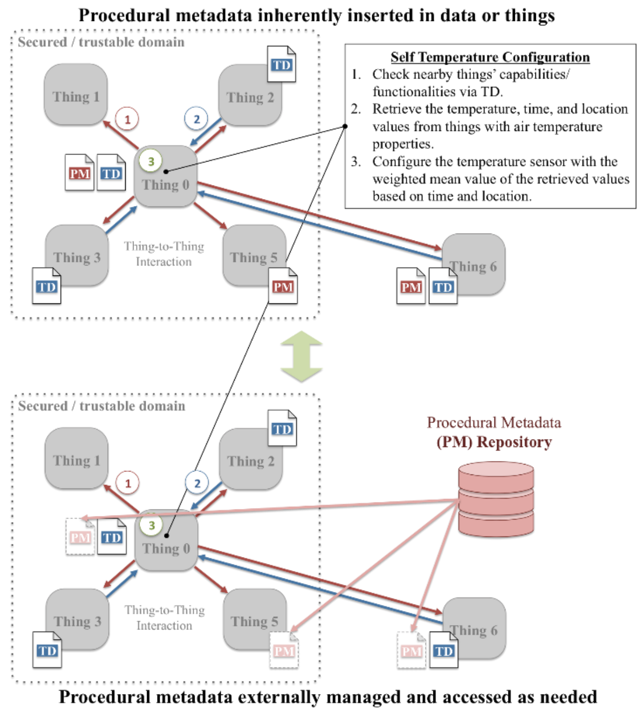

Thing Description (TD) Repository
Procedural Metadata (PM) Repository
TD & PM
Register/ Update/ Delete/ …
Distribute/ Share/ …
TD
PM
TD
PM
Thing
Thing
Thing
Thing-to-Thing Interaction

Fig. 3 – Applying procedural metadata in W3C's TD concept

However, in order to reduce the memory consumption of devices and overheads of the data, the procedural metadata can be externally managed out of data, a device, and a system, and it can be accessed and applied based on the needs shown in the lower part of Fig. 4. Moreover, the predefined procedures can be flexibly extended according to needs while providing skeletons of the common descriptions of procedures.

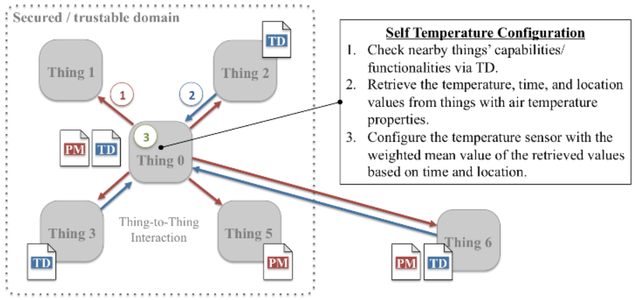
Security is another critical issue of IoT. Especially, in the case of procedural metadata, security can be vulnerable since it supports modifications of data, devices, and systems. Hence, the security issue has to be considered along with interoperability. For data, devices, and systems that adopt procedural metadata, security can be supported by controlling the access, which can be managed with procedural metadata as well. As in the example in Fig. 5, every object needs to be cleared for each action if it wants to access a certain system domain to read or write on data, devices, or the system. That object


Procedural metadata inherently inserted in data or things
Secured / trustable domain
Thing 1
Thing 2
Thing 0
Thing 3
Thing 5
Thing 6
Thing-to-Thing Interaction
1
2
3
PM
TD
Secured / trustable domain
Procedural Metadata (PM) Repository
Procedural metadata externally managed and accessed as needed

Self Temperature Configuration

1. Check nearby things' capabilities/ functionalities via TD.
2. Retrieve the temperature, time, and location values from things with air temperature properties.
3. Configure the temperature sensor with the weighted mean value of the retrieved values based on time and location.

Fig. 4 – Example of procedural metadata embedment



Self Temperature Configuration

1. Check nearby things' capabilities/ functionalities via TD.
2. Retrieve the temperature, time, and location values from things with air temperature properties.
3. Configure the temperature sensor with the weighted mean value of the retrieved values based on time and location.

Fig. 5 – Example of security ensuring procedural metadata

needs to be already affiliated with and registered in the secured and trusted domain, then it can access the system according to its clear level. Otherwise, that object must go through security and trust appreciations according to appropriate procedural metadata. By denying unsecured or untrusted access, the data, devices, and systems can ensure their security.

4. PROOF OF CONCEPT

The proof of concept is implemented in Matlab based on a scenario analysis in a smart-home environment. The feasibility of the proposed metadata to be applied to the web are shown as well. Self-defined metatags are used to ease the implementation process. However, different types of data formats, models, languages, vocabularies, ontologies, etc. can be used for procedural metadata as long as they support a micro-formatted tag structure. Hence, the procedural metadata used in the scenario analysis and implementations is provided in the form of a pseudo code to describe the execution workflows.

4.1 Scenario analysis

A scenario analysis in a smart-home environment is provided to validate the applicability of the procedural metadata concept. This scenario describes how metatags and procedural metadata can be used for automatic electricity consumption pattern estimation.

4.1.1 System environment

The system environment is designed to have data storage (database), a convergence system (data analyzer and organizer), and actors (smart home, system services, and 3rd party service providers). Connected devices in the smart home generate data, which is transferred to a database and data analyzer according to procedural metadata. The data analyzer processes the metatags and their values, with raw data if needed, to examines the relations, rules, and policies as described in the procedural metadata. Based on data analysis results and procedural metadata, the data organizer creates, updates, and manages the metatags, then it makes decisions on the current conditions. According to analyzed contexts, the system service and 3rd party service providers take proper actions with up-to-date descriptions of the current situations.

4.1.2 Consumption pattern estimation scenario

For the electricity consumption pattern estimation task, it is assumed that an advanced metering infrastructure (AMI) device exists in the smart-home environment and that it periodically collects the electricity consumption data.

With the collected data, the data analyzer performs estimation tasks according to the procedural metadata as follows:

- Retrieve pattern and peak time tag data for the past 7 days.
- Organize the tag data into a sequence.
- Search the top 5 similar sequences in the database.
- Choose the pattern value of the day after the end of each sequence as a candidate pattern.

- Use K-nearest-neighbourhood (Knn) method to select the most probable pattern candidate.

Then, the data organizer generates, updates, and modifies the tag values, then takes actions or makes decisions based on the procedural metadata as the follows:

- Generate/update the estimated pattern tag and insert the most recent analysis result.
- Organize the tag values and update the database with changed information.
- Take actions specified in the procedural metadata, such as actuate an AMI device to shut down the electricity, notify the current situation to the service providers, or even call other procedural metadata.

The overall steps of this scenario are described in Fig. 6. The procedural metadata of this scenario analysis is depicted as a pseudo-code since the procedural metadata is an executable metadata coherent with other web technologies, which is not limited to a single form.

4.2 Proof of concept implementation

As a proof of concept, the functionalities of the analyzed scenario are implemented in Matlab, and applicability of the procedural metadata concept on the web is validated. The dataset used in the implementations is real world AMI data metered for 900 days during 2014–2016 in Gwangju, South Korea. It contains 700 low voltage residential users, and the electricity consumption load data is recorded every 15 minutes.

Without processing the original raw data, the implementations use defined procedural metadata in a form of execution code to perform pattern estimation tasks only with the preprocessed metatags: consumption-pattern tag and peak-time tag. The tags are categorized, processed, and assigned in advance according to several criteria through statistical analysis and machine learning techniques. The consumption-pattern tags are acquired through clustering [20]. The peak-time tags are obtained with a peak searching process. The entire dataset is organized with proper consumption-pattern and peak-time tags before implementation. In the implementation, only the tags are used for electricity consumption pattern estimation.

4.2.1 Concept implementation in Matlab

In Matlab, the procedural metadata concept is implemented as a function call. The main code calling an external function code is considered as a system retrieving and activating procedural metadata from an external repository. By utilizing daily consumption-pattern tags and peak-time tags of previous days as inputs, the main code performs pattern estimation based on the pseudo-code described in Table 3. Fig. 7 shows a part of the generated results as an example.

4.2.2 Validation on the web

To show the validity of the proposed concept of procedural metadata on the web, the tags used for the example implemented in Matlab are validated in web environments. Fig. 8 is the results of pattern estimation shown on the web to prove the applicability and feasibility of the procedural metadata concept on the web. As mentioned above, different types of web data model languages, vocabularies, ontologies, etc. can be used for procedural metadata as long as they support micro-formatted tag structures. Therefore, this proof of concept implementation can be generalized for various web environments.

5. CONCLUSION

In this paper, it is argued that establishing common descriptions of roles of data and devices in smart systems, specifically in the part of decision making and performing tasks, is the driver in the development of autonomic and interoperable WoT. To support WoT with common descriptions of logic and workflows, this paper proposed a concept of procedural metadata. As a type of metadata containing executable procedures to make decisions and perform tasks, procedural metadata provides accumulated knowledge information for interoperable, autonomic systems. This information can be shared by heterogeneous devices and systems which utilize different types of data formats, languages, vocabularies, ontologies, etc. as long as they support micro-formatted tag structures available on the web. Therefore, heterogeneous IoT environments can be integrated partially and entirely through the proposed concept. Hence, procedural metadata enables automatic and interoperable data, device, and system engagements in WoT. The viability and applicability of the proposed concept are shown through scenario analysis and proof of concept implementations. Due to the characteristics of procedural metadata that it exploits data and devices and facilitates functionality modification, feasibility and security are the critical issues. However, it is discussed that these issues can be resolved by managing the procedural metadata both internally and externally and registering secured and trustable devices and domains. For future work, the study needs to proceed on enhancing security and trust appreciation algorithms specifically for devices and systems that adopt procedural metadata.

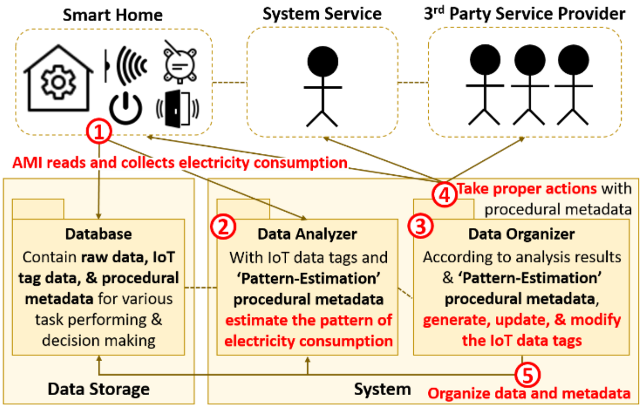

Smart Home
System Service
3rd Party Service Provider
① AMI reads and collects electricity consumption
④ Take proper actions with procedural metadata
Database
Contain raw data, IoT tag data, & procedural metadata for various task performing & decision making
② Data Analyzer
With IoT data tags and 'Pattern-Estimation' procedural metadata estimate the pattern of electricity consumption
③ Data Organizer
According to analysis results & 'Pattern-Estimation' procedural metadata, generate, update, & modify the IoT data tags
Data Storage
System
⑤ Organize data and metadata

Fig. 6 – Scenario for pattern-estimation procedural metadata

Table 3 – Example of pattern-estimation procedural metadata

Pattern-estimation

```
To perform pattern estimation (Knn method)
For previous 7 days from the day to be estimated
    check <data>-<pattern>
    check <data>-<peak>-<max>-<time>
    do make them as a sequence
For n = 1~5
    find the n^th similar pattern and peak time sequences
    check <data>-<pattern> of the day after the sequences
    if the found <data>-<pattern> is also unkown
        repeat
    else
        <pattern>-<candidate>-<n> = found <data>-<pattern>
<data>-<pattern>-<estimated> = mode(<pattern>-<candidate>)
condition to <data>-<pattern>-<estimated> do <action>
```

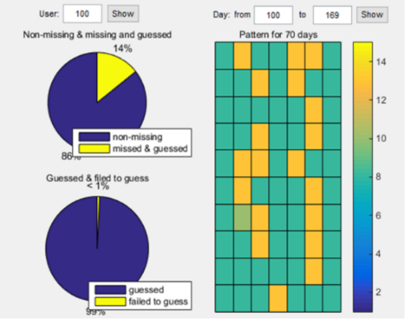
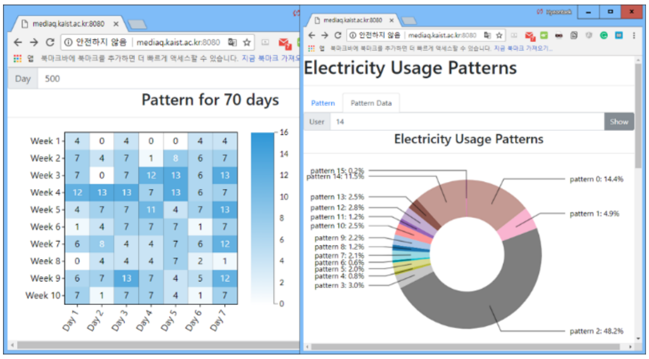


Fig. 7 – Estimation result with procedural metadata in Matlab



	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7
Week 1	4	0	4	0	0	4	4
Week 2	7	4	7	1	8	6	7
Week 3	7	0	7	12	13	6	13
Week 4	12	13	13	7	13	6	7
Week 5	4	7	7	11	4	7	13
Week 6	1	4	7	7	7	1	7
Week 7	6	8	4	4	7	6	12
Week 8	0	4	4	4	7	2	1
Week 9	6	7	13	7	4	5	12
Week 10	7	1	7	7	4	1	7

Fig. 8 – Estimation result with procedural metadata on web

ACKNOWLEDGEMENT

This work was supported by the Institute for Information & Communications Technology Promotion (IITP) grant funded by the Korea government (Ministry of Science and ICT). (No.2018-0-00261, GDPR Compliant Personally Identifiable Information Management Technology for IoT Environment)

REFERENCES

[1] D. Miorandi, S. Sicari, F. De Pellegrini, and I. Chlamtac, (2012). Internet of Things: Vision, applications and research challenges. Ad Hoc Networks, vol. 10, no. 7, pp. 1497–1516, 2012.

[2] I. Jacobs, N. Walsh, (2004). Architecture of the World Wide Web, Volume One. W3C. https://www.w3.org/TR/webarch/

[3] D. Raggett, (2015). The Web of Things: Challenges and Opportunities. Computer, vol. 48, no. 5, pp. 26-32, May 2015.

[4] K. Kajimoto, M. Kovastsch, U. Davuluru, (2017). Web of Things (WoT) Architecture. W3C. https://www.w3.org/TR/2017/WD-wot-architecture-20170914/

[5] A. Kamilaris, S. Yunusak, M. I. Ali, (2016). WOTS2E: A Search Engine for a Semantic Web of Things. World Forum on Internet of Things (WF-IoT), 12-14 Dec. 2016.

[6] F. Scioscia, M. Ruta (2009). Building a Semantic Web of Things: issues and perspectives in information compression. International Conference on Semantic Computing (ICSC). 14-16 Sept. 2009.

[7] M. Janssen, E. Estevez and T. Janowski, Interoperability in Big, Open, and Linked Data--Organizational Maturity, Capabilities, and Data Portfolios. Computer, vol. 47, no. 10, pp. 44-49, Oct. 2014.

[8] J. Riley (2017). Understanding Metadata: What is Metadata, and What is it For?: A Primier. National Information Standards Organization (NISO). Jan. 2017.

[9] M. Compton, P. Barnaghi, L. Bermudez, R. GarciaCastro, O. Corcho, et al.. (2012). The SSN ontology of the W3C semantic sensor network incubator group. Web Semantics: Science, Services and Agents on the World Wide Web, vol. 17, pp. 25–32, 2012.

[10] M. Ganzha, M. Paprzycki, W. Pawlowski, P. Szmeja, K. Wasielewska, (2016). Semantic technologies for the IoT – an Inter-IoT perspective. International Conference on Internet-of-Things Design and Implementation (IoTDI), 4-8 Apr. 2016.

[11] M. Bermudex-Edo, T. Elsaleh, P. Barnaghi, K. Taylor (2017). IoT-Lite: a lightweight semantic model for the internet of things and its use with dynamic semantics. Personal and Ubiquitous Computing, vol. 21, no. 3, pp. 475-487, Feb. 2017.

[12] FIESTA-IoT, FIESTA-IoT Concept. http://fiesta-iot.eu/index.php/iot-experiments-as-a-service/

[13] INTER-IoT, INTER-IoT Apprach. http://www.inter-iot-project.eu/approach

[14] V. Charpenay, J. Hund, S. Kabisch, T. Kamiya, M. Koster, (2018). WoT Current Practices. W3C. Mar. 2018. http://w3c.github.io/wot/current-practices/wot-practices.html

[15] S. Kaebisch, T. Kamiya, (2018). Web of Things (WoT) Thing Description. W3C. Apr. 2018. https://www.w3.org/TR/wot-thing-description/

[16] Z. Kis, K. Nimura, D. Peintner, J. Hund, (2018). Web of Things (WoT) Scriting API. W3C. Sept. 2018. https://www.w3.org/TR/wot-scripting-api/

[17] A. E. Kjaled, A. Helal, W. Lindquist, C. Lee, (2018). IoT-DDL-Device Description Language for the "T" in IoT. IEEE Access. vol. 6. pp. 24048-24063, Apr. 2018.

[18] A. Palavalli, D. Karri, S. Pasupuleti, (2016). "Semantic Internet of Things." IEEE Tenth International Conference on Semantic Computing (ICSC), 4-6 Feb. 2016.

[19] V. V. Patel, J. D. McGregor, S. Goasguen, (2010). SysML-Based Domain-Specific Executable Workflows. IEEE International Systems Conference. 5-8 Apr. 2010.

[20] N. Kim, S. Park, J. Lee, J. K. Choi, (2018). Load Profile Extraction by Mean-Shift Clustering with Sample Pearson Correlation Coefficient Distance. Energies, vol. 11(9), 2397, 2018.