Cultivating Service Knowledge Models for IoT-Based Systems Adaptability

Aradea, Rianto* and Husni Mubarok
Email: aradea@unsil.ac.id, rianto@unsil.ac.id, husni.mubarok@unsil.ac.id
*Coresponding Author

Department of Informatics, Faculty of Engineering
Siliwangi University, Kode Pos 46115, Tasikmalaya, Indonesia.

Keywords: knowledge meta-model, service knowledge, self-adaptive service, IoT systems

Received: December 15, 2021

Service models have been widely developed and applied to the Internet of Things (IoT) systems. However, current service models tend to emphasize the need for various types of services based on certain IoT service domains. Hence, the limitations of this service model are not prepared to meet the general objectives of IoT-based systems so that services cannot adapt to various IoT domains. Besides, developers should redefine service requirements and specifications. This paper introduces a service knowledge model, where meta-model elements are defined more generically. The control loops pattern of a self-adaptive model as a service-forming component and behavior regulator are deployed as the investigative approach. The developed service knowledge model encompasses five main classes, nine sub-classes, twelve object properties, and eighty-nine axioms. Meta-model evaluation results revealed that the level of completeness and consistency of 100% related to the structure, language, and syntax of a knowledge model. Additionally, the proposed model has an architecture adaptability index (AAI) level = 0.89. Hence, it can reduce the uncertainty of IoT services at runtime.

Povzetek: Model znanja o storitvah je oblikovan kot vzorec prilagajanja krmilne zanke in kontekstualno znanje za prilagodljivost različnim domenam storitev interneta stvari.

1 Introduction

Generally speaking, the Internet of Things (hereafter, IoT) is integrating a variety of processes, such as identifying, sensing, networking, and computing so that it provides added values for personalized services by users' interaction with a variety of "things" [1]. Thus, IoT systems are distributed services and collaborate with others [2]. Currently, several IoT applications can be categorized into various domains, such as smart cities, smart metering, process automation, remote monitoring, retails, agriculture, logistics, health, traffic, etc. [1]. However, the system requirements for diverse types of IoT application domains remain ill-defined [2]. Park et al. [3] contend that developing adaptive IoT services through existing frameworks requires a lot of time and effort since most frameworks are limited to a single system. In addition, IoT services are generally very dependent on certain domains and are not easy to be referred to by other domains [3]. This is related to how the service models can meet the specifications of miscellaneous IoT service domains. This situation demands that the IoT service models should possess adaptability.

Numerous researchers have proposed IoT service models. As an example, Urbieta et al. [4] introduced the adaptive service composition framework to support dynamic reasoning. Service models were represented as context-aware specifications so that they were capable of thinking related to user tasks. Likewise, Service models indicated service behaviors that deal with environmental diversity. In another instance, Chatfield et al. [5] promoted an IoT framework about performance with case studies analyzing digital technology, IoT cybersecurity, etc. This work focuses on one IoT domain, namely IoT smart governance. Furthermore, Mocrii et al. [6] have explored elements in IoT (e.g. IoT-based architecture, components of management systems, communication technologies, privacy, and security). This work accentuates IoT-based smart homes. Ahmed et al. [7] maintained that one of the main challenges in managing IoT services currently is service adaptation. This is needed because IoT systems are fast-changing, heterogeneous, highly dynamic, and subject to risks and failures. With this in mind, the results of system development should have the ability to adjust at runtime. Maurya et al [8] proposed an IoT architecture and algorithm based on Servive Oriented Architecture (SOA) to address energy efficiency in IoT. In the simulation only the necessary devices will be active while the others remain in an inactive state, resulting in a more efficient use of energy.

Researchers have overcome problems of developing IoT systems by adopting a Model-Driven Development (MDD) approach. For example, Sosa-Reyna et al. [9] proposed a method based on MDD and Service-Oriented Architecture (SOA). In this case, they assumed that such

a method enables to guide the process of fostering service-oriented applications from the conceptual model to the application code and selected technology platform. In another study, Hussein et al. [10] applied MDD to capture system functionality and adaptation requirements. System functionality is modeled with the SysML4IoT profile. This system modeled the required data by following the publish/subscribe paradigm while the runtime adaptation model adopts a state machine approach. Likewise, Brambilla et al. [11] developed a user interface for the IoT system based on the MDD approach through the expansion of OMG's standard IFML. Also, it implemented a code generator prototype for developing IoT applications. However, the scrutiny focuses on the MDD approach tends to be more concerned with the process of transforming high-level models into software code. To realize system adaptability, it is also necessary to define a generic design pattern to control the requirements of various domains.

Recently, Park et al. [3] proposed a cloud-based middleware framework for self-adaptive IoT collaboration services grounded in the MAPE cycle (Monitor, Analyze, Plan, Execute). This cycle is a generic design pattern for adaptation requirements at run-time. MAPE is deployed to each IoT component to separate components depending on the domain from the existing framework layers. The proposed framework focuses on providing reference architecture. While the requirements elicitation for specifying IoT service artifacts (knowledge) have not been covered. Dealing with this, Achtaich et al. [12] designed a framework for smart IoT adaptations as a methodology in the stages of requirements elicitation, identifying dimensions for IoT systems, selecting a self-adaptation mechanism, and designing a fleet as a DSPL (Dynamic Software Product Lines). Unfortunately, this framework still requires further validation and its implementation in various domains, including defining a generic reference architecture.

Yen et al. [13] have tried to cultivate a more generic IoT service model. It extends the software service model to support the specifications of IoT services and things. Conversely, adaptability factors have not been addressed in the scope of this study. In a similar vein, Yen et al. [14] followed up on their work [13] by analyzing the differences between IoT and software services to define IoT service requirements in the software service model. This expands the OWL-S service for the IoT-specific service model. Moreover, it accentuates IoT service discovery based on discovery routing approaches. Further, the requirements for the self-adaptive mechanism of IoT services are not the main focus. Table 1 shown related work with using approach of each proposed model.

Grounded in the abovementioned investigative issues, there is still a gap motivating to conduct further scrutiny, namely how to develop a knowledge model as requirements for an IoT service that can adapt to various domains. Therefore, the present study aims at introducing an IoT service knowledge model developed based on a self-adaptive systems approach [15] [16]. In this case, the adaptation pattern of control loops and context elements become the main class in the form and regulating service

Author	Model	Description
Urbieta et al. [4]	Adaptive service composition framework: context-aware, user task, service behaviours, dynamic reasoning	Focus on one of services IoT domain
Chatfield et al. [5]	IoT framework: digital technology analysis, IoT cybersecurity, etc.	Focus on IoT smart governance
Mocrii et al. [6]	IoT-based architecture: management systems, communication model, privacy and security	Focus on IoT-based smart homes
Sosa-Reyna et al. [9]	IoT-MDD model: model-driven development, service-oriented architecture	Focus on development of IoT Achitecture
Hussein et al. [10]	IoT systems: model-driven development, SysML4IoT, state machine approach	Focus on identification of IoT system functionality
Brambilla et al. [11]	IoT systems: user interface, model-driven development, OMG's standard IFML, code generator	Focus on transformation of high level model to software code
Park et al. [3]	Cloud-based middleware framework for IoT: IoT collaboration services, MAPE control loops	Focus on provision of architecture reference
Achtaich et al. [12]	IoT framework: smart IoT adaptations, dimensions for IoT systems, dynamic software product lines	Focus on methodology (requirements, identification, and mechanism)
Yen et al. [13][14]	Generic IoT service model: specifications of IoT services and things, OWL-S service, IoT service discovery	Focus on requirement of software IoT services
Supriana et al. [30]	Self-adaptive cyber-city system (SACCS): belief-desire-intention (BDI) model, event-condition-action, IoT policy engine, MAPE-K control loops	Focus on adaptability needs of cyber-city system based on IoT services artifact
Nasiri et al [31]	IoT Layer Architecture: perception layer, network layer, and application layer	Focus on IoT-based IoT integrated healthcare system
Benkhaled et al [32]	Ontology IoT model : semantic interoperability, contextual approach, Cross-Domain Ontology	Focus on reasoning across IoT domains and infer new knowledge required in cross-domain applications

Table 1: Related works.

behavior. Additionally, the organization of this study covers the second section which describes the proposed model, the third section which designates the application of the model, the fourth section which contains empirical evaluations, and the last section which draws conclusions and future directions of this study.

2 The proposed method

The proposed method lied in an ontological knowledge base adapted from a joint statement of TBox and ABox meta-model [17] [18] [19]. Service knowledge was developed as a scheme (TBox) depicting a set of concepts from service element properties and adaptation behavior regulators. Additionally, IoT domains represented instances of TBox generated from an IoT application. More specifically, Figure 1 illustrates service knowledge in an ontological knowledge base.

The service knowledge meta-model consists of service artifacts, control loops, and context. The service artifact is adapted from [13]. At the same time, the control loops and context components are an extension of our previous work [20] [21] [22]. Service knowledge with adaptability was developed from two main models, namely TBox and ABox:

a. TBox is terminological knowledge describing the general properties of a concept.
b. ABox is assertional knowledge that is specific to individuals from the system domain.

TBox is defined to represent all service artifacts (e.g. all concepts related to service knowledge). In particular, metadata is all attributes and relationships between services and objects deployed to generate ABox [19]. The developed service artifact comprises forming elements of model requirements that correspond to concepts or classes in ontology. Besides, it is divided into three superclasses, namely service artifact (process, profile, and grounding), control loop (monitor, analyze, plan and execute), and context (entity, dimension, and status) as outlined subsequently:

a. the process is a declarative statement of service operations, including service composition and monitoring [13],
b. the profile is a declarative statement of functional property (input, output, prerequisite, and effect) and non-functional services (categorization and quality rating) [13],
c. grounding is a declarative statement of technical details for accessing services, including mapping the attributes of specific domain entities to the property that can be observed by sensors [13],
d. monitoring is a class for monitoring and collecting data/context information based on the system environment [23],
e. analyzing & planning is a class for analyzing and planning actions that are appropriate for the system [23],
f. executing is a class for determining adaptation actions at run-time [23].
g. contexts are abstractions of relevant domain properties based on a set of environmental assumptions [24][25],

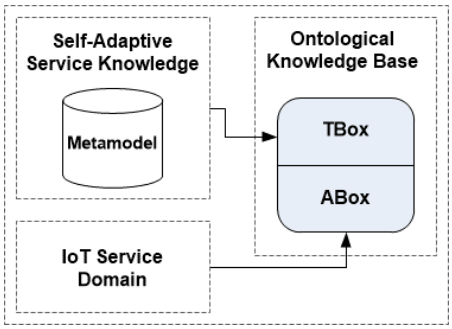


Figure 1: Service knowledge based on an ontological knowledge base.

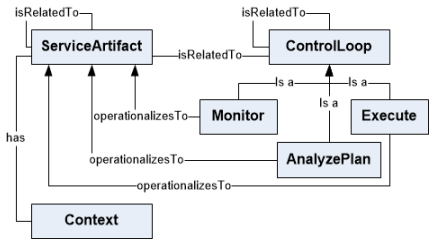
Classes	Meta-model Fragment
Super classes: Things. Sub classes: Service Artifact; Context; Control Loop.	

Table 2. Service knowledge meta-model

h. context variability dimension is an aspect representing changes in domains at run-time [25],
i. context status is an expression to define the status of a domain in certain circumstances (e.g. activating and deactivating) [25].

Table 2 displays the relationships between classes of service knowledge meta-model through various object properties and data properties. Service artifact class serves to capture the instance or concrete IoT service encompassing three subclasses, namely process, profile, and grounding. Service classes can have contexts representing a set of IoT environmental assumptions. This class has three subclasses, namely dimension, service entity, and service status. Each class is connected to the control loops class to realize self-adaptive service capabilities.

Restrictions for service knowledge meta-models are displayed in Table 2:

a. *hasControlLoop* some ControlLoop
b. *hasMonitor* some Monitor
c. *hasAnalyzePlan* some AnalyzePlan
d. *hasExecute* some Execute
e. *hasContext* some Context
f. *hasServiceReason* some ServiceReason
g. *operationalizesTo* some ServiceArtifact
h. *isConflict* some ServiceArtifact
i. *isMandatory* some boolean
j. *isOptional* some boolean
k. *isValid* some boolean

The control loops class covers three subclasses, namely monitor, analyze, plan, and execute. This class captures service descriptions and manages its operations based on the context that applies to IoT service activation.

Domain	Properties	Range
ServiceArtifact	hasControlLoop	ControlLoop
ControlLoop	operationalizesTo	SeviceArtifact
ServiceArtifact	hasMonitor	Monitor
ServiceArtifact	hasAnalyzePlan	AnalyzePlan
ServiceArtifact	hasExecute	Execute
ServiceArtifact	hasContext	Context
Context	isContext	ServiceArtifact
ServiceEntity	hasEntity	ServiceArtifact
ServiceArtifact	isConflict	ServiceArtifact
	throwErrorException	ErrorException
ServiceArtifact	hasServiceReason	ServiceReason
ServiceReason	isServiceReason	ServiceArtifact

Table 3: Object properties.

Domain	Properties	Range
ServiceArtifact	isMandatory	Boolean
ServiceArtifact	isOptional	Boolean
ServiceReason	isValid	Boolean

Table 4: Data properties.

Service knowledge relationships represent relationships between classes or instances (object properties) and relation instances in classes with data values (data type properties). Tables 3 and 4 reports a summary of the object properties and data type properties. Based on this meta-model design, service knowledge has been directed as self-adaptive service systems where IoT service elements are captured through service artifact classes and context classes. On the other hand, self-adaptive behavior is realized through control loops classes. Thus, this meta-model forms a self-adaptive service knowledge.

3 Result

Self-adaptive service knowledge for IoT service domains was developed using Protégé 5.2.0. Class taxonomy modeling. To illustrate, Figure 2 and Figure 3 delineated the visualization of the structures and their relations. There were a number of predominant classes through root Thing and each class has its sub-classes. Regarding this, the ControlLoop class is equivalent to the ServiceArtifact class, while the Context class is equivalent to the ServiceEntity class. These classes represent class equivalents formulated to speed up the query process between classes. As a matter of fact, querying an instance in a particular class allows for other equivalent classes. Similarly, the specifications of a particular class can be defined by other equivalent classes based on the applicable restrictions.

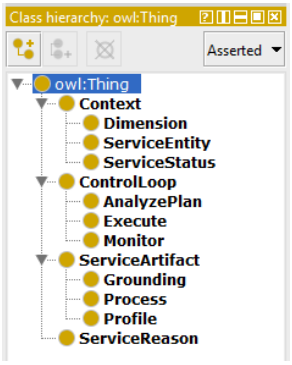


Figure 2: Class hierarchy of self-adaptive service knowledge.

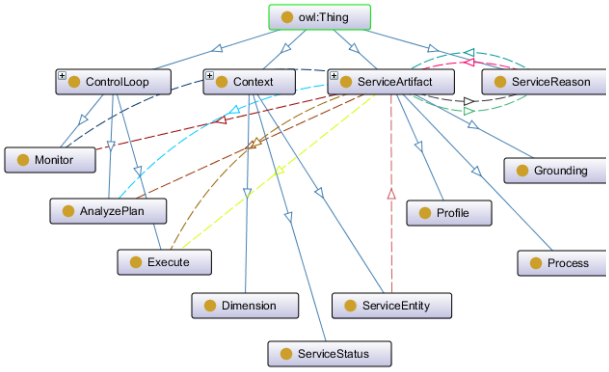


Figure 3: OntoGraf of self-adaptive service knowledge.

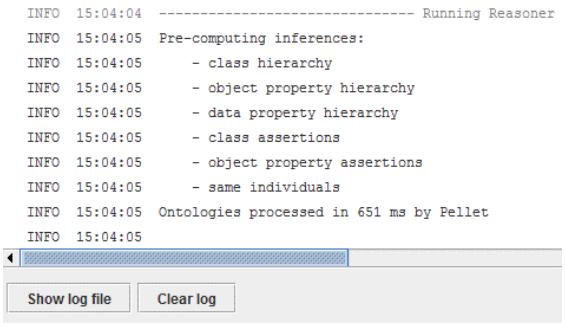


Figure 4: Reasoning logs based on Pellet.

The meta-model for IoT service adaptability encompasses five major classes, nine sub-classes, twelve object properties, three data properties, and ninety-one axioms. The evaluation of the meta-model was performed by verifying the inference process and observing the consistency of structure, language, and syntax. The evaluation process was executed by utilizing the Pellet plugin through a reasoning process in the Protégé 5.2.0 tools. More technically, Figure 4 reported results that are consistent and complete. In other words, there are no errors in terms of structure, language, and syntax in the knowledge model. In addition, the inference process to the class and its relations (both relations between classes/ instances and their relationships with data) revealed that no errors were detected.

No	Element	Description
1	Purpose	Evaluating responses and fragment constraints for the variability of IoT services in the model. Evaluating the adaptability of each element in the model.
2	Object of study	Service specifications in the model. Service artifact in the model.
3	Domains	IoT Service (Cyber-City) System.
4	Focus	Adaptation strategies in the model. Adaptability specifications of each element in the model.
5	Research question	How do the responses and fragment constraints for the variability of IoT services in the proposed model compare with other models? How is the adaptation capability of IoT services in the proposed model compared to other models?
6	Variables (V)	Responses (V1-service access failure; V2-service application error; V3-service provider failure; V4-new stimulus). Fragment constraints (V5-mandatory; V6-variants specific). Architecture adaptability index (V7-AAI).

Table 5: The design of an empirical evaluation.

4 Empirical evaluation

The approach utilized in evaluating activities is to adopt empirical evaluation guidelines from Wohlin et al. [26]. In particular, the guideline employed the domain Quality Attribute Scenarios (dQAS) [27][28] and i* Metrics Definition Framework Method (iMDFM) [29]. On the other hand, dQAS was exerted as a quality attribute to evaluate service variability features while iMDFM was applied to measure the adaptability of each service element. Table 5 deciphered the experimental evaluation design. More specifically, three quality attributes (response (R), fragment constraints (FC), and architecture adaptability index (AAI)) are compared as the dependent variable. On the other hand, the independent variable is the model being compared (e.g. the proposed model in this research) and the model of the results of previous studies (e.g. Self- Adaptive Cyber-City System (SACCS) [30].

The purpose of this evaluation is to obtain empirical evidence of adaptability in the IoT environment to deal with uncertainty based on the design support provided by each model. An overview of IoT service modeling for cyber-city systems [29] based on the goal model approach (Figure 5 and Table 6) revealed the quality attribute (dQAS) scenario mapped from the IoT service modeling as exemplified in Figure 5.

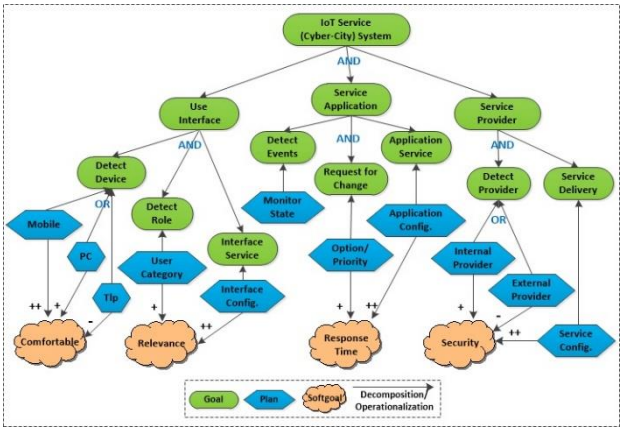


Figure 5: Modeling IoT services based on the goal model.

No	dQAS Element	Description
1	Source (SO)	[SO1] User interface sensor; [SO2] Application sensor; [SO3] Service provider sensor
2	Stimulus (ST)	Service access failure: [ST1] Sensor device failure; [ST2] Changes to user roles Service application error: [ST3] Application error; [ST4] Error for request change Service provider failure: [ST5] Service mismatch; [ST6] Failure to change service New Stimulus: [ST7] Failure to define a new service context
3	Artifact (A)	[A1] Fragments of the user interface [A2] Fragments of the service applications [A3] Fragments of the service providers
4	Environment (E)	[E1] The system is running/run-time [E2] The system is being designed (design-time)
5	Variants (V) & Valid QAS Configurations (VC)	[V1] Detect device; [V2] Detect role; [V3] Interface service; [V4] Detect events; [V5] Request for change; [V6] Application service; [V7] Detect provider; [V8] Service delivery [VC1] V1 ∧ V2 ∧ V3 [VC2] V4 ∧ V5 ∧ V6 [VC3] V7 ∧ V8
6	Response (R)	[R] Activity of the response made by the system after the arrival of the stimulus.
7	Fragment Constraints (FC)	[FC] Constraints of fragment selection for the provision of IoT services.
8	Architecture Adaptability Index (AAI)	[AAI] A measure of the adaptability of IoT services based on the architecture adaptability index.

Table 6: Quality attribute scenarios for cyber-city systems.

dQAS Element	SACCS Model	Proposed Model
Response (R)	[R1] Service access failure is detected from external system properties not registered in the symptom repository (predefined). [R2] Service application error is detected from internal system properties not registered in the symptom repository (predefined). [R3] Service provider failure is detected from service level agreement (SLA) that is not in accordance with (predefined) rules. [R4] New stimulus is detected from operating components that are configured at design-time.	[R1] Service access failure is detected from the service artifact based on the context dimension through the service reason (automatically). [R2] Service application errors are detected from the service artifact based on the context dimension through the service reason (automatically). [R3] Service provider failure is detected from service level agreement based on service reason (automatically). [R4] New stimulus is detected from operating components configured at run-time based on service reason
Fragment Constraints (FC)	[Mandatory] {SO1, SO2, SO3} ∧ {A1, A2, A3} ∧ {E2} ∧ {V7, V8}; [Variant VC1] {ST1, ST2} ∧ {R1} ∧ {E2}; [Variant VC2] {ST3, ST4} ∧ {R2} ∧ {E2}; [Variant VC3] {ST5, ST6} ∧ {R3} ∧ {E2}; [Variant VC4] {ST7} ∧ {R4} ∧ {E2}.	[Mandatory] {SO1, SO2, SO3} ∧ {A1, A2, A3} ∧ {E1} ∧ {V7, V8}; [Variant VC1] {ST1, ST2} ∧ {R1} ∧ {E1}; [Variant VC2] {ST3, ST4} ∧ {R2} ∧ {E1}; [Variant VC3] {ST5, ST6} ∧ {R3} ∧ {E1}; [Variant VC4] {ST7} ∧ {R4} ∧ {E1};
Architecture Adaptability Index (AAI)	[AAI] EAI (goal) = 8 + EAI (task) = 11 + EAI (soft-goal) = 0 / Total elements = 27; AAI = 19/ 27 = 0.70.	[AAI] EAI (goal) = 11 + EAI (task) = 11 + EAI (soft-goal) = 2 / Total elements = 27; AAI = 24/ 27 = 0.89.

Table 7: The results of the comparison of the attributes [R], [FC], and [AAI].

5 Discussion

Evaluation control for each quality attribute, namely [R], [FC], and [AAI] are sketched out in Table 7. The evaluation results informed that based on the existing VC combination, the proposed model responded to each stimulus automatically through the service reason features. On the other hand, the model SACCS does this predefined through a list of actions in the symptom repository. Therefore, the proposed model can respond to new stimuli emerging in the IoT environment at run-time. Further, the SACCS model handles it at design-time re-engaging the role of the software designers.

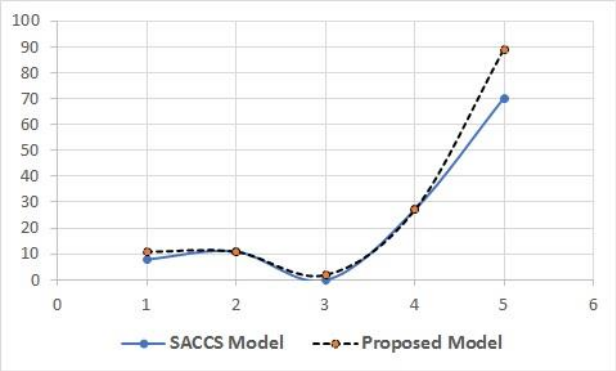


Figure 6: Comparation accuracy level of adaptability model.

Figure 6 shown accuracy of architecture adaptability index (AAI) between proposed model and the comparison model. x axle represent artifact of used IoT services, namely 1:goal; 2: task; 3: soft-goal; 4: EAI (element adaptability index), and AAI. y axle is value index that use to determine adaptability level of compared model. The adaptability of each system element in the proposed model has an AAI level of 0.89, while SACCS = 0.70. The results designated that the proposed model reduced the uncertainty service of IoT caused by variability at runtime better than the SACCS model.

This proposed model is generic so it can be applied for various cases in different IoT service domains. In this case, software developers can define elements of instances in service knowledge showing the ability to adapt based on the behavior of the most suitable IoT service components. Likewise, the defined relationships and attributes have also been prepared to fulfill all adaptability functions in IoT services.

6 Conclusion and further studies

This scrutiny highlights the importance of preparing IoT services with adaptability functions for a variety of IoT domains. The proposed model defines an IoT service based on the self-adaptive systems approach where the forming and controlling elements of the service are fostered through a generic meta-model. The meta-model is formulated as an adaptation pattern of control loops and context elements with classes. Also, it relates to and accommodates the needs of various IoT domains. The evaluation results indicated that the proposed meta-model has a level of completeness and consistency of 100% related to the structure, language, and syntax of a knowledge model. Empirical evaluation results showcase that the proposed model can respond to each stimulus automatically through the service reason feature. On the other hand, the other models (SACCS) [30] are predefined. In addition, the proposed model has an AAI level of 0.89. Meanwhile, other models (SACCS) [30]

implemented as the main comparison are 0.70. As a result, the proposed model can better reduce the uncertainty of IoT services at runtime. For future studies, this model should ideally be followed up by developing a tool that can integrate declarative knowledge with reasoning services mechanisms. Further, some alternatives to the artificial intelligence approach can be considered to enrich this investigation.

Acknowledgements

This investigative project was supported by the Institute for Research, Community Sevices and Education Quality Assurance, Siliwangi University, and Ministry of Research, Technology and Higher Education of the Republic of Indonesia (No. 233/UN58.21/PP/2019).

References

[1] Alem Čolaković, and Mesud Hadžialić (2018). Internet of Things (IoT): A review of enabling technologies, challenges, and open research issues, *Computer Networks*, 144, pp. 17-39. https://doi.org/10.1016/j.comnet.2018.07.017

[2] Leila Fatmasari Rahman, Tanir Ozcelebi, and Johan Lukkien (2018). Understanding IoT systems: a life cycle approach, *Procedia Computer Science*, 130, pp 1057-1062. https://doi.org/10.1016/j.procs.2018.04.148

[3] Soojin Park and Sungyong Park (2019). A cloud-based middleware for self-adaptive IoT-collaboration services, *Sensors (Basel, Switzerland)*, 19(20), 4559. https://doi.org/10.3390/s19204559.

[4] Aitor Urbieta, Alejandra González-Beltrán, Sonia Ben Mokhtar, M Anwar Hossain, and Licia Capra (2017). Adaptive and context-aware service composition for IoT-based smart cities, *Future Generation Computer Systems*, 76, pp. 262-274. https://doi.org/10.1016/j.future.2016.12.038

[5] Akemi Takeoka Chatfield and Christopher G. Reddick (2018). A framework for Internet of Things-enabled smart government: a case of IoT cybersecurity policies and use cases in U.S. federal government, *Government Information Quarterly*, 36(2), pp 1-12. https://doi.org/10.1016/j.giq.2018.09.007.

[6] Dragos Mocrii, Yuxiang Chen, and Petr Musilek (2018). IoT-based smart homes: A review of system architecture, software, communications, privacy and security, *Internet of Things*, 1(2), pp. 81–98. https://doi.org/10.1016/j.iot.2018.08.009.

[7] Abdelmuttlib Ibrahim Abdalla Ahmed, Abdullah Gani, Siti Hafizah Ab Hamid, Abdelzahir Abdelmaboud, Hassan Jamil Syed, Riyaz Ahamed Ariyaluran Habeeb Mohamed, and Ihsan Ali (2019). Service management for IoT: requirements, taxonomy, recent advances, and open research challenges, *IEEE Access*, 7, pp. 155472-155488, https://doi.org/10.1109/ACCESS.2019.2948027

[8] Sudhanshu Maurya and Kuntal Mukherjee (2019). An energy efficient architecture of IoT based on service oriented architecture (SOA), *Informatica*, 43(1), pp. 87-94. https://doi.org/10.31449/inf.v43i1.1790.

[9] Claudia Maricela Sosa-Reyna, Edgar Tello-Leal, David Lara-Alabazares (2018). Methodology for the model-driven development of service-oriented IoT applications, *Journal of Systems Architecture*, 90, pp. 15-22. https://doi.org/10.1016/j.sysarc.2018.08.008

[10] Mahmoud Hussein, Shuai Li, and Ansgar Radermacher (2017). Model-driven Development of Adaptive IoT Systems, *Satellite Events*, MODELS, United States, pp. 17-23.

[11] Marco Brambilla, Eric Umuhoza, and Roberto Acerbis (2017). Model-driven development of user interfaces for IoT systems via domain-specific components and patterns, *Journal of Internet Services and Applications*, 8, p.14. https://doi.org/10.1186/s13174-017-0064-1.

[12] Asmaa Achtaich, Nissrine Souissi, Raul Mazo, Camille Salinesi, and Ounsa Roudies (2017), Designing a Framework for Smart IoT Adaptations, *International Conference on Emerging Technologies for Developing Countries*, Springer, Marrakesh-Morocco, pp. 57-66. https://doi.org/10.1007/978-3-319-67837-5_6

[13] I-Ling Yen, Farokh Bastani, San Yih Hwang, Wei Zhu, and Guang Zhou (2017). From software services to IoT services: the modeling perspective, *International Conference on Serviceology*, Springer, Viena-Austria, vol 10371. https://doi.org/10.1007/978-3-319-61240-9_20

[14] I-Ling Yen, Farokh. Bastani, Wei Zhu, Hessam Moeini, San Yih Hwang and Yuqun Zhang (2018). Service-oriented IoT modeling and its deviation from software services, *IEEE Symposium on Service-Oriented System Engineering (SOSE)*, IEEE, Bamberg-Germany, pp. 40-47. https://doi.org/10.1109/SOSE.2018.00014

[15] Aradea, Iping Supriana, Kridanto Surendro, and Irfan Darmawan (2017), Variety of approaches in self-adaptation requirements: a case study, *Recent Advances on Soft Computing and Data Mining*. Springer, pp. 253-262. https://doi.org/10.1007/978-3-319-51281-5_26

[16] Aradea, Iping Supriana, Kridanto Surendro, and Irfan Darmawan (2017), Integration of Self-Adaptation Approach on Requirements Modeling, *Recent Advances on Soft Computing and Data Mining*, Springer, pp. 233-243. https://doi.org/10.1007/978-3-319-51281-5_24

[17] Asunción Gomez-Perez, Mariano Fernández-López, and Oscar Corcho (2004). *Ontological Engineering: with examples from the areas of knowledge management, e-commerce and the semantic web*, Springer-Verlag London Berlin Heidelberg. https://doi.org/10.1007/b97353

[18] Christian Wende, Katja Siegemund, Edward Thomas, Yuting Zhao, Jeff Z. Pan, Fernando Silva Parreiras, Tobias Walter, Krzysztof Miksa, Pawel Sabina, Uwe Assmann (2013). *Ontology reasoning for consistency-preserving structural modeling*, Springer-Verlag Berlin Heidelberg. https://doi.org/10.1007/978-3-642-31226-7_9

[19] Katja Siegemund,(2015). *Contributions to ontology-driven requirements engineering*, Dissertation, Technischen Universität Dresden, Fakultät Informatik, Lehrstuhl Softwaretechnologie.

[20] Aradea, Iping Supriana, and Kridanto Surendro (2018). Self-adaptive software modeling based on contextual requirements, *Telecommunication, Computing, Electronics and Contro (Telkomnika)*, Universitas Ahmad Dahlan, 16(3), pp. 1276-1288. http://dx.doi.org/10.12928/telkomnika.v16i3.7032

[21] Aradea, Iping Supriana, and Kridanto Surendro, (2018). *ARAS: Adaptation requirements for adaptive systems – handling run-time uncertainty of contextual requirements*," Technical Report, School of Electrical Engineering and Informatics, Bandung Institute of Technology, Indonesia.

[22] Aradea, Iping Supriana, and Kridanto Surendro (2019). *Quality evaluation of adaptation requirements knowledge based on ontological approach*, Technical Report, School of Electrical Engineering and Informatics, Bandung Institute of Technology, Indonesia.

[23] Y Yousef Abuseta, and Khaled Swesi (2015). Design patterns for self-adaptive systems engineering, *International Journal of Software Engineering & Applications (IJSEA)*, Arxiv, 6(4), pp. 11-28. https://doi.org/10.5121/ijsea.2015.6402

[24] Alessia Knauss, Daniela Damian, Xavier Franch, Angela Rook, Hausi A Müller, and Alex Thomo (2016). ACon: A learning-based approach to deal with uncertainty in contextual requirements at runtime, *Information and Software Technology*, Elservier, 70, pp. 85-99. https://doi.org/10.1016/j.infsof.2015.10.001

[25] Alexei Lapouchnian, and Jhon Mylopoulos (2011) . Capturing contextual variability in i* models, *CEUR Proceedings of the 5th International i* Workshop, iStar*, 96–101.

[26] Claes Wohlin, Per Runeson, Martin Host, Magnus C Ohlsson, Björn Regnell, and Anders Wesslen (2012). *Experimentation in software engineering*, Springer-Verlag Berlin Heidelberg. https://doi.org/10.1007/978-3-642-29044-2.

[27] Nadeem Abbas, Jesper Andersson, and Danny Weyns (2020) "ASPLe – A methodology to develop self-adaptive software systems with reuse, *Journal of Systems and Software*, Elsevier, 167, pp. 110626. https://doi.org/10.1016/j.jss.2020.110626.

[28] Nadeem Abbas, Jesper Andersson, and Danny Weyns (2012). Modeling variability in product lines using domain quality attribute scenarios, *IEEE/IFIP Conference on Software Architecture (WICSA), and the Sixth European Conference on Software Architecture (ECSA) 2012*, Association for Computing Machinery (ACM), Helsinki-Finland, pp. 135–142. https://doi.org/10.1145/2361999.2362028

[29] João Pimentel, Xavier Franch, and Jaelson Castro (2011). Measuring architectural adaptability in i* models, *Proceedings of the 14th Ibero-American Conference on Software Engineering (CIbSE)*, Rio de Janeiro, Brazil, pp. 115-128.

[30] Iping Supriana, Kridanto Surendro, Aradea, and Edwin Ramadhan (2017). Self-adaptive cyber-city system, *International Conference On Advanced Informatics: Concepts, Theory And Application (ICAICTA)*, IEEE, Penang-Malaysia. https://doi.org/10.1109/ICAICTA.2016.7803142.

[31] Somayeh Nasiri, Farahnaz Sadoughi, Afsaneh Dehnad, Mohammad Hesam Tadayon, and Hossein Ahmadi (2021). Layered architecture for internet of things-based healthcare system: a systematic literature review, *Informatica*, Slovene Society Informatika, 45(4), pp. 543-561. https://doi.org/10.31449/inf.v45i4.3601

[32] Sihem Benkhaled, Mounir Hemam, Meriem Djezzar and Moufida Maimour (2022). An ontology – based contextual approach for cross-domain applications in internet of things, *Informatica*, 46(1), pp. 39-48. https://doi.org/10.31449/inf.v46i5.3627