Comparison of general-purpose and domain-specific modeling languages in the IoT domain: A case study from the OMiLAB community

Arianna Fedeli[2,*], Nils Beutling[1], Emanuele Laurenzi[1,*] and Andrea Polini[2]

[1]*University of Applied Sciences and Arts Northwestern, Olten, Switzerland*

[2]*University of Camerino, Computer Science Division, Camerino, Italy*

Abstract

The Internet of Things (IoT) is a revolutionary concept that has rapidly transformed how we interact with technology and the world around us. In response to the inherent complexity and heterogeneity of the IoT domain, there has been a surge in the development of modeling languages and supporting platforms for developing IoT applications. Among the many modeling options available, one can distinguish between General-Purpose Modeling Languages (GPML) and Domain-Specific Modeling Languages (DSML). Each language has unique characteristics, offering distinct levels of abstraction and expressiveness crucial for effective IoT solution modeling. The challenge of selecting the most suitable language remains, with developers needing to weigh the benefits and drawbacks of each option carefully. This paper compares GPML and DSML regarding their characteristics, benefits, and drawbacks. By identifying key factors to consider when choosing a modeling language for IoT solutions, this research aims to provide valuable insights for a decision-making framework to help practitioners with this choice. To validate the findings and practical implications, a practical workshop was conducted. After creating a smart room scenario using the *X-IoT* DSML, the participants confirmed the advantages of DSML regarding user-friendliness, higher abstraction, improved communication, faster development, and the ability for non-experts to participate in the IoT application development process.

1. Introduction

Conceptual modeling is a fundamental and crucial technique in various fields, including software engineering, information systems, and business analysis. It involves creating abstract representations of a system, process, or domain to provide a clear and structured understanding of its key concepts, relationships, and functionalities [1]. Modeling languages come into play to express these conceptual models in a precise and standardized manner. Modeling languages serve as indispensable tools in the field of information systems, enabling the visual representation of complex systems, processes, and data structures, making it easier for designers, developers,

and stakeholders to understand, communicate, and collaborate on a shared vision [2]. Modeling languages provide a standardized set of symbols, syntax, and semantics for capturing and documenting various system aspects [3]. The used languages can be categorized into general-purpose modeling languages (GPML) and domain-specific modeling languages (DSML). GPML, such as the Unified Modelling Language (UML) [4], is designed to apply to any domain and application. In contrast, DSMLs are tailored to a specific domain or application. The choice of a modeling language significantly impacts the modeling process' efficiency, effectiveness, and the resulting model's quality and accuracy [2].

In the Internet of Things (IoT) domain, providing such a type of modeling solution introduces complexities such as the hybrid of the physical and the digital world, the high heterogeneity of the components, such as devices, sensors, actuators, and interfaces, the dynamic environment, the connectivity at multiple and extreme scales, and the system autonomy [5]. Furthermore, there is a complexity in integrating various devices across domains [6]. The use of appropriate modeling languages is required for the model-based development of IoT solutions, as it can substantially impact the efficiency and effectiveness of the development process. Understanding the characteristics, strengths, and limitations of modeling notation languages is crucial, especially when considering the unique challenges posed by IoT solutions. To enable stakeholders to make informed decisions, it is crucial to identify the factors that allow a comparison between modeling languages, especially in developing IoT solutions. However, a clear explanation of which type of model can better represent the IoT domain and a guiding high-level view is still missing.

Based on the problem statement, this work aims to answer the following main research question:

RQ: *How can general-purpose and domain-specific modeling languages be compared in terms of their suitability for the development of IoT solutions?*

The main research question can be subdivided into further sub-research questions for its elaboration:

Sub-RQ 1: *What are the key characteristics of GPML and DSML and their weaknesses?*

Sub-RQ 2: *What factors must be considered when deciding on a modeling language for developing IoT solutions?*

Sub-RQ 3: *How do users perceive the usage of a DSML to develop an IoT solution?*

Answering these research questions, this paper provides a theoretical comparison between GPMLs and DSMLs, especially regarding their benefits for developing IoT solutions. This paper aims to provide a decision framework to help stakeholders make informed decisions about which modeling language to use to develop IoT solutions. To do this, we highlighted several key factors to consider in modeling IoT solutions. Finally, a user evaluation of using the *X-IoT* DSML [7] for modeling IoT solutions is provided. *X-IoT* was developed inside the OMiLAB community[1] that offers a digital ecosystem bringing together open technologies to investigate and apply

---

[1]OMiLAB website: https://www.omilab.org/

conceptual modeling methods for varying purposes and domains. Those conceptual modeling methods, such as *X-IoT*, can be developed inside the ADOxx metamodelling platform[2] produced by OMiLAB. The structure of the paper is the following. Section 2 aims to answer *Sub-RQ 1* through a review of publications concerning the theoretical characteristics of GPML and DSML and their applicability to model IoT solutions. Section 3 responds to *Sub-RQ 2*, providing guidance and recommendations for practitioners and researchers, highlighting several key factors when choosing a modeling language for an IoT solution. To evaluate the findings of the literature review and answer the *Sub-RQ 3*, in Section 4 is reported a workshop focused on exploring the usage of the *X-IoT* DSML [7] to develop IoT solutions. Finally, Section 5 concludes the paper by providing insights for future work.

2. Related Work

In this section, we provide an overall view of General-Purpose Modeling Languages (GPMLs) for the IoT in Section 2.1 and Domain-Specific Modeling Language (DSML) for the IoT in Section 2.2, highlighting their characteristics and weak points. The findings discussed in this section answer the *Sub-RQ1* and summarize the essential characteristics of GPML and DSML, shown in Table 1 and highlighted in italics in the following text.

2.1. General-Purpose Modelling Languages for the IoT

GPMLs, by their inherent definition, transcend the confines of a restricted *Scope*, rendering them *Easy of Use* and universally applicable across diverse domains and applications. This intrinsic quality provides a *Standardized* methodology for representing and elucidating intricate systems, enabling enhanced comprehension and fostering the potential for streamlined reuse of existing models and components across disparate projects and organizations [8]. The high degree of *Applicability* of GPML allows modeling IoT solutions as well. Inside the IoT domain, they can capture the system architecture, data flow and communication, and behavior and interactions [9]. As the complexity of IoT solutions is characterized by the heterogeneity of devices and the involvement of stakeholders with different technical knowledge [10], GPML can provide a high level of *Abstraction*, thanks to a common language that facilitates its *Adaptability* and *Reuse* across domains and can capture the variety of components.

One of the most used includes the established Unified Modeling Language (UML) [4]. UML is a standardized visual modeling language used in software engineering and systems analysis to represent, design, and document software-intensive systems. It provides a set of graphical notations and diagrams to depict various aspects of a system's structure, behavior, and interactions. UML provides a set of diagrams, including but not limited to Activity Diagrams, State Machines, Use-Case, Class-Diagram, and many others. In addition, the System Modelling Language (SysML)[3], defined as an extension of a subset of UML 2, as it incorporates the UML structure and provides additional diagrams to support systems engineering applications. It has been employed to support the model design, analysis, verification, and validation of a broad

---

[2]ADOxx Metamodelling Platform https://www.adoxx.org/live/home
[3]SysML: https://sysml.org/

range of complex systems, including those in aerospace and automotive industries where IoT elements are included [11]. By transcending concepts, they facilitate *Communication* across different fields. Different works [12, 13, 9] propose UML extensions incorporating IoT-related elements. However, such extensions are only targeting IoT application programmers, thus not employable for effective communication among the various stakeholders involved in the development of IoT systems, e.g., domain expert, application expert, hardware expert [14].

Since GPMLs are typically well-established and have a sizeable user community, they often have a wealth of *Supporting Toolset* available, including documentation, tutorials, and third-party tools and libraries. Meta-modeling technologies and platforms (e.g., Eclipse Modeling Framework (EMF)[4]) are often used in industry nowadays, facilitating the design of modeling languages and the development of their supporting toolset. Several works [15, 16] provide IoT solutions development upon those platforms. These tools, through their standardization, allow easily *Interoperability* between other GPMLs and platforms.

However, the syntax and tooling mechanisms of such GPMLs require a steep *Learning Curve* for practitioners (especially for non-programmers) [2]. Further, they do not provide modeling elements covering all the representational requirements of an IoT system (i.e., *Expressiveness* issue). Hence general elements must be adapted for representing the IoT domain [15]. The latter requires multiple exchanges among the IoT expert and the language engineer, which makes the development of an IoT solution upon a GPML an activity with high *Cost and Time*. Finally, regarding *Maintenance and Evolution*, the use of GPMLs to develop IoT solutions, which by definition evolve really fast, may need to be continuously updated and maintained [12].

2.2. Domain-Specific Modelling Languages for the IoT

Researchers and practitioners have explored using Domain-Specific Modeling Languages to provide a more specific modeling language tailored for a target complex *Scope* such as the IoT and to address various stakeholders for the efficient and effective development of these solutions. For their definition, DSML requires an abstract syntax, a concrete syntax, and the semantics of the language, whereas the abstract syntax consists of the domain's concepts and rules [2].

According to [17], domain-specific modeling is a software engineering approach that raises the level of *Abstraction* by specifying the solution directly using problem domain concepts while narrowing down the design space – e.g., to a single range of products for a single company in the case of high specificity. The resulting high-level specifications can automatically generate final products. As a result, DSML leads to better productivity, better quality applications, and more accessible introduction and facility for new developers [8]. Indeed, compared to GPMLs, they generally have a shallower *Learning Curve*, making them *Easy to Use* for domain experts due to the IoT familiar domain terminology [7]. A broad range of DSMLs targeting the IoT domain was developed, starting from the usage of de-facto notations such as the Business Process Modelling Notation (BPMN)[5] [18, 19, 20]; its language extension, e.g., for a smart mobility planning [21]; to the development of DSMLs targeting a specific scope [5, 6, 7, 22, 23]. Thanks to their specific *Applicability*, DSML reduces the risk of errors by eliminating illegal designs upfront and detecting specifications that may lead to poor performance during product build,

---

[4]EMF: https://projects.eclipse.org/projects/modeling.emf.emf
[5]BPMN: https://bpmn.io

becoming efficient in terms of *Development Costs and Time* and resources required for IoT application development [17, 24, 25]. DSML also allows a facilitated *Communication* between experts across different IoT domains due to the higher level of representation of IoT elements [26]. This is especially true considering the specialized nature of DSMLs, designed with a focus on particular domains, encapsulating domain-specific concepts, semantics, and abstractions, making it easier for domain experts to understand and exchange complex information. They improve IoT application quality by including domain-specific correctness rules and mapping to a lower abstraction level, reducing the need for *Maintenance and Evolution* [7]. Due to the use of DSMLs within small communities based on a specific application domain, they need a specific *Supporting Toolset*. As an example, the OMiLAB community[1] is an active community of practice that offers a digital ecosystem bringing together open technologies to investigate and apply conceptual modeling methods for varying purposes and domains [27, 28, 29]. OMiLAB permits users to develop DSMLs through the ADOxx Platform[2], a metamodelling-based development and configuration environment, to create domain-specific modeling tools. Currently, more than seventy DSML modeling tools were developed by the community users[6], each focusing on a specific domain.

However, DSML also has drawbacks. Due to the low degree of *Adaptability* in a different context, DSMLs may lack *Standardization*, limiting *Reuse* possibility, and decreasing *Interoperability* between IoT domains, they tend to be kept in-house by enterprises who develop them based on their strict necessities [17]. To solve these lacks, a DSML should include elements that fully represent IoT applications and can be deployed on multiple IoT platforms [30], without rewriting each application for each deployment context. The work in [7] proposes a reusable DSML that allows deployment among different IoT platforms through the model-driven engineering approach.

3. Key Factors to Consider when Deciding on a modeling language for IoT Solutions

Before deciding on a modeling language for developing IoT solutions, the requirements and purposes of the solution must be defined and used to evaluate which modeling language provides the most significant benefit and has the appropriate specificity. In the following, to answer the *Sub-RQ 3*, we propose several factors that must be considered when deciding on a modeling language to assess its suitability. These factors are based on the literature review made in Section 2.

Factor 1 - Involved stakeholders. Models are practical communication tools. Regarding expressiveness and abstraction, whether they should be suitable for communication between experts of different domains must be considered: the more diverse the domains, the less specific the language. Communication with external stakeholders commonly requires a language that is easy to understand. The disadvantages of GPML and DSML could be tackled by combining them into a symbiotic solution that provides different views of the same model. The intended users are also crucial. For example, a highly specified modeling language with detailed rules

---

[6]OMiLAB Modeling Tools: https://www.omilab.org/activities/projects/

Characteristic	General-Purpose Modelling Languages (GPML)	Domain-Specific Modelling Languages (DSML)
Scope	Represent concepts across a wide range of domains	Formalize the structure, behavior and requirements within a domain
Applicability	Applicable across various domains and applications	Specific to a particular domain
Easy of Use	Often more accessible to a wider audience due to their general nature	More intuitive and easier to use within their domain, but might require domain expertise.
Communication	Usually operate at a higher level of abstraction, enabling broader modeling capabilities.	Facilitate effective communication within the targeted domain
Expressiveness	May lack the depth of expressiveness because they must cater to diverse applications	Offer higher expressiveness within their domain, capturing intricate details and relationships relevant to the specific problem
Learning Curve	Generally have a steeper learning curve as they cover a broad range of functionalities.	Users with domain knowledge may find them easier to learn and work with due to their focus on specific problems.
Reuse	Enable reuse of existing models and components	Limited reuse on different use-cases as their specific focus
Abstraction	High-level of abstraction as do not incorporate domain-specific concepts	Raise the level of abstraction by using problem domain concepts
Adaptability	More adaptable to different scenarios due to their generality	Well-suited for specific domains but may require modification for use in other areas.
Interoperability	Tend to be more interoperable with other general-purpose tools and systems	Might face challenges in integrating with other modeling languages or systems outside their domain.
Standardization	Some may have standardized syntax and semantics to ensure consistency	Standardization is less common, as they are often developed for specific projects or organizations
Supporting Toolset	Often benefit from extensive tooling and a large community for support	May have specialized tooling and a smaller, more focused community
Development Cost and Time	Generally require more effort and time to develop, given their broader scope	Can be quicker and more cost-effective to develop, especially for specialized domains
Maintenance and Evolution	Typically involve ongoing maintenance and evolution due to their broader user base and more extensive use cases	May require maintenance, but their narrow focus might lead to less frequent update
Advantages	• Standardized approach with a large user community with available resources and support • Unlimited applicability across domains • Can be used to specify the metamodel of a DSML	• Better productivity and application quality • Higher customized • Executable • Shorter development time • Lower error rate • Allow different allocation of work • Better productivity and application quality
Disadvantages	• Lack of domain-specific concepts • Limited specificity to a particular domain • Require mapping into further models for implementation	Need for adaptations as the domain and requirements change • Often kept in-house, limiting reuse and interoperability • Development time and cost for DSML • Difficulty in evaluating their value

Table 1
Comparison of GPML and DSML regarding their main characteristics

allows non-IoT experts to be involved as developers, as illegal designs are prevented, and less knowledge is required. The development of such a DSML is complex and must emphasize user-friendliness.

Factor 2 - Purpose and Scope. Clearly define the purpose of the model and its scope. Determine what aspects of the system or process you need to model and what specific questions or problems the model should address. Identifying the purpose and scope of a modeling task is the foundational step in selecting the appropriate modeling language. This involves precisely defining the model's objectives and understanding the specific system, process, or concept to be represented. By clarifying the scope, one can determine the level of detail required, the relevant stakeholders, and the intended audience for the model. For instance, the modeling purpose in software development may involve visualizing the system's architecture and interactions between components. On the other hand, in business process modeling, the goal might be to represent workflows and optimize process efficiency. By clearly defining the scope and purpose, decision-makers can narrow the choices and find the modeling language that best suits the intended outcomes and aligns with the project's goals.

Factor 3 - Targeted domain and complexity of IoT solution. The chosen language must be able to represent the different IoT devices, platforms, and tasks concerning the application domain of the solution. The narrower the domain and the lower the heterogeneity, the more specific the language can be. However, this makes reusability in other areas and integrating additional devices and platforms very difficult. The application domain also defines the complexity, which can be better handled with the help of DSML. IoT solutions are usually highly complex, and research shows a lack of languages that fully support the standard IoT requirements. As a result, solutions usually offer low reusability and must therefore be developed from scratch. Also, the ambiguity in interpreting industry standards among providers is problematic. A GPML may be missing the domain concepts but excels regarding reusability in different domains being able to represent high heterogeneity.

Factor 4 - Availability of time and resources. Usually, a company incorporates the expected return on investment and resource feasibility in its decision-making process. For GPML are typically many resources available, including documentation, tutorials, and tools which allow instant usage. Suppose an existing, more unspecific DSML is chosen. In that case, the time required to develop a DSML is eliminated, costs are generally lower, and different systems and tools may support the language. However, this bears the risk of vendor lock-in and a lack of customization. Nevertheless, a customized DSML enables faster development through automation, requires less maintenance, and increases the quality of the solutions. So, one must weigh the quicker availability of the language against the shorter development time and evaluate whether the development of a DSML can be compensated in the long run. Additionally, the reusability of a DSML also impacts the return on investment.

Factor 5 - Collaboration and competition. From a strategic view, whether the language should enhance competitiveness or enable open collaboration must be considered. In terms of competitiveness, the productivity of the language plays a crucial role. A sophisticated DSML can enable a faster time-to-market with higher quality, providing a competitive advantage. But such a DSML is usually kept secret, which limits cooperation significantly. In contrast, a more general, open-accessible modeling language has a higher reusability and facilitates collaboration with other companies.

Factor 6 - Tool Support. Ensure that there are suitable tools available that support the chosen modeling language. Modeling tools can significantly assist in creating, managing, and analyzing models. Selecting the appropriate modeling language for a project involves considering tool support as crucial. Modeling tools are essential for creating, editing, and analyzing models efficiently. The chosen tool should be compatible with the selected modeling language and provide a user-friendly interface supporting team members' collaboration. It should also offer validation and analysis features to ensure model correctness and completeness. Generating reports and documentation is essential for effective communication with stakeholders. Customization options and community support further enhance the tool's capabilities. By evaluating tool support, decision-makers can streamline the modeling process, improve productivity, and ensure that the chosen language aligns with the project's needs and objectives.

4. User experience regarding the use of a DSML

To answer the third Sub-RQ 3, we effectively evaluate the possibility of modeling an IoT solution through a DSML conducting a practical workshop. We organized one day workshop for master students of the University of Applied Sciences and Arts Northwestern in Switzerland in the Emerging Topics in Information Systems module.

Design. We arranged a usability experiment where participants were asked to use an already-developed DSML to model an IoT application. To this end, we gave them a theoretical lesson about general concepts such as the IoT, modeling solutions, and DSMLs to introduce participants to the topic. Then we detail the aim and usage of the *X-IoT* tool[7], a tool that allows modeling a DSML to represent and deploy an IoT solution. The *X-IoT* tool was developed as a modeling tool for the OMiLAB community and upon the ADOxx metamodelling platform. At the end of the theoretical lesson, participants were aware of using the *X-IoT* tool and asked to work on modeling a smart room scenario.

A total of 30 male and female subjects participated in the experiment, divided into five balanced groups of six participants each. Each group played the role of an IoT modeler. The instruments that were used to experiment are as follows:

- *A demographic questionnaire*: a set of questions to know the level of the users' experience in modeling and the IoT domain;
- *Work description*: the description of the activities that the subjects should carry out, i.e., using the provided tool to model and deploy the smart room scenario;
- *A NASA-TLX questionnaire [31]*: it was used to evaluate the perceived mental/physical/temporal demand, performance, effort, and frustration. This questionnaire was extended with an additional open question to collect suggestions and improvement points.
- *A time form*: it was defined to capture the start and completion times of the proposed activities.

---

[7]X-IoT Tool: https://www.omilab.org/activities/projects/details/?id=226
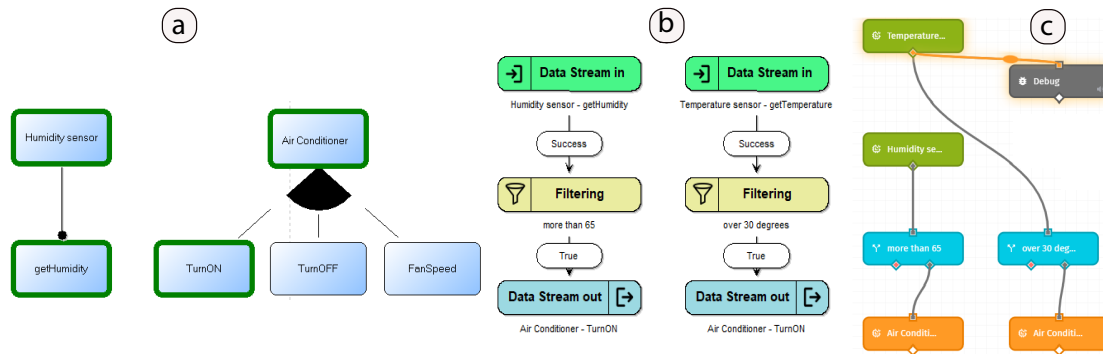
Figure 1: (a) Task 1: IoT device selection, (b) Task 2: IoT Application behavior modeling, performed inside the *X-IoT* tool; and (c) Task 3: IoT Application Deployment result inside the IoT Platform

The participants utilized the *X-IoT* DSML to develop a smart room scenario with a set of predetermined devices aided by a user manual. In detail, this workshop requires participants to perform three tasks, defined as follows.

Task 1. *Selection of IoT devices.* Users must decide which functionalities and IoT devices they want to include inside their IoT application.

Task 2. *Modelling of the IoT application behavior.* Users must model the logic of the entire IoT Application to perform the desired functionalities chosen by themselves.

Task 3. *Deployment of the IoT application in a chosen IoT platform.* Once the modeling is complete, the users must deploy the solution inside the chosen IoT Platform.

Results. The results of the NASA-TLX questionnaire are shown in Table 2. The highest scores represent the worst results. Therefore, mental load, physical demand, temporal demand, effort, and frustration are rated very low (value 0) and high (value 100). Performance is rated between good (value 0) and bad (value 100). Table 2 shows the average (*Avg*), the best result (*Best*), and the worst outcome (*Worst*). For task comparison, the NASA-TLX method suggests computing a weighted overall workload score for each task [31]. To facilitate the comprehension of this composite score, [32] delivers a detailed examination of more than 1000 global NASA-TLX scores extracted from over 200 published works. This assessment yielded an average composite score of 48.74, with the recorded scores spanning from a minimum of 8 to a maximum of 80. As demonstrated in Table 2, the computed global workload for each task falls below the average deduced from this analysis, thereby allowing us to characterize the achieved outcomes as favorable. In the following, we will describe the results provided by the workshop. In detail, we explain which key factors, discussed in section 3, were highlighted by users using the X-IoT tool. This allows responding to *Sub-RQ 3*, describing whether using a DSML can support the development of IoT solutions. The participants, who were familiar with modeling languages such as BPMN and UML, but had no previous experience in modeling and developing IoT solutions, successfully created a functional smart room scenario in about two hours. At the end of the modeling activity, the users could deploy the models directly inside IoT platforms to test their models. Figure 1 is reported as one result of the modeling activity. Detailing, in Figure 1(a), users define which devices and their operations want to include inside the application

(i.e., Humidity sensor and Air conditioner), as required by **Task 1**. Then Figure 1(b) presents the result of **Task 2**, as the behavior modeling of the application, defining rules that can be applied to the device's data (i.e., if the humidity data is greater than 65, the air conditioner turns on). Finally, in Figure 1(c), the result of **Task 3** as the IoT Application deployment inside an IoT Platform is illustrated. The outcomes depicted in Table 2 distinctly outline that Task 2 was perceived as the most intricate assignment. This observation finds justification in the fact that Task 2 necessitated users to model the comprehensive behavior of the IoT application. Delving deeper, the data indicates that this complexity was predominantly associated with the time invested in task completion while not significantly affecting the effort exerted, as elucidated in the table information.

Discussion. At the end of the experiments, through the questionnaire, the participants highlighted *X-IoT* as a user-friendly and efficient approach for IoT solution development thanks to the graphical interface provided by the ADOxx metamodelling platform (*Factor 6*). The difficulty of developing an IoT solution was notably decreased due to the graphical language and the facility of use of the *X-IoT* DSML (*Factor 2*). This allowed a high level of communication (*Factor 5*) for open discussions among participants about the design and optimization of the solution purpose by making the current state transparent – e.g., deciding which devices to insert inside the solution (*Factor 1*). Users pinpoint that a textual representation, instead of the graphical one presented through the ADOxx platform, would have been significantly more challenging and may have suffered regarding comprehensiveness. The DSML learning curve to understand the modeling language would have been greater. Additionally, presenting all possible elements prevented the overlooking of essential functions or devices. No coding knowledge was necessary to develop the scenario, making *X-IoT* easy to use and suitable for the participants who had different backgrounds and often lacked coding experience. The automated generation of the following models based on the selected devices and functions facilitated the process, and the implementation of domain rules prevented errors (*Factor 4*).

Groups suggest enhancing the automation provided inside *X-IoT*, enabling user support in making the selection of functions easier or more automatic. For example, users highlight that if a device had functions for turning it on and off, both could have been selected automatically. However, they had to be manually chosen at the actual stage, as reported in Figure 1(a). Regarding the modeling perspective, the participants experienced the typical benefits of a DSML, including higher abstraction, increased expressiveness and accuracy, improved communication, faster development through automation and implemented rules and the ability for non-IoT experts to participate (*Factor 3*). The workshop highlighted the advantages of *X-IoT* and provided positive feedback and successful outcomes underscoring the value of using DSMLs to model IoT applications. In addition, the possibility to directly deploy the final models inside running IoT platforms enhanced the total perception of the *X-IoT* tool. Future enhancements like further automation and usability improvements may contribute to even more seamless and intuitive modeling experiences.

Task	Task 1 Device Selection			Task 2 IoT Application Behavior Modeling			Task 3 IoT Application Deployment		
Param.	*Avg*	*Best*	*Worst*	*Avg*	*Best*	*Worst*	*Avg*	*Best*	*Worst*
Mental Load	25.50	10	60	22.50	20	75	13.50	10	30
Physical Demand	9.50	5	30	20.50	5	50	10.5	5	30
Temporal Demand	14.50	5	50	23.50	10	60	8	5	20
Performance	19	5	55	22.50	5	55	12.50	5	40
Effort	18.50	10	50	18	5	70	12.50	10	30
Frustration	15	5	50	18.50	5	55	9.50	5	20
Global	*21.8*			*28.5*			*19.5*		

Table 2
NASA-TXL questionnaire results - Avg: Average

5. Conclusions and Future Work

This work discusses the theoretical key characteristics of GPML and DSML, with insights from the IoT solution modeling. From this comparison, this work presented several key factors to handle when deciding on GPML or DSML to develop IoT solutions. Finally, through a practical workshop, participants confirmed several benefits and drawbacks of using a DSML (*X-IoT*) to model a smart room scenario. One avenue for future research is to create a holistic framework that integrates the identified factors and aids in decision-making or DSML development systematically. Another promising area is exploring how GPML and DSML can complement each other to achieve more significant benefits. Additionally, it is essential to investigate the interoperability of IoT solutions and the reusability of DSML for different scenarios.

Acknowledgments

References

[1] M. Boman, J. A. Bubenko Jr, P. Johannesson, B. Wangler, Conceptual modelling, Prentice-Hall, Inc., 1997.

[2] D. Karagiannis, R. A. Buchmann, P. Burzynski, U. Reimer, M. Walch, Fundamental Conceptual Modeling Languages in OMiLAB, 2016, pp. 3–30.

[3] X. He, Z. Ma, W. Shao, G. Li, A metamodel for the notation of graphical modeling languages, in: 31st Annual International Computer Software and Applications Conference (COMPSAC 2007), volume 1, IEEE, 2007, pp. 219–224.

[4] H.-E. Eriksson, M. Penker, B. Lyons, D. Fado, UML 2 toolkit, John Wiley & Sons, 2003.

[5] M. W. Aziz, M. Rashid, Domain specific modeling language for cyber physical systems, in: 2016 International Conference on Information Systems Engineering (ICISE), 2016, pp. 29–33.

[6] F. Corradini, A. Fedeli, F. Fornari, A. P. and, X-IoT: A Model-Driven Approach for Cross-Platform IoT Applications Development, in: SAC'22: The 37th ACM/SIGAPP Symposium on Applied Computing, ACM, 2022, pp. 1448–1451.

[7] F. Corradini, A. Fedeli, F. Fornari, A. Polini, B. Re, L. Ruschioni, X-iot: a model-driven approach to support iot application portability across iot platforms, Computing (2023).

[8] S. Arslan, M. Ozkaya, G. Kardas, Modeling languages for internet of things (iot) applications: A comparative analysis study, Mathematics 11 (2023).

[9] K. Thramboulidis, F. Christoulakis, Uml4iot-a uml profile to exploit iot in cyber-physical manufacturing systems, arXiv preprint arXiv:1512.04894 (2015).

[10] I. S. Udoh, G. Kotonya, Developing iot applications: challenges and frameworks, IET Cyber-Physical Systems: Theory & Applications 3 (2018) 65–72.

[11] M. Hause, et al., The sysml modelling language, in: Fifteenth European Systems Engineering Conference, volume 9, 2006, pp. 1–12.

[12] T. Eterovic, E. Kaljic, D. Donko, A. Salihbegovic, S. Ribic, An internet of things visual domain specific modeling language based on uml, in: 2015 XXV International Conference on Information, Communication and Automation Technologies (ICAT), IEEE, 2015, pp. 1–5.

[13] F. Ihirwe, D. Di Ruscio, S. Mazzini, A. Pierantonio, Towards a modeling and analysis environment for industrial iot systems, arXiv preprint arXiv:2105.14136 (2021).

[14] F. Corradini, A. Fedeli, F. Fornari, A. Polini, B. Re, Floware: a model-driven approach fostering reuse and customisation in iot applications modelling and development, Software and Systems Modeling (2022) 131–258.

[15] F. Pramudianto, I. R. Indra, M. Jarke, Model driven development for internet of things application prototyping., in: SEKE, 2013, pp. 703–708.

[16] F. Corradini, A. Fedeli, F. Fornari, A. Polini, B. Re, Floware: An approach for iot support and application development, in: Enterprise, Business-Process and Information Systems Modeling, Springer International Publishing, Cham, 2021, pp. 350–365.

[17] S. Kelly, J.-P. Tolvanen, Domain-Specific Modeling: Enabling Full Code Generation, 2008.

[18] L. Buffoni, L. Ochel, A. Pop, P. Fritzson, N. Fors, G. Hedin, W. Taha, M. Sjölund, Open source languages and methods for cyber-physical system development: Overview and case studies, Electronics 10 (2021) 902.

[19] P. Valderas, V. Torres, E. Serral, Towards an interdisciplinary development of iot-enhanced business processes, Business & Information Systems Engineering (2022) 1–24.

[20] V. Torres, E. Serral, P. Valderas, V. Pelechano, P. Grefen, Modeling of IoT devices in business processes: A systematic mapping study, 22nd IEEE Conference on Business Informatics,

CBI 2020 1 (2020) 221–230.

[21] E. Laurenzi, O. Ruggli, A. V. D. Merwe, E. Laurenzi, O. Ruggli, A. V. D. Merwe, BPMN4MoPla: Mobility Planning Based on Business Decision-Making, Springer, Cham, 2022, pp. 617–638.

[22] F. Corradini, A. Fedeli, F. Fornari, A. Polini, B. Re, Dtmn a modelling notation for digital twins, in: T. P. Sales, H. A. Proper, G. Guizzardi, M. Montali, F. M. Maggi, C. M. Fonseca (Eds.), Enterprise Design, Operations, and Computing. EDOC 2022 Workshops, Springer International Publishing, Cham, 2023, pp. 63–78.

[23] F. Corradini, A. Fedeli, A. Polini, B. Re, Towards a digital twin modelling notation, in: 2022 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress, 2022, pp. 1–6.

[24] B. Akesson, J. Hooman, J. Sleuters, A. Yankov, Chapter 10 - reducing design time and promoting evolvability using domain-specific languages in an industrial context, in: B. Tekinerdogan, Önder Babur, L. Cleophas, M. van den Brand, M. Akşit (Eds.), Model Management and Analytics for Large Scale Systems, Academic Press, 2020, pp. 245–272.

[25] A. J. Salman, M. Al-Jawad, W. Al Tameemi, Domain-specific languages for iot: Challenges and opportunities, in: IOP Conference Series: Materials Science and Engineering, IOP Publishing, 2021, p. 012133.

[26] M. Mohamed, G. Kardas, M. Challenger, Model-driven engineering tools and languages for cyber-physical systems–a systematic literature review, IEEE Access PP (2021) 1–1.

[27] D. Karagiannis, R. A. Buchmann, W. Utz, The omilab digital innovation environment: Agile conceptual models to bridge business value with digital and physical twins for product-service systems development, Comput. Ind. 138 (2022) 103631.

[28] D. Karagiannis, R. A. Buchmann, X. Boucher, S. Cavalieri, A. Florea, D. Kiritsis, M. Lee, Omilab: A smart innovation environment for digital engineers, in: Boosting Collaborative Networks 4.0 - 21st IFIP WG 5.5 Working Conference on Virtual Enterprises, series, volume 598, Springer, 2020, pp. 273–282.

[29] I. Vaidian, A. Jurczuk, Z. Misiak, M. Neidow, M. Petry, M. Nemetz, Challenging digital innovation through the omilab community of practice, in: D. Karagiannis, M. Lee, K. Hinkelmann, W. Utz (Eds.), Domain-Specific Conceptual Modeling - Concepts, Methods and ADOxx Tools, Springer, 2022, pp. 41–64.

[30] I. Abouzid, Y. K. Bekali, R. Saidi, Modelling iot behaviour in supply chain business processes with bpmn: A systematic literature review, Journal of ICT Standardization (2022) 439–468.

[31] S. G. Hart, L. E. Staveland, Development of nasa-tlx (task load index): Results of empirical and theoretical research 52 (1988) 139–183.

[32] R. A. Grier, How high is high? a meta-analysis of nasa-tlx global workload scores, in: Proceedings of the Human Factors and Ergonomics Society Annual Meeting, volume 59, SAGE Publications Sage CA: Los Angeles, CA, 2015, pp. 1727–1731.