Affordance Representation and Recognition for Autonomous Agents

Habtom Kahsay Gidey[1,*], Niklas Huber[2], Alexander Lenz[1] and Alois Knoll[1]

[1]Technische Universität München, München, Germany
[2]Jessy Works, München, Germany

Abstract

The autonomy of software agents is fundamentally dependent on their ability to construct an actionable internal world model from the structured data that defines their digital environment, such as the Document Object Model (DOM) of web pages and the semantic descriptions of web services. However, constructing this world model from raw structured data presents two critical challenges: the verbosity of raw HTML makes it computationally intractable for direct use by foundation models, while the static nature of hardcoded API integrations prevents agents from adapting to evolving services.

This paper introduces a pattern language for world modeling from structured data, presenting two complementary architectural patterns. The *DOM Transduction Pattern* addresses the challenge of web page complexity by *distilling* a verbose, raw DOM into a compact, task-relevant representation or world model optimized for an agent's reasoning core. Concurrently, the *Hypermedia Affordances Recognition Pattern* enables the agent to dynamically enrich its world model by parsing standardized semantic descriptions to discover and integrate the capabilities of unknown web services at runtime. Together, these patterns provide a robust framework for engineering agents that can efficiently construct and maintain an accurate world model, enabling scalable, adaptive, and interoperable automation across the web and its extended resources.

Keywords

Autonomous Agents, Affordances, World Modeling, Design Patterns, Web Automation.

1. Introduction

Recent advances and the abundance of foundation models are driving a new generation of software agents performing complex cognitive tasks in dynamic, mixed-reality ecosystems [1, 2, 3]. For these agents to operate effectively, they must perceive and understand their environment to build an internal *world model*, an actionable representation that guides their reasoning and planning [4, 5, 6]. Although significant research has focused on visual perception from pixels, many digital environments are built upon rich, structured data sources like the hypermedia of the Document Object Model (DOM) of web pages and the descriptive interfaces of web services [7, 8]. To this end, leveraging this explicit structure offers a more efficient and deterministic path to building a world model [9, 10].

However, directly consuming this structured data presents two critical architectural challenges that hinder the development of truly autonomous agents [11, 2]. First, the verbosity of raw HTML is a major bottleneck for agents using foundation models for reasoning and planning [1]. Empirical studies of DOM pruning show that the majority, often 80 - 90%, of tokens in raw HTML consist of non-semantic markup such as scripts, styles, and trackers [9, 7], which can overwhelm the model's, i.e., LLMs, context window, degrade reasoning quality, and incur high computational costs [12]. This is the fundamental *challenge of representation*, where the raw state of the world is too complex for the agent's cognitive core to process efficiently [10, 13].

Second, agents must interact with a dynamic ecosystem of services and devices in the increasingly interconnected Web of Things (WoT) and microservice architectures [14, 15, 16]. In such cases, the

The Second International Workshop on Hypermedia Multi-Agent Systems (HyperAgents 2025), in conjunction with the 28th European Conference on Artificial Intelligence (ECAI 2025); October 26, 2025, Bologna, Italy

*Corresponding author.
habtom.gidey@tum.de (H. K. Gidey); niklas.huber@jessyworks.com (N. Huber); alex.lenz@tum.de (A. Lenz); knoll@tum.de (A. Knoll)
0000-0001-5802-2606 (H. K. Gidey); 0000-0003-4840-076X (A. Knoll)

traditional approach of hardcoding binary interfaces, such as API endpoints and interaction logic, creates brittle, tightly coupled systems that cannot adapt when a service changes or a new device is introduced in the web microcosm [17, 11]. This is the *challenge of interoperability, adaptability, and discovery*, i.e., recognition of affordances, where the agent's world model is static and cannot be updated to reflect a changing digital environment [12, 10].

This paper introduces preliminary work on a pattern language [18] that addresses these challenges by framing them as problems of world model construction [4, 5, 6]. We present two architectural design patterns that provide reusable solutions for building and enriching an agent's world model from structured data:

1. **The DOM Transduction Pattern:** A pattern for distilling a complex, raw DOM into a compact, task-relevant world model optimized for an agent's reasoning core.
2. **The Hypermedia Affordances Recognition Pattern:** A pattern for dynamically enriching the world model by parsing standardized semantic descriptions of web services to discover and integrate their capabilities at runtime.

Combined with other percepts, these patterns provide a robust framework for engineering agents that can efficiently and adaptively interact with the structured web and its connected extended resources and the web of things [15, 5].

2. Background

This research work covers several domains; in particular, it is situated at the intersection of hypermedia multi-agent systems, world modeling, and cognitive automation [19, 14, 4, 15, 5, 6].

2.1. Web Agents Observation Space

The challenge of processing verbose HTML for LLM-based agents has become a significant area of research [11, 2]. The core problem is that the agent's observation space, when represented by a raw DOM, is misaligned with the LLM's processing capabilities. As a result, this has led to the development of various techniques for DOM cleaning and simplification [9, 7, 10]. Recent work on LLM-based web agents such as WebVoyager, Agent-E, and AgentOccam has highlighted the critical importance of managing the agent's observation space [20, 8, 7]. The primary challenge lies in handling complex and verbose HTML, which motivates approaches such as DOM distillation and HTML pruning. Several approaches have emerged, collectively known as DOM distillation or HTML pruning, which aim to simplify the DOM to make it more tractable for LLMs. For example, AgentOccam focuses on refining the observation space to better align with the LLM's capabilities [7]. More recent work, such as HtmlRAG, has introduced block-tree-based pruning methods and concrete methods to clean and compress HTML while preserving its structure [9]. This concept of webpage segmentation has a long history in information extraction, aiming to break down a page into semantically related parts to improve downstream tasks [20, 8, 7]. These approaches demonstrate the critical importance of preprocessing the DOM. Our DOM Transduction Pattern then aims to formalize these emerging best practices into a reusable architectural solution.

2.2. Hypermedia Multi-agent Systems and World Models via Hypermedia

The second challenge, interoperability, is addressed by principles from hypermedia multi-agent systems [14]. The core idea is *Hypermedia as the Engine of Application State (HATEOAS)*, a fundamental constraint of the REST architectural style [21]. HATEOAS states that a client should navigate an application entirely through links provided dynamically by the server, eliminating the need for a developer in the middle or hardcoded endpoints and thus decoupling the client from the server. While the adoption of HATEOAS in general web APIs has been debated, its principles are ideally suited for autonomous agents that require dynamic discovery and adaptation of affordances.

The most concrete and standardized application of these principles is the W3C Web of Things (WoT) framework [22]. Its central component, the *Thing Description (TD)*, is a JSON-LD document that provides machine-readable "capability knowledge" for any given "Thing," such as a device, service, or other web resource. A TD specifies a resource's metadata and its *Interaction Affordances*, the 'Properties', 'Actions', and 'Events' it exposes, along with the specific 'Protocol Bindings' (e.g., HTTP, MQTT) required for interaction. By parsing a TD, an agent can dynamically learn what a service can do and how to communicate with it without prior, hardcoded knowledge. This is complemented by *WoT Discovery* mechanisms, which define how agents can find relevant Thing Descriptions, for instance, through a searchable Thing Description Directory (TDD) [23, 16].

Our Hypermedia Affordances Recognition Pattern formalizes this HATEOAS-based discovery process as a key mechanism for enriching an agent's world model at runtime.

These ideas connect to the notion of a cognitive map or a world model, grounded in the foundational work on world models [4, 5, 6].

3. Pattern Language for Structured Perception

To systematically address the challenges of building world models from structured data, here we adopt the established software engineering methodology, i.e., design patterns [24, 25, 18]. A pattern is a reusable design or architectural solution to a recurring problem within a given context, forming a "pattern language" that captures and communicates expert architectural knowledge. By applying this methodology, we can systematically investigate and codify solutions for agents' recurring perceptual challenges when interacting with structured digital environments.

Consequently, to ensure a rigorous description of each pattern, we employ a comprehensive cataloging template adapted from standard pattern documentation formats. This template organizes each pattern into three main sections: the *Problem Space*, which defines the context, problem, and motivating forces; the *Solution Space*, which details the solution's description, components, flow, and formal constraints; and *Application and Evaluation*, which discusses consequences and implementation. This format is deliberately preferred to facilitate future formalization and ensure the reproducibility of our proposed solutions [25, 26].

3.1. The DOM Transduction Pattern

3.1.1. Problem

An LLM-based agent must understand and interact with a web page, but the raw HTML DOM is too verbose and noisy. It exceeds the LLM's context window, contains irrelevant information that degrades reasoning, and incurs high computational costs. To that end, the agent needs a simplified yet structurally coherent representation of page affordances to build its world model [11, 2].

3.1.2. Solution

This pattern introduces a *DOM Transformer* component within the agent's perception module. As illustrated in Fig. 1, this component ingests raw DOM and applies a series of transformations to distill and represent it in a compact, task-relevant world model with affordances. This process typically involves the following.

1. **Cleaning:** Removing universally irrelevant tags like '`<script>`' and '`<style>`', which can substantially shrink the DOM size [9, 7].
2. **Pruning:** Intelligently removing content that is irrelevant to the current task, for example, block-tree-based pruning strategies or embedding-based relevance filtering [10, 13].
3. **Compact Representation:** Converting cleaned HTML into token-efficient encodings such as Emmet notation [9].

4. **LLM-as-Transformer:** Using a smaller LLM to summarize the DOM before passing it to a larger reasoning model [11].

The output is a simplified DOM that preserves the essential structure needed for interaction while being optimized for LLM processing. Fig. 1 illustrates the flow and components of the perception pipeline for the *DOM Transduction Pattern*.

Figure 1: *Flow and components of the DOM Transduction Pattern. Raw DOM is distilled into an affordance representation, the Page Affordance Model (PAM), and then fused with other structured percepts, such as service contracts and WoT Things, to update the Cognitive Map, the agent's world model.*

3.1.3. Architectural Constraints

1. The input must be a structured DOM tree.
2. The transformation is a unidirectional data flow from Raw DOM into a structured representation of the Page Affordance Model.
3. The output, Page Affordance Model, must preserve the essential structure of task-relevant interactive elements.
4. The DOM Transformer must be a decoupled perception component.

3.1.4. Application and Evaluation

- **Consequences:**
  - *Benefits:* Enables automation on complex pages by overcoming context window limitations; significantly reduces cost and latency; improves agent reliability by providing a cleaner, more focused context.
  - *Liabilities:* Designing a robust DOM Transformer is a non-trivial engineering task; an overly aggressive pruning strategy can lead to critical failures by removing necessary elements.
- **Implementation Considerations:**
  - *Rule-Based Filtering:* Pattern-matching and rule-based parsing techniques can support the selective removal of predefined tags and attributes from the DOM.
  - *Block-Based Pruning:* Partitioning the DOM into semantic blocks, combined with task-aware relevance scoring, for example, embedding similarity to task descriptions, can provide effective strategies for discarding irrelevant content [9].
  - *Compact Representation:* Structural encoding methods, such as token-efficient notations, can provide compressed forms of the cleaned DOM while preserving essential hierarchy and relationships [9].
  - *LLM-as-Transformer:* Cascaded model strategies, where a smaller model distills or summarizes the DOM before forwarding it to a more capable reasoning model, can offer efficiency gains [11].

3.2. The Hypermedia Affordances Recognition Pattern

3.2.1. Problem

An agent must operate in a dynamic ecosystem of distributed services and IoT devices (the "Web of Things"). Hardcoding the API and having a developer in the middle for each service makes the agent brittle and unable to adapt to new or updated services. It must have perceptual skills to dynamically discover, recognize, and understand the capabilities, or affordances, of any web service or resource it encounters [14, 17].

3.2.2. Solution

This pattern, based on HATEOAS principles [21], requires that services expose their capabilities through standardized, machine-readable semantic descriptions. The agent's perception module contains an *Affordance Parser* that fetches and interprets these descriptions. The canonical implementation is the *W3C WoT Thing Description (TD)*, a JSON-LD document that specifies two key elements [22]:

- **Interaction Affordances:** The 'Properties' (readable/writable state), 'Actions' (invokable functions), and 'Events' (subscribable notifications) the service offers.
- **Protocol Bindings:** These are the specific technical instructions that detail how an agent can interact with each of a service's capabilities or affordances.

By parsing a TD, the agent dynamically learns how to interact with services at runtime. As shown in Fig. 2, this enables adaptability and robustness, allowing agents to autonomously integrate new devices and services without prior hardcoding.

Figure 2: The flow of the Hypermedia Affordances Recognition Pattern involves the agent discovering a semantic description and parsing it into an affordances catalog which updates the Cognitive Map.

3.2.3. Architectural Constraints

1. The agent and the resource it interacts with must be fully decoupled with no pre-configured, hardcoded dependencies on each other.
2. The resource must expose its capabilities and make them known by publishing a standardized semantic description.
3. All agent interactions must be driven by affordances discovered in the description.
4. The resource's description dictates the interaction protocol and all communication specifics, which are determined by the resource itself, not the agent.

3.2.4. Application and Evaluation

- **Consequences:**
  - *Benefits:* Enables extreme adaptability and robustness, allowing agents to autonomously integrate new devices and services on the fly; simplifies the development of large-scale, interoperable systems.
  - *Liabilities:* Requires device and service providers to correctly implement and host a Thing Description, which can be a barrier to adoption; a poorly written TD can lead to agent errors.
- **Implementation Considerations:**
  - Affordance parsing can be implemented using JSON-LD libraries and WoT toolkits.
  - Agents must include client libraries for common protocols (HTTP, MQTT, etc.), and can use WoT Discovery mechanisms such as TD directories to find services [22].

4. Cognitive Map as a Unified World Model

The two patterns embody two complementary modes of perception: one focused on distilling and representing a known, complex environment, and the other on discovering and integrating unknown entities. Jointly, they contribute to the construction of a unified cognitive map or a world model, as emphasized in the foundational work on world models [4, 5, 6, 27].

Their flow can be illustrated with a simplified practical example. An agent tasked with booking a hotel first applies the *DOM Transduction Pattern* to parse the booking website, producing a simplified world model of the form. Within this model, it discovers a hyperlink labeled *"Smart Room Controls"*. Following this link, the agent receives a W3C Thing Description for the room's environmental controls. It then switches to the *Hypermedia Affordances Recognition Pattern* to parse the description, dynamically enriching its world model with new capabilities, for example, discovering a *'thermostat'* and a *'setTemperature'* action. The agent can now not only complete the booking but also offer to pre-set the room temperature, an affordance discovered and integrated entirely at runtime.

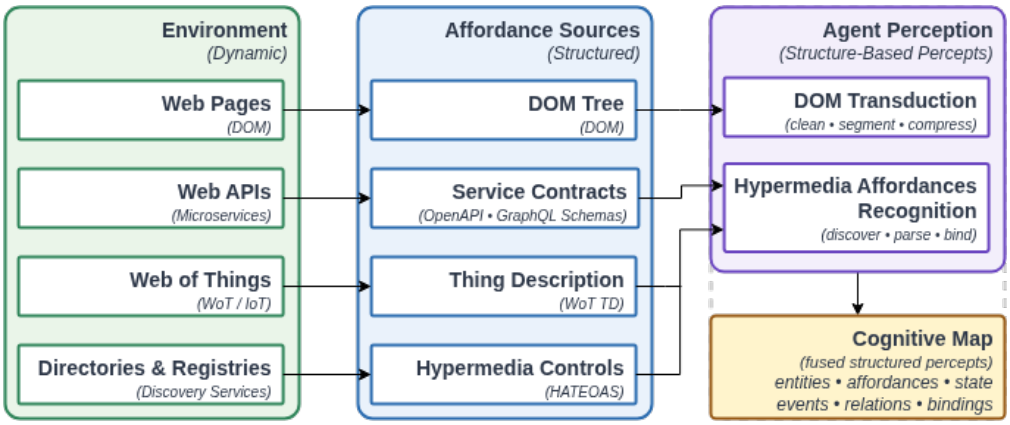

Figure 3: High-level flow from structured environments (web services, devices) to a unified cognitive map. Vision is excluded; only structured inputs are considered.

This composition of patterns is central to the agent's perception architecture, as shown in Fig. 3. The diagram illustrates how different percepts and affordances, such as DOM trees, thing descriptions, and service contracts, are fused into a unified cognitive map. The DOM Transduction Pattern processes HTML structures, while the Hypermedia Affordances Recognition Pattern interprets service descriptions. Concurrently, these complementary perceptual streams provide the agent with an adaptable and semantically rich representation of its environment in the evolving ecosystem.

5. Conclusion and Outlook

This paper introduced a preliminary pattern language to address key challenges in web and service interaction by presenting two architectural patterns that enable autonomous agents to construct and maintain an actionable world models from structured data. The DOM Transduction Pattern distills complex web pages into tractable affordance representations for LLM-based reasoning, while the Hypermedia Affordances Recognition Pattern enables dynamic discovery of service capabilities, ensuring interoperability and adaptability.

The primary contribution of this work is a principled, reusable framework for engineering hypermedia multi-agent systems that can build and maintain accurate world models from the explicit structure of their environment. This enables the development of agents that are more efficient, scalable, resilient to change, and capable of predictive reasoning, compared to those relying on brittle, hardcoded logic.

As an outlook, this work represents one half of a larger vision. Our future research will extend these structure-based patterns with visual counterparts. The long-term goal is a comprehensive pattern language for multi-modal perception, enabling agents to fuse structured and visual percepts. This will allow an agent, for example, to use the DOM Transduction pattern on a website but switch to visual parsing when structured representations are unavailable. This ability to intelligently select and adapt perceptual modalities will advance the next generation of autonomous agents toward human-level competence in digital environments.

Declaration on Generative AI

During the preparation of this work, the authors used Grammarly and ChatGPT in order to: Grammar and spelling check, Paraphrase and reword. After using this tool/service, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] I. Gur, H. Furuta, A. Huang, M. Safdari, Y. Matsuo, D. Eck, A. Faust, A real-world webagent with planning, long context understanding, and program synthesis, arXiv (2023). URL: https://arxiv.org/abs/2307.12856. doi:10.48550/arXiv.2307.12856. arXiv:2307.12856.

[2] L. Ning, Z. Liang, Z. Jiang, H. Qu, Y. Ding, W. Fan, X. Wei, S. Lin, H. Liu, P. S. Yu, Q. Li, A survey of webagents: Towards next-generation AI agents for web automation with large foundation models, in: Proceedings of the ACM Web Conference 2025 (WWW '25), 2025. URL: https://dl.acm.org/doi/10.1145/3711896.3736555. doi:10.1145/3711896.3736555.

[3] J. Macedo, H. K. Gidey, K. B. Rebuli, P. Machado, Evolving user interfaces: A neuroevolution approach for natural human-machine interaction, in: C. Johnson, S. M. Rebelo, I. Santos (Eds.), Artificial Intelligence in Music, Sound, Art and Design: 13th International Conference, EvoMUSART 2024, Held as Part of EvoStar 2024, Aberystwyth, UK, April 3–5, 2024, Proceedings, volume 14633 of *Lecture Notes in Computer Science*, Springer, Cham, 2024, pp. 246–264. URL: https://doi.org/10.1007/978-3-031-56992-0_16. doi:10.1007/978-3-031-56992-0_16.

[4] D. Ha, J. Schmidhuber, World models, arXiv preprint arXiv:1803.10122 2 (2018).

[5] Y. LeCun, A path towards autonomous machine intelligence, arXiv preprint arXiv:2205.01761 (2022). URL: https://arxiv.org/abs/2205.01761.

[6] J. Richens, T. Everitt, D. Abel, A. Bellot, General agents need world models, in: Proceedings of the 42nd International Conference on Machine Learning (ICML 2025), Vancouver, Canada, 2025. URL: https://icml.cc/virtual/2025/poster/44620.

[7] K. Yang, Y. Liu, S. Chaudhary, R. Fakoor, P. Chaudhari, G. Karypis, H. Rangwala, Agentoccam: A simple yet strong baseline for LLM-based web agents, arXiv (2024). URL: https://arxiv.org/abs/2410.13825. arXiv:2410.13825.

[8] H. He, W. Yao, K. Ma, W. Yu, Y. Dai, H. Zhang, Z. Lan, D. Yu, Webvoyager: Building an end-to-end web agent with large multimodal models, arXiv preprint arXiv:2401.13919 (2024). URL: https://arxiv.org/abs/2401.13919.

[9] J. Tan, Z. Dou, W. Wang, M. Wang, W. Chen, J. Wen, Htmlrag: HTML is better than plain text for modeling retrieved knowledge in RAG systems, in: Proceedings of the ACM Web Conference 2025 (WWW '25), ACM, Sydney, 2025, pp. 1733–1746. URL: https://dl.acm.org/doi/10.1145/3696410.3714546. doi:10.1145/3696410.3714546.

[10] J. Liu, J. Hao, C. Zhang, Z. Hu, Wepo: Web element preference optimization for LLM-based web navigation, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 39, AAAI Press, 2025, pp. 26614–26622. URL: https://ojs.aaai.org/index.php/AAAI/article/view/34863.

[11] R. Assouel, T. Marty, M. Caccia, I. H. Laradji, A. Drouin, S. R. Mudumba, H. Palacios, Q. Cappart, D. Vázquez, N. Chapados, M. Gasse, A. Lacoste, The unsolved challenges of llms as generalist web agents: A case study, in: NeurIPS 2023 Workshop on Foundation Models for Decision Making (FMDM@NeurIPS 2023), 2023. URL: https://openreview.net/forum?id=jt3il4fC5B.

[12] X. H. Lù, G. Kamath, M. Mosbach, S. Reddy, Build the web for agents, not agents for the web, arXiv (2025). URL: https://arxiv.org/abs/2506.10953. arXiv:2506.10953.

[13] S. Kim, N. Kim, Y. Jeong, Next-eval: Next evaluation of traditional and LLM web data record extraction, arXiv (2025). URL: https://arxiv.org/abs/2505.17125. arXiv:2505.17125.

[14] F. Gandon, Merry hmas and happy new web: A wish for standardizing an ai-friendly web architecture for hypermedia multi-agent systems, in: Dagstuhl-Seminar 21072: Autonomous Agents on the Web, 2021, p. 3.

[15] M. Castellucci[1], S. Burattini[1], A. Ciortea, J. Lemée, D. Vachtsevanou, A. Ricci[1], S. Mayer, Towards agents' embodiment in hypermedia multi-agent systems, in: Multi-Agent Systems: 21st European Conference, EUMAS 2024, Dublin, Ireland, August 26–28, 2024, Proceedings, volume 15685, Springer Nature, 2025, p. 361.

[16] H. K. Gidey, M. Kesseler, P. Stangl, P. Hillmann, A. Karcher, Document-based knowledge discovery with microservices architecture, in: Intelligent Systems and Pattern Recognition: ISPR 2022, volume 1589, Springer, Cham, 2022, pp. 146–161. URL: https://doi.org/10.1007/978-3-031-08277-1_13. doi:10.1007/978-3-031-08277-1_13.

[17] H. K. Gidey, P. Hillmann, A. Karcher, A. Knoll, Towards cognitive bots: Architectural research challenges, in: Artificial General Intelligence, AGI 2023, Springer, 2023. URL: https://doi.org/10.1007/978-3-031-33469-6. doi:10.1007/978-3-031-33469-6.

[18] H. K. Gidey, D. Marmsoler, J. Eckhardt, Grounded architectures: Using grounded theory for the design of software architectures, in: 2017 IEEE International Conference on Software Architecture Workshops (ICSAW), IEEE, Gothenburg, Sweden, 2017, pp. 141–148. URL: https://doi.org/10.1109/ICSAW.2017.41. doi:10.1109/ICSAW.2017.41.

[19] H. K. Gidey, P. Hillmann, A. Karcher, A. Knoll, User-like bots for cognitive automation: A survey, in: Machine Learning, Optimization, and Data Science, LOD 2023,, Springer, 2023. URL: https://doi.org/10.1007/978-3-031-53966-4. doi:10.1007/978-3-031-53966-4.

[20] T. Abuelsaad, D. Akkil, P. Dey, A. Jagmohan, A. Vempaty, R. Kokku, Agent-e: From autonomous web navigation to foundational design principles in agentic systems, arXiv preprint arXiv:2407.13032 (2024).

[21] M. Kelly, JSON Hypertext Application Language, Internet-Draft draft-kelly-json-hal-11, Internet Engineering Task Force, 2023. URL: https://datatracker.ietf.org/doc/draft-kelly-json-hal/11/.

[22] M. Lagally, R. Matsukura, M. McCool, K. Toumura, Web of Things (WoT) Architecture 1.1, W3C Recommendation REC-wot-architecture11-20231205, World Wide Web Consortium, 2023. URL: https://www.w3.org/TR/wot-architecture11/.

[23] Web of Things (WoT) Discovery, W3C Recommendation, World Wide Web Consortium (W3C), 2023. URL: https://www.w3.org/TR/wot-discovery/.

[24] K. Beck, Using pattern languages for object-oriented programs, in: OOPSLA-87 workshop on the Specification and Design for Object-Oriented Programming, 1987.

[25] H. K. Gidey, A. Collins, D. Marmsoler, Modeling and verifying dynamic architectures with factum studio, in: Formal Aspects of Component Software,FACS 2019,, Springer, 2019. URL: https://doi.org/10.1007/978-3-030-40914-2. doi:10.1007/978-3-030-40914-2.

[26] H. K. Gidey, D. Marmsoler, FACTum Studio, https://habtom.github.io/factum/, 2018.

[27] H. K. Gidey, D. Marmsoler, D. Ascher, Modeling adaptive self-healing systems, CoRR abs/2304.12773 (2023). URL: https://arxiv.org/abs/2304.12773. doi:10.48550/arXiv.2304.12773. arXiv:2304.12773.