Exploiting Interaction Affordances: On Engineering Autonomous Systems for the Web of Things

Andrei Ciortea, Simon Mayer, Olivier Boissier, Fabien Gandon

Exploiting Interaction Affordances: On Engineering Autonomous Systems for the Web of Things

Position Paper for the Second W3C Workshop on the Web of Things

Andrei Ciortea
University of St. Gallen
St. Gallen, Switzerland
Inria, Université Côte d'Azur, CNRS, I3S, Wimmics
Sophia Antipolis, France
andrei.ciortea@unisg.ch

Simon Mayer
University of St. Gallen and ETH Zürich
St. Gallen, Switzerland
simon.mayer@unisg.ch

Olivier Boissier
MINES Saint-Étienne, Université de Lyon, LabHC, CNRS
Saint-Étienne, France
olivier.boissier@emse.fr

Fabien Gandon
Inria, Université Côte d'Azur, CNRS, I3S, Wimmics
Sophia Antipolis, France
fabien.gandon@inria.fr

ABSTRACT

Interaction Affordances are central to the *W3C Web of Things (WoT)*: they allow *Consumers* to identify and use the capabilities provided by *Web Things*.[1] Ideally, interaction affordances would allow consumers to *arrive-and-operate* in any W3C WoT environment: given an entry URI, consumers would be able to achieve their tasks *in an autonomous manner* by navigating the hypermedia and deciding among the various options presented to them at run time. A central challenge then, which is not typically within the scope of Web engineering, is how to *design*, *program*, *debug*, *monitor*, and *regulate* such autonomous consumers of Web Things. The engineering of similar autonomous systems has been studied to a large extent in research on *multi-agent systems (MAS)*, and we believe that tapping into the large body of MAS research holds great promise for unlocking the full potential of the W3C WoT. In this position paper, we motivate and present our vision for autonomous systems in the WoT, and support this vision with a prototype for industrial manufacturing. We then discuss some of the challenges and opportunities raised by bringing *autonomy* to the WoT.

KEYWORDS

Web of Things, Hypermedia, Multi-Agent Systems, Semantic Web

1 INTRODUCTION

The Web of Things (WoT) fosters innovation and rapid prototyping in the Internet of Things: developers can use standard Web technologies to build applications on top of mash-ups of devices and digital services (a.k.a. *WoT mashups* or *physical mashups* [10]). Two paradigms that were very successful early-on in showcasing these benefits were inspired by *dataflow programming* and *rule-based systems*: in the former approach, developers and/or end-users would use a visual programming language to create directed graphs of devices and digital services (e.g., with Node-RED[2]), while in the latter approach they would define *event-condition-action* rules where an event triggers the execution of one or more actions (e.g., with IFTTT[3]). Both approaches ease the development of *reactive* WoT systems that are very responsive to sensory input, but their main drawback is that – once deployed – these systems cannot adapt their behavior to changes in the environment or to new user requirements at run time: they are built on static WoT mashups that have to be defined and maintained manually. The W3C WoT helps mitigate these limitations to great extent through *interaction affordances* and *hypermedia controls*: it enables WoT mashups to be defined in terms of abstract interaction patterns rather than specific protocols and device APIs. The resulting WoT mashups are then more flexible as they are loosely coupled to the underlying device APIs, but they still have to be defined and maintained manually.

To avoid the use of static WoT mashups, some WoT researchers turned to automated planning [14, 16]: given a design goal or a user-specified goal, it is left to the WoT system to infer how to achieve that goal using a reasoner and interaction affordances discovered at run time. Such systems are more adaptable (the WoT mashups are created at run time), but they are also less responsive: automated planning is computationally costly, and environmental changes are taken into account only before the planning phase (the inferred plan can become invalid during execution).

The W3C WoT already provides essential building blocks for engineering flexible WoT systems. We believe these building blocks, and in particular the introduction of *interaction affordances*, also hold the potential to unlock new practical use cases for autonomous systems on the Web [8]: many of the underlying research questions that we are now confronted with, such as *how to balance reactive and goal-directed behavior in WoT systems* or *how to enable goal-directed behavior in resource-constrained systems*, have already been explored to a large extent in the scientific literature on distributed artificial intelligence and, in particular, *multi-agent systems (MAS)* [23]. This area of research provides models, languages, and tools that could potentially be used to *design*, *program*, *debug*, *monitor*, and *regulate* systems of autonomous consumers of Web Things.

[1] This terminology is defined at W3C: https://www.w3.org/TR/wot-architecture/.
[2] http://nodered.org/
[3] http://www.ifttt.com/

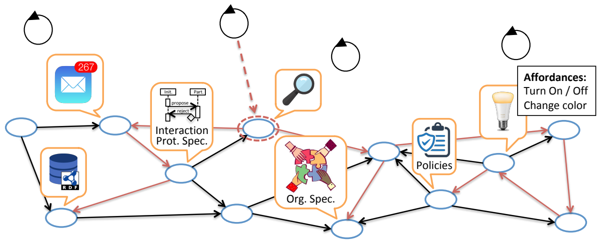

Figure 1: Hypermedia MAS: Agents are situated in an environment designed as a distributed hypermedia application (sourced from [8]).

In what follows, Section 2 briefly presents our vision for a new class of MAS for the WoT (and the Web in general), which we call *Hypermedia MAS* [7, 8][4]. Section 3 then illustrates and supports this vision with a prototype for flexible industrial manufacturing [9]. We discuss opportunities and challenges raised by bringing *autonomy* to the WoT in Section 4.

2 HYPERMEDIA MULTI-AGENT SYSTEMS

In AI research, an *agent* is commonly defined as "a computer system, situated in some environment, that is capable of flexible autonomous action in order to meet its design objectives" [13]. *Autonomy* is central to this definition and refers to the agent's ability to operate on its own, without the need of direct intervention from people or other agents. An *autonomous agent* thus has control over its actions and internal state. The agent is typically situated in an external environment that it can perceive via sensors and influence via actuators.[5] In distributed AI, a *multi-agent system (MAS)* is then a system conceptualized in terms of agents that are situated in a *shared environment* and interact with one another to achieve their design objectives [13, 23].

We envision a new class of MAS that are aligned with the Web architecture to inherit the properties of the Web as a *world-wide*, *open*, and *long-lived* system. These envisioned Web-based MAS use hypermedia as a general mechanism to support *uniform interaction* among *heterogeneous entities* (e.g., agents, devices, digital services, knowledge repositories, organizations, datasets, any Web resource) – and we call them *Hypermedia MAS* [7, 8]. Figure 1 illustrates our vision: we conceive of Hypermedia MAS as socio-technical systems composed of *people* and *autonomous agents* – henceforth *agents* – that are situated and interact in a shared *hypermedia environment* distributed across the Web. To achieve their goals, agents navigate the hypermedia environment to discover other entities in the MAS that they could interact with and the interaction affordances provided by those entities. Agents could navigate and search the hypermedia environment themselves, or they could use hypermedia search engines (cf. Figure 1) – an approach that has proven successful on the Web.

[4]https://hyperagents.org/

[5]A distinctive feature of an autonomous agent is its flexibility in the pursuit of some design objectives [13]: the agent is *reactive* by responding to changes in the environment in a timely fashion, *proactive* by exhibiting goal-directed behavior and taking the initiative when appropriate, and *social* by interacting with humans or other agents in order to achieve complex tasks that would otherwise overcome its own capabilities.

Here, we consider the *hypermedia environment* as a *first-class abstraction* in the MAS, which is a view based on a long line of research in engineering MAS [21, 24]: it is a key component designed and programmed with clear-cut responsibilities, such as mediating interaction among agents and access to the deployment context (e.g., devices, digital services), and providing an abstraction layer for modeling, representing, and programming non-autonomous entities (e.g., *artifacts* that agents can use [21]). In this vision, the Web provides the underpinning that interconnects *all entities* – within and across MAS.

3 DEMONSTRATOR: AGENT-BASED MANUFACTURING FOR THE WOT

To illustrate and support our vision for Hypermedia MAS in the context of the W3C WoT, in this section we present briefly a prototype for flexible industrial manufacturing that integrates automated planning with *multi-agent oriented programming* [4] for WoT systems. This prototype was developed at Siemens Corporate Technology and is described in detail in [9]. A demonstrator video of our deployment is available online.[6]

3.1 Flexible Assembly of Customized Furniture

The scenario considered for this prototype is the assembly of customized furniture. This market seems particularly appealing for flexible manufacturing systems both due to high potential and high demand for customization: even when restricting our scope to only a few different types of furniture, colors, leg configurations (e.g., three-legged vs. four-legged stools), and types of floor protector pads, the manufacturing line already faces several hundreds of product variations. This scenario thus warrants the use of manufacturing systems that are capable of switching between product variants and can be extended with additional industrial devices. The concrete example we use is switching between the assembly of several different variants of a wooden stool: our system should be able to continuously manufacture *stools with three legs*; this behavior can then be interrupted by requests to assemble stools with a *four-leg configuration*, which does not require any adjustment of the manufacturing line itself but merely of the behaviors of involved agents (i.e., a manufacturing robot and a human worker, in our example); furthermore, the system should react appropriately to assembly requests for stools with *padded legs* (in any leg configuration) – in this case, the system should transparently add a third agent (e.g., a second robot) that delivers felt pads to the assembly station.

3.2 System Design and Implementation

Figure 2 depicts an overview of the layers of abstraction used in our approach. The various *programming abstractions* – agents, artifacts, organizations – were inspired by the JaCaMo meta-model for *multi-agent oriented programming* [4]. We motivate and present in detail these abstractions in the context of the WoT in [6].

The *agent environment* (cf. Figure 2) is composed of *workspaces*, which are dynamic sets of artifacts that agents can use to achieve their goals. *Artifacts* can be both physical entities (e.g., devices on

[6]https://youtu.be/tfAVDpYn_ow

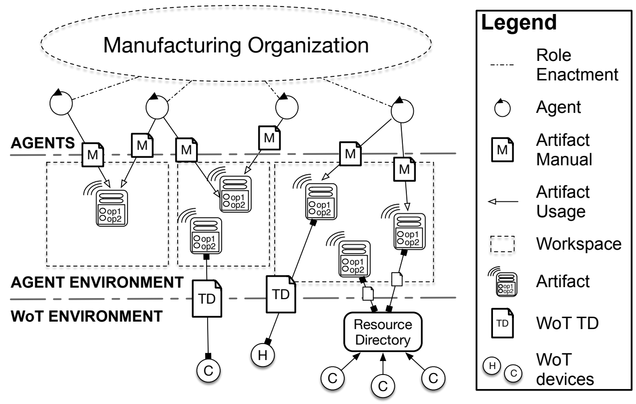

Figure 2: The various layers of abstraction in our agent-based manufacturing system for the WoT (sourced from [9]).

the shop floor) and digital entities (e.g., an automated planner). Physical artifacts are loosely coupled to devices via *W3C WoT Thing Descriptions (WoT TD)*.[7] To achieve their manufacturing goals, autonomous agents either use production plans that have already been programmed by an engineer, or they use an automated planner to synthesize plans from semantic descriptions of artifacts available in their environment at run time (the *artifact manuals* in Figure 2). All production plans, either programmed or inferred, are represented in *AgentSpeak* [5], a high-level programming language for cognitive agents. Production engineers can thus use the same programming language both for writing plans and for inspecting or editing plans synthesized by agents at run time.

We use multi-agent *organizations* [12] (cf. Figure 2) to coordinate all agents in our system (i.e., factory workers and autonomous agents), and to hot-deploy manufacturing specifications at run time – which allows production engineers to repurpose the manufacturing system on-the-fly (see [9] for details). Production engineers use a graphical Web front end to define and deploy manufacturing organizations, to configure and monitor the system, and to write, inspect, and edit production plans.

We implemented the autonomous agents, the artifacts they use, and the multi-agent organizations with the JaCaMo platform for MAS [4]. The semantic models used in our system (e.g., organizational specifications, agent descriptions) are stored and managed with the *Open Semantic Framework (OSF)* [15], a standards-compliant industrial knowledge management platform.

3.3 Deployment

We deployed the presented system in a prototypical production cell containing two handling robots that interact with human workers. The robots are controlled by systems that are representative for a state-of-the-art production cell in a real manufacturing environment. The first of the two handling robots is a *Universal Robotics UR5* robot with an attached *Robotiq 2-Finger Gripper* that is connected to the rest of the system via ROS [20], a Linux-based robot programming framework that includes modules for movement planning, optimization, and visualization. The second robot is a *Fischertechnik ROBO TX Automation Robot* that is controlled via an industrial controller (a *Siemens S7-300 PLC*) and is programmed by means of the *STEP7* language using ladder logic. Both robot controllers expose REST HTTP APIs for integration with the rest of the system.[8]

The robots interact with human workers through an *augmented reality (AR)* interface for *Microsoft HoloLens*[9] devices.[10] This setup allows us to demonstrate that our system enables *collaborative robots* (i.e., the UR5), which are widely perceived as important future participants in industrial production processes, to interact with people as well as with devices controlled via *conventional factory automation systems* such as PLCs. Furthermore, by building on top of the WoT, our system can be seamlessly extended with any Web-enabled devices that expose WoT TDs for their Web APIs.

4 AUTONOMY IN THE WOT: CHALLENGES AND OPPORTUNITIES

Weaving Hypermedia MAS into the open Web raises a broad range of technical, social, and ethical challenges (see also [8]). In this section, we discuss two immediate challenges that we believe are of particular interest in the context of the W3C WoT.

4.1 Interaction as a first-class abstraction

The W3C WoT TD and other recent initiatives such as Hydra[11] pave the way for declarative specifications of interactions on the Web: consumers of Web APIs no longer have to rely on hard-coded interactions, but instead they can reliably interpret and use interaction affordances provided to them at run time. Further evolutionary steps would be (i) to enable consumers of Web APIs to reason on how to achieve their design goals using the interaction affordances provided to them at run time, and (ii) to enable any software agent on the Web to discover and interact not only with Web APIs, but also with people and other agents on the Web. Ideally, interaction would become *a first-class abstraction* on the Web: both people and autonomous agents would be able to *publish*, *discover*, *interpret*, *enact*, and *reason on* interaction protocols, and when applicable to *modify* the protocols as they evolve throughout their lifetime. We believe this is particularly important in the context of the WoT due to the open and dynamic nature of WoT systems.

The declarative specification and enactment of interactions has been studied to large extent in MAS research (we refer the interested reader to [2] for an introduction). One research opportunity would be to investigate the reuse of this previous work to further support and promote *interaction as a first-class abstraction* in the WoT (and on the Web in general).

[7]https://www.w3.org/TR/wot-thing-description/

[8]For more technical details about the setup of the laboratory production cell we refer interested readers to [17].

[9]https://www.microsoft.com/en-us/hololens/

[10]Human workers are proxied by software agents that forward to humans all their obligations (e.g., to achieve a specific goal) and signal their fulfillment on behalf of humans. Obligations are displayed to human workers in textual form, and workers use voice commands to notify when they have fulfilled their obligations.

[11]http://www.hydra-cg.com/spec/latest/core/

4.2 Regulation as a first-class abstraction

The *Robots Exclusion Protocol* (a.k.a. robots.txt)[12] allows website administrators to convey to crawlers policies for polite crawling. Rate limiting is also a common practice for Web APIs, but many times such policies are only presented in human-readable documentation. Another challenge for data consumers are licensing policies, which can hinder the reuse of data [22]. We expect these challenges will remain relevant in the context of the WoT, in particular as consumers of Web Things become increasingly autonomous. To address these challenges, consumers should be able to reliably identify, interpret, and reason on any norms applicable to their interactions with Web Things and any other resources on the Web. If norms are expressed unambiguously and in machine-readable formats, autonomous behavior can then be monitored and regulated. Without such mechanisms, however, the benefits of bringing autonomy to the WoT may be negated by intentional, accidental or erroneous risks raised by autonomous agents.

Regulation has been studied to a large extent in research on *normative MAS* [1], where *regulative norms* [18] or *prescriptions* [3] are generally used to specify who does what, in what context, and as subject to what deontic modality (e.g., obligation, prohibition, permission). Regulative norms, therefore, affect agent behaviour in an indirect manner: agents can decide to conform or not to regulative norms, for instance in order to balance internal motivation versus external consequences. Therefore, in order to be effective, regulative norms require enforcement mechanisms, for instance to sanction norm violators. Two types of enforcement mechanisms that have been studied in MAS research are [18]: (i) *social enforcement*, where control emerges from the independent actions of agents (e.g., approaches based on computational trust and reputation [19]), and (ii) normative organizations, where agent behavior is monitored and enforced in a top-down manner by an authority [12]. The various regulation models and mechanisms proposed by MAS research over the past decade can now be used to support and promote *regulation as a first-class abstraction* in the WoT (and on the Web in general).

5 CONCLUSIONS

The W3C WoT defines the building blocks required to create more flexible WoT systems. In particular, we believe that the introduction of *interaction affordances* is an important step in the evolution of the Web that can promote and motivate the need for autonomous agents (cf. original Semantic Web vision [11]). The last decade of research on MAS produced results that can now be directly applied to *design*, *program*, *debug*, *monitor*, and *regulate* autonomous agents for the WoT. All the elements required to build and deploy the Hypermedia MAS presented in our vision are already available. We believe that tapping into the large body of MAS research holds great promise for unlocking the full potential of the W3C WoT.

REFERENCES

[1] Giulia Andrighetto, Guido Governatori, Pablo Noriega, and Leendert WN van der Torre. 2013. *Normative multi-agent systems*. Vol. 4. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

[2] Matteo Baldoni, Cristina Baroglio, K Copra Amit, Akin Günay, et al. 2018. Interaction Protocols. In *Handbook of Normative Multiagent Systems*. College Publications. http://hdl.handle.net/2318/1685680

[3] Tina Balke, Célia da Costa Pereira, Frank Dignum, Emiliano Lorini, Antonino Rotolo, Wamberto Vasconcelos, and Serena Villata. 2013. Norms in MAS: Definitions and Related Concepts. In *Normative Multi-Agent Systems*, Giulia Andrighetto, Guido Governatori, Pablo Noriega, and Leendert W. N. van der Torre (Eds.). Dagstuhl Follow-Ups, Vol. 4. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 1–31. https://doi.org/10.4230/DFU.Vol4.12111.1

[4] Olivier Boissier, Rafael H. Bordini, Jomi F. Hübner, Alessandro Ricci, and Andrea Santi. 2013. Multi-agent oriented programming with JaCaMo. *Science of Computer Programming* 78, 6 (2013), 747 – 761. https://doi.org/10.1016/j.scico.2011.10.004

[5] Rafael H. Bordini, Jomi F. Hübner, and Michael Wooldridge. 2007. *Programming multi-agent systems in AgentSpeak using Jason*. John Wiley & Sons.

[6] Andrei Ciortea, Olivier Boissier, and Alessandro Ricci. 2017. Beyond Physical Mashups: Autonomous Systems for the Web of Things. In *Proc. 8th International Workshop on the Web of Things (WoT 2017)*. ACM, New York, NY, USA, 16–20. https://doi.org/10.1145/3199919.3199924

[7] Andrei Ciortea, Olivier Boissier Boissier, and Alessandro Ricci. 2018. Engineering World-Wide Multi-Agent Systems with Hypermedia. In *6th International Workshop on Engineering Multi-Agent Systems (EMAS 2018)*. http://emas2018.dibris.unige.it/images/papers/EMAS18-17.pdf

[8] Andrei Ciortea, Simon Mayer, Fabien Gandon, Olivier Boissier, Alessandro Ricci, and Antoine Zimmermann. 2019. A Decade in Hindsight: The Missing Bridge Between Multi-Agent Systems and the World Wide Web. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS 2019, Montreal, Canada, May 13-17, 2019*. International Foundation for Autonomous Agents and Multiagent Systems.

[9] Andrei Ciortea, Simon Mayer, and Florian Michahelles. 2018. Repurposing Manufacturing Lines on the Fly with Multi-agent Systems for the Web of Things. In *Proceedings of the 17th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS '18)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 813–822. http://dl.acm.org/citation.cfm?id=3237383.3237504

[10] Dominique Guinard, Vlad Trifa, Thomas Pham, and Olivier Liechti. 2009. Towards physical mashups in the web of things. In *Networked Sensing Systems (INSS), 2009 Sixth International Conference on*. IEEE, 1–4.

[11] James Hendler. 2001. Agents and the semantic web. *IEEE Intelligent systems* 16, 2 (2001), 30–37.

[12] Jomi F. Hübner, Olivier Boissier, Rosine Kitio, and Alessandro Ricci. 2010. Instrumenting multi-agent organisations with organisational artifacts and agents. *Autonomous Agents and Multi-Agent Systems* 20, 3 (01 May 2010), 369–400. https://doi.org/10.1007/s10458-009-9084-y

[13] N. R. Jennings and M. Wooldridge. 1998. *Applications of Intelligent Agents*. Springer Berlin Heidelberg, Berlin, Heidelberg, 3–28. https://doi.org/10.1007/978-3-662-03678-5_1

[14] M. Kovatsch, Y. N. Hassan, and S. Mayer. 2015. Practical semantics for the Internet of Things: Physical states, device mashups, and open questions. In *2015 5th International Conference on the Internet of Things (IOT)*. 54–61. https://doi.org/10.1109/IOT.2015.7356548

[15] S. Mayer, J. Hodges, D. Yu, M. Kritzler, and F. Michahelles. 2017. An Open Semantic Framework for the Industrial Internet of Things. *IEEE Intelligent Systems* 32, 1 (Jan 2017), 96–101. https://doi.org/10.1109/MIS.2017.9

[16] S. Mayer, N. Inhelder, R. Verborgh, R. Van de Walle, and F. Mattern. 2014. Configuration of smart environments made simple: Combining visual modeling with semantic metadata and reasoning. In *2014 International Conference on the Internet of Things (IOT)*. 61–66. https://doi.org/10.1109/IOT.2014.7030116

[17] Simon Mayer, Dominic Plangger, Florian Michahelles, and Simon Rothfuss. 2016. UberManufacturing: A Goal-Driven Collaborative Industrial Manufacturing Marketplace. In *Proceedings of the 6th International Conference on the Internet of Things*. ACM, 111–119.

[18] Sascha Ossowski. 2013. *Agreement technologies*. Vol. 8. Springer Science & Business Media. https://doi.org/10.1007/978-94-007-5583-3

[19] Isaac Pinyol and Jordi Sabater-Mir. 2013. Computational trust and reputation models for open multi-agent systems: a review. *Artificial Intelligence Review* 40, 1 (01 Jun 2013), 1–25. https://doi.org/10.1007/s10462-011-9277-z

[20] Morgan Quigley, Ken Conley, Brian P. Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y. Ng. 2009. ROS: an open-source Robot Operating System. In *ICRA Workshop on Open Source Software*, Vol. 3.

[21] Alessandro Ricci, Michele Piunti, and Mirko Viroli. 2011. Environment programming in multi-agent systems: an artifact-based perspective. *Autonomous Agents and Multi-Agent Systems* 23, 2 (2011), 158–192.

[22] Serena Villata and Fabien Gandon. 2012. Licenses Compatibility and Composition in the Web of Data. In *Third International Workshop on Consuming Linked Data (COLD2012)*. Boston, United States. https://hal.inria.fr/hal-01171125

[23] Gerhard Weiss. 2000. *Multiagent systems: a modern approach to distributed artificial intelligence*. MIT press.

[24] Danny Weyns, Andrea Omicini, and James Odell. 2007. Environment as a first class abstraction in multiagent systems. *Autonomous agents and multi-agent systems* 14, 1 (2007), 5–30.

[12] http://www.robotstxt.org/robotstxt.html