Domain-Expert Configuration of Hypermedia Multi-Agent Systems in Industrial Use Cases

Extended Abstract

Jérémy Lemée
University of St. Gallen
St. Gallen, Switzerland
jeremy.lemee@unisg.ch

Samuele Burattini
Università di Bologna
Cesena, Italy
samuele.burattini@unibo.it

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

Andrei Ciortea
University of St. Gallen
St. Gallen, Switzerland
andrei.ciortea@unisg.ch

ABSTRACT

Based on the analysis of two real-world use cases for agriculture and manufacturing, we suggest that Hypermedia Multi-Agent Systems (MAS) are a viable option to interconnect and coordinate devices, services, machine-learning systems, and people in industrial scenarios. We propose and implement an architecture based on three components: an infrastructure that manages Web of Things environments and executes Hypermedia MAS, a visual development environment for programming agents, and a goal specification interface for end-users. While the infrastructure manages information flows between the system components and provides an environment for agents, the visual language enables domain experts to configure the behaviour of the system leveraging agent-oriented programming abstractions both at design time and run time, and the goal specification interface permits users to delegate goals to the running Hypermedia MAS while re-using domain vocabulary.

CCS CONCEPTS

• Information systems → World Wide Web; **• Human-centered computing → Ubiquitous and mobile computing**; **• Computing methodologies → Multi-agent systems**.

KEYWORDS

Hypermedia Multi-Agent Systems; Web of Things

ACM Reference Format:
Jérémy Lemée, Samuele Burattini, Simon Mayer, and Andrei Ciortea. 2023. Domain-Expert Configuration of Hypermedia Multi-Agent Systems in Industrial Use Cases: Extended Abstract. In *Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), London, United Kingdom, May 29 – June 2, 2023*, IFAAMAS, 3 pages.

1 INTRODUCTION

By deploying the Internet Protocol stack on physical devices, including constrained devices, the Internet of Things (IoT) has fostered interoperability between devices and services on the *network layer*. The principles of the World Wide Web can be applied to IoT systems in order to extend this interoperability to the application layer [14], which has led to the definition of the Web of Things (WoT) by the World Wide Web Consortium (W3C)[1], a standardization effort aimed at defining a *uniform interface* for heterogeneous IoT devices based on the notion of Interaction Affordances[2].

We propose an architecture for the use of Hypermedia Multi-Agent Systems (Hypermedia MAS) to flexibly coordinate industrial cyber-physical systems (CPS) that may include devices, services, machine-learning systems, and even people. This architecture has been applied to two real-world use cases in the domains of *agriculture* and *industrial manufacturing*. The use cases helped us identify the functional and non-functional properties that our system should have and that are reflected in the resulting architecture. The heterogeneity of the two use cases suggests that our approach can support a wide range of scenarios in-between them and beyond.

2 USE CASES

In both use cases, agents are configured by domain experts to achieve a delegated goal by interacting with devices and services in the system (e.g., tractors, robots, machine-learning systems, etc.) which are all described using W3C WoT Thing Descriptions (TD) [13]. Agents then integrate and coordinate different types of services, including artificial intelligence systems, with sensors and actuators in the real world. While our objective is to preserve the autonomy of the system, in both use cases, the agents may involve humans as a fallback option when every other method fails.

Agriculture. In the agriculture use case, autonomous agents control self-driving tractors, selecting the right tractor among those available based on the its implements (e.g., sprayer, plough), specifying the waypoints where the tractors need to go, and operating the tractor in the field. The tractor has a camera to detect and avoid obstacles using a machine-learning system. The agent can further request the intervention of a remote human pilot in situations that the machine-learning system cannot handle, such as when there are unexpected situations on the planned course (e.g., animals).

Manufacturing. In the manufacturing use case, different wooden or metal work pieces are engraved with custom text at a given

Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), A. Ricci, W. Yeoh, N. Agmon, B. An (eds.), May 29 – June 2, 2023, London, United Kingdom.

[1] https://www.w3.org/WoT/
[2] https://www.w3.org/TR/wot-thing-description11/#interactionaffordance

position. When an agent receives a goal to produce an engraved piece, it controls a robot to fetch the piece from storage and insert it into an engraver. In this use case, cameras are used to perceive workpiece positions and a machine-learning system is used to compute optimal grab spots for the robot. In case of uncertainty, the system may request a human to actuate the robot.

3 ARCHITECTURE

We propose an architecture for industrial systems that can be easily configured by domain experts. The approach builds on top of Hypermedia MAS [6] to coordinate machines and services. The proposed architecture was developed to enable the following features:

- **F1** The system allows publishing of W3C WoT TDs of devices and services in the environment and the discovery of these W3C WoT TDs by other system components.
- **F2** The system implements the JaCaMo meta-model [2] and enables the run-time deployment of agents and artifacts.
- **F3** Domain experts, who have knowledge of the use case, may configure autonomous agents using end-user-friendly tools.
- **F4** End users (e.g., a farmer in the smart farming use case), can define goals using their domain vocabulary and can delegate these goals to agents.

Our proposed architecture includes the following components:

- A *Hypermedia MAS Infrastructure* used to program and run Hypermedia MAS.
- A *Web-based Agent IDE* with a visual programming system that domain experts without programming experience can use it to configure and deploy a Hypermedia MAS.
- A *Goal Specification User Interface* that is used by end users to send the goal to be achieved to a deployed MAS.

3.1 Hypermedia MAS Infrastructure

The Hypermedia MAS Infrastructure supports the creation of Hypermedia MAS following the JaCaMo meta-model and is an extension of the Yggdrasil framework [6] with: the Jason interpreter for agents programmed in AgentSpeak [3]; the CArtAgO framework for programming agent environments [12]; and the Moise framework for implementing multi-agent organizations [7].

The Hypermedia MAS Infrastructure supports the execution of computational artifacts, as defined in the JaCaMo meta-model [2]: It allows agents to instantiate artifacts, exposes HTTP interfaces for interacting with the artifacts, and generates W3C WoT TDs for the instantiated artifacts; these can be complemented by TDs for external devices or services. The resulting hypermedia environment is represented using the Resource Description Framework (RDF), which allows agents to query the environment at the knowledge level. Agents can furthermore create organizational artifacts [8] in the Hypermedia MAS environment. These are used to instantiate *Moise* organizations and support the coordination of agents in complex MAS. Finally, Jason agents can also be executed upon request on the Hypermedia MAS Infrastructure. The agents expose a message-box API to enable direct communication from other components both within or outside the Hypermedia MAS. The different abstractions of the JaCaMo meta-model can thus be programmed and used via the Hypermedia MAS Infrastructure.

3.2 Web-based Agent IDE

We hypothesize that the BDI abstractions [4] can be leveraged to simplify the programming of autonomous agents for people without programming expertise due to their roots in human practical reasoning. We hence developed a visual programming language [9] for Hypermedia MAS. This programming language targets domain experts (e.g., industrial engineers) who can use it to define Hypermedia MAS behavior. Our Web-based Agent IDE [5] is implemented as a Web application and lets users program BDI agents with a block language [1, 10] based on AgentSpeak [11].

To let the agents interact with available devices and services, W3C WoT TDs are leveraged: a TD is a way to define, at design time, a generic interface that may be satisfied by concrete *Things* at run time through specific protocol bindings. In this way, agent logic can be implemented against Interaction Affordances defined in a TD while resolving the protocol binding only at run time. This avoids compiling against bindings that might change at run time and hence break the application. The Web-based IDE interacts with the Hypermedia MAS Infrastructure to (i) retrieve the TD of the devices and services in the system for agents to interact with; (ii) deploy and manage the agents running on the platform.

3.3 Goal Specification User Interface

The Goal Specification UI is a component that allows people to specify the goals they want the Hypermedia MAS to achieve. This (simple) component is the only part of the proposed architecture that needs to be adapted from one use case to another. This was a conscious design decision, as end users can be ideally satisfied by integrating the possibility to specify domain goals with an interface that they are already familiar with (e.g., a farm management system such as `FarmOS`). Through the UI, the user can select parameters of domain-specific goal templates (which are represented as JSON documents satisfying a given JSON schema), and these goals are then sent to a Manager Agent inside the Hypermedia MAS that runs on the Hypermedia MAS infrastructure. This Manager Agent decides whether the goal can be processed by the Hypermedia MAS and can then delegate the goal to the agents running on the Hypermedia MAS Infrastructure. This is especially useful because it decouples the UI from the actual agents that realize the goal.

4 CONCLUSION

We identified the need for more intelligent and autonomous behaviour in CPS that involve the cooperation of devices, services, different machine-learning systems, and humans. At the same time, the system should remain under the control of human workers, even if they are not experts in programming. To support this vision, we conceptualize and implement an architecture based on three components: a *Hypermedia MAS Infrastructure* managing both the WoT-based environment and the execution of software agents, a *Web-based Agent IDE* supporting agent programming with a visual language, and a *Goal Specification UI* tailored to each use case to be better understandable by the end users and let them delegate goals to the Hypermedia MAS. The proposed architecture enables domain experts to create, configure, deploy, and reconfigure (at run time) a Hypermedia MAS to control a realistic industrial shopfloor and a smart farm through the WoT.

ACKNOWLEDGEMENTS

This research has received funding from the European Union's Horizon 2020 research and innovation program under grant No. 957218 (*IntellIoT*) and the Swiss National Science Foundation under grant No. 189474 (*HyperAgents*).

REFERENCES

[1] David Bau, Jeff Gray, Caitlin Kelleher, Josh Sheldon, and Franklyn Turbak. 2017. Learnable programming: blocks and beyond. *Commun. ACM* 60, 6 (2017), 72–80.

[2] 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.

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

[4] Michael E Bratman, David J Israel, and Martha E Pollack. 1988. Plans and resource-bounded practical reasoning. *Computational intelligence* 4, 3 (1988), 349–355.

[5] Samuele Burattini, Alessandro Ricci, Simon Mayer, Danai Vachtsevanou, Jérémy Lemee, Andrei Ciortea, and Angelo Croatti. 2022. Agent-Oriented Visual Programming for the Web of Things. In *Engineering Multi-Agent Systems - 10th International Workshop, EMAS 2022*.

[6] Andrei Ciortea, Olivier Boissier, and Alessandro Ricci. 2018. Engineering world-wide multi-agent systems with hypermedia. In *International Workshop on Engineering Multi-Agent Systems*. Springer, 285–301.

[7] Jomi Fred Hübner, Jaime Simao Sichman, and Olivier Boissier. 2002. MOISE+ towards a structural, functional, and deontic model for MAS organization. In *Proceedings of the first international joint conference on Autonomous agents and multiagent systems: part 1*. 501–502.

[8] Rosine Kitio, Olivier Boissier, Jomi Fred Hübner, and Alessandro Ricci. 2007. Organisational artifacts and agents for open multi-agent organisations:"giving the power back to the agents". In *International Workshop on Coordination, Organizations, Institutions, and Norms in Agent Systems*. Springer, 171–186.

[9] Brad A Myers. 1990. Taxonomies of visual programming and program visualization. *Journal of Visual Languages & Computing* 1, 1 (1990), 97–123.

[10] Erik Pasternak, Rachel Fenichel, and Andrew N Marshall. 2017. Tips for creating a block language with blockly. In *2017 IEEE blocks and beyond workshop (B&B)*. IEEE, 21–24.

[11] Anand S Rao. 1996. AgentSpeak (L): BDI agents speak out in a logical computable language. In *European workshop on modelling autonomous agents in a multi-agent world*. Springer, 42–55.

[12] Alessandro Ricci, Mirko Viroli, and Andrea Omicini. 2006. CArtAgO: A framework for prototyping artifact-based environments in MAS. In *International Workshop on Environments for Multi-Agent Systems*. Springer, 67–86.

[13] Kaebisch Sebastian, McCool Michael, Korkan Ege, Kamiya Takuki, Charpenay Victor, and Kovatsch Matthias. 2020. *Web of Things (WoT) Thing Description*. Technical Report. W3C. https://www.w3.org/TR/wot-thing-description/

[14] Erik Wilde. 2007. Putting things to REST. (2007).