Report from Dagstuhl Seminar 23081

Agents on the Web

Olivier Boissier*[1], Andrei Ciortea*[2], Andreas Harth*[3], Alessandro Ricci*[4], and Danai Vachtsevanou†[5]

1 **Ecole des Mines – St. Étienne, FR.** olivier.boissier@emse.fr
2 **Universität St. Gallen, CH.** andrei.ciortea@unisg.ch
3 **Fraunhofer IIS – Nürnberg, DE.** andreas@harth.org
4 **Università di Bologna, IT.** a.ricci@unibo.it
5 **Universität St. Gallen, CH.** danai.vachtsevanou@unisg.ch

Abstract

Recent standardization on the Web of Things and (Social) Linked Data unlocks new practical use cases and new opportunities for research on Web-based multi-agent systems. While existing research on multi-agent systems can contribute to engineering adaptive and flexible Web-based systems, increased deployment of systems following the recent standards can bring new insight into engineering large-scale and open multi-agent systems. These developments motivate the need for a broader perspective that can only be achieved through a concerted effort of the research communities on *Web Architecture and Web of Things*, *Semantic Web and Linked Data*, and *Autonomous Agents and Multi-Agent Systems*. Thus, the main objective of the *Dagstuhl Seminar 23081 on Agents on the Web* was to investigate these new research opportunities, to support the transfer of knowledge and results across the different communities, and to create a network of leading scholars and practitioners around these topics. This report documents the seminar's program and outcomes. To continue the joint work after the seminar, the seminar participants created the *W3C Autonomous Agents on the Web (WebAgents) Community Group*.

Seminar February 19–24, 2023 – https://www.dagstuhl.de/23081
2012 ACM Subject Classification Information systems → World Wide Web; Computing methodologies → Multi-agent systems; Human-centered computing → Ubiquitous computing
Keywords and phrases Web Architecture, Web of Things, Semantic Web, Linked Data, Multi-Agent Systems
Digital Object Identifier 10.4230/DagRep.13.2.71

1 Executive Summary

Olivier Boissier (Ecole des Mines – St. Étienne, FR)
Andrei Ciortea (Universität St. Gallen, CH)
Andreas Harth (Fraunhofer IIS – Nürnberg, DE)
Alessandro Ricci (Università di Bologna, IT)

The recent evolution towards hypermedia-driven services, Linked Data, and the Web of Things is turning the Web into a homogeneous hypermedia fabric that interconnects everything – devices, physical objects, documents, or abstract concepts. The latest standards on the Web of Things and (Social) Linked Data allow automated software clients to *navigate*, *query*, *observe*, and *act on* this uniform hypermedia fabric. Use cases that have long been envisioned

* Editor / Organizer
† Editorial Assistant / Collector

for artificial agents on the Web – as published in the original Semantic Web vision in 2001 and going back to the early days of the Internet – are now closer to their practical implementation and deployment. Nevertheless, the engineering of such modern Web-based systems poses research challenges that have yet to be addressed.

Today's Web-based systems are inherently complex, heterogeneous, and increasingly dynamic (e.g., especially in the context of the Web of Things). Moreover, the World Wide Web was designed to be a decentralized system that spans not only geographical boundaries but also organizational boundaries. In such settings, traditional paradigms for engineering Web-based systems become impractical. Many of the research questions underlying these challenges – such as how to design software agents able to cope with complex and dynamic environments, or how to design and govern interactions in decentralized systems – have been investigated in research on autonomous agents and multi-agent systems.[1] To design a new generation of Web-based systems, we thus need a broader perspective that can only be achieved through a concerted effort of several research communities – which, for the purpose of this seminar, we identified as the communities around *Web Architecture and Web of Things*, *Semantic Web and Linked Data*, and *Autonomous Agents and Multi-Agent Systems*.

The *Dagstuhl Seminar 23081 on Agents on the Web* continued to investigate the research opportunities identified in the *Dagstuhl Seminar 21072 on Autonomous Agents on the Web* in order to consolidate the discussions and to continue the transfer of knowledge and results across the involved research communities. Concretely, the seminar pursued the following objectives:

- to identify and align the various research threads in the targeted communities that are relevant for advancing the research on Web-based multi-agent systems;
- to work towards a shared conceptualization and shared theoretical underpinnings for Web-based multi-agent systems;
- to identify representative use cases in different domains that would help demonstrate the potential impact of this joint research effort on society and the economy;
- to evaluate the state of technologies available for prototyping and deploying Web-based multi-agent systems.

The main motivation for this seminar was to consolidate a network of senior and young researchers around the topics. To continue the joint work after the seminar, participants created the *W3C Autonomous Agents on the Web (WebAgents) Community Group*.[2]

The Seminar Format

The seminar brought together 39 participants across the above-mentioned research communities. The 5-day seminar was a blend of presentations, live demonstrators, and group work structured around three types of sessions:

- *presentations of position statements*: sessions organized on the first day based on position statements submitted by the seminar participants;
- *Demos & Tech sessions*: sessions organized on the afternoon of the second day for presenting demonstrators and technologies relevant to the seminar;
- *working group sessions*: sessions organized throughout the week for focused group work on specific topics identified during the seminar.

Figure 1 Concept boards used to organize the seminar topics into working groups.

The seminar started with presentations of position statements submitted by participants to help bootstrap the discussions. Participants were invited to read all submitted position statements before the seminar and to prepare a 1-minute presentation of their position statements. To help create a narrative throughout the day, a selection of topics was invited for 5-minute presentations. The first day ended with a brainstorming session, in which participants used several concept boards to organize the seminar topics and to form working groups for the rest of the week.

The first half of the second day started with working group sessions, and the afternoon was reserved for the Demos & Tech sessions. The objective of these sessions was to ground the conceptual discussions from the working groups and to paint a picture of what can already be achieved with existing technologies. In total, these sessions attracted 16 demonstrators, out of which 7 submitted abstracts for this report (see Section 4).

The rest of the week continued with working group sessions. Beginning of each day, participants were invited to pitch new ideas for working groups. In addition, a synchronization session was organized in the middle of the week to review the progress, reinforce bridges across working groups, and reorganize the working groups if needed. In total, seven working groups were created and their consolidated discussion summaries are presented in Section 5.

Overview of the Report

This report is organized into four main parts. Section 3 includes the position statements submitted by seminar participants that were presented during the first day. Section 4 includes a list of abstracts for several of the demonstrators presented during the seminar. Section 5 includes the reports submitted by the working groups created during the seminar.

[1] For a broad overview of this area, we refer the interested reader to: Gerhard Weiss (ed.), *Multiagent Systems (Second Edition)*, MIT Press, ISBN 978-0262018890, 2013.

[2] `https://www.w3.org/community/webagents/`

2 Table of Contents

Working Group Reports

3 Overview of Position Statements

3.1 How to Ensure Agents on the Web are Behaving as Expected?

Cleber Jorge Amaral (Universidade Federal de Santa Catarina, BR)

Agents have access to an abundance of resources and interactions on the Web. At present, complex Web agents such as trading bots and personal assistants utilize a combination of data-driven and symbolic techniques to develop their reasoning capabilities [1]. For the Multi-Agent Systems (MAS) community, Agent-Oriented Programming (AOP) languages based on the BDI model of agency, which offers abstractions for knowledge, intentions, objectives, and plans, forms an excellent foundation for combining different symbolic, stochastic, and sub-symbolic AI techniques [2]. However, it can be difficult to ensure the behavior of autonomous and proactive software artifacts is reliable given the diversity of techniques and environment richness.

Indeed, as the Web offers an unpredictable and dynamic environment and agents are usually part of a system that contains several agents, it is tough to reproduce and control interactions of agents with other agents and humans [3]. Besides, it can be hard to test and even to understand agents as they may present behaviors that have no formal specification and they can have several alternative plans for achieving a particular goal. Other aspects such as scaling to numerous autonomous software artifacts running relatively freely in the environment that can also contain malign agents can be even more problematic [4].

Given the aforementioned difficulties, the following questions can be raised:

- How to ensure teleo-reactive agents on the Web are behaving as expected considering a multiplicity of course of actions that they can take for achieving a particular goal?
- How can it be ensured that a new version of a certain MAS is strengthening rather than weakening in the many scenarios to which the preceding version was subjected?
- How can be create reproducible Web environments that are rich in resources, interactions, and dynamic that can be used as a development sandbox and to compare various techniques?

References

1 Tarek R. Besold and Artur d'Avila Garcez and Sebastian Bader and Howard Bowman and Pedro Domingos and Pascal Hitzler and Kai-Uwe Kuehnberger and Luis C. Lamb and Daniel Lowd and Priscila Machado Vieira Lima and Leo de Penning and Gadi Pinkas and Hoifung Poon and Gerson Zaverucha. *Neural-Symbolic Learning and Reasoning: A Survey and Interpretation*. 2017. `https://arxiv.org/abs/1711.03902`

2 Bordini, Rafael H. and El Fallah Seghrouchni, Amal and Hindriks, Koen and Logan, Brian and Ricci, Alessandro. *Agent programming in the cognitive era*. Autonomous Agents and Multi-Agent Systems, 2020. `https://doi.org/10.1007/s10458-020-09453-y`

3 Linz, Tilo. *Testing autonomous systems*. The Future of Software Quality Assurance, Springer International Publishing, 2020. `https://doi.org/10.1007/978-3-030-29509-7_5`

4 Koopman, Philip and Wagner, Michael. *Challenges in autonomous vehicle testing and validation*. SAE International Journal of Transportation Safety, 2016. `https://www.jstor.org/stable/26167741`

3.2 Swarm Intelligence for Agents in Smart Edge Environments

Jean-Paul Calbimonte (HES-SO Valais – Sierre, CH)

Motivation

Smart devices, sensors, autonomous robots, drones, and other similar instruments have increasingly become pervasive in industrial and home environments, for numerous use-cases and scenarios. These devices are not limited to the acquisition of data as mere observers of their surroundings, but they are also capable of actuation, and of potentially complex processing of rapidly produced data flows. Given their limitations in terms of computational, storage, and power capabilities, some of these processing features can be delegated to intermediate edge nodes in a network of smart devices, integrated with cloud interactions when required. While in simple cases a manual and explicit set up of this infrastructure is possible, the reality is that more and more use-cases are subject to highly dynamic changes in the number and type of devices, as well as the nature of their interactions. Under these conditions it becomes necessary to use higher-level abstractions that allow the self-organization of smart devices, bringing intelligence to the edge nodes, and allowing the collaborative distribution of their computational tasks through swarm-inspired behavioral patterns.

Challenges and open issues

Considering the potential of swarm agents for self-organizing edge nodes and smart devices, we identify a number of relevant challenges:

- Heterogeneous agent knowledge: Multiple schemas and ontologies exist for representing sensor and IoT devices and data. Are these enough to allow self-organization? [1, 5, 2]
- Are there swarm-inspired models that can be propagated to smart edge agents? Can these act as mediators among devices according to their capabilities? [4]
- What is the role of social coordination of agents in a swarm environment? Can the Web serve as a common place to enable the establishment of inter-deployment knowledge exchanges? [8]
- Negotiation in the context of swarm agents can bring both advantageous optimization patterns, as well as delays and potential conflicts. It may become crucial to identify ways of reducing the risks, and to find compromises that adapt to changing situations and redefinition of goals. [7, 6]
- In this context the risks of manipulation and mischievous behaviors is more than plausible. The conception of integrity, confidence, and transparency mechanisms need to be integrated into this research road-map.
- Edge and sensing devices are often required to acquire and handle sensitive data, which should be subject to strict privacy constraints, while keeping the computation and processing goals. [3, 9]

Opportunities

Addressing these challenges, swarm intelligent agents for smart edge may open a number of research opportunities:

- Semantic representation of shared goals, organization patterns, and behaviors have the potential to facilitate agent coordination and negotiation in swarm smart edge environments.
- Implementation of low-code solutions can help abstracting the complexity of smart edge agents, and their deployment under heterogeneous conditions.
- Smart edge agents necessarily need to go well beyond the Web as a means to interact. Nevertheless, the Web offers a solid and standard mechanism to enable the interoperability, discovery, and accessibility among different swarms.
- Fully decentralized swarms of agents can make use of existing approaches for self-organization that have been successfully used in previous works in other contexts.
- Domain-specific deployments can help drive the requirements and implementation of this idea on different areas including automation, self-driving vehicles, robotics, domotics, eHealth, etc.
- Federated learning and decentralized processing can be adapted to run on swarm agents for smart edge environments, further expanding their applicability.

References

1 M. Bermudez-Edo, T. Elsaleh, P. Barnaghi, and K. Taylor. Iot-lite: a lightweight semantic model for the internet of things. In *International IEEE conferences on ubiquitous intelligence & computing*, pages 90–97. IEEE, 2016.

2 V. Charpenay and S. Käbisch. On modeling the physical world as a collection of things: The w3c thing description ontology. In *The Semantic Web: 17th International Conference, ESWC 2020, Heraklion, Crete, Greece, May 31–June 4, 2020, Proceedings 17*, pages 599–615. Springer, 2020.

3 L. Chen, S. Fu, L. Lin, Y. Luo, and W. Zhao. Privacy-preserving swarm learning based on homomorphic encryption. In *Algorithms and Architectures for Parallel Processing: 21st International Conference, ICA3PP 2021, Virtual Event, December 3–5, 2021, Proceedings, Part III*, pages 509–523. Springer, 2022.

4 C. Guéret, S. Schlobach, K. Dentler, M. Schut, and G. Eiben. Evolutionary and swarm computing for the semantic web. *IEEE Computational Intelligence Magazine*, 7(2):16–31, 2012.

5 K. Janowicz, A. Haller, S. J. Cox, D. Le Phuoc, and M. Lefrançois. Sosa: A lightweight ontology for sensors, observations, samples, and actuators. *Journal of Web Semantics*, 56:1–10, 2019.

6 N. Kouka, R. Fdhila, and A. M. Alimi. Multi objective particle swarm optimization based cooperative agents with automated negotiation. In *Neural Information Processing: 24th International Conference, ICONIP 2017, Guangzhou, China, November 14–18, 2017, Proceedings, Part IV 24*, pages 269–278. Springer, 2017.

7 F. Lorenzi, D. S. dos Santos, and A. L. Bazzan. Negotiation for task allocation among agents in case-base recommender systems: a swarm-intelligence approach. In *2005 International workshop on multi-agent information retrieval and recommender systems*, pages 23–27, 2005.

8 S. Mokarizadeh, A. Grosso, M. Matskin, P. Kungas, and A. Haseeb. Applying semantic web service composition for action planning in multi-robot systems. In *2009 Fourth International Conference on Internet and Web Applications and Services*, pages 370–376. IEEE, 2009.

9 B. Zhao, X. Liu, A. Song, W.-N. Chen, K.-K. Lai, J. Zhang, and R. H. Deng. Primpso: A privacy-preserving multiagent particle swarm optimization algorithm. *IEEE Transactions on Cybernetics*, 2022.

3.3 On Information and Interactions on the Web

Victor Charpenay (Mines Saint-Étienne, FR)

Information on the Web is (primarily) symbolic

In the past few years, most spectacular advances in research have been made possible by the Web, to a large extent. The field of computer vision owes much to ImageNet, a dataset compiled with images found on the Web[3]. In natural language processing, large language models such as BERT or GPT are trained on large corpi found on the Web, including Wikipedia and CommonCrawl[4] data.

Yet, the Web holds value in itself not because it makes it easy to publish and access large dumps of data (most open data is hard to use because it is heterogeneous and lacks contextual information) but because pieces of information coming from different sources can easily be interlinked.

That makes the Web a source of information that is essentially relational. The Resource Description Framework (RDF) captures that essence by restricting its data model to triples of the form $\langle resource, relation, resource \rangle$, which generalizes hyperlinks. As a consequence, Web agents *should* be able to deal with relational data, if not RDF data.

Information on the Web is (sometimes) approximate

It is legitimate to ask oneself whether the Web would bring anything new to Multi-Agent System (MAS) design. Because of the genericity of its architectural principles, any MAS can be implemented on the Web [3]. But is there a MAS that can *only* be implemented on the Web? If there is one, it will most likely be at a large scale (both with respect to the size of the environment and to the complexity of agent behaviors).

As soon as data is exchanged at a large scale, there is a long-tail phenomenon when it comes to its quality. This phenomenon is well-known among Semantic Web researchers, who have accepted the fact that information found on the Web is sometimes approximate: few publishers will spend a significant amount of time on the quality of the data they publish but most publishers won't. This was the starting point for the Pedantic Web group[5], that aimed at increasing data quality with low effort (mostly via better tooling).

Accepting approximations requires including a certain amount of statistical inference and learning in agent design. With the coming of Knowledge Graphs (KG), large collections of well-known facts [4], it is possible to learn vector representations of resources and relations, such that an agent can estimate the truth value of unknown $\langle resource, relation, relation \rangle$ statements. Web agents *should* be able to leverage KG latent representations.

Interactions on the Web are (strictly) discrete

To act on the Web, agents must fill in forms and send individual requests. This uniform layer for both agent-to-environment and agent-to-agent interactions requires to describe actions in terms of input, output, preconditions and effects (IOPE). It also requires that agents have discrete, symbolic models of perception and action, very different from e.g. models based on physical simulations.

[3] `https://image-net.org/about.php`
[4] `https://commoncrawl.org/`
[5] `https://pedantic-web.org/`

Such as statement summons “orthodox” approaches in MAS research, whose common objective has been to find a logical language that was both easy to understand and expressive enough to describe any MAS. One can mention e.g. Michael Wooldridge’s Logic of Rational Agents ($\mathcal{LORA}$) [2] and Michael Fisher’s METATEM [1]. These two languages have not been widely adopted for specifying agent behaviors but they remain highly relevant for verifying the behavior of agents that use non-symbolic latent representations.

Both $\mathcal{LORA}$ and METATEM rely on temporal logics. The Web Ontology Language (OWL), which should be the language of choice to model information on the Web, does not align well with temporal logics. It does, however, easily capture other useful modalities to express beliefs, knowledge, morality, etc. Web agents *should* be able to interpret a mixture of temporal and OWL specifications and have rational (i.e. logical) behaviors with respect to these specifications.

References

1 M. Fisher, An Introduction to Practical Formal Methods Using Temporal Logic. Chichester, UK: John Wiley & Sons, Ltd, 2011.
2 M. J. Wooldridge, Reasoning about rational agents. Cambridge, Mass: MIT Press, 2000.
3 V. Charpenay, T. Käfer, and A. Harth, “A Unifying Framework for Agency in Hypermedia Environments,” in Engineering Multi-Agent Systems, Springer International Publishing, 2022.
4 A. Hogan et al., “Knowledge Graphs,” ACM Compututer Surveys, 2021.

3.4 Towards Decentralized Applications based on Protocols, Norms, and Accountability

Amit K. Chopra (Lancaster University, GB)

Web applications were envisaged to support social processes, to facilitate interactions between their users. However, a Web application is a conceptually centralized entity that mediates and, as exemplified by popular social media platforms, influences and controls their interactions. Noting this loss of control for users, efforts such as Berners-Lee led Solid are part of the growing trend toward decentralized data in the Web. However, current efforts do not go far enough because they ignore the larger issue of how to accommodate the autonomy of the users. Accommodating autonomy requires not only data decentralization, but also the representation and decentralization of decision making by users [1].

We advocate thinking of applications fundamentally in terms of decentralized multiagent systems (MAS), where each agent represents a user and embodies not only control over the user’s data but also their decision making. We advocate modeling a MAS in terms of social abstractions such as the norms and protocols that apply to interaction between users. The norms could pertain to business interactions, privacy, and in general, governance. Norms give grounding for accountability relationships between users, which are crucial to understanding reasons for norms violations by users. The big question is how to realize such a decentralized multiagent system using Web technologies such the HTTP and HATEAOS. Data semantics promotes interoperability, but we also need interaction semantics, which is where MAS research shines and provides a basis for logical decentralization.

References

1 Amit K. Chopra and Munindar P. Singh *From Social Machines to Social Protocols: Software Engineering Foundations for Sociotechnical Systems*. In Proceedings of the 25th International Conference on World Wide Web, pages 903–914, Montreal, 2016.

3.5 Principled Design of Web-based Multi-Agent Systems

Andrei Ciortea (Universität St. Gallen, CH)

There has been significant work on engineering Web-based multi-agent systems (MAS), but several points remain open. Several ideas from the involved research communities appear fundamental for deriving a principled way forward.

Environment as a First-class Abstraction

In MAS, multiple agents interact in a shared environment – and thus require a supporting infrastructure to discover other agents, to exchange messages, to coordinate, etc. This led to the view of the *environment as a first-class abstraction* in MAS [12] – and even as a *programming abstraction* [9] that allows agents to shape their environment such that it better fits their needs. The separation of concerns between agents and the environment abstraction is fundamental for engineering Web-based MAS. First, it simplifies integrating architectures for MAS with the Web Architecture. Second, it allows agents and components in their environment to be developed, deployed, and evolve independently from one another. Third, it ensures forward compatibility in long-lived Web-based MAS: the environment abstraction can reify other existing or future conceptual dimensions in MAS. Together with colleagues, we have been developing the idea of hypermedia-based environments as a first-class abstraction in Web-based MAS – and as a conceptual bridge between MAS and the Web [4, 2, 1, 3].

Asynchronous Interaction

It is widely accepted that synchronous interactions are insufficient for engineering MAS (e.g., see [10]). The Web was originally built on synchronous interactions between components – but since then, a plethora of methods and protocols have been developed to support asynchronous interaction on the Web. Such extensions address requirements beyond the classic Web Architecture and it is not always obvious how they fit within a REST-style Web. The reason is that REST, as a coherent closed set of architectural design decisions, was defined in the mid-90s to meet the needs of the Web at that time – and asynchronous interaction was not among them [5]. Several generations of researchers have continued to extend the insights of REST to meet requirements beyond the classic Web Architecture [6]. For example, one such extension is *Asynchronous REST* [7], which allows clients to observe resources. A similar method is implemented by CoAP [11].[6]

[6] The Constrained Application Protocol (CoAP) is a Web protocol for constrained devices and networks developed in the context of the Web of Things [11].

Uniform Knowledge-Level Abstraction

The Web defines a hypermedia-driven, uniformly-accessible, knowledge-level abstraction of the world.[7] If we consider the environment as a first-class abstraction in Web-based MAS, where the environment can also reify all the other conceptual dimensions in the MAS, then we have all the elements we need to provide agents with a knowledge-level abstraction of their system that they can also interact with [1]. What remains is to ensure that agents are able to both *perceive* and *act on* their hypermedia-based environment in a uniform way.

Situatedness and Embodiement

The dominant view in AI research is that intelligent behavior is closely related to the environment an agent occupies and is not disembodied [13]. This view emerged in the late '80s in close relationship with research on intelligent robots [8], which are naturally *situated* and *embodied* in a physical environment. On the Web, there is no implicit support for situatedness and embodiment: e.g., multiple agents can browse the same website without being aware of other agents. However, there are also no inherent limitations: if we consider the environment as a first-class abstraction in Web-based MAS, then the hypermedia-based environment can be designed to provide agents with various levels of support for situatedness and embodiment – it is merely a design choice.

References

1 Andrei Ciortea, Olivier Boissier, and Alessandro Ricci. 2019. Engineering World-Wide Multi-Agent Systems with Hypermedia. In Engineering Multi-Agent Systems, Danny Weyns, Viviana Mascardi, and Alessandro Ricci (Eds.). Springer International Publishing, Cham, 285–301.

2 Andrei Ciortea, Olivier Boissier, Antoine Zimmermann, and Adina Magda Florea. 2017. Give Agents Some REST: A Resource-Oriented Abstraction Layer for Internet-Scale Agent Environments. In Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems (São Paulo, Brazil) AAMAS '17. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 1502–1504.

3 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 (Montreal QC, Canada) AAMAS '19. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 1659–1663.

4 Andrei Ciortea, Antoine Zimmermann, Olivier Boissier, and Adina Magda Florea. 2016. Hypermedia-Driven Socio-Technical Networks for Goal-Driven Discovery in the Web of Things. In Proceedings of the Seventh International Workshop on the Web of Things (Stuttgart, Germany) WoT '16. Association for Computing Machinery, New York, NY, USA, 25–30.

5 Roy Thomas Fielding. 2000. Architectural styles and the design of network-based software architectures. Ph. D. Dissertation. University of California, Irvine.

6 Roy T. Fielding, Richard N. Taylor, Justin R. Erenkrantz, Michael M. Gorlick, Jim Whitehead, Rohit Khare, and Peyman Oreizy. 2017. Reflections on the REST Architectural Style and "Principled Design of the Modern Web Architecture" (Impact Paper Award).

[7] `https://www.w3.org/DesignIssues/Abstractions.html`

In Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering (Paderborn, Germany) ESEC/FSE 2017. Association for Computing Machinery, New York, NY, USA, 4–14.
7 R. Khare, and R.N. Taylor. 2004. Extending the Representational State Transfer (REST) architectural style for decentralized systems. In Proceedings of the 26th International Conference on Software Engineering. 428–437.
8 Pattie Maes. 1993. Modeling Adaptive Autonomous Agents. Artificial Life, 1(1_2), 135–162.
9 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), 158–192.
10 Onn M Shehory. 1998. Architectural properties of multi-agent systems.
11 Zach Shelby, Klaus Hartke, and Carsten Bormann. 2014. The Constrained Application Protocol (CoAP). RFC 7252 (Proposed Standard). `http://www.ietf.org/rfc/rfc7252.txt`.
12 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), 5–30.
13 Michael Wooldridge. 2000. In Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence (Second Edition), Gerhard Weiss (Ed.). MIT press.

3.6 Towards a Low Entry Barrier for Agents on the Web

Stephen Cranefield (University of Otago, NZ)

Introduction

The Web was explicitly designed to allow sharing of information in an open, extensible and scalable way. When considering how agents might enhance the Web, or vice versa, it is useful to revisit Hewitt's requirements for open systems [1]: **(1)** *continuous change and evolution*, **(2)** *arm's-length relationships and decentralized decision making*, **(3)** *accommodation of perpetual inconsistency among knowledge bases*, **(4)** the need for *negotiation among system components*, and **(5)** recognition of *the inadequacy of the closed-world assumption*. These challenges have long been tackled by MAS research, which raises the question of what new challenges must be addressed to make agents on the Web a viable technology.

Fielding [2, Ch. 4] presents and motivates the following requirements of the Web architecture: **(a)** a *a low entry barrier*, **(b)** *extensibility*, **(c)** a *distributed hypermedia* model (in which application control information [is] embedded within, or as a layer above, the presentation of information"), and **(d)** the ability to work at *internet scale*, implying a need for **(i)** *anarchic scalability* (essentially graceful handling of unanticipated situations) and **(ii)** *independent deployment* of architectural elements in a partial, iterative fashion". In the MAS context, the dynamic application control that emerges from the proactivity and autonomy of agents can be seen as a natural generalisation of the traditional Web's distributed hypermedia model.

I believe that requirements **b, c** (in its generalised form) and **d(ii)** have been well addressed in MAS research (outside the specific context of the Web), but requirements **a** (low entry barrier) and, to a lesser extent, **d(i)** have lacked significant attention by MAS researchers.

What does a low entry barrier look like for agents on the Web?

It is my observation that over time, much MAS research has shifted away from a peer-to-peer model of agent interaction, in which coordination mechanisms and social intelligence are located within the agents, towards a combination of agent-level and *system-level* design, where the desired properties of an MAS are supported by specific system architectures, services and protocols. While this is entirely appropriate for specific closed application areas (e.g. smart traffic control), I am sceptical that this approach will provide a low entry barrier to making agents a viable and accepted technology (in general) on the Web. Furthermore, we need to make it easier for non-specialist programmers to create and deploy agents on the Web, and limit the requirement to learn specialist agent programming languages and middleware. To this end, I make the following recommendations:

- Provide convenient interfaces between the agent model and conventional programming languages, information models and coordination tools (e.g. [3, 4]).
- Avoid creating MAS-specific middleware and instead use mainstream or at least domain-independent alternatives (e.g. [5, 6]).
- If really necessary, make any newly proposed "Web agent" standards as generic and simple as possible.
- Where possible, locate social intelligence within agents without reliance on "the system", but . . .
- provide individual agent reasoning components as application- and domain-independent libraries or services (e.g. [7]).

References

1 Carl Hewitt. The challenge of open systems. In Derek Partridge and Yorick Wilks, editors, *The Foundations of Artificial Intelligence*, pages 383–395. Cambridge University Press, 1990. doi: 10.1017/CBO9780511663116.038.

2 Roy Thomas Fielding. *Designing the Web Architecture: Problems and Insights.* PhD thesis, University of California, Irvine, 2000. URL `https://www.ics.uci.edu/~fielding/pubs/dissertation/web_arch_domain.htm`.

3 Stephen Cranefield and Martin K. Purvis. UML as an ontology modelling language. In *Proceedings of the IJCAI-99 Workshop on Intelligent Information Integration, Sixteenth International Joint Conference on Artificial Intelligence*, volume 23 of *CEUR Workshop Proceedings*. CEUR-WS.org, 1999. URL `http://ceur-ws.org/Vol-23/cranefield-ijcai99-iii.pdf`.

4 Stephen Cranefield and Surangika Ranathunga. Embedding agents in business processes using enterprise integration patterns. In Massimo Cossentino, Amal El Fallah Seghrouchni, and Michael Winikoff, editors, *Engineering Multi-Agent Systems*, volume 8245 of *Lecture Notes in Computer Science*, pages 97–116. Springer, 2013.

5 Stephen Cranefield. Reliable group communication and institutional action in a multi-agent trading scenario. In Frank Dignum, Rogier M. van Eijk, and Roberto A. Flores, editors, *Agent Communication II*, volume 3859 of *Lecture Notes in Computer Science*, pages 258–272. Springer, 2006.

6 Stephen Cranefield. Enabling BDI group plans with coordination middleware: semantics and implementation. *Autonomous Agents and Multi Agent Systems*, 35(2):45, 2021. doi: 10.1007/s10458-021-09525-7.

7 Surangika Ranathunga, Stephen Cranefield, and Martin K. Purvis. Integrating expectation monitoring into BDI agents. In Louise A. Dennis, Olivier Boissier, and Rafael H. Bordini, editors, *Programming Multi-Agent Systems – 9th International Workshop, ProMAS 2011*, volume 7217 of *Lecture Notes in Computer Science*, pages 74–91. Springer, 2011.

3.7 Society = Autonomy + Adaptation

Jérôme Euzenat (INRIA Grenoble Rhône-Alpes, Saint Ismier, FR & Univ. Grenoble-Alpes, FR)

Main reference Jérôme Euzenat: "A map without a legend", Semantic Web, Vol. 11(1), pp. 63–68, 2020.
URL http://dx.doi.org/10.3233/SW-190376

What makes a true lively society is the capability of their members to autonomously adapt to others. It is not a set of norms cast in iron, be they programming norms or 'legal norms'. It is a set of beings trying to behave with others.

This behaviour may lead to explicit norms that make explicit what should not have/need to be reinvented, but they may well remain implicit, hence continuously adapted.

We should design software agents so that they are able to elaborate what drives their (social but not only) behaviours. They should be allowed to try, to make mistakes, and to transmit what they know. This is the ground on which evolution may happen. This capacity is what should be built in agents in order for them to behave without breaking too many things.

The goal is not to reach a static equilibrium: in an open-ended agent space there are always opportunities to learn new things, meet new people and visit new places. Hence, rather than the state reached by agents, this is they ability to surf a dynamic disequilibrium that must be sought.

This statement is somewhat made for triggering reactions within the seminar. It reacts to the apparent loss of autonomy of agents. It also extend the one I did for the previous seminar.

3.8 Multi-Agent Systems on the Web as a Special Kind of Knowledge Graphs Ecosystems

Catherine Faron (Université Côte d'Azur – Sophia Antipolis, FR)

I am interested in the management of knowledge graphs ecosystems on the Web, i.e. the modelling and exploitation of the relationships between not only resources within knowledge graphs, but also the particular resources that are knowledge graphs themselves. By nature, a knowledge graph represents and organises within it descriptions of resources of all types: factual knowledge (from data), procedural knowledge (rules, business processes), domain knowledge (schemas, thesauri, ontologies). At the same time, these knowledge graphs are themselves resources, which have links between each other and need to be described for their management and exploitation. The same models and languages can be used for intra- and inter-graph knowledge management.

My assumption is that Web agents and their interactions within Multi-Agents Systems can be represented and managed as a special kind of knowledge graphs ecosystems, relying on the Solid Protocol specification.

References

1 Pierre Maillot, Olivier Corby, Catherine Faron, Fabien Gandon, Franck Michel. IndeGx: A Model and a Framework for Indexing RDF Knowledge Graphs with SPARQL-based Test Suits. Journal of Web Semantics, 2023, https://hal.inria.fr/hal-03946680

2 Pierre Maillot, Olivier Corby, Catherine Faron, Fabien Gandon, Franck Michel. KartoGraphI: Drawing a Map of Linked Data. ESWC 2022 – 19th European Semantic Web Conferences, May 2022, Hersonissos, Greece. Springer. https://hal.inria.fr/hal-03652865

3.9 Governing Communities of Autonomous Agents and People on the Web Using Social Norms

Nicoletta Fornara (University of Lugano, CH)

Social norms and policies are fundamental for governing communities of autonomous agents and people on the Web. Various research communities have developed various models designed to formalise different types of norms and policies and to offer various types of automatic reasoning services on these norms. The MAS community has been focused on proposing models for the formal specification of social norms and contracts able to express different types of normative concepts, e.g. obligations, prohibitions, and permissions and on the definition of frameworks for reasoning on norms. An important connection between the MAS and the Semantic Web communities is represented by those norm models that use Semantic Web technologies for defining some components of the norms and for defining their operational semantics, like the OWL-POLAR framework [4] and the T-Norm model [1, 2]. Another policy language based on semantic web technologies is ODRL 2.2 [8] which is a W3C Recommendation since 15 February 2018. It is a policy expression language that can be used to represent permitted, prohibited, and obliged actions over a certain asset. ODRL 2.2 has no formal semantics and I am currently cooperating with other members of the ODRL Community Group for proposing a Formal Semantics[9] for the language.

All these studies can be a starting point for the engineering and development of mechanisms for governing the interactions of autonomous agents and people on the Web.

It is probably unthinkable that we would reach an agreement on a common model for specifying norms and policies for all web applications, perhaps it is more reasonable to take the path of studying which norms can be formalised with which languages and how to translate norms from one formalisation to another. An initial study in this direction is [3].

It is also difficult to imagine that people not specifically trained could formalise the norms and policies governing the use of their data or of their interactions, so methods have to be devised that can assist such formalisation or vice versa that can describe in natural language the content of norms written in a formal language so that human beings can understand them.[10]

[8] https://www.w3.org/TR/odrl-model/
[9] https://w3c.github.io/odrl/formal-semantics/
[10] Funded by the SNSF (Swiss National Science Foundation) grant no. 200021_175759/1.

References

1 N. Fornara, S. Roshankish, and M. Colombetti. A framework for automatic monitoring of norms that regulate time constrained actions. In A. Theodorou, J. C. Nieves, and M. D. Vos, editors, *Coordination, Organizations, Institutions, Norms, and Ethics for Governance of Multi-Agent Systems XIV – International Workshop, COINE 2021, London, UK, May 3, 2021, Revised Selected Papers*, volume 13239 of *Lecture Notes in Computer Science*, pages 9–27. Springer, 2021.

2 N. Fornara and M. Sterpetti. An architecture for monitoring norms that combines OWL reasoning and forward chaining over rules. In E. M. Sanfilippo et al., editors, *Proceedings of the Joint Ontology Workshops 2021 Episode VII: The Bolzano Summer of Knowledge co-located with the 12th International Conference on Formal Ontology in Information Systems (FOIS 2021), and the 12th International Conference on Biomedical Ontologies (ICBO 2021), Bolzano, Italy, September 11-18, 2021*, volume 2969 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2021.

3 S. Roshankish and N. Fornara. A methodology for formalizing different types of norms. In D. Baumeister and J. Rothe, editors, *Multi-Agent Systems*, pages 348–363, Cham, 2022. Springer International Publishing.

4 M. Sensoy, T. J. Norman, W. W. Vasconcelos, and K. P. Sycara. OWL-POLAR: A framework for semantic policy representation and reasoning. *Journal of Web Semantics*, 12:148–160, 2012.

3.10 Finding the Critical MAS(S): Resources and Representations Needed for Weaving a Web that Hosts Linked Multi-Agent Systems

Fabien Gandon (INRIA – Sophia Antipolis, FR)

A small but viral first hMAS. Linked Multi-Agent Systems (MAS) is a vision where MAS are linked through their resources and representations on the Web to allow us to turn the Web into an architecture supporting a network of interoperable MAS. One of the hardest tasks for Tim Berners-Lee in the early 90s was to make people imagine a world with a fully deploy Web and that we have the same cold-start problem with the Agents on the Web [1]. I still believe this to be a strategic question in 2023. In Hypermedia Multi-Agent Systems (hMAS), agents operate on an homogeneous hypermedia fabric that interconnects heterogeneous resources. The Web is the distributed hypermedia that has become the primary software architecture for applications on the Internet but not for Multi-Agent Systems, at least not yet. We are getting closer but we are not there and, in my opinion, a key step is to identify at least one use case that, not only could demonstrate hMAS but, more importantly, that has the potential to reach a critical mass of usage, the tipping point of a network effect like it was the case in the past successes of the Web [2].

The right metadata soil for growing hMAS. A lot of effort has been invested in knowledge acquisition and knowledge publication. We participated to that effort, with methods to index (IndeGx) [4], visualize (Kartographi) and annotate (Metadatamatic) linked data(sets)[5]. But for a first viral hMAS to happen we need the right breeding ground, we need to target metadata that have an impact on targeted adopters, largest users' community, etc. Moreover, with native mechanisms such as conneg (content negotiation over HTTP) the Web can do much more and support adaptative knowledge exchange of customized representations for human and software agents. The Web has the potential to support profile-based knowledge negotiation for AI methods to obtain or contribute the type of knowledge they can process. Such an open negotiation could also be a key enabler for forward compatibility in open hMAS.

The position of hMAS in the effort of (re)decentralization. Even more than AI, it is distributed AI that has a rendezvous with the Web [3]. By nature, the distributed artificial intelligence paradigm of MAS can participate to the (re-)decentralization of applications and their architectures in general, and on the Web in particular with hMAS. This requires positioning hMAS w.r.t. other initiatives like SOLID. Decentralization in general, has to consider many fronts (architecture, applications, data, schemata) at the same time and hMAS, in particular, will have to consider all of them too.

Extend and make easier the existing solutions. hMAS must be conceived as an extension of the Web and not as another Web or as a Web apart. The futures of the Web must be suited both for "software- and human-agents" [6] and it may reopen the discussion on the relation between humans and agents in MAS both conceptually and practically. This may also echo discussions on other current trends in computer science such as the topic of digital twins (for artefacts, for users, etc.). Finally I would like to conclude on a "meta-risk": In the history of computer science, we have many examples of a language, an architecture or a tool being reinvented because the previous ones had become too complex for newcomers, for simple use cases, etc. In what we will propose we must thrive to avoid the Déjà Vu of a technology stack that becomes too complex with an adoption cost too high and making the bed for the temptation to create something initially seemingly simpler on the side but that will end up being the first layer of a new technology stack continuing the cycle of reinventing languages, architectures, formats, etc. again and again.

References

1 Fabien Gandon. Merry hMAS and Happy New Web: A Wish for Standardizing an AI-Friendly Web Architecture for Hypermedia Multi-Agent Systems. Dagstuhl-Seminar 21072 : Autonomous Agents on the Web, Dagstuhl Schloss, Feb 2021, Dagstuhl, Germany. pp.3. https://hal.inria.fr/hal-03960471

2 Fabien Gandon, Wendy Hall. A Never-Ending Project for Humanity Called "the Web". WWW 2022 – ACM Web Conference, Apr 2022, Lyon, France. https://hal.inria.fr/hal-03633526

3 Fabien Gandon. Distributed Artificial Intelligence and Knowledge Management: Ontologies And Multi-Agent Systems For A Corporate Semantic Web. PhD Thesis, INRIA, Université Nice Sophia Antipolis, November 2002. https://tel.archives-ouvertes.fr/tel-00378201

4 Pierre Maillot, Olivier Corby, Catherine Faron, Fabien Gandon, Franck Michel. IndeGx: A Model and a Framework for Indexing RDF Knowledge Graphs with SPARQL-based Test Suits. Journal of Web Semantics, 2023, https://hal.inria.fr/hal-03946680

5 Pierre Maillot, Olivier Corby, Catherine Faron, Fabien Gandon, Franck Michel. KartoGraphI: Drawing a Map of Linked Data. ESWC 2022 – 19th European Semantic Web Conferences, May 2022, Hersonissos, Greece. Springer. https://hal.inria.fr/hal-03652865

6 Bettina Berendt, Fabien Gandon, Susan Halford, Wendy Hall, Jim Hendler, Katharina E Kinder-Kurlanda, Eirini Ntoutsi, Steffen Staab. Web Futures: Inclusive, Intelligent, Sustainable The 2020 Manifesto for Web Science. Dagstuhl Manifestos, 2021. https://hal.inria.fr/hal-03189474

3.11 About the Place of Agents in the Web

Jomi Fred Hübner (Federal University of Santa Catarina, BR)

In an integration perspective, we can face two problems when considering agents and the web:

1. how agents can use web resources and
2. how web applications can use agents.

While the former problem is addressed by some proposals, the latter case is mostly ignored. Of course, these problems can not be addressed considering the web simply as a kind of transport layer. A proper integration requires a better conceptualization. These problems are related to the third research question of the seminar: "How to design, represent, and reason about inter-actions among autonomous agents, people, and any other resources on the Web?"

An important aspect regarding the second problem is how to present agents to the web keeping their main property: *autonomy*. Moreover, it would be interesting to keep agents as cognitive entities and interact with them such as. This problem brings thus some initial questions:

- Is an agent web interface (like REST API) enough?
- Is it possible to include an agent in the web presenting it by means of a Thing Description?
- What kind of 'resource' is an autonomous agent?
- Can we conceive and develop tools (on the agent side) that can help this integration?

Ideally, we can imagine a scenario where an ordinary web developer is able to reuse available agents on the web while developing an application. We may require that he/she knows what an agent is. However, we should not require that he/she transform his/her application into an agent.

My preference to approach this problem is from an engineering perspective and based on application scenarios.

My assumption is that agents will be useful for the web only if they can be easily used by the web. Moreover, I would avoid a reductionist approach (e.g., transforming agents into services).

3.12 Towards an Analysis-oriented Perspective on Agent-oriented Abstractions for the Web

Timotheus Kampik (Umeå University, SE & SAP Signvavio – Berlin, DE)

When studying the engineering of multi-agent systems, the notion of an agent is typically seen as a *design* abstraction, *i.e.*, as a tool that directly facilitates implementation. However, one may argue that in the context of complex, large-scale socio-technical systems such as the Web, the suitability of an abstraction as a facilitator of run-time *analysis* is just as or even more important than design-time advantages: after all, it is only during run-time that a software artifact is fully exposed to the technical and social complexities of the larger system, often in unanticipated ways.

Generally, this idea is not new and has, indeed, informed fundamental research on the foundations of knowledge representation and reasoning that multi-agent systems rest upon. For example, in a seminal 1985 paper, the principle of cautious monotonicity in expert systems was motivated by the need to determine whether a system whose domain-specific purpose is unknown or has been obscured still behaves "logically" (in the colloquial sense) [4]. From an application-oriented perspective, recent research results have provided first evidence that the notion of an agent can potentially aid the discovery of symbolic models from system logs, and in particular facilitate the mining of Petri nets that represent behavioral models of agents' actions and interactions [6, 7], following the established research direction of *agent mining* [3].

However, the design abstractions that the academic multi-agent systems community advocates for appear to be primarily based on philosophical concepts of reasoning, such as the well-known belief-desire-intention model [1, 2], and not on real-world requirements for facilitating run-time interpretability of behaviors in socio-technical systems. Consequently, we consider the following research question as particularly promising in the context of "agents on the Web" research:

> How can agent-oriented abstractions facilitate the run-time analysis of complex socio-technical systems on the Web?

For example, the notions of *beliefs* and *belief revision* [5] can potentially be used to explain misaligned behaviors of several agents (caused by inconsistent beliefs), as well as apparently inconsistent behavior by one particular agent (caused by belief change) over time. Yet, beliefs are rarely used as abstractions in software engineering. Although much has been written about the mathematical foundations of belief revision, no comprehensive body of works that puts beliefs into the context of "mainstream", large-scale software engineering seems to exist. This apparent gap in the literature may warrant further investigation.

References

1 Sameera Abar, Georgios K. Theodoropoulos, Pierre Lemarinier, and Gregory M.P. O'Hare. Agent based modelling and simulation tools: A review of the state-of-art software. *Computer Science Review*, 24:13–33, 2017.

2 Michael Bratman. *Intention, Plans, and Practical Reason.* Cambridge: Cambridge, MA: Harvard University Press, 1987.

3 Longbing Cao, Gerhard Weiss, and Philip Yu. A brief introduction to agent mining. *Autonomous Agents and Multi-Agent Systems*, 25(3):419–424, 2012.

4 D. M. Gabbay. Theoretical foundations for non-monotonic reasoning in expert systems. In Krzysztof R. Apt, editor, *Logics and Models of Concurrent Systems*, pages 439–457, Berlin, Heidelberg, 1985. Springer Berlin Heidelberg.

5 Peter Gärdenfors and Hans Rott. Belief revision. In *Handbook of logic in artificial intelligence and logic programming (Vol. 4) epistemic and temporal reasoning*, pages 35–132. Oxford University Press, 1995.

6 Andrei Tour, Artem Polyvyanyy, and Anna Kalenkova. Agent system mining: Vision, benefits, and challenges. *IEEE Access*, 9:99480–99494, 2021.

7 Andrei Tour, Artem Polyvyanyy, Anna Kalenkova, and Arik Senderovich. Agent miner: An algorithm for discovering agent systems from event data (pre-print), 2022.

3.13 Agent-based Information Systems Built on Read-Write Linked Data

Tobias Käfer (KIT – Karlsruher Institut für Technologie, DE)

The web architecture has scaled to a global information exchange infrastructure, but in contrast to other architectures, it comes with peculiar constraints, summarised in the REST architectural style [8]. Augmented with knowledge representation using semantic technologies, thus forming Read-Write Linked Data [1], the web presents us with a substrate for integration on the component interaction and data level, i. e. a substrate for interoperability. As such, Read-Write Linked Data is built on standards that are simple yet powerful, such as HTTP and RDF [7, 5]. The underpinnings, their simplicity and formal properties allow for layering approaches for behaviour on top, such as [10, 11].

As the scalability and the adoption of the web has been attributed to its basic standards [8], I would argue that if we want to bring agents to the web, the community should only cautiously extend the underpinnings by inventing extensions and alternatives to the standards for data transmission (HTTP) and data description (RDF), and rather try to embrace the power of what exists and built agent-based systems on top:

Alternatives to HTTP data transmission include SPARQL over HTTP [6], SPARQL with web preemption [12], Linked Data Fragments [14], RDF streaming [13]; RDF is currently being extended to RDF-star [9]. Instead, Linked Data Notifications [3] work in a RESTful manner avoiding the need for streaming for notifications about updates, and the proposal of the data interoperability panel [2] allows for describing subdivisions of data that are accessible in a RESTful manner, instead of querying for triple fragments.

With a cleanly defined substrate, and I prefer Read-Write Linked Data, let us go and connect techniques from agent systems and multi-agent systems intelligently with the web architecture [4].

References

1 T. Berners-Lee. *Read-Write Linked Data*. Design Issues. 2009. URL: `http://www.w3.org/DesignIssues/ReadWriteLinkedData.html`.

2 J. Bingham, E. Prud'hommeaux and elf Pavlik. *Solid Application Interoperability*. Editor's Draft. 2023. URL: `https://solid.github.io/data-interoperability-panel/specification/`.

3 S. Capadisli and A. Guy. *–Linked Data Notifications*. Recommendation. W3C. 2017. URL: `https://www.w3.org/TR/ldn/`.

4 A. Ciortea, S. Mayer, F. L. Gandon, O. Boissier, A. Ricci and A. Zimmermann. "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 Multi-Agent Systems (AAMAS)*. 2019.

5 R. Cyganiak, D. Wood and M. Lanthaler. *–RDF 1.1 Concepts and Abstract Syntax"*. Recommendation. W3C. 2014. URL: `https://www.w3.org/TR/rdf11-concepts/`.

6 L. Feigenbaum, G. Williams, K. Clark and E. Torres. *SPARQL 1.1 Protocol*. Recommendation. W3C. 2013. URL: `http://www.w3.org/TR/sparql11-protocol/`.

7 R. Fielding and J. Reschke. *–Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing"*. RFC 7230 (Proposed Standard). Internet Engineering Task Force, 2014. URL: `http://www.ietf.org/rfc/rfc7230.txt`.

8 R. Fielding. "Architectural Styles and the Design of Network-based Software Architectures". PhD thesis. University of California, Irvine (CA), USA, 2000.
9 O. Hartig, P.-A. Champin, G. Kellogg and A. Seaborne. *RDF-star and SPARQL-star*. Community Group Report. W3C. 2021. URL: `https://w3c.github.io/rdf-star/cg-spec/2021-12-17.html`.
10 T. Käfer and A. Harth. "Specifying, Monitoring, and Executing Workflows in Linked Data Environments". In: *Proceedings of the 17th International Semantic Web Conference (ISWC)*. 2018.
11 T. Käfer, B. Jochum, N. Aßfalg and L. Nürnberg. "Specifying and Executing User Agents in an Environment of Reasoning and RESTful Systems using the Guard-Stage-Milestone Approach". In: *Journal on Data Semantics* 10.1-2 (2021).
12 T. Minier, H. Skaf-Molli and P. Molli. "SaGe: Web Preemption for Public –SPARQL˝ Query Services". In: *Proceedings of the 28th World Wide Web Conference (WWW)*. ACM, 2019.
13 E. D. Valle, S. Ceri, F. van Harmelen and D. Fensel. "It's a Streaming World! Reasoning upon Rapidly Changing Information". In: *–IEEE˝ Intelligent Systems* 24.6 (2009).
14 R. Verborgh, M. V. Sande, P. Colpaert, S. Coppens, E. Mannens and R. V. de Walle. "Web-Scale Querying through Linked Data Fragments". In: *Proceedings of the 7th International Workshop on Linked Data on the Web at the 23rd International World Wide Web Conference (WWW)*. 2014.

3.14 Old Agents, New Environments: Open Issues and Unexpected Opportunities

Stefano Mariani (University of Modena, IT)

Main reference Alessandro Ricci, Angelo Croatti, Stefano Mariani, Sara Montagna, Marco Picone: "Web of Digital Twins", ACM Trans. Internet Techn., Vol. 22(4), pp. 101:1–101:30, 2022.
URL https://doi.org//10.1145/3507909

Hypermedia provides for a new infrastructure and paradigm to design agents' *virtual* environments, as well as (mediated) interactions. The Web of Things pushes this further to *physical* environments, through actuators encapsulated as hypermedia resources. Should the web bear responsibility for enabling, mediating, and controlling access to them? Probably not. *Digital Twins* could, in turn providing services as hypermedia resources. But can they cope with the web *openness* and *dynamism*? Can agents, while preserving *autonomy*? Probably yes, provided they can *learn*.

Acting through hypermedia

Quoting from the seminar "manifesto":

> "The latest standards allow clients not only to browse and query, but also to observe and act on this hypermedia fabric."

In a Web of Things perspective what does it even mean for agents to *act* in a hypermedia environment? What can agents *expect* as action feedback and outcomes? Are there *implications* for physical resources? What are the *commitments* agents should hold to when providing hypermedia-accessible resources and services? What is the impact on agents' architecture, if any?

Openness + dynamism = autonomy + _?

Given that:

> "[...] the dynamic and open nature of these systems requires components to be deployed and to evolve independently from one another."
> "Autonomy is [...] agent's ability to operate on its own, without the need of direct intervention of other people or other agents."
> "[...] hard-coding rules into agents [...] would be impractical in an environment as open and complex as the Web."

what is the missing factor in the heading equation, if any?

Digital Twins, Agents' other half

Digital Twins have already been framed within MAS as a novel abstraction to engineer virtual environments deeply bonded with the physical one. Hypermedia can fit in the picture by standardasing the means by which such environment is distributed, accessed to, and operated on.

Learning to the rescue

Openness + dynamism = autonomy + *learning*. Do we agree that learning is the best way to deal with open and dynamic environments in *autonomous* way? If so, has hypermedia any value to add to agents' learning experience? Hypermedia can be the *shared learning environment* where agents learn (i) the structure of the world, (ii) how do they capabilities match the environment affordances, (iii) how to achieve their own goals given so, (iv) whether other agents can help or interfere. Most importantly, agents can make the learnt knowledge, behaviours, and decision making policies available to others as hypermedia resources and services, thus communicating via the very same environment they live within, in a stigmergic way.

Conclusion

I hope that the role of mediation between agents, an hypermedia environment, and the physical world can be discussed and clarified, especially with respect to the Digital Twin abstraction. I also hope that learning, as the cornerstone form of adaptation, is not left out of the picture, as it would be a missed opportunity.

3.15 Pervasive Autonomous Systems: So Much to Learn from One Another

Simon Mayer (Universität St. Gallen, CH)

I would like to share a particularly stimulating finding that could act as an example and also as a motivator in our attempt to find a common language to identify, discuss, and solve the issues on our path to Internet-scalable communities of autonomous agents and people who work together to seamlessly allocate concerns and reach their objectives. To support more efficient agent-environment interaction, we recently proposed that the environment might contain signification that is *personalized* at run time to each agent that roams it [9, 5]. This was inspired by affordance theory [3] and computer scientists, philosophers, and psychologists who built on top of it (e.g., [7, 6, 1], and the basic idea is visible all around us: in (well-designed) signs, furniture, and, famously, door handles [6]. When designed well, signifiers become environmental cues that can be *intuitively and reliably* discovered and interpreted to provide guidance to agents who roam an environment about *what* are the possible behaviors and *how* these behaviors can be performed [9]. However, importantly, while in classical ecology signifiers are assumed to be run-time static and can only be designed a priori and with respect to agent *stereotypes* rather than actual agent features that are measured at run time, we may drop these constraints in virtual environments and, more and more, in *physical environments as well.* For this reason – and driven by advancements from material science to human-computer interaction – I expect that the findings of our Dagstuhl community for supporting autonomous agents have a high potential to be re-applied to human-computer interaction and, using HCI as a vehicle, to classical ecology. We expect that already in the near future, personalized content will be delivered *to humans* not only through Web browsers and other digital mediation (e.g., mobile apps) but through technologies that mediate individual or group experiences of *physical* reality as well. This delivery may happen through projected or head-worn Mixed Reality, but also through other sensory modalities such as audio [12], haptics [4], or vestibular stimulation [8]; objects in our physical environments with communication and processing abilities [10] might also alter experienced realities directly – e.g., in the context of self-balancing bicycles [11] where it has been shown that users prefer to experience artificially decreased tilting when turning – agent-personalized physics?

Regarding the cognitive dimension, some of these concerns are certainly best left to statistical approaches, in particular if abundant training data is available and can be processed relatively efficiently; others are to be allocated to symbolic systems, not only if more easily accessible explanations of the system's behavior are desired; we should further support neurosymbolic combinations – to this end, I find the current developments in the field of semantic scene understanding (cf. [2]) particularly well-accessible and transferable; and finally, possibly, there will be tasks that humans are well-suited to solve, and that we also enjoy solving. Let us together work on an architecture that will permit the integration of such heterogeneous systems, including means for environments to support autonomous behavior in agent-agent and agent-environment interaction, means to design and govern communities of autonomous entities towards achieving organizational goals, and means to foster the adoption potential of our approaches into more real use cases.

Acknowledgments

Leading to this and other insights, I have very much enjoyed the discussions with members of my research team and beyond, where I'd like to name specifically Jérémy Lemée, Jannis Strecker, Danai Vachtsevanou, Ganesh Ramanathan, Samuele Burattini, Kenan Bektaş, Kimberly Garcia, and Andrei Ciortea.

References

1 A. Chemero and M. Turvey. Complexity, Hypersets, and the Ecological Perspective on Perception-Action. *Biological Theory*, 2(1):23–36, 2007.

2 Y. Cong, M. Y. Yang, and B. Rosenhahn. RelTR: Relation Transformer for Scene Graph Generation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–16, 2023.

3 J. J. Gibson. *The Ecological Approach to Visual Perception: Classic Edition.* Psychology press, 2014.
4 T. Grosshauser and T. Hermann. Augmented Haptics – An Interactive Feedback System for Musicians. In *Proceedings of the 4th International Conference on Haptic and Audio Interaction Design*, HAID '09, pages 100–108, Berlin, Heidelberg, Sept. 2009. Springer-Verlag.
5 J. Lemée, D. Vachtsevanou, S. Mayer, and A. Ciortea. Signifiers for Affordance-driven Multi-Agent Systems, 2022. Paper presented at the International Workshop on Engineering Multi-Agent Systems (EMAS) at the 21st International Conference on Autonomous Agents and Multiagent Systems (AAMAS).
6 D. Norman. *The Design of Everyday Things: Revised and Expanded Edition.* Basic Books, 2013.
7 D. A. Norman. The Way I See It: Signifiers, Not Affordances. *Interactions*, 2008.
8 M. Sra, A. Jain, and P. Maes. Adding Proprioceptive Feedback to Virtual Reality Experiences Using Galvanic Vestibular Stimulation. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, pages 1–14, New York, NY, USA, May 2019. ACM.
9 D. Vachtsevanou, A. Ciortea, S. Mayer, and J. Lemée. Signifiers as a First-class Abstraction in Hypermedia Multi-Agent Systems. In *Proceedings of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2023.
10 M. Weiser. The Computer for the 21st Century. *Scientific American*, 1991.
11 P. Wintersberger, A. Shahu, J. Reisinger, F. Alizadeh, and F. Michahelles. Self-Balancing Bicycles: Qualitative Assessment and Gaze Behavior Evaluation. In *Proceedings of the 21st International Conference on Mobile and Ubiquitous Multimedia*, MUM '22, page 189–199, New York, NY, USA, 2022. Association for Computing Machinery.
12 J. Yang, A. Barde, and M. Billinghurst. Audio Augmented Reality: A Systematic Review of Technologies, Applications, and Future Research Directions. *Journal of the Audio Engineering Society*, 70(10):788–809, Nov. 2022.

3.16 Federated Learning in Goal-oriented WoT Environments

Mahda Noura (TU Chemnitz, DE)

Joint work of Mahda Noura, Martin Gaedke

Goal-oriented user interaction is a promising new interaction paradigm for collaborative environments of heterogeneous distributed human, software, and hardware agents in the context of IoT. Unlike traditional programming-based approaches, which necessitates knowing and identifying available and potentially transient agents as well as explicitly giving them commands, goal-oriented approaches automatically identify a solution to fulfill a human user's goal [2, 4]. The goal can be expressed in a multimodal fashion through channels such as voice, visual programming, gestures or haptic interaction [5].

There are several categories of approaches used for determining an appropriate sequence of actions for the available agents to fulfill the user's goals such as *semantic reasoning*, symbolic AI *automated planners*, or *reinforcement learning*. Planners perform well when they have a complete understanding of how the world evolves as a result of various actions. However, the deployment of planning for dynamic WoT environments is algorithmically complex and can result in unacceptable response delays, especially when executed on energy-efficient edge devices [3]. The Reinforcement Learning paradigm, on the other hand, does not require prior knowledge and instead involves exploration of the state-action space [1]. In WoT application scenarios actions of physical devices have an effect on the physical environments, therefore applying reinforcement learning is often impossible as this would result in unexplainable system behaviors via arbitrary device activations.

A novel idea would be to consider exploration not only in isolated multi-agent environments, but also in large-scale distributed multi-agent environments, allowing for exploration to be distributed and knowledge to be shared across them. This would reduce the negative impact on individual environments while also allowing for the optimization of complex decision making via large-scale distributed agent collaboration. Furthermore, due to the decentralized and distributed nature of the learning process, it would help to preserve privacy and data security, especially important in applications involving sensitive data.

Exploration from environments that are different could cause the inferred decisions to be misjudged. However, we argue that for environments like smart homes which are characterized by many different homes providing a large amount of learning data, a limited number of device types, and a limited number of device instances. Due to these characteristics, the chances of goals overlapping are higher, enabling autonomous learning with heuristics. Some research challenges that need to be considered are data aggregation on edge devices, variation of the environments, semantic interoperability, noisy and incomplete data, and data anonymization.

An idea worth discussion in the seminar would be identifying dimensions of the suitability of the different methods across three targeted application areas.

References

1 Jiménez, S., De La Rosa, T., Fernández, S., Fernández, F., Borrajo, D. A review of machine learning for automated planning. The Knowledge Engineering Review 27(4), 433–467 (2012).

2 Mayer, S., Verborgh, R., Kovatsch, M., Mattern, F. Smart configuration of smart environments. IEEE Transactions on Automation Science and Engineering 13(3), 1247–1255 (2016).

3 Noura, M., Gaedke, M. An automated cyclic planning framework based on plan-do-check-act for web of things composition. In: Proceedings of the 10th ACM Conference on Web Science. pp. 205–214 (2019).

4 Noura, M., Heil, S., Gaedke, M. Growth: Goal-oriented end user development for web of things devices. In: International Conference on Web Engineering. pp. 358–365. Springer (2018).

5 Noura, M., Heil, S., Gaedke, M. Natural language goal understanding for smart home environments. In: Proceedings of the 10th International Conference on the Internet of Things. pp. 1–8 (2020).

3.17 Integrating Multi-Agent Systems within Web-based Microservices Architectures – Multi-Agent MicroServices (MAMS)

Eoin O'Neill (University College Dublin, IE)

In [2], Hubner discussed the need for not only agents interacting with services, but also the ability for services to be able to interact with agents. This is the vision of the Multi-Agent MicroServices [4] architecture. Although interaction between agents and Web services is not a new concept, with [3] being an example of such a system, the key distinction is that the agents in those systems are external to their environment and are not directly accessible entities. Our goal with the MAMS architecture was to introduce the idea of immersing agents within the environment of the Web and allowing them to expose individual aspects of their agent "body" as directly queryable *virtual resources*. By integrating Multi-Agent Systems (MAS) with Microservices (MS) the MS community benefits from the plethora of research that has been done on fully encapsulated entities interacting and how to handle the interactions between individual, fully-encompassed entities, while the MAS community benefits by enabling agents to utilise the ecology of tools built to support the MS community.

One of the key inspirations for this vision is the HATEOAS principle set forth by Roy Fielding in his description of the REST architectural paradigm [1]. In order to adhere to this principle, each *virtual resource* that the agent exposes, must have a hypermedia representation with actionable links available in order to facilitate state transitions of the resource. By providing a hypermedia body for each resource of the agent, we are embodying the agent within the Web environment, making it identifiable to the other entities that inhabit it.

Each *virtual resource*, an example of which could be to display a history of interactions with other entities as a log in order to to represent a public persona to other entities in the environment, or an inbox resource that entities can use to communicate with the agent. In order to immerse agents within a Web-based environment, a hypermedia representation is necessary. The current industry standard in microservices architectures for providing a hypermedia representation of the possible actions for a microservice is the OpenAPI specification. However, by using Linked Data representations, such as the Hydra specification [5], service descriptions can provide a level of detail and ontological context that you cannot achieve by strictly using data representations such as XML or JSON. The Hydra specification allows users to define the inputs and outputs of a resource, the HTTP operations that are accepted in its current state.

The importance of this level of detail is apparent when you start to envision components built as microservices, agent enabled or not, with a description of what it can provide that is consumable by other entities within the Web environment. This can allow interactions with Web-baed microservices to occur in an ad-hoc fashion, with all the necessary information for interaction being available to construct requests being directly available in the hypermedia. Additionally, by utilising Linked Data compliant hypermedia representations, the MAS community can make use of the abundance of tools developed for the Semantic Web community to store and manipulate Linked Data. By enabling autonomous agents with the ability to explore resources, reason about their capabilities, see how they line up with their goals and then utilise them if necessary all through the use of hypermedia, we are allowing machines to utilise the Web in the same manner as humans. This is a step towards the serendipitous use of the Web from the perspective of machines.

References

1 Fielding, R.T.: Architectural styles and the design of network-based software architectures. University of California, Irvine (2000)

2 Hubner, J.: About the place of agents in the web. vol. Web p. 44

3 Minotti, M., Santi, A., Ricci, A.: Developing web client applications with jaca-web. In: WOA (2010)

4 W. Collier, R., O'Neill, E., Lillis, D., O'Hare, G.: Mams: Multi-agent microservices. In: Companion Proceedings of The 2019 World Wide Web Conference. pp. 655–662 (2019)

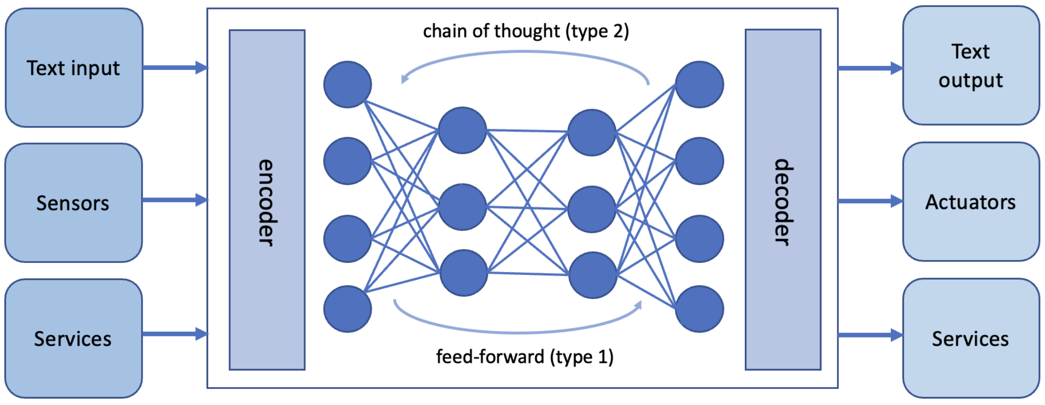


Figure 2 architecture for neurosymbolic cognitive agents.

3.18 Human-like AI for Artificial General Intelligence

Dave Raggett (W3C – United Kingdom, GB)

According to Arthur C. Clarke, any sufficiently advanced technology is indistinguishable from magic, and this increasingly applies to the remarkable advances in the field of artificial intelligence, in particular, to large language models and image generators. There are lots of opportunities for the Intelligent Web of Things as agents that can communicate, learn and reason like we do, supporting human-machine collaboration to boost productivity and standards of living.

Symbolic AI is falling behind relative to neural networks, but both have strengths and weaknesses. Symbolic AI is hard to scale up due to a dependency on expensive hand-crafted knowledge. Neural networks, by contrast, can be easily scaled using larger datasets. Neurosymbolic AI combines the strengths of both approaches, yielding better explainability and enabling semantic interoperability for inter-system communication. There are plenty of opportunities for scaling across the computing continuum from the network edge to the cloud.

Stanovich's tripartite model of mind distinguishes the autonomous mind, the algorithmic mind and the reflective mind. The first is associated with type 1 cognition, which is fast, automatic and opaque. The second and third use type 2 cognition, which is slow, deliberative and open to introspection. This is formed by linking together type 1 processes using working memory to form a chain of thought.

Neurosymbolic systems combine neural networks with symbolic approaches, including back-end information technology systems, sensors and actuators. A natural next step would be to integrate neural networks with cognitive databases that combine symbolic and sub-symbolic approaches.

The Plausible Knowledge Notation (PKN) is an example of how to provide richer semantics compared to W3C's Resource Description Framework (RDF). PKN supports properties, relations and implications, combined with qualitative metadata and scopes; imprecise concepts and quantifiers, along with analogical reasoning.

Plausible reasoning is a form of argumentation for everyday knowledge, i.e. knowledge that is often uncertain, imprecise, incomplete and inconsistent. In place of logical proof, we have multiple lines of argument for and against the premise in question, just as in the courtroom.

The large language model ChatGPT is remarkably good at plausible reasoning, but can make human-like errors due to careless assumptions. Reinforcement learning with human feedback is proving effective in training large language model for specific tasks, e.g. to solve a range of math and physics problems.

Some outstanding research challenges include:

- Faster and smarter learning by mimicking human learning using a combination of type 1 and type 2 cognition.
- Taming catastrophic interference when learning new tasks.
- Earning trust through citing provenance and avoiding careless mistakes.
- Integrating episodic memory and continual learning through instruction, observation and experience.
- Reflective cognition and support for theory of mind for self and others.
- Acquisition and use of implicit, and explicit, behavioural norms.
- Shared cognitive databases for enabling hive minds where knowledge gained by one agent is immediately available to all other agents in the hive.
- Agent-Client confidentiality and policy-based data sharing.

Researchers should be encouraged by Jeremy Howard and Sylvain Gugger (FastAI), who say that breakthrough work in deep learning absolutely does not require access to vast resources, elite teams or advanced math!

Human-like AI will be hugely disruptive to web search, improved personal privacy and ecosystems of services. This includes personal agents that safeguard your privacy, help you with your health, financial affairs, education and so forth; policing fake news and malicious posts; agents communicating with other agents to find and provide services in open decentralised ecosystems; agents on the Web, in the Metaverse, as robots and embedded in other devices, including cars; in short, the intelligent Web of Things!

For more details, see: https://www.w3.org/2023/02-Raggett-towards-AGI.pdf/

3.19 Ontologies with Temporal Logics Allow Hypermedia Agents to Make Plans

Daniel Schraudner (Universität Erlangen-Nürnberg, DE)

Hypermedia agents are agents that are situated in a hypermedia environment, like e.g. the Web. These agents can interact with artifacts using HTTP methods on resources that are managed by Web servers and interconnected via hypermedia.

Possible interactions for agents are mediated by affordances which in the Web usually are links and forms. I.e. hypermedia agents can follow links to get information about artifacts and can submit forms in order to change the current state of an artifact.

In hypermedia graphs like the web the number of affordances for an agents usually gets quite large very fast. Agents can find out which links to follow in order to get certain information by utilizing ontologies. An ontology could e.g. have the following two axioms:

$$\text{Coffee} \sqsubseteq \text{Beverage} \sqcap \exists.\text{madeOutOf}.\text{CoffeeBean}$$

$$\text{CoffeeBean} \sqsubseteq \exists.\text{hasSort}.\{\text{arabica}, \text{robusta}\}$$

This allows an agent that knows the resource of a coffee to infer that it has to follow the links of types madeOutOf and hasSort in order to find out the bean sort of the coffee.

Manipulating the state of artifacts, however, is not that straight-forward. A coffee Web service e.g., will need an agent to place a valid order by changing the state of a resource through a form. Afterwards the artifacts will carry out actions that again change their state, e.g. by creating a finished coffee resource (reactivity of artifacts).

Getting a finished coffee resource is usually the goal of an agent placing a coffee order – however, hypermedia agents have no possibility to infer that a valid coffee order will eventually lead to a finished coffee resource by using an ontology.

To solve this problem, we suggest using Temporal Description Logics (ontologies enriched with Temporal Logics) in order to give agent information about how to manipulate the state of artifacts in order to get the wanted reaction of the artifacts. Temporal Description Logics offer the possibility to define temporal concepts, like e.g.:

$$\Diamond \text{Coffee} \equiv \text{OrderedCoffee} \sqcap \exists \text{hasPayment.ValidPayment}$$

This axiom defines that a coffee order that has a valid payment will eventually lead to a finished Coffee resource (as per reaction of the artifact).

An agent that has the goal to get a coffee can now utilize the ontology to infer how it has to manipulate its environment using forms in order to reach its goal.

Having a method to reason about the consequences of their actions will allow hypermedia agents to combine multiple actions to make sophisticated plans in order to achieve complex goals (actually getting a coffee already might require a two-step plan of an agent as it might need to create a OrderedCoffee and a ValidPayment resource by using different forms).

3.20 Rethinking Agents and Meaning over the Web of Things

Munindar P. Singh (North Carolina State University – Raleigh, US)

This position paper takes a fresh look at how we ought to understand autonomy and meaning on the web in relation to the needs of flexibility and governance and with respect to system architecture.

Positions

I take two positions that are well-aligned with the objectives of this Dagstuhl Seminar but nevertheless are departures from what the communities represented in this seminar have primarily pursued.

First, we ought to find new ways to unite the distinct concerns of autonomy, interaction, meaning, and sensing to produce well-governed systems of agents on the web. This position is that our conceptions of these elements must be rethought (and refactored and reconstructed) with awareness of the kinds of usage scenarios that we wish to support.

Second, we should dispense with the traditional conception of a layered system architecture. Black-box layering is dead, or ought to be.

Concepts

Many of the common approaches to multiagent systems (MAS) are conventional in their outlook: they adopt traditional programming abstractions for the most part albeit with a dose of AI, such as through logic programming (with facts interpreted as knowledge and inference rules as plans). This MAS-lite attitude is intended to facilitate adoption of MAS technologies by practitioners versed in traditional methods. The attitude has produced mixed success at best. We would be better served by emphasizing rather than hiding our strengths.

Autonomy

The common realization of autonomy in MAS allows interference from organizational constraints and from the infrastructure. Agents can be blocked from violating norms by the so-called governors in organizations. The infrastructure is set up to control the order in which an agent observes events even when they is no physical reason for such ordering (e.g., because the events have occurred and information about them has already arrived). The net result of focusing on and adopting traditional software assumptions is that autonomy, which would be the main contribution of the MAS field, is sacrificed. Once it is sacrificed, the traditional approaches are perfectly adequate: why would anyone include any MAS at all?

Further, the increasing realization of the importance to ethics in AI provides a challenge and an opportunity. Our current techniques don't facilitate support moral autonomy in its general Kantian sense [9, 11] nor do they correspond to human behavior (beliefs and desires are in general not precursors to behavior).

Meaning

This seminar's description and indeed much of computer science talks of semantics as the only kind of meaning. But classically, that is not so [5, 8]. *Semantics* is the aspect of the meaning of a representation that can be computed from the representation. But what matters more is *pragmatics*, which is the meaning of a representation that relies upon the context [8]. Even something as simple as understanding pronouns thus involves pragmatics.

Semantics is valuable when we can get it. Semantics gives us reusability and easier standardization but at the cost of flexibility. In particular, in multiagent systems on the web, a large part of the context is built by the agents themselves. How language is used and how its usage and meaning evolve fall in the realm of pragmatics.

For agents on the web, meaning would include practices that they establish (e.g., norms and conventions) or which emerge through their interactions. For example, if a bank processes deposits to your account each night before withdrawals, the outcome would be different than if it did so in the reverse order or in the order in which deposits and withdrawals arrived.

What we need are methods to induce semantics from the pragmatics, which would give us gains in transparency and interpretability. Such methods would apply continually to account for the evolution of a sociotechnical system.

Interaction

Conventionally, interaction is modeled from the perspective of its initiator: e.g., as the caller of a method or the sender of a (synchronous or asynchronous) message.

But what an interaction is depends on how it is construed. An agent may act morally, violate social norms, suffer a loss, inflict pain or pleasure by taking an action in a particular setting. Moreover, two or more agents may jointly invent a new language or introduce or deprecate some meanings by their collaborative actions. Thus, how we understand interaction interplays with autonomy and meaning. Also, interaction can provide a basis for conceptualizing more elaborate agents out of ensembles of agents based on how they interact with effects on their abilities [6] and intentions [7], to mention two works from the early days of the field.

Architecture

A key guiding principle in system design is the end-to-end argument. If a functionality is potentially needed at one layer, it ought not be supported from a lower layer: either the functionality is superfluous and never used or incomplete and the upper layer needs to redo it.

Time and again, software library developers and even standards groups find themselves contravening the end-to-end argument. They give arguments such as the need for performance of some shape or form. I am reminded here of Don Knuth's dictum on premature optimization being the root of all evil.

But perhaps these seemingly unprincipled developers (and groups) are on to something. Taking as a given that the needs that motivate these developers are legitimate as is the end-to-end argument, we should move away from the notion of layers that are fixed across application environments. That is, if developers find it necessary to break the layer encapsulation, maybe that wasn't such a great abstraction to begin with.

We need architectures in which policies addressing different aspects of interaction can be composed. Let's consider retrying transmissions as an example. An interaction may (or may not) be retried upon failure depending on the pragmatics of the application and knowledge of the state of the infrastructure. For example, an action of a physician, patient, and pharmacy to refill a prescription may be retried by just the patient and pharmacy (if the physician's component had progressed sufficiently). Or, the action may be retried with the physician's assistant on the physician's behalf. Or, the retry may be dropped if the patient has met their need through another source. That is, high-level reasoning may be used to handle something as simple as retrying. This view contrasts with many communication protocols that hardcode this functionality, thereby violating the end-to-end argument.

Traditionally, the layers corresponded to different potential businesses. The physical layer is about wires and radios to get a signal across; the network layer is about routing packets; and so on. Thus the layers were standardized and different businesses could build products for them. That is a desirable feature. But we could potentially support greater transparency into each layer as well as greater control. For example, visibility into the nonfunctional properties of a lower layer may help an upper layer adapt better in itself and control the lower layer better.

I expect that these challenges are being addressed in the networking community at the lower layers but we need analogous and open solutions with respect to MAS and governance. Along these lines, we might consider new composition operators for protocols [4] that support additional details to be interposed where appropriate, going beyond existing approaches to contextualize a protocol [2].

Governance

I think of governance of a sociotechnical system as combining the above concerns. Traditional computer scientists in general have produced strong capabilities in the technical tier. The MAS community has investigated flexibility in the social tier through the study of interaction and autonomy and abstractions based on cognition, norms, and economic behavior. Other communities talk of social meaning but do so in a largely informal way or with low-level abstractions and in the aggregate, such as in social networks. Thus current work suffers from limitations but provides directions to address the flexibility we desire.

But the most interesting part of a sociotechnical system is how the social and technical tiers interplay. Computer scientists of all stripes leave this interplay largely ad hoc with legendary fiascoes in the case of technologies such as blockchain [1].

An essential direction therefore is to capture varieties of this interplay to be able to govern a sociotechnical system effectively, e.g., by encoding the social tier explicitly [3] and providing ways to specify expectations rigorously while enabling them to the violated (a crucial requirement of autonomy) [10].

Meta Motivation

A potential counterargument is that we have our hands full dealing with the concepts as they are. Suppose we were more ambitious? Wouldn't that be even harder?

My meta position is that it would be easier. For inspiration, I draw your attention to the *Inventor's Paradox*. This is a situation arising commonly enough in problem solving and was identified by George Pólya. The "paradox" simply is that the more general problem is easier to solve.

In essence, by generalizing a problem, we can bring forth structure in the problem that is otherwise invisible. That structure leads to a more natural solution becoming available.

The prevalent narrow conceptions of autonomy, interaction, semantics on the one hand, and rigid architectures of the other restrict outcomes.

Acknowledgments

I have benefited from discussions with Amit Chopra and Samuel Christie on related topics. Thanks also to the US National Science Foundation (grant IIS-1908374) for supporting this research.

References

1 Vitalik Buterin. Critical update re: DAO vulnerability, June 2016. URL: `https://blog.ethereum.org/2016/06/17/critical-update-re-dao-vulnerability/`.
2 Amit K. Chopra and Munindar P. Singh. Contextualizing commitment protocols. In *Proceedings of the 5th International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 1345–1352, Hakodate, Japan, May 2006. ACM Press. `doi:10.1145/1160633.1160884`.
3 Özgür Kafalı, Nirav Ajmeri, and Munindar P. Singh. DESEN: Specification of sociotechnical systems via patterns of regulation and control. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 29(1):7:1–7:50, February 2020. `doi:10.1145/3365664`.
4 Ashok U. Mallya and Munindar P. Singh. An algebra for commitment protocols. *Journal of Autonomous Agents and Multi-Agent Systems (JAAMAS)*, 14(2):143–163, April 2007. `doi:10.1007/s10458-006-7232-1`.
5 Charles Morris. *Foundations of the Theory of Signs.* University of Chicago Press, Chicago and London, 1938.
6 Munindar P. Singh. Group ability and structure. In Yves Demazeau and Jean-Pierre Müller, editors, *Decentralized Artificial Intelligence, Volume 2*, pages 127–145. Elsevier/North-Holland, Amsterdam, 1991. Revised proceedings of the *2nd European Workshop on Modeling*

Autonomous Agents in a Multi Agent World (MAAMAW), St. Quentin en Yvelines, France, August 1990.

7 Munindar P. Singh. The intentions of teams: Team structure, endodeixis, and exodeixis. In *Proceedings of the 13th European Conference on Artificial Intelligence (ECAI)*, pages 303–307, Brighton, United Kingdom, August 1998. John Wiley.

8 Munindar P. Singh. The pragmatic web. *IEEE Internet Computing (IC)*, 6(3):4–5, May 2002. Instance of the column *Being Interactive*. `doi:10.1109/MIC.2002.1003124`.

9 Munindar P. Singh. Consent as a foundation for responsible autonomy. *Proceedings of the 36th AAAI Conference on Artificial Intelligence (AAAI)*, 36(11):12301–12306, February 2022. Blue Sky Track. `doi:10.1609/aaai.v36i11.21494`.

10 Munindar P. Singh and Amit K. Chopra. Computational governance and violable contracts for blockchain applications. *IEEE Computer*, 53(1):53–62, January 2020. `doi:10.1109/MC.2019.2947372`.

11 Roger J. Sullivan. *An Introduction to Kant's Ethics*. Cambridge University Press, Cambridge, UK, 1994.

3.21 Challenges for Query Agents on the Decentralized Web

Ruben Taelman (Ghent University, BE)

While the Web was originally envisioned as a decentralized information space, it has evolved to a place where the majority of data is flowing towards isolated *data silos*. Since these data silos are in the hands of large companies and organizations, this leads to issues related to privacy and vendor lock-in.

To counter these problems, decentralization initiatives such as Solid are gaining popularity, which allows users to store any kind of data in their own personal data vault, which they fully control. These data vaults are personal Knowledge Graphs that can be represented as collections of Linked Data documents and more expressive query APIs.

The presence of such data vaults results in a large-scale distribution of data, where applications involving multiple individuals' data require accessing thousands or even millions of APIs across different data vaults on the Web. These applications cannot effectively be built today due to the lack of querying techniques that can hande the requirements of decentralized environments like Solid.

A promising paradigm to tackle this query problem is *Link Traversal Query Processing (LTQP)*, which is based on the *follow-your-nose principle* of Linked Data where query execution is done over a continuously growing range of documents by autonomously following hyperlinks between documents.

In order to apply LTQP effectively to decentralized environments, several open challenges need to be tackled:

1. To enable clients to select APIs with the most suitable query capabilities, there is a need to **represent the capabilities of query APIs using self-descriptive hypermedia descriptions**. Current approaches have limited expressivity.
2. To enable clients to select and discover APIs matching their query needs, there is a need to **describe the contents of the data exposed through query APIs via hypermedia**, which could be cardinality-based, shape-based, or approximate summaries. Current approaches lack the discovery and privacy-preservation.

3. To enable efficient client-side query execution, there is a need for **algorithms that provide (adaptive) query planning and execution over the heterogeneity of query APIs** in terms of capabilities, contents, and client-side policies. Current techniques suffer from query termination problems caused by unselective link following techniques and inefficient query plans caused by static non-adapting query planning.

3.22 Signifying Affordances for Effective Interaction of Agents on the Web

Danai Vachtsevanou (Universität St. Gallen, CH)

Having declarative specifications of interaction in Web-based Multi-Agent Systems (MAS) could support autonomous agents in discovering and reasoning about affordances[11] on the Web towards effectively achieving their goals [2]. One possible path forward is to consider *signifiers* as a first-class abstraction in Web-based MAS for modelling *cues and signs that reveal information about how to exploit affordances* offered to agents in hypermedia environments [8]. Signifiers enable agents to interact with shared artifacts and other resources in a hypermedia-driven manner, and recommend under which circumstances interaction should take place based on an agent's *abilities* and the agent-environment *context.* As a result, the exploitability and relevance of an affordance can be dynamically evaluated to guide interaction even in affordance-rich and open environments, for instance, based on the agent's ability to reason and act using specific methods and mechanisms, the agent's ability to handle abstractions and processes of a given domain, the agent's goals or the artifact's state.

Further, evaluating interaction relevance could be dynamically distributed among agents, and other entities in the environment that *manage the exposure of signifiers* [6, 8]. Therefore, interaction guidance would remain effective (despite the heterogeneity of components' context and features[12]) across the spectrum from Web-like environment-driven opportunism to complex agent-driven reasoning and planning about action: From the Web perspective, this offers a bridge between local and global hypermedia-driven guidance, e.g. by enabling the step-by-step provisioning of selected interaction specifications (e.g., of W3C Web of Things Interaction Affordances [13]) with respect to valid state transitions and agents' goals. For example, a BDI agent [3] that desires to pick and place an item, may perceive the signifier for moving the gripper of a robotic arm only if the gripper in empty and in range of the item. From the MAS perspective, signifiers provide a bridge between hypermedia-driven affordance exploitation and methods for reasoning about action as examined in MAS research, since hypermedia are defined in terms of abstractions relevant to autonomous agents (e.g. action preconditions and postconditions, agent goals and roles etc.). For example, an agent capable

[11] Affordances in this context denote interactions that become possible for agents to enact based on the run-time context and agents' abilities. The term is inspired from affordance theory [1]

[12] For example, signifier exposure could be adjusted with respect to artifacts that implement one native application logic, or the Thin Server architecture [4], and agents capable of simple reflex actions, or automated planning.

[13] A W3C Web of Things Interaction Affordance is metadata that shows how agents can interact with physical devices modelled as Things [5].

of planning how to pick and place may perceive signifiers for most of the affordances of the robotic arm (including information about action preconditions and postconditions) towards enriching its planning domain.

Given that declarative interaction specifications aim to support *interaction effectiveness* in hypermedia environments (e.g. through the contextual run-time exposure of signifiers), it is interesting to investigate how agents could evaluate such effectiveness. To begin with, addressing the latter requires that interaction specifications capture the full complexity of agents' interactions, which should be realized and, thus, evaluated within an (partially) *observable stateful system* and within a *social context*. For example, agents should be able to reason about how they can *interact for perception*, i.e. with the purpose of affecting their percepts. Percepts are essential for supporting agents' goal deliberation and decision-making processes, including deciding when and how to act towards modifying their perceived environment, or where to focus for evaluating whether their actions succeeded and contributed to their goals. To this end, and given the large size and the dynamicity of the state space, interaction specifications should reveal information about how agents can interact for directing their perception scope, and how interactions for perception relate to interactions for modifying the environment state. Upon exposure of such information, agents would perceive the environment state, and, then, reason about how well their expectations and intentions have been met [14].

Similarly, agents should be able to discover and interpret how to interact towards perceiving and modifying the current *social state*, for instance, in the context of an organization or an interaction protocol. For example, based on [8], signifiers could be exposed based on agent abilities that derive from the roles that agents hold within an organization. However, agents still cannot reason about policies and norms that capture the behavioral and social expectations that relate to affordance exploitation on the Web, e.g. reason about what organizational goals to adopt. Upon discovering interaction specifications, agents should be able to interpret information for perceiving the social state, acting towards modifying the social state in a regulated environment, and finally, reasoning about how well the behavioral and social expectations have been met.

Overall, towards defining and evaluating interaction effectiveness in Web-based MAS, it is interesting to investigate the following:

- How to model and represent interaction specifications that reveal information about how to interact for perception, and how interaction for perception relates to interaction for modifying the environment state in Web-based MAS.
- How to model and represent interaction specifications that reveal information about how to interact within a social context in Web-based MAS, e.g. based on the norms that apply in the scope of an organization or an interaction protocol.
- How to define and evaluate interaction effectiveness with respect to the expected or desired changes on the environment and social state in Web-based MAS.

References

1 A. Chemero and M.T. Turvey. *Complexity, Hypersets, and the Ecological Perspective on Perception-Action.* Biological Theory, 2(1), 2007.

2 A. Ciortea, S. Mayer, O. Boissier and F. Gandon. *Exploiting Interaction Affordances: On Engineering Autonomous Systems for the Web of Things.* In 2nd W3C Workshop on the Web of Things The Open Web to Challenge IoT Fragmentation, 2019.

[14] Per Norman, determining how well expectations and intentions have been met upon affordance exploitation amounts to "crossing the Gulf of Evaluation" [7].

3 M.P. Georgeff and A.L. Lansky. *Reactive Reasoning and Planning.* In Proceedings of the 6th National Conference on Artificial Intelligence (AAAI), 1987.
4 M. Kovatsch, S. Mayer and B. Ostermaier. *Moving Application Logic from the Firmware to the cloud: Towards the Thin Server Architecture for the Internet of Things.* In 2012 6th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing, 2012.
5 M. Kovatsch, M. McCool, S. Käbisch, V. Charpenay and T. Kamiya. *Web of Things (WoT) Thing Description. W3C Recommendation.* W3C, 2020. Available online: https://www.w3.org/TR/2020/REC-wot-thing-description-20200409/.
6 J. Lemée, D. Vachtsevanou, S. Mayer and A. Ciortea. *Signifiers for Affordance-driven Multi-Agent System.* Paper presented at the International Workshop on Engineering Multi-Agent Systems (EMAS) at the 21st International Conference on Autonomous Agents and Multiagent Systems (AAMAS), 2022.
7 D.A. Norman. *The Design of Everyday Things.* BasicBooks, 2013.
8 D. Vachtsevanou, A. Ciortea, S. Mayer and J. Lemée. *Signifiers as a First-class Abstraction in Hypermedia Multi-Agent Systems.* In Proceedings of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS), 2023.

3.23 Agents for the Decentralized Web of Data

Jan Van den Bussche (Hasselt University, BE)

On the decentralized Web, personal data is managed in so-called "pods". The idea (as seen, e.g., in the Solid project) is that a pod is an online data store that can give access to resources to specific parties. To manage the various data sharing contracts that exist with many different parties, one could use a "Web agent". Such an agent can not only keep track of which parties have which access to which resources, but can provide mediator services for finer-grained data sharing. The mediator can map the data we are willing to share with a party to a virtual or materialized RDF graph, directly in a format that is most useful for the party. Methods from the field of information integration, such as schema mappings and query rewriting, can be applied in this context. Yet, the adaptation of these methods to the RDF setting presents new challenges, such as reasoning about conformance of SPARQL queries to shape (e.g., SHACL) schemas.

A separate function that the Web agent can fulfill is to keep parties up-to-date about changes in the personal data. If the shared resource is virtual, this leads to interesting reasoning tasks about indepence of queries from updates. These tasks have been investigated in the field of database systems, but much less in the SPARQL context.

3.24 Socio-Technical Networking Platforms for Multi-Agent Collaboration on the Web

Antoine Zimmermann (Ecole des Mines – St. Étienne, FR)

A lot of collaboration happening on the Web between people consists in exchanging written messages. In fact, for years, it was the only way people could interact via the Web. Now, since the Web has become a software platform through which we can build and propose any kind of applications, collaborations happen in many different ways, and people on the Web have an open-ended set of possibilities of interaction with the Web and with other people. The long-term goal of the Agents on the Web effort should be that artificial agents can deal with all these possibilities, with sufficient guidelines provided by explanatory resources like metadata, ontologies, etc. However, a framework that would allow all agents to exploit any and all possible interactive resources on the Web in their generality would be as difficult to design as a general AI.

I propose that we instead start the effort of bringing agents to the Web by devising *socio-technical networking platforms* (STN platforms)[15] where agents can easily:

- retrieve specific written messages posted by other agents and isolate them from other content like navigation, titles, dates, authors, etc. (e.g. discussion threads, blog posts, comments, replies, forum messages, wiki discussions, Git issues, reviews, Q&As);
- distinguish different types of messages (e.g., distinguish an original post from a response to another message);
- register to the platform autonomously (possibly with validation by a human person that could be accountable for the agent's actions);
- post messages, either as original posts or as replies to others;
- connect with other agents on the platform, so as to filter which content matters vs what's ignored.

Devising a framework that would allow agents to autonomously make use of this limited set of operations, in a way that is generic and can be instantiated on any social networking platform, is attainable in a near future. As a next step, there should be regulation put in place with the same objective of genericity: devise a framework where agents can autonomously determine what permissions/prohibitions there are on the operations listed above, and have certain agents playing the roles of policy enforcement agents.

When all this will be in place, then we can consider more general modes of interactions on the Web, such as making use of arbitrary software artifact that functions on the Web, or use real time discussion tools.

[15] STN platforms already exist, e.g., Wikipedia where humans and bots collaborate to either ensure better content or to help regulate the platform.

4 Overview of Demonstrators

4.1 Developing Multi-Agent Systems Using JaCaMo-REST

Cleber Jorge Amaral (Federal Institute of Santa Catarina – São José, BR)
Jomi Fred Hübner (Federal University of Santa Catarina, BR)
Timotheus Kampik (Umeå University, SE & SAP Signvavio – Berlin, DE)

In recent years, researchers have started to explore the integration of Multi-Agent Oriented Programming (MAOP) paradigm [1] and resource-oriented web architecture (REST) [2, 3, 4]. The present demo further advances this research line, presenting a novel approach for setting up the integration between a MAS and a range of different kinds of web resources. The demonstration of this approach and the steps to execute it are available at `https://github.com/jacamo-lang/jacamo/tree/master/demos/integration`. In particular, we demonstrate the use of the routing and mediation tool Node-Red (`https://nodered.org/`), for setting up JaCaMo-REST, a resource-oriented web-based abstraction for the Multi-Agent System (MAS).

JaCaMo-REST defines endpoints and provides web infrastructure for exposing MAS entities as web resources. Node-Red provides an integrating engine and an interface for defining flows to wire-up heterogeneous web resources. Its interface allows users to define data flow and transformation sequences from an input endpoint to an output endpoint.

Among the features of JaCaMo-REST, we highlight *proxy agents* and *proxy artifacts*. They allow MAS application agents to access web resources considering them as either agents or artifacts. Thus agents are able to interact with web resource in usual agent-to-agent or agent-to-environment interaction. In this sense, according to the external resources that the MAS must be integrated with, it is possible to define abstractions to interface web resources and MAS internal entities. The referred *proxy* abstractions can be defined using Node-Red flows. Indeed, Node-Red handles protocol transformations through its extensive library of connectors and encapsulates integration specifications such as the address of the endpoints and specific data transformation flows.

In the demonstration, two examples are exploited. In the first example, an external resource that is available via an MQTT *topic* is represented in the MAS as an *proxy agent*. An MAS agent, named *Bob*, can thus send messages to the resource using its Agent Communication Language (ACL). In this example, *Bob* can interact with the external resource without knowing, for instance, its Unique Resource Identifier (URI) or communication protocol, showing that an MAS application can be defined without integration concerns. In another example, *Bob* interacts with a proxy artifact (perceiving and acting upon it).

References

1 Olivier Boissier and Rafael H. Bordini and Jomi F. Hübner and Alessandro Ricci and Andrea Santi. *Multi-agent oriented programming with JaCaMo.* Science of Computer Programming, 78, 6, p. 747-761, 2013. `https://doi.org/10.1016/j.scico.2011.10.004`
2 Amaral, Cleber Jorge and Hübner, Jomi Fred and Kampik, Timotheus. *Towards Jacamo-rest: A Resource-Oriented Abstraction for Managing Multi-Agent Systems*, 2020. `https://arxiv.org/abs/2006.05619`

3 Ciortea, Andrei and Zimmermann, Antoine and Florea, Adina Magda. *Give Agents Some REST: Hypermedia-driven Agent Environments*, Springer, 125-141, 2018. https://doi.org/10.1007/978-3-319-91899-0_8
4 W. Collier, Rem and O'Neill, Eoin and Lillis, David and O'Hare, Gregory. *MAMS: Multi-Agent MicroServices*, Association for Computing Machinery, 655-662, 2019. https://doi.org/10.1145/3308560.3316509

4.2 Test-Driven Development for Agent-Oriented Programming

Cleber Jorge Amaral (Federal Institute of Santa Catarina – São José, BR)
Jomi Fred Hübner (Federal University of Santa Catarina, BR)
Timotheus Kampik (Umeå University, SE & SAP Signvavio – Berlin, DE)

Agent-Oriented Programming (AOP) [1] is a paradigm that provides first-class abstractions for instilling autonomous behavior into software systems. While this paradigm has not yet been widely adopted by the software engineering mainstream, from an academic perspective, the technology ecosystem can be considered thriving [2, 3].

In modern software engineering, developers commonly apply Test-Driven Development (TDD) approaches, in which a large portion of the tests is written during or even ahead of the implementation of the actual program code. The assumption is that specifying the exact desired behavior of a software component before or alongside the implementation facilitates a more rigorous assessment of the component and ensures testing is not cut short because of time shortage caused by incorrect or imprecise estimations. This applies in particular in the context of autonomous and distributed agent-oriented systems, where reliable governance is a key concern [4] and implementation or operation errors can have disastrous consequences [5]. Although first works that are concerned with the development of QA-related capabilities for AOP have recently emerged [6, 7], so far, no testing utilities for any agent-oriented programming language appear to exist.

This demonstration shows how to specify and perform tests for JaCaMo agents using the AgentSpeak programming language [9, 10]. It is derived from the tutorial is available at: `https://github.com/jacamo-lang/jacamo/tree/master/doc/tutorials/tdd`, as well as from a recent AAMAS demonstration paper [8]. The approach uses a novel goal-oriented testing feature that enables an agent-oriented perspective on automated software testing and test-driven development.

The developed testing tool provides facilities to assert whether an expected outcome is being produced at the unitary level of an agent's inference rules, to test plans for the achievement of the agent's desired goals, and to test agents' integration, in which the facility uses the testing pipeline to assure that the interactions among agents are occurring as expected. The tool also allows for the instantiation of tester agents, which, besides inheriting the features of the agent under test, can run tests to assert whether the rules, plans, and beliefs of the agent under test are as expected. It is possible, for instance, to test if the agent behaves as expected while a particular plan is performed and changes the agent's internal state (e.g., belief adoption). When the main concern is the agent's logic, it is possible to use mock artifacts to avoid interaction with the environment. Additionally, mock agents can be instantiated to be part of a system to test agents' interactions.

The testing tool has several advantages compared to the common practice of using debug messages on production code: (i) the test code can be written separately from the production code, which results in cleaner code and a clearer separation of concerns; (ii) the expected outcomes of agents' functionalities are easier to understand; and (iii) it facilitates the automation of the tests, for instance, using Continuous Integration (CI) tools that enhance collaboration and quality assurance of projects developed by multiple engineers.

References

1 Shoham, Yoav. *Agent-oriented programming.* Artificial Intelligence, 60, 1, p. 51-92, 1993. `https://doi.org/10.1016/0004-3702(93)90034-9`

2 Mascardi, Viviana and Weyns, Danny and Ricci, Alessandro and Earle, Clara Benac and Casals, Arthur and Challenger, Moharram and Chopra, Amit and Ciortea, Andrei and Dennis, Louise A. and Díaz, Álvaro Fernández and El Fallah-Seghrouchni, Amal and Ferrando, Angelo and Fredlund, Lars-Åke and Giunchiglia, Eleonora and Guessoum, Zahia and Günay, Akin and Hindriks, Koen and Iglesias, Carlos A. and Logan, Brian and Kampik, Timotheus and Kardas, Geylani and Koeman, Vincent J. and Larsen, John Bruntse and Mayer, Simon and Méndez, Tasio and Nieves, Juan Carlos and Seidita, Valeria and Teze, Baris Tekin and Varga, László Z. and Winikoff, Michael. *Engineering Multi-Agent Systems: State of Affairs and the Road Ahead.* Association for Computing Machinery, New York, NY, USA, 44, 1, p. 18-28, 2019. `https://doi.org/10.1145/3310013.3322175`

3 Pal, Constantin-Valentin and Leon, Florin and Paprzycki, Marcin and Ganzha, Maria. *A review of platforms for the development of agent systems.* 2020. arXiv preprint arXiv:2007.08961

4 Kampik, Timotheus and Mansour, Adnane and Boissier, Olivier and Kirrane, Sabrina and Padget, Julian and Payne, Terry R. and Singh, Munindar P. and Tamma, Valentina and Zimmermann, Antoine. *Governance of Vol. Web: Challenges and Opportunities.* Association for Computing Machinery, 22, 4, 2022. `https://doi.org/10.1145/3507910`

5 Kampik, Timotheus and Amaral, Cleber Jorge and Hübner, Jomi Fred. *Developer Operations and Engineering Multi-agent Systems.* Engineering Multi-Agent Systems, Springer International Publishing, p. 175-186, 2022. `https://doi.org/10.1007/978-3-030-97457-2_10`

6 Amaral, Cleber Jorge and Hübner, Jomi Fred. *Jacamo-Web is on the Fly: An Interactive Multi-Agent System IDE.* Engineering Multi-Agent Systems, Springer International Publishing, p. 246-255. 2020. `https://doi.org/10.1007/978-3-030-51417-4_13`

7 Amaral, Cleber Jorge and Kampik, Timotheus and Cranefield, Stephen. *A Framework for Collaborative and Interactive Agent-oriented Developer Operations.* Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems, Auckland, New Zealand. 2020. `https://dl.acm.org/doi/10.5555/3398761.3399086`

8 Amaral, Cleber Jorge and Hübner, Jomi Fred and Kampik, Timotheus. *TDD for AOP: Test-Driven Development for Agent-Oriented Programming* (to appear). Proceedings of the 22nd International Conference on Autonomous Agents and MultiAgent Systems, London, UK. 2023.

9 Boissier, Olivier and Bordini, Rafael H and Hubner, Jomi and Ricci, Alessandro. *Multi-agent oriented programming: programming multi-agent systems using JaCaMo.* MIT Press, Cambridge, Massachusetts, USA. 2020.

10 Bordini, Rafael H. and Hübner, Jomi Fred and Wooldridge, Michael. *Programming Multi-Agent Systems in AgentSpeak Using Jason (Wiley Series in Agent Technology.* John Wiley & Sons, Inc., Hoboken, NJ, USA. 2007.

4.3 Robust JaCaMo Applications via Exceptions and Accountability

Matteo Baldoni (University of Turin, IT)
Cristina Baroglio
Roberto Micalizio
Stefano Tedeschi

Robustness, *"the degree to which a system or component can function correctly in the presence of invalid inputs or stressful environmental conditions"* – generally called *perturbations* [11], is a crucial requirement of distributed software systems [12, 13, 10, 9]. Multi-Agent Systems (MAS) [15] are an effective approach to realize distributed systems by means of heterogeneous, and autonomous agents. Agent *organizations* (MAO), in particular, provide abstractions for modularizing code spread over many components, and orchestrate their execution by way of norms. JaCaMo [8] is one of the best-known platforms for implementing MAOs, but it focuses on providing the means for capturing the normal, correct behavior of the system and lacks of structural mechanisms allowing agents to exchange and propagate information (feedback) when they face perturbations. As in [1], the availability of *feedback* about perturbations is crucial to build robust distributed systems. Also MAS robustness should ground on the ability to convey feedback about perturbation to the agents that can handle it. But since agents generally are peers, and are not related by relationships like caller-callee or parent-child, the realization of robustness should occur through the definition of distributions of responsibilities among the agents, that become part of the MAO. This demonstration presents two extensions to the JaCaMo platform which allow building robust agent organizations. The first borrows from software engineering the concepts of *exception* and *exception handling*, while the second relies on the notion of *accountability*. Exception handling is suitable for treating perturbations anticipated at design time (i.e., exceptions) by activating handlers, that are also specified at design time. The details of the implementation can be found in [14, 6, 2, 3], source code available at `http://di.unito.it/moiseexceptions`.

Accountability, instead, defines feedback "channels" that agents can use at runtime to gain situational awareness about what occured and then take actions. Raising and handling exceptions as well as asking and returning for an account will be tasks, under the responsibility of specific agents. The two extensions provide the means for representing such tasks as goals, and for distributing the reponsibilities of such goals to the capable agents. Note that each such goal can be assigned to many agents, specifying the minimum and maximum cardinality of how many agents need to achieve the goal (as standard in a JaCaMo organization specifications). Moreover, we allow specifying many raising and handling goals for each exception, and many requesting, accounting and treatment goals (possibly involving many agents) for each accountability. Concretely, we leveraged the model presented in [4] and the formalization from [5, 7], source code available at `http://di.unito.it/moiseaccountability`.

References

1 David L. Alderson and John C. Doyle. Contrasting views of complexity and their implications for network-centric infrastructures. *IEEE Tr. on Sys., Man, and Cyber.*, 40(4), 2010.
2 Matteo Baldoni, Cristina Baroglio, Olivier Boissier, Roberto Micalizio, and Stefano Tedeschi. Demonstrating Exception Handling in JaCaMo. In Frank Dignum, Juan M. Corchado, and Fernando De La Prieta, editors, *Advances in Practical Applications of Agents, Multi-Agent Systems, and Social Good. The PAAMS Collection – 19th International Conference, PAAMS*

2021, Salamanca, Spain, October 6-8, 2021, Proceedings, volume 12946 of *Lecture Notes in Computer Science*, pages 341–345. Springer, 2021.

3 Matteo Baldoni, Cristina Baroglio, Olivier Boissier, Roberto Micalizio, and Stefano Tedeschi. Distributing Responsibilities for Exception Handling in JaCaMo. In Ulle Endriss, Ann Nowé, Frank Dignum, and Alessio Lomuscio, editors, *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '21, pages 1752–1754. International Foundation for Autonomous Agents and Multiagent Systems, 2021.

4 Matteo Baldoni, Cristina Baroglio, Roberto Micalizio, and Stefano Tedeschi. Reimagining Robust Distributed Systems through Accountable MAS. *IEEE Internet Computing*, 25(6), 2021.

5 Matteo Baldoni, Cristina Baroglio, Roberto Micalizio, and Stefano Tedeschi. Robustness Based on Accountability in Multiagent Organizations. In Ulle Endriss, Ann Nowé, Frank Dignum, and Alessio Lomuscio, editors, *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '21, pages 142–150. International Foundation for Autonomous Agents and Multiagent Systems, 2021.

6 Matteo Baldoni, Cristina Baroglio, Roberto Micalizio, and Stefano Tedeschi. Exception handling as a social concern. *IEEE Internet Computing*, 26(6):33–40, 2022.

7 Matteo Baldoni, Cristina Baroglio, Roberto Micalizio, and Stefano Tedeschi. Accountability in multi-agent organizations: from conceptual design to agent programming. *Autonomous Agents and Multi-Agent Systems*, 37(1):1–37, 2023.

8 Olivier Boissier, Rafael H. Bordini, Jomi Hübner, and Alessandro Ricci. *Multi-agent oriented programming: programming multi-agent systems using JaCaMo*. MIT Press, 2020.

9 Samuel H. Christie, Amit K. Chopra, and Munindar P. Singh. Bungie: Improving fault tolerance via extensible application-level protocols. *Computer*, 54(5):44–53, 2021.

10 Samuel H. Christie, Amit K. Chopra, and Munindar P. Singh. Mandrake: multiagent systems as a basis for programming fault-tolerant decentralized applications. *Autonomous Agents and Multi-Agent Systems*, 36(1), 2022.

11 ISO/IEC/IEEE. Systems and software engineering – Vocabulary. *24765:2010(E) – ISO/IEC/IEEE International Standard*, 2010.

12 Anuj K Jain, Manuel Aparico IV, and Munindar P Singh. Agents for process coherence in virtual enterprises. *Communications of the ACM*, 42(3):62–69, 1999.

13 Anup K. Kalia and Munindar P. Singh. Muon: designing multiagent communication protocols from interaction scenarios. *Autonomous Agents and Multi-Agent Systems*, 29(4):621–657, 2015.

14 Stefano Tedeschi. *Exception Handling for Robust Multi-Agent Systems*. PhD thesis, Università degli Studi di Torino, Dipartimento di Informatica, Torino, Italy, 2021.

15 Michael Wooldridge. *An introduction to multiagent systems*. John Wiley & Sons, 2009.

4.4 Agent-Oriented Visual Programming for the Web of Things

Samuele Burattini (University of Bologna, IT)

Joint work of Samuele Burattini, Simon Mayer, Alessandro Ricci, Danai Vachtsevanou, Jérémy Lemée, Angelo Croatti

Overview

This demo presents the outcome of the work on *Agent-Oriented Visual Programming* [1] carried out in the context of the European project IntellIoT[16].

The demo is available on GitHub[17] and shows how a multi-agent system controlling two autonomous tractors that expose Web of Things (WoT) Thing Descriptions[18] can be defined and managed through a visual programming interface on a Web application.

More details about the demo can be found on the repository `README.md`.

Agent-Oriented Visual Programming

The Agent-Oriented paradigm, allows the definition of software systems using high-level abstractions (i.e. software agents). Cognitive architectures for software agents are inspired to the mental abilities of humans, hence we argue that, as it is easy for developers to describe the world in terms of objects in Object-Oriented Programming, it should be even easier to model autonomous behaviour in terms of agents.

In the Belief-Desire-Intention (BDI) model, agents are described in terms of mental qualities that are inspired by *human practical reasoning*[3]. We argue that it is possible to exploit this alignment between human practical reasoning and the BDI architecture to create systems that people can effectively use to program MAS since they more closely match their everyday experience in interacting with artifacts and other agents.

Our endeavor is furthermore motivated by two concrete issues experienced in an industrial scenario based on the Web of Things (WoT). The first one is the ever-increasing interest in forms of end-user programming that shall enable not only experienced programmers but ideally domain experts without programming experience to create or modify software systems of different complexity. The second one is the need to create or modify solutions featuring different degrees of autonomy of software components in performing tasks in a flexible way, dealing with open, dynamic, distributed WoT environments. From this angle, at the same time, the use of semantic-web technologies allows to discover high-level actions at run-time, which promotes the serendipitous creation of applications in such environments – given a proper level of abstraction for exploiting them.

Then, the question is how to design an interface that could reinforce the alignment of agent and human reasoning, hiding the technicalities, to enable individuals without experience in programming to express the behaviour of software agents in the most natural way.

Our approach is built on a blocks-based visual programming environment to create Multi-Agent Systems (MAS) that are then executed on top of the JaCaMo[2] platform and can interact with Web of Things environments.

[16] `www.intelliot.eu`
[17] `https://github.com/samubura/dagstuhl-demo-wot-autonomous-tractors`
[18] `https://www.w3.org/TR/wot-thing-description/`

References

1 Burattini, S., Ricci, A., Mayer, S., Vachtsevanou, D., Lemee, J., Ciortea, A. and Croatti, A., 2022. *Agent-Oriented Visual Programming for the Web of Things.*

2 Boissier, O., Bordini, R.H., Hübner, J.F., Ricci, A. and Santi, A., 2013. *Multi-agent oriented programming with JaCaMo.* Science of Computer Programming, 78(6), pp.747-761.

3 Bratman, M.E., Israel, D.J. and Pollack, M.E., 1988. *Plans and resource-bounded practical reasoning.* Computational intelligence, 4(3), pp.349-355.

4.5 Interaction-Oriented Programming

Amit K. Chopra (Lancaster University, GB)
Samuel H. Christie V
Munindar P. Singh (North Carolina State University – Raleigh, US)

Main reference Amit K. Chopra, Samuel H. Christie V., Munindar P. Singh: "Interaction-Oriented Programming: Intelligent, Meaning-Based Multiagent Systems", in Proc. of the 2023 International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2023, London, United Kingdom, 29 May 2023 – 2 June 2023, pp. 3041–3043, ACM, 2023.
URL https://doi.org//10.5555/3545946.3599166

Interaction-Oriented Programming (IOP) is a novel approach for designing decentralized multiagent systems based on models of interactions. Currently, IOP supports specifying multiagent systems via norms (e.g., commitments) and protocols. It includes tools that can be query databases for norm states, tools for verifying protocols, and programming models based on protocols that can be used to implement agents. IOP's strength is that it supports flexible decision making by agents based on the meaning of interactions. For more details, see our AAMAS Demo paper.
Our software repository is here: `https://gitlab.com/masr/`.
Some software is here: `https://github.com/akchopr/Cupid/`.

4.6 Hypermedia Multi-Agent Systems

Olivier Boissier (Ecole des Mines – St. Étienne, FR)
Andrei Ciortea (Universität St. Gallen, CH)
Fabien Gandon (INRIA – Sophia Antipolis, FR)
Simon Mayer (Universität St. Gallen, CH)
Alessandro Ricci (Università di Bologna, IT)

Main reference Andrei Ciortea, Olivier Boissier, Alessandro Ricci: "Engineering World-Wide Multi-Agent Systems with Hypermedia", in Proc. of the Engineering Multi-Agent Systems – 6th International Workshop, EMAS 2018, Stockholm, Sweden, July 14-15, 2018, Revised Selected Papers, Lecture Notes in Computer Science, Vol. 11375, pp. 285–301, Springer, 2018.
URL https://doi.org//10.1007/978-3-030-25693-7_15

We are working towards defining a new class of Web-based Multi-Agent Systems (MAS) that can inherit the architectural properties of the Web (scalability, heterogeneity, evolvability, etc.), preserve the architectural properties of MAS (adaptability, openness, robustness, etc.), and are human-centric (usable, transparent, accountable, etc.). Our aim is to leverage the full potential of the Web as a middleware in MAS. In our approach, we consider the environment as a first-class abstraction in MAS – and we design the agents' environment as a distributed hypermedia application guided by the design rationale of the Web architecture [1]. The distributed hypermedia environment provides agents with a uniform, knowledge-level abstraction of the system that they can navigate, query, observe, and act upon. The hypermedia-based design rationale reduces coupling and enhances the scalability, openness, and evolvability of the MAS. We refer to Web-based MAS that follow this design rationale as Hypermedia MAS.

During the seminar, we showed a demonstrator in which Belief-Desire-Intention (BDI) agents are situated in an open and evolvable hypermedia-based environment distributed across two locations: Schloss Dagstuhl (Germnay) and the University of St.Gallen (Switzerland). Given a set of semantic models and a single entry URI into the system, the agents are able to achieve their design objectives by navigating the hypermedia environment to discover, create, perceive, and act on artifacts: the agents are able to create and use a digital counter for coordination, and they can control robotic arms located in St.Gallen. The distributed hypermedia environment is represented in RDF and agents are able to susbcribe to triple patterns to receive fine-grained notifications of environmental changes. The demonstrator was built using Yggdrasil[19], JaCaMo[20], Corese[21], and the hardware infrastructure provided by the Chair for Communication- and Interaction-based Systems at the University of St.Gallen[22].

References

1 Andrei Ciortea, Olivier Boissier, and Alessandro Ricci. 2019. Engineering World-Wide Multi-Agent Systems with Hypermedia. In Engineering Multi-Agent Systems, Danny Weyns, Viviana Mascardi, and Alessandro Ricci (Eds.). Springer International Publishing, Cham, 285–301.

4.7 Hypermedia MAS Simulation

Rem Collier (University College Dublin, IE)

Agent-Based Modelling (ABM) is an approach to implementing population-based simulations. A standard ABM simulation consists of two key types of entity: a set of agents (representing the population) and an environment (that provides a shared context for interaction between the agents). A range of toolkits are available for implementing ABM simulations covering both small scale desktop approaches and large scale cluster and cloud based approaches [1].

This talk introduces a new approach to implementing simulations using ABM that builds on the Multi-Agent MicroServices (MAMS) architectural style [2]. The approach, known as Hypermedia MAS Simulation [3] decomposes the environment part of an ABM into a set of sub-environments that are implemented as web resources that are hosted by microservices. The state of each sub-environment is then exposed using REpresentational State Transfer using Linked Data representations. The hyperlinks embedded in the linked data representations capture the relationships between sub-environments (and potentially other relationships too).

[19] `https://github.com/interactions-hsg/yggdrasil`
[20] `https://github.com/jacamo-lang/jacamo`
[21] `https://github.com/wimmics/corese`
[22] `https://interactions.ics.unisg.ch/`

A key novelty of the approach is that the linked data representations, when specified using Semantic Web languages, form a knowledge graph of the environment. Agents enter the environment by joining one of the sub-environments. Where appropriate, the agents transition between sub-environments and receive periodic updates of the state of the sub-environment they are currently in. These updates include the URL of that sub environment, which the agent can use as an entry point into the environment knowledge graph. All agent-environment interaction is realised through HTTP.

In the talk, the approach is illustrated using a simple worked example of a road network that is decomposed into streets and junctions. For simplicity, a single standard model of a street and a junction is defined and the corresponding environment is implemented using two microservices: one for streets and one for junctions. Hyperlinks connect the start and end of streets with associated junctions. Agents enter the environment by connecting to an junction (their starting point) and navigate the road network based on internal goals.

References

1 Abar, Sameera and Theodoropoulos, Georgios K and Lemarinier, Pierre and O'Hare, Gregory MP.: Agent Based Modelling and Simulation tools: A review of the state-of-art software. In: Computer Science Review 24, pp. 13–33, Elsevier (2017)

2 Collier, Rem, O'Neill, Eoin, Lillis, David, O'Hare, Gregory: Mams: Multi-agent microservices. In: Companion Proceedings of The 2019 World Wide Web Conference. pp. 655–662 (2019)

3 Collier, Rem and Russell, Seán and Ghanadbashi, Saeedeh and Golpayegani, Fatemeh: Towards the Use of Hypermedia MAS and Microservices for Web Scale Agent-Based Simulation, SN Computer Science 3(6), Springer (2022)

4.8 Building Multi-Agent Microservices with ASTRA

Eoin O'Neill (University College Dublin, IE)

Integrating Multi-Agent Systems into larger software systems is difficult. Often, the result is to attempt to re-engineer the system in a way that makes it agent-centric. Such approaches are received negatively by the software engineering community because: (1) it requires the adoption of technology stacks that are not well understood or widely used, and (2) most systems already exist in some form and the cost of transforming it to be *agent-ready* is simply too high.

The emergence of microservices architectures [2] has changed this. Among the many benefits of microservices is the notion of polyglot computing – the construction of systems using multiple languages / technology stacks. Polyglot computing works well with microservices because of other principles, such as the discrete boundaries principle (microservices should be independently deployable). When combined with other architectural styles, such as REpresentational State Transfer (REST), the view emerges (from a deployment perspective) of a microservice as a black box with a uniform interface that can simply be connected into the larger system architecture. Details of the technology stack used become less important, so long as they only impact on the developer of the microservice rather than its users.

In response to this, the Multi-Agent MicroServices (MAMS) architectural style has been proposed [4] and a prototype implementation developed using the ASTRA agent programming language [1] and CArtAgO [3]. The MAMS architectural style promotes the idea of an agent being represented using a hypermedia body, composed of a set of virtual resources that it chooses to expose. Each virtual resource, accessible at an endpoint embodies an element of the agent, such as it's inbox. Based on the HATEOAS constraint of the REST architectural style, the state of the agent can determine what virtual resources are exposed and when. Any and all changes are, at all times represented in the hypermedia representation of the agent.

In this demonstration, a number of simple applications are presented that have been built using the MAMS prototype. In the first example we demonstrate an implementation of a Vickrey Auction using the MAMS architectural style. The Agent Oriented Micro Service (AOMS) exposed two virtual resources that allowed external entities to issue HTTP POST requests to, in order to register as a bidder or to sell an item. Internal agents that represent an Auctioneer and a Bidder agent per client then perform the auction and the bidder is informed by a response issued to the web hook they provide.

In the second example we demonstrate how a MAMS agent can interact with an external microservice that implements a single Tic-Tac-Toe board. The board is represented using hypermedia that the agent perceives in order to play the game. This level of loose-coupling and hypermedia-driven interaction between agents and Plain Old Micro Services (POMS) is the goal of our architectural style.

References

1 Collier, R.W., Russell, S., Lillis, D.: Reflecting on agent programming with agentspeak (l). In: PRIMA 2015: Principles and Practice of Multi-Agent Systems: 18th International Conference, Bertinoro, Italy, October 26-30, 2015, Proceedings 13. pp. 351–366. Springer (2015)

2 Dragoni, N., Giallorenzo, S., Lafuente, A.L., Mazzara, M., Montesi, F., Mustafin, R., Safina, L.: Microservices: yesterday, today, and tomorrow. Present and ulterior software engineering pp. 195–216 (2017)

3 Ricci, A., Piunti, M., Viroli, M., Omicini, A.: Environment programming in cartago. Multi-agent programming: Languages, tools and applications pp. 259–288 (2009)

4 W. Collier, R., O'Neill, E., Lillis, D., O'Hare, G.: Mams: Multi-agent microservices. In: Companion Proceedings of The 2019 World Wide Web Conference. pp. 655–662 (2019)

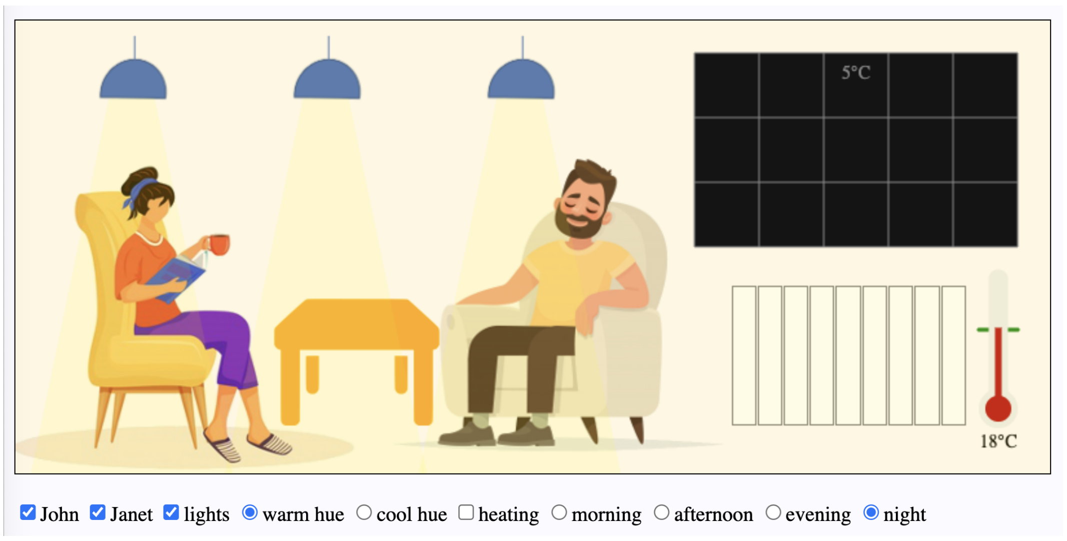
4.9 Web-based Demos of Cognitive AI

Dave Raggett (W3C – United Kingdom, GB)

I presented three web-based demos, two of which use chunks and rules, inspired by John Anderson's cognitive architecture named ACT-R, and another demo introducing plausible reasoning.

A chunk is a collection of properties along with a type and a chunk identifier. Chunk rules have a conjunction of conditions expressed as chunks that must match designated cognitive buffers where each buffer holds a single chunk. A rule is stochastically selected from the set of matching rules, and its actions applied. Actions either synchronously update the buffers or invoke asynchronous operations that indirectly update the buffer analogous to CRUD operations with HTTP. Chunk memory mimics human memory, e.g. the forgetting curve, spreading activation and stochastic recall.

See: `https://w3c.github.io/cogai/`



[✓ John] [✓ Janet] [✓ lights] [✓ warm hue] cool hue heating morning afternoon evening [✓ night]

Figure 3 screenshot of smart home demo.

Smart Homes

This includes a real-time model of heat flow between a room and the outside world, using a cognitive agent to manage the lighting and heating according to the preferences of the room's current participants. Janet prefers a warm room and warm lighting, whilst John prefers a cooler room and bluer lighting. If both Janet and John are present at the same time, then Janet has precidence for the lighting and John for the temperature.

This demo features default reasoning using chunk rules, triggered by events such as people entering or leaving the room, a change in the time of day, and the room temperature rising above or falling below the target temperature. The preferences are expressed as facts.

See: `https://www.w3.org/Data/demos/chunks/home/` and Figure 1.

Manufacturing Cell

This features a cognitive agent that controls a robot, two conveyor belts, a bottle filling station, and a bottle capping station. Empty bottles are take from the first conveyor belt, then filled and capped before being packed into a box, on the second conveyor belt, that takes six bottles. Chunk rules are used to express event driven behaviour. The rules also specify the target state for the robot gripper, delegating real-time control to the robot which smoothly accelerates and decelerates each actuator as required. The demo features matching sound effects.

See: `https://www.w3.org/Data/demos/chunks/robot/` and Figure 2.

Plausible Reasoning

This demo explores the potential for plausible reasoning with imperfect knowledge, i.e. knowledge that may be uncertain, imprecise, incomplete and inconsistent. Logical proof is replaced by plausible arguments for and against a premise, drawing upon work by a long line of philosophers on argumentation theory going all the way back to Ancient Greece, and as used in courtrooms and everyday reasoning. The plausible knowledge notation (PKN) features properties, relationships and implications, along with qualitative metadata that

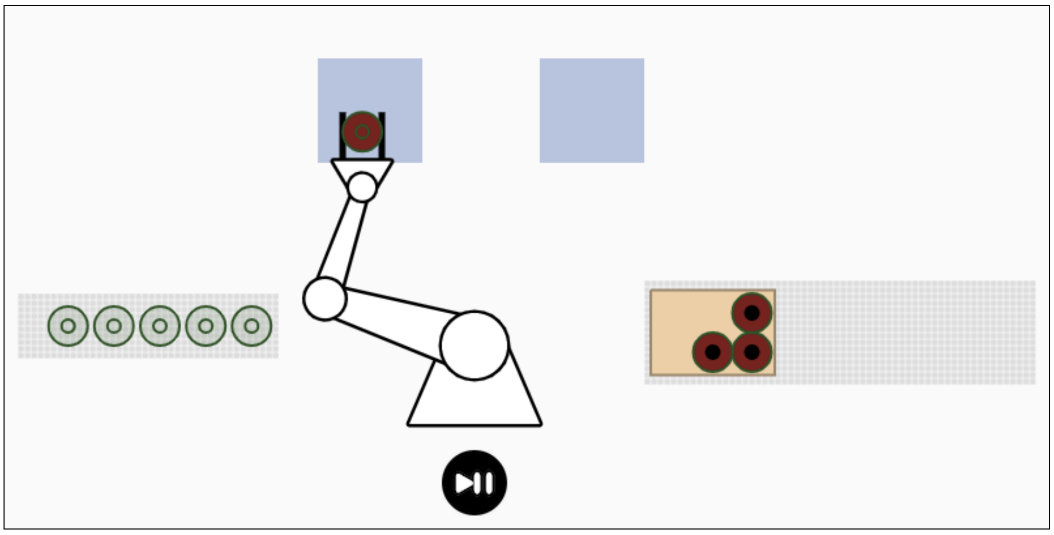


Figure 4 screenshot of robot demo.

reflects prior knowledge. PKN embraces plausible inferences based upon relationships and implications, fuzzy reasoning, fuzzy quantifiers, qualitative reasoning and analogical reasoning. PKN can be considered as a knowledge representation at a level above RDF with additional semantics for imperfect knowledge.

See: `https://www.w3.org/Data/demos/chunks/reasoning/`

4.10 Using MOSAIK for a Decentralized Transportation System

Daniel Schraudner (Universität Erlangen-Nürnberg, DE)

Joint work of Andreas Harth, Sebastian Schmid, Daniel Schraudner
URL https://github.com/wintechis/mosaik-runtime-documentation

We investigate possibilities for implementing the decentralized control of transporters with Semantic Web agents to fulfill a given transportation task. We present our demo of the MOSAIK framework [2] as a system to build and simulate simple reflex agents [1] to control transporters using stigmergy for indirect communication via the environment [3] and self-organize based on local decisions. We use Semantic Web technologies as the communication paradigm of stigmergy directly maps to the REST constraints of the application architecture of the web.

In MOSAIK, we discern active components, agents, and reactive components, called artifacts that respond to agents' actions, and form together the agents' environment. As demo scenario, we present a square shop floor with distinct floor tiles and four stations, each with one of four different colors. A station accepts only products of its own color and randomly produces a product of a different color. Three transporters have to bring colored products to the correct station of the same color without being able to perceive more than only their adjacent fields. The transporters. as artifacts, are each controlled by our software agents that communicate with each other via stigmergy: by placing information about the position of stations in the environment, on the shop floors, the agents help each other to build gradients that lead directly to the stations. With this, no global knowledge of the environment is necessary and the adjustment of the system emerges from local interactions.

As tools, we use BOLD (see https://github.com/bold-benchmark/bold-server) and Linked-Data Fu (ldfu) [4]. BOLD is a simulation environment that implements artifacts with multiple RDF graphs, collected in a single named graph representing the environment state. Agents can read and manipulate artifacts via HTTP requests to HTTP resources under the URI of the named graph, which BOLD updates via defined SPARQL INSERT / DELETE queries. The data processing system ldfu can retrieve, process and modify Linked Data based on logical rules and production rules in Notation3. We use N3 to implement the condition-action rules describing the behaviour of the agent which controls the transporters, and interact with the artifacts via RESTful interfaces.

The decentralized transportation system achieves self-organization by implementing a combination of simple reflex web agents that coordinate using web resources as environment for stigmergy.

References

1 Stuart Russell and Peter Norvig, *Artificial Intelligence: A Modern Approach*, 2009, Prentice Hall Press, USA

2 Victor Charpenay, Daniel Schraudner, Thomas Seidelmann, Torsten Spieldenner, Jens Weise, René Schubotz, Sanaz Mostaghim, and Andreas Harth, *MOSAIK: A Formal Model for Self-Organizing Manufacturing Systems*, IEEE Pervasive Computing, 2021

3 Francis Heylighen, *Stigmergy as a universal coordination mechanism I: Definition and components*, Cognitive Systems Research, 2016

4 Andreas Harth and Tobias Käfer, *Linked Data Techniques for the Web of Things: Tutorial*, 2018, Proceedings of the 8th International Conference on the Internet of Things, Santa Barbara, California, USA

4.11 Querying over Solid Pods via Link Traversal, with Comunica

Ruben Taelman (Ghent University, BE)

Main reference Ruben Taelman, Ruben Verborgh: "Evaluation of Link Traversal Query Execution over Decentralized Environments with Structural Assumptions", CoRR, Vol. abs/2302.06933, 2023.
URL https://doi.org//10.48550/arXiv.2302.06933

In this demonstration, we showed a technique to enable SPARQL query execution over one or more Solid pods in a decentralized environment using the client-side Comunica query engine. The method is a form Link Traversal Query Processing (LTQP) where the structural properties of the Solid ecosystem are taken into account [1] to provide earlier and more complete results.

The demonstrator is based on SolidBench, which is a benchmark that provides a large number of simulated Solid pods containing social network data, and queries over this data. Some of the demonstrated queries cover just one pod, while others span multiple pods.

The demo indicates that a traversal-based query method is an effective way for querying over Solid pods, without having to centralize all of this data beforehand. However, more work is needed to optimize more complex queries are queries spanning multiple pods.

The demo remains available online at `https://comunica.dev/research/link_traversal/`, together with the open-source implementation of the query engine.

References

1 Taelman, Ruben, and Ruben Verborgh. "Evaluation of Link Traversal Query Execution over Decentralized Environments with Structural Assumptions." arXiv preprint arXiv:2302.06933 (2023).

5 Working Group Reports

5.1 Decentralized Hypermedia Ecosystems

Olivier Boissier (Ecole des Mines – St. Étienne, FR)
Pierre-Antoine Champin (INRIA – Sophia Antipolis, FR)
Amit K. Chopra (Lancaster University, GB)
Nicoletta Fornara (University of Lugano, CH)
Fabien Gandon (INRIA – Sophia Antipolis, FR)
Ruben Taelman (Ghent University, BE)
Jan Van den Bussche (Hasselt University, BE)

Inspired by the recent focus of decentralized data for Web applications, as supported in infrastructures such as SOLID, this group discussed the idea of decentralized hypermedia applications in general. In particular, this working group focused on understanding how to integrate the technologies and models used and proposed by three areas of research targeted by the seminar.

1. The notion of Hypermedia, Hypermedia as the Engine of Application State (HATEOAS), and ecosystems developed by the Web of Things community;
2. The notion of Linked Data Platform (a W3C recommendation) together with open specifications such as SOLID proposed by the Semantic Web and Linked Data community.
3. Decentralized abstractions studied in multiagent systems. These include the concept of agents, norms or policies, protocols, artifact, workspace, and organization together with processes of regimenting and enforcing norms.

This integration was investigated by first discussing a list of crucial terms used in each of the communities involved, in order to achieve a good level of mutual understanding. Subsequently, by discussing and specifying a scenario in which all the technologies and concepts proposed by each of the communities involved can be used at different levels of abstraction.

Crucial Terms

- **Hypermedia**: a non-linear multimedia medium where resources are linked by hyperlinks i.e. a digital reference to data that the can be followed to discover new data
- **Hypermedia as the Engine of Application State (HATEOAS)**: is a constraint of the REST application architecture where a client interacts with a network application whose application servers provide information dynamically through hypermedia.
- **Decentralized System**: is a type of system where every node makes its own decision, there is not a central coordinating or governing unit or server.

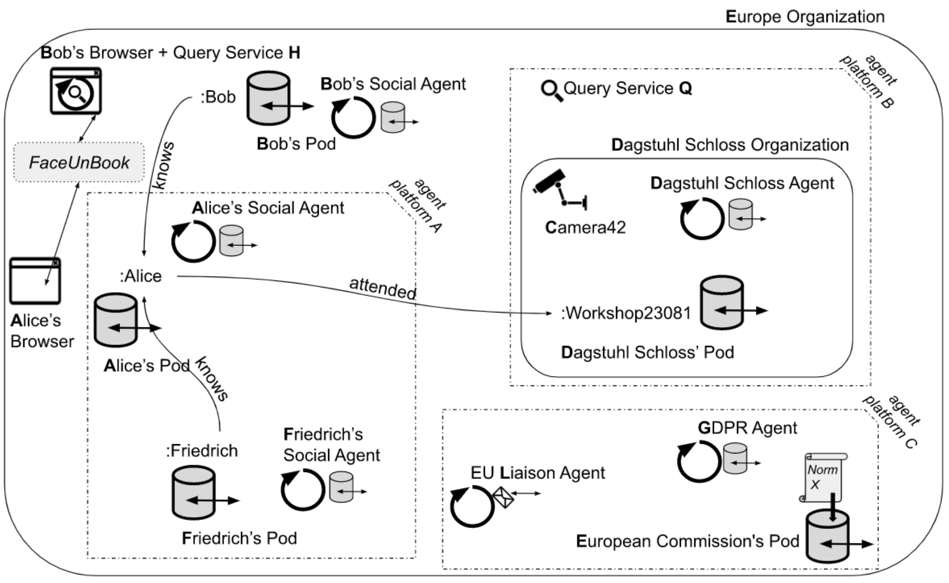
- **SOLID**: Solid is a set of standards that enables a decentralization of data, with interoperable applications over this decentralized data.
- **SOLID ecosystem**: is a specific deployment of a community of users and agents using a given set of applications on top of their Solid pods, and complying with a set of technical and social norms.
- **SOLID pods**: Solid is a specification that lets people store their data securely in decentralized personal online data stores called pods. Pods are like secure personal web servers for data. When data is stored in someone's Pod, they control which people and applications can access it.
- **Pody**: is a term forged during the Dagstuhl Seminar to identify the Pod of an agent that embodies some of its features. This approach is referred to as the empodiment of the agent.
- **Web ID**: URI that identify a person / an agent
- **Agents** : autonomous entities able to perceive, act and interact with other agents, making decisions, subject to norms.
- **Norms**: Capture social expectations, they are by definition violable. Norm types include commitment (obligation), prohibition, permission and so on.
- **Artifacts**: non autonomous entities.
- **Workspace**: a way to organize the artifacts, the artifacts are situated in workspace, agents join workspaces, workspace scope the observability for an agent.
- **Enforcing**: applying sanctions (rewards or fines) when norms are fulfilled or not fulfilled
- **Regimenting**: hindering the violations of norms by agents.
- **ODRL**: the Open Digital Rights Language (ODRL) is a W3C recommendation defining a policy expression language. ODRL policies are used to represent permitted and prohibited actions over a certain asset, and obligations required to be met by stakeholders.

Scenario: Social Media Sharing Application in a Decentralized Hypermedia Ecosystem

We designed and described a plausible decentralized architecture that broadly combines above themes with reference to a social media sharing application. In contrast to the traditional approach of realizing an application as a Web service that mediates interactions between its users, here each user is represented by its own agent who interacts with other agents based on an understanding of applicable norms.

The Scene

Alice is a person sharing her photos on the social network FaceUnBook running this app in her browser and using her pod to host her data and her agent to manage her photo sharing preferences (among other tasks). **B**ob is a friend of **A**lice and is following her activity stream on FaceUnBook. His agent is in charge of collecting the activity of all his friends, which will be summarized in the pod of **B**ob. This summary can then improve discovery when **B**ob uses the FaceUnBook app, which is backed by the personal query engine of Bob. It combines all search capabilities it can identify to plan and solve this aggregation query. **A**lice attended the seminar 23081 at Schloss **D**agstuhl. Among other things the agent of that organization enforces the norms of the organization that are stored in the organization's pod. The agent also collects pictures taken by the Camera42. This happens in **E**urope which is an organization with norms. The agent **G**DPR agent is in charge of enforcing the GDPR norms for every member of Europe. To make all members in Europe findable by the GDPR agent, Europe has a norm saying that all pods in Europe must be registered to Europe's



pod. **F**riedrich is a kid accompanying Alice in Dagstuhl. The agents, pods and resources in general are hosted on different platforms but discover and interact with each other through the hypermedia fabric and its standards. To support the discovery of resources, pods and their contents, **Q** is a query service available in the environment as one possibility to read and write within the hypermedia fabric, used by either software agents, or by human agents through client-side apps. This scenario follows the empodiment principle i.e. the agents communicate through their pods (pody) both through the inbox container of the pody or through other containers of the pody.

An Interaction to Share Photos May Process as Follows:

Alice took photos at **D**agstuhl, in which herself and **F**riedrich are depicted. She runs the FaceUnBook application in her browser, and uses it to share the photo with her friends. The application posts the photo in **A**lice's pod, and (HTTP) posts a notification in **B**ob's inbox (a specific container in his pod). **D**agstuhl publishes on its pod ODRL policies about photo sharing that apply to all seminar participants. **A**lice tags the photo with the tag "Dagstuhl Seminar 23081", then the FaceUnbook application also reads the policy and asks **A**lice to confirm that she complies with that policy (including the rule that no photo of a children must be published without consent from a caretaker – in that case, Alice is the caretaker and consents). Then the application (HTTP) posts a notification in **D**agstuhl's inbox. **B**ob likes **A**lice's photo; he improves the contrast in an image editor, and republishes it on his own pod (notifying **B**ob's friends, including **A**lice). **A**lice's agent notices that the republishing violates **A**lice's preferences and (HTTP) posts a warning to the pods of **A**lice and **B**ob.

Moving Forward

People in the group have independently worked on different aspects of the above architecture. A way forward would be to actually realize this application as broadly outlined above, see how things fit together and identify the gaps. Although in the above description **A**lice acts in accordance with the norms, in another situation, she may act in violation of the norms, e.g., if she posts a photo of a child who needs medical attention. FaceUnbook must not prevent such sharing but must support recording her violation, based on which the child's parent, for example, may demand an explanation from her. This brings in accountability and a richer notion of social processes than currently possible.

5.2 Abstractions for Agents on the Web

Andrei Ciortea (Universität St. Gallen, CH)
Catherine Faron (Université Côte d'Azur – Sophia Antipolis, FR)
Jomi Fred Hübner (Federal University of Santa Catarina, BR)
Eoin O'Neill (University College Dublin, IE)
María Poveda-Villalón (Technical University of Madrid, ES)
Alessandro Ricci (Università di Bologna, IT)
Antoine Zimmermann (Ecole des Mines – St. Étienne, FR)

This document reports on the findings of a Working Group at the Dagstuhl Seminar on *Agents on the Web* relative to what abstractions must be considered when designing multi-agent systems on the Web.

We first consider the case of a single agent that makes use of the Web, consuming or producing information. Then, we examine the case of multiple agents that work together as a system where the Web plays the central role of the environment in which they are situated. Finally, we provide preliminary ideas on how to instantiate the elicited abstractions in practice, relying on Semantic Web models and technologies and the Solid Protocol for Social Linked Data.

5.2.1 Introduction

Agents that are (inter)acting autonomously on the Web and with the Web must make use of the information available in this environment to gather knowledge and make decisions. Different kinds of information correspond to different types of abstractions that we want to formally identify such that they can be represented explicitly and systematically. With such representations of relevant abstractions, artificial agents can replicate activities on the Web that have been so far reserved to human agents: navigating hypermedia environments with a purpose, choosing what next links to follow based on promising signifiers, and collaborating with one another via collaborative Web platforms.

In what we consider here in the context of agents on the Web, an abstraction is a notion that can be defined as a formal structure (such as a mathematical object) which is relevant for a system of autonomous agents to function adequately. If the agents are humans, it is usually not necessary to formalise these abstractions because people have many ways of getting an intuitive understanding of ways to cooperate. For instance, when a community of people discusses issues on a Web forum, the fact that there is a coloured rectangle with the words "New thread" is sufficient to alert humans that this is a button for starting a new textual discussion. When one sees a blue underlined word that says "Reply" under the last message of a thread, one easily understands that this is a link that allows participants to reply to the thread, or to the last post. On the contrary, artificial agents may not be able to interpret these visual cues. If we would like to allow such agents to participate in a discussion, it is possible to hard code the interactions, e.g., a developer translates the actions needed to read and post messages on a specific forum. However, this is very limiting in terms of autonomy. Instead, we would like to define more abstractly what generic notions have to be known by the agent in order to choose its interactions with Web resources. As an example, online discussions can be characterised by the fact that there are discussion threads, each composed of individual messages that one can respond to. Messages are authored by someone (or something) at a date and time. These concepts are common to any Web forum, to blogs and microblogging platforms, to Git issues, chats, etc.

Once identified, the concepts that pertain to one type of abstractions can be formally defined as terms of an ontology, further described and constrained by logical axioms. In turn, a web platform can make use of such terms to describe itself for autonomous agents. The aim of this document is to report on the findings of a Working Group at the Dagstuhl Seminar on *Agents on the Web* relative to what abstractions must be considered when designing multi-agent systems on the Web. Consequently, this can serve as a starting point to decide what ontologies may be used to describe where, what, how resources are to be used by autonomous agents, for themselves or for a collective goal.

To do this, we start by considering the case of a single agent that makes use of the Web, consuming or producing information. In this first case, the Web is only used as a tool that is assumed to evolve independently of all agents (Section 5.2.2). Then, we examine the case of multiple agents that work together as a system where the Web plays the central role of the environment in which they are situated (Section 5.2.3). Then, we provide preliminary ideas on how to instantiate the abstractions in practice (Section 5.2.4).

5.2.2 Single Agents Interacting with the Web

Before considering the more arduous challenges of coordinating multiple agents on and via the Web, we first tackle the case where a single agent makes use of the Web as if it was a large warehouse full of documents and tools. The agent may exploit the Web in getting information from it: from the Web of documents, the Web of data, or the Web of knowledge. From this, it can derive or update its own beliefs. Or the agent may act upon resources leading to altering the state of the Web itself (e.g., posting a new message adds text to a Web page), or altering the state of the physical world itself (e.g., moving a robotic arm that offers a Web interface using Web of Things technologies).

To be fully autonomous on these types of tasks, Web agents must start their exploration from somewhere. At this point, there must be indications of where to find relevant information. This should lead them to online platforms where they may ask themselves: *What can I do on this platform? How can I do it?* Since the Web is evolving, agents may also be interested in changes occurring, without constantly checking for differences. An agent can be partly or fully guided by hard-coded knowledge provided at implementation time, such as the URIs of a preselected set of resources. However, in general, the agent has to interact with an entry point that serves as a hub towards any relevant resources. Such hub can be a search engine, a query endpoint, etc. that provides, at the minimum, links to resources that match the agent's request. Usually, the resource that the agent is looking for is not atomic: it corresponds to a compound resource that gives multiple options to the agent. For instance, the agent may be looking for online forums. Each online forum is a resource that provides discussion threads, search functionalities, posting abilities, etc. We call the abstraction of such compound resource a *platform*. The functionalities enabled by a platform and that the agent can use are affordances, which can be indicated to the agent via signifiers [10].

In summary, some competency questions must be answerable using the right abstractions, as follows:

- *Where do I find relevant resources?* Entry hubs, links to platforms.
- *What can I do on this platform?* Signifiers, links to affordances.
- *How can I do it?* Action possibilities, links to Manual.
- *How can I know what changes?* Notifications.

5.2.3 Multiple Agents Interacting on the Web

In the case of a single agent interacting with the Web, all online resources are considered as artifacts or tools that can evolve according to an independent life cycle. When agents need to collaborate, cooperate, or at least behave with awareness of each other on the Web, additional abstractions need to be introduced.

The concept of situatedness with regards to Web agents in a *multiagent* context can be defined in relation to the scope of environment that they exist within. As defined in Section 5.2.2, any tools that an agent utilises in its goal-directed behaviour provides a level of situatedness as each interaction will be between two entities that can be identified using the global, unique naming scheme provided by the use of IRIs. This also pertains to agent-to-agent interaction as when agents collaborate and coordinate in order to achieve system-wide, or individual, goals they interact via the Web, with each agent being identifiable by a unique IRI.

In order for collaboration and coordination between Web agents to be a possibility in a multiagent context, the embodiment of an agent within a hypermedia environment is a crucial abstraction that needs to be defined. This allows for the agent to maintain a presence in a Web context which enables other agents to be aware of its existence. However, agents can be embodied in multiple Web contexts, the same way a human would have multiple profiles on different social media platforms. This embodiment of an agent in a particular Web context can represent an identity of a Web agent. The agent's profile can be a certain persona that this particular agent wants to portray within that context. This abstraction can contain the agent's preferred methods of interaction via its *communication interface*, which can be defined as or subject to a set of *norms*. These *norms* can be defined within the hypermedia environment, as resources, that can be directly associated with the Web agent's embodiment.

When discussing multiple Web agents in a hypermedia environment, the scope of the environment is a relevant topic due to the computational limitations of the agents themselves. As a result, an abstraction for specific collections of resources can be viewed as an aggregation of related resources. This abstraction can be viewed as an *area* within a Web-based environment. If we take for example a set of related resources that represents elements of a building (Rooms, Floors etc.), these can be aggregated into an *area*. This example makes logical sense as the relation can be easily visualised and the relations envisioned, however there can be Web resources that are related, but in a less explicit manner and so this abstraction can provide a general way of aggregating resources to provide agents with scope as to the environments they inhabit.

5.2.4 Instantiating the Abstractions

The Semantic Web relies on the Resource Description Framework (RDF), a graph model to structure data by expressing relations between entities, and on RDF Schema and the Web Ontology Language to represent the ontologies used in RDF knowledge graphs, thus providing semantics to them. The Linked Data principles are a set of best practices to publish RDF data on the Web, namely: 1) use URIs to name things; 2) use HTTP URIs so that names can be looked up; 3) describe things using standards (RDF) so useful information is provided for URIs; and 4) include links to other URIs in things descriptions. Ontologies and linked data together provide the means by which an agent can reliably interpret resources described on the Web, whether they are digital resources or real-world resources. Additionally, with links, a web resource leads to other resources, and so forth, so as to make agents aware of the environment that the Web constitutes.

Besides, the Solid Protocol [2] aims at decentralising personal data management in such a way that Web users regain ownership and control over their data. At the core of it, there is the Solid *pod* (**p**ersonal **o**nline **d**ata store) that hosts the user's data and is implemented as a Linked Data Platform [9] with access control on top of it. Solid pods can host any kind of data but are designed in particular to easily manage RDF datasets with fine-grained read/write operations. Overall, the Solid Protocol specifies authentication, storage, access control, and interactions that must be implemented by Solid pods and Solid platforms in order to interoperate with each other and with applications that build on them.

Solid pods containing RDF knowledge graphs can be used to implement the abstractions introduced in Section 5.2.3. In order to represent the different levels and types of knowledge defining those abstractions it is necessary to combine a number of ontologies. For example the Web of Things model [7] might be used to describe the affordances while the Open Digital Rights Language (ODRL [6]) would be useful to define access policies to data or resources. Other existing ontologies might be extended or specialised to represent particular needs in the agents domain like the Organization ontology [8] or sensor and actuator related ontologies as SSN&SOSA [5] and SAREF [4]. Finally, specific domain ontologies related to the agent's tasks should be develop or reused. The Solid protocol states that "an agent is a person, social entity, or software identified by a URI; e.g., a WebID denotes an agent". We then assume that such a URI would dereference to an entry point for the data pod of the agent, where an **Agent Description** would be provided as an RDF graph, in addition to the mandatory credentials for authenticating the agent. We call the Solid pod implementing an agent's Web body a *pody* [12].

5.2.5 Conclusion

The dominant view in AI research is that intelligence is defined in relation to the environment an agent occupies [11]. The agents we have on the Web today also reflect the nature of their environment: they mostly solve the problem of finding and curating information in a Web of Documents. The Web, however, was designed to provide different levels of abstraction – going from computer networks to a knowledge-level representation of the world [1]. With recent standardisation efforts for Linked Data and subsequently the Web of Thing, the Web now extends to the physical world and provides agents with a uniform, knowledge-level hypermedia fabric that allows them "to browse and manipulate reality" – a vision that can be traced back to the early days of the Web.[23] This evolution unlocks new practical use cases for more intelligent agents on the Web, but such agents have to be provided with a proper level of abstraction that allows them to discover and interact with one another – and with the world in general. This report presents an initial proposal for such a set of abstractions with a focus on *situatedness* and *embodiment*. We invite the research community to join us in further investigating and developing these abstractions.

[23] See the keynote of Sir Tim Berners-Lee at the First International Conference on the World Wide Web (WWW'94): `https://videos.cern.ch/record/2671957`

Acknowledgements

Andrei Ciortea and Antoine Zimmermann were supported by funding from the HyperAgents project (grant from French national research agency ANR-19-CE23-0030-01 and from Swiss National Science Foundation No. 189474). María Poveda-Villalón was supported by the Spanish National project KnowledgeSpaces: Técnicas y herramientas para la gestión de grafos de conocimientos para dar soporte a espacios de datos (PID2020-118274RB-I00). Eoin O'Neill is supported by the Science Foundation of Ireland Strategic Partnerships Programme (16/SPP/3296) and is co-funded by Origin Enterprises Plc.

References

1 Tim Berners-Lee. Levels of Abstraction: Net, Web, Graph. Design Issues, 2007. `https://www.w3.org/DesignIssues/Abstractions.html`

2 Sarven Capadisli, Tim Berners-Lee, Ruben Verborgh and Kjetil Kjernsmo. Solid Protocol, W3C Solid Community Group working draft, 2022. `https://solidproject.org/TR/2021/protocol-20211217`

3 Sarven Capadisli and Amy Guy. Linked Data Notification. W3C Recommendation, 2 May 2017. `https://www.w3.org/TR/2017/REC-ldn-20170502/`

4 Laura Daniele, Raúl García-Castro, Maxime Lefrançois and María Poveda-Villalón. SAREF: the Smart Applications REFerence ontology. ETSI TS 103 264, v3.1.1, 2020. `https://saref.etsi.org/core/v3.1.1/`

5 Armin Haller, Krzysztof Janowicz, Simon Cox, Danh Le Phuoc, Jamie Taylor and Maxime Lefrançois. Semantic Sensor Network Ontology, W3C Recommendation, 2017. `https://www.w3.org/TR/2017/REC-vocab-ssn-20171019/`

6 Renato Ianella and Serena Villata. ODRL Information Model 2.2, W3C Recommendation, 2018 `https://www.w3.org/TR/2018/REC-odrl-model-20180215/`

7 Sebastian Kaebisch, Michael McCool and Ege Korkan. Web of Things (WoT) Thing Description 1.1, W3C Candidate Recommendation Snapshot, 2023. `https://www.w3.org/TR/2023/CR-wot-thing-description11-20230119/`

8 Dave Reynolds. The Organization Ontology, W3C Recommendation, 2014. `http://www.w3.org/TR/2014/REC-vocab-org-20140116/`

9 Steve Speicher, John Arwe and Ashok Malhotra. Linked Data Platform 1.0, W3C Recommendation 2015. `https://www.w3.org/TR/2015/REC-ldp-20150226/`

10 Danai Vachtsevanou, Andrei Ciortea, Simon Mayer, and Jérémy Lemée. Signifiers as a First-Class Abstraction in Hypermedia Multi-Agent Systems. In Proc. of the 22nd International Conference on Autonomous Agents and MultiAgent Systems, AAMAS 2023.

11 Michael Wooldridge. Intelligent Agents. In *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*, MIT Press, 2000. p27–72.

12 Antoine Zimmermann, Andrei Ciortea, Catherine Faron, Eoin O'Neill and María Poveda-Villalón. Pody: a Solid-based Approach to Embody Agents in Web-based MAS. To appear in Proc. of Engineering Multiagent Systems, EMAS 2023.

5.3 Representation Formats Supporting Goal-oriented Decision-Making on the Web

Cleber Jorge Amaral (Universidade Federal de Santa Catarina, BR)
Samuele Burattini (University of Bologna, IT)
Jean-Paul Calbimonte (HES-SO Valais – Sierre, CH)
Rem Collier (University College Dublin, IE)
Matthias Kovatsch (Siemens Schweiz AG – Zug, CH)
Mahda Noura (TU Chemnitz, DE)
Daniel Schraudner (Universität Erlangen-Nürnberg, DE)
Danai Vachtsevanou (Universität St. Gallen, CH)

5.3.1 Overview

One of the key outcomes from this working group was a consensus on the importance of identifying a general representation format to enable both interoperability between different Multi-Agent System (MAS) implementations and adaptability to unexpected differences in the environment for agents that can engage in goal-oriented decision-making on the Web. In the first three days of the seminar, this working group focused on aligning the necessary terminology, investigating the definition of high-level goals, identifying different use cases and decision-making procedures, and establishing the required abstractions for supporting plan execution. On the fourth day, this working group further merged with working group 5 which was focusing on the synthesis of MAS, with a particular focus on strategy identification for agents on the Web. A brief summary of the broad range of topics discussed is provided in the following.

5.3.2 Terminology Alignment

Early in the working group discussions, it became apparent that there was a divergence in the terminology used. This section outlines key terms along with definitions that were agreed by the group.

- *Environment:* Generally viewed as the space that an agent inhabits. Environments can be physical, virtual, or a combination of the two. In MAS, environments are seen as being first-class abstractions in that they are as important as the agents that inhabit them [2]. In this working group, given the scope of the seminar, the environment was considered to be a set of resources that are published on the Web; are accessible through the Web; and are linked to one another (where relevant) via hyperlinks. Those resources could relate to physical (e.g. sensors, actuators) or virtual (e.g. databases, services) resources.
- *Workspace:* In general, a workspace is part of an environment that is oriented towards a specific (set of) tasks or activities. Workspaces are an intermediary layer that adds structure and organisation to the environment. Agents typically operate across one or more workspaces. For the working group, a workspace was viewed as a set of resources that are somehow related (e.g. the resources relating to a smart room or building) and should be monitored by agents operating in that workspace.
- *Situatedness:* This refers to the relationship between an agent and its environment. Situated agents are able to interact with their environment via sensors and actuators. Situated agents are considered to be closely coupled to their environment, and their

behaviours can often be defined in terms of their effect on the environment. In terms of the working group, agents are situated in the Web environment described above. Agents are able to identify resources through Uniform Resource Identifiers and are able to interact with them using HTTP operations.

- *Artifact/Thing: Artifacts* are an abstraction of the non-agent entities that exist within an environment. The concept arises from the Agents & Artifacts meta-model [2]. Artifacts typically adhere to a well-defined interface that can be used by agents. Artifacts can be anything (e.g. an diary, a shared whiteboard, or a web browser). In terms of the Web, one possible application of artifacts is to use them as abstractions of resources [3], providing a enabling medium for agent-resource interaction. In such an approach, the artifact would be responsible for exposing the state of the resource and providing the necessary functionality to interact with the resource. In contrast, a *Thing* is a specific type of environment entity; typically a physical sensor / actuator device. The term comes from the *Internet of Things (IoT)* community. One refinement of this view comes from the *Web of Things (WoT) community*, where things are exposed on the Web as a resource. Access to sensor data or control functions is exposed through the Web, and the interface is usually described by an associated *Thing Description.* In this sense, a Thing can be viewed as a stylised Web resource whose representations are constrained by agreed standards.
- *Action/Operation:* Agents perform *actions* in the environments they inhabit. They do so by invoking actuators that implement the primitive mechanisms for affecting the environment. Agents choose the action they wish to perform through a variety of approaches, such as stigmergy, reinforcement learning or symbolic (logical) reasoning. In the context of the working group, actions was taken to generally refer to the submission of HTTP Requests to Web resources. An *Operation* is a type of actuator that is associated with the Agents & Artifacts meta-model. It is part of the well-defined interface described earlier. Each artifact has an associated set of operations that represent the different ways that an agent can interact with the artifact. In terms of the working group, it was agreed that agents perform actions by invoking operations on artifacts. Further the set of operations associated with an artifact representing a Web resource would be the set of valid HTTP Requests that can be used with that resource.
- *Effects/Pre-/Post-Conditions:* In order for some types of agent to decide what action to perform next, they need a description of the context in which an action can be performed and possibly description of the impact/effect of that action. The terminology for describing these descriptions is commonly drawn from early work on planning and they are often defined in terms of logical formulae, but this is not a requirement. The context in which an action may be performed is often defined as a partial description of the state that the environment must be in for the action to be applicable. The action context is often defined as a set of *pre-conditions* that is part of an action description that also includes unique identifier for each action. In addition to providing a context in which the action is applicable, many action descriptions also provide additional information about the *Effect* of the action. This is often described in terms of the changes that the action makes to the environment (both addition and removal). Effects are also known as *Post-conditions.* The action descriptions are developed for specific scenarios as part of the domain model. In the context of this work package, pre- and post- conditions are expected to be defined in terms of the current state of one or more resources and the changes that a given HTTP Request will have on that (set of) resources. How to express a state of multiple web resources was not completely clear.

- *Goals:* A goal is generally viewed as a future state of affairs that the agent (or system) is trying to achieve. Achieving a goal involves the creation of a plan of action. A plan is a set of actions whose cumulative effect is the achieve the goal. The task of constructing a plan of action to achieve a goal is known as planning. Goals that identify future states that the agent wishes to achieve are broadly known as declarative goals. This type of goal contrasts with the idea of procedural goals – goals that encapsulate some behaviour that is to be achieved. For example, the task of paying a bill assigned to an agent can be expressed declaratively via the goal <paid bill> or procedurally via the goal <pay the bill>. The former refers to the state of having paid the bill and the later refers to the procedure of paying the bill. Generally, planning involves the use of declarative goals as it allows the planning problem to be expressed as a search through a space of possible states. States are represented as nodes, and actions are associated with edges that transition the system from one state (node) to another (node) as defined by the pre- and post- conditions of the action description. In terms of this working group, much of the discussion focused on goals in a more abstract sense or around declarative goals.
- *Global Guidance:* Global guidance refers to the creation of high-level plans to achieve domain specific goals. It is primarily concerned with goal refinement and is expected to utilise planning techniques in conjunction with any domain models that have been developed for the scenario. Largely, global guidance is concerned with the strategic decision-making activities of the deployed system. The idea of global guidance is refined further in section 5.3.4.
- *Local Guidance:* Local guidance refers to the refinement of high-level plans based on the context in which the agent is operating. It is primarily concerned with the reification of the abstract domain model derived plans with the actual configuration of the environment. In contrast with global guidance, local guidance is primarily concerned with tactical decision-making and is also discussed further in section 5.3.4.
- *Protocols:* Protocols define structured interactions between agents. They define a set of messages that should be sent between two or more agents with a specified ordering that can be global or partial in nature. Protocols are a building block for defining valid patterns of interaction between agents. In the context of this work package, the existence of protocols was assumed, but was not considered in detail.
- *Organisation:* In Multi-Agent Systems research, an organisation is broadly defined as a collection of agents that work together within an agreed context that is based around a set of well defined roles with clear lines of communication and responsibilities. Organisations apply structure to a Multi-Agent System and agents can play different roles within that structure at different times depending on their own goals and objectives. In the context of this working group, the function of an organisation was not discussed extensively.
- *Norms:* Norms define the standard behaviour of a system (e.g. all cars going in the same direction drive on the same side of the road). In societies, norms often emerge from the behaviour of the individuals within that society (everybody else is doing it, so should we). Norms can be formalised as laws, with an associated penalty, that can be used to enforce compliant behaviour. Like organisations, norms can be used to provide additional structure around the behaviour of agents. In some respect, norms can be seen as a less rigid than organisations because that provide a kind of social guidance that each agent can either adhere to or not. In the context of the working group, there was little discussion around the use of norms.

The provided definitions enabled us to discuss primary abstractions that are commonly encountered in a MAS. Such abstractions have been identified as potential candidates for representation in a common format towards facilitating the interoperability and adaptability of agents on the Web.

5.3.3 Use Case

In this section, we present a scenario to motivate the definition and representation of proper abstractions to support agents' interactions in hypermedia environments. Given that agents can execute a variety of tasks that call for coordination and communication among them, and that the group was interested in challenges posed by a real-world scenario with agents interacting through the Web with an evolving physical environment, a building automation scenario can provide a suitable environment for testing a concept for modelling abstractions of MAS.

In such a scenario, a human user may have a set of preferences relating to environmental conditions stored in a personal Solid Pod[24]. Agents in the MAS can use the data from the Pod to adopt goals towards increasing the user's comfort with respect to the given set of preferences. In the most simple scenario, we can imagine having a single agent managing a room in the smart building, identifying users, retrieving information from the Pods of the identified users, and striving to implement the desired environmental preferences. Of course, the agent might also need to adapt its behavior based on constraints about energy consumption, or towards supporting users' productivity or health and safety, and so on.

Given a generic specification of the user comfort level as a goal, the agent will be delegated to navigate the hypermedia environment and find the WoT Thing Descriptions of the things it can interact with and adjust the conditions accordingly. Table 1 lists some things and services the agent might need to interact with to achieve its goal.

Table 1 Things and services that allow to modify the comfort level in a smart room scenario.

Thing	Description	Illustrative use
Temperature sensor	Provides the temperature of the room (e.g. in Celsius Degrees)	An agent can find in the thing's description how to read the temperature sensor of the room, monitor it, and possibly actuate over things such as the air conditioning to achieve a target temperature.
Humidity Sensor	Provides the relative humidity of the room	In order to attain a target humidity, an agent can find instructions in the thing's description for how to read the room's humidity sensor, monitor it, and possibly actuate devices like the air conditioner.
Illuminance sensor	Provides the illuminance of the room (e.g. in Lumens)	In the thing's description, an agent can find how to read the room's illuminance sensor, monitor it, and possibly act on items like the lighting and window blinds to reach a desired level of illumination.
Lighting Bulbs	Actuator that affects the illuminance of the room	The power over this device can be adjusted over a range of values by an agent to increase or reduce its effectiveness. This will change the amount of light in the room.
Window Blinds	Actuator that can cover or uncover a window affecting the illuminance and visibility of the room	This actuator can be set in a range of values by an agent for changing the lighting and visibility in the room, opening or closing the covering.
Weather service	Provides the forecast for temperature and humidity in the region in which the room is located	This service can provide the forecast to the next hours. This can be used by an agent that has a threshold in energy consumption which may decide to use natural light for some time even reducing the human comfort (e.g. due to sun glare)

[24] https://solidproject.org

Of course, there are other ways to implement this use case without having a single central agent capable of managing everything. We could for example have different personal agents representing different users, that will try to negotiate the best environmental condition for each user. For instance, for finding a common sense for different preferences of multiple human representative agents can adopt an existent protocol for auctions [1]. Also, an agent representing the room's provider might be present, and have its own preferences (e.g. a certain threshold in energy consumption) in conflict with the comfort level thus they should be negotiated and prioritized according to some extra definitions.

Another possibility may be to also have multiple agents controlling certain room conditions. In each room there might be agents in charge of adjusting the temperature, humidity, and illumination of a room, then the task would no longer require an agent to navigate the hypermedia environment directly looking for things, but instead to navigate the MAS to find which agents are responsible and interact with them. For instance, if the current room temperature is not as a human prefers, its representative agent can ask the agent responsible for controlling the room temperature to adjust it accordingly.

When the room is occupied by more than one agent, the interactions and coordination issues can be even more complex. In this scenario setup, the agents can interact with each other to properly coordinate their actions over the things of the room.

To monitor and control things, and to negotiate a suitable condition for the room, the agents must employ common abstractions to describe their knowledge, intentions, goals, and plans. Table 2 shows possible interactions between agents and things and among agents, as well as necessary abstraction that they must share.

Table 2 Kind of interactions and necessary abstractions.

Interaction	Abstraction description	Examples of circumstances
Agent representing a human / Environment described with Thing's Description	The agent must understand what is described in the Thing's Description.	(i) For understanding the value of the temperature, the agent must know in which unit it is being represented. (ii) For setting up the temperature the agent must know if a particular action would increase or decrease the power over a cooler or a heater.
Agent representing a human / Agents controlling room's conditions	The agents that are interacting must have a common understanding of perceptions, actions, desires and intentions.	(i) The agent that controls the temperature may increase the power of the heater to achieve a request made by an agent representing an human in the room. (ii) The agent that controls the humidity refuses the request of a human representative after realising that achieving the requested humidity conflicts with a higher priority target that is already set.
Agent representing a human / Agent representing another human	The agents that are interacting must negotiate the final desired outcome. Therefore, they must have a common understanding of relevant interaction protocol.	(i) The agent that controls the temperature may reply to a second human representative agent that it will set the temperature for the average of these two, or they must negotiate another target.

5.3.4 Strategic and Tactical Planning

For this section, we assume that we already have a way of expressing goals for agents as well as actions, which agents can take, and their effects on the environment at an abstract level. We will call this a domain model. An example of an abstract representation of an agent's goal in the domain of building automation would be "I want to make the room brighter". The domain model would then comprise all possible actions and their effects, e.g. "I can set the power of light bulbs to adjust the brightness" or "I can lift the blinds of a window in order to adapt the brightness of the room to the outside brightness".

Agents that want to achieve such a goal in hypermedia environments like the Web usually have to make plans on two different levels. We call them strategic planning and tactical planning. Strategic plans are plans on a high level of abstraction, and they are made a priori, have a global scope, and are based on a domain model (which could e.g. captured with an OWL ontology). A strategic plan in the given domain for the goal "I want to make the room brighter" could, for example, be "I need to find a light bulb in the room, then I can adjust its power". We make no assumptions about the mechanics of the plan creation or acquisition here (it could be classical PDDL planning, but it could also be a neural network generating the plan).

To actually carry out a plan, an agent has to interact with its environment. In the case of hypermedia environments, these interactions usually are either following a link or submitting a form to a server. This means that agents have to refine their strategic plan to be able to execute it. Agents employ tactical planning to refine their strategic plans. Tactical plans, as opposed to strategic plans, are made at run time, have a local scope, and are based on what the agent found out about the environment (e.g. based on WoT Thing Descriptions that have been found). An example of a tactical plan would be "I send an HTTP POST request with the payload '`<> rdf:value 10 .`' to http://ex.org/room1/bulb4/properties/power". A comparison between strategic and tactical plans can be seen in Table 3.

We need the separation between the two levels of plans because hypermedia environments are usually highly dynamic environments and (strategic) plans that are too detailed get obsolete very fast.

The environment can help agents in creating tactical plans by providing metadata about the available links and forms that the agent can understand (e.g. "If you follow this link, you will get a list of all devices in the room"). The environment could even take a more active role in guiding an agent by presenting them only links and forms which the environment knows that they are useful to the agent (e.g. "I know you are an agent for handling the room brightness, thus I will only show you links to the devices that can influence brightness").

What parts of a plan should belong to the tactical side and what to the strategic side is not fixed but lies on a continuum. The exact position on the continuum depends on the use application scenario. At one end of the continuum, plans in agents would be hard-coded: the agent already has a very detailed "strategic" plan of what exact steps to carry out in which order (first send this HTTP request, then that HTTP request) based on the domain model before it starts executing it. This works well in a static and deterministic environment, and it is very efficient there, but hard-coded plans are not flexible and cannot adapt to unforeseen changes in the environment at all.

On the other end of the continuum, agents do their best effort link following: the agent has no strategic plan at all (maybe because it does not have knowledge about the domain) and just follows the links and submits the forms that at the moment seem most useful to them; it just uses tactical plans. This approach is very flexible (as it makes no assumptions about the environment) but usually also not very efficient.

For the design of Hypermedia MAS, it clearly is a challenge for system designers to determine where on the described continuum the different agents should be located. It will also be important to find out, how the transitions between the two planning tactics work and how they influence each other.

Table 3 Comparison of the different aspects of strategic and tactical planning.

	Strategic Planning	Tactical Planning
Time	A priori	At run time
Scope	Global	Local
Based on	Domain model	Environment
Focus on leads to	High efficiency, low flexibility	Low efficiency, high flexibility

5.3.5 Designing MAS for Strategic and Tactical Planning

The discussion about strategic and tactical planning presented in the previous section moved our focus from the external data that should be made available to the agent to the internal decision-making processes. We believe that a better understanding of the patterns that agents could use to reason and adapt to changes in their environment is crucial to also identifying how to sufficiently describe such environments with a suitable representation.

We then decided to join our efforts with the working group focused on synthesis in MAS since both groups shared an interest in the activities of the other and felt they were going towards similar objectives.

We ended up with a shared view of the main abstractions that are needed to design a MAS in order to account for many possible strategies to refine tactical planning. We named such strategies *Course Check and Revision Strategies* (CCRS) which are further described in our shared report.

5.3.6 Conclusion and Planned Work

Our initial focus on examining representation formats for describing goals and environments led us to explore ways to enable heterogeneous agents to collaborate effectively on the Web and to enhance agents' autonomous decision-making through tactical planning. Eventually, we shifted our attention towards designing a system that accounts for divergences between the design time abstraction of an environment and its run-time implementation.

The collaboration between our team and the working group on the synthesis on MAS has proven to be fruitful. We are pleased with the outcomes of our joint efforts and intend to pursue this research direction further. Specifically, we aim to investigate the use of CCRS in agent architectures, and how these strategies can be supported by detailed descriptions of the hypermedia environment, the goals of a MAS, and the other agents present within the system.

References

1 Fabio Bellifemine and Agostino Poggi and Giovanni Rimassa. *Developing multi-agent systems with a FIPA-compliant agent framework*. Software: Practice and Experience, 2001

2 Andrea Omicini and Alessandro Ricci and Mirko Viroli. *Artifacts in the A&A meta-model for multi-agent systems*. Autonomous Agents and Multi-Agent Systems, 17, 432-456, 2008.

3 Andrei Ciortea, Olivier Boissier, Alessandro Ricci. *Engineering World-Wide Multi-Agent Systems with Hypermedia*. International Workshop on Engineering Multi-Agent Systems, 285-301, Springer, 2018.

5.4 Course Check and Revision Strategies for Autonomous Hypermedia Navigation

Cleber Jorge Amaral (Universidade Federal de Santa Catarina, BR)
Samuele Burattini (University of Bologna, IT)
Jean-Paul Calbimonte (HES-SO Valais – Sierre, CH)
Rem Collier (University College Dublin, IE)
Andreas Harth (Fraunhofer IIS – Nürnberg, DE)
Matthias Kovatsch (Siemens Schweiz AG – Zug, CH)
Brian Logan (University of Aberdeen, GB & Utrecht University, NL)
Simon Mayer (Universität St. Gallen, CH)
Mahda Noura (TU Chemnitz, DE)
Sebastian Schmid (Universität Erlangen-Nürnberg, DE)
Daniel Schraudner (Universität Erlangen-Nürnberg, DE)
Danai Vachtsevanou (Universität St. Gallen, CH)

5.4.1 Overview

This working group report sums up the activities of the joint sessions between working group 4 (Representation Formats) and working group 5 (MAS synthesis). The groups merged due to similar interests in the relationship of design-time plans and run-time adaptation to real hypermedia environment for autonomous software agents. An extended version of this work has been submitted to the 2023 ACM Hypertext Conference.

5.4.2 Introduction and Background

Supported by Web browsers and online services such as search engines and recommender systems, many people today excel at efficiently navigating the Web's hypermedia structure towards achieving their goals. The essence of navigating hypermedia is that a user is able to match a discovered – local – hypermedia control with their – global – application goal. For instance, the control `HTTP PUT /cart { "isbn" = "978-1452654126" }` should be matched with the goal to *buy the book "The Design of Everyday Things"*). Users accomplish this matching using information in the representation of the current Web resource – in the above example, this might be a button labelled *Add to Shopping Cart!* or a cart icon. The user hence needs to be able to find out that a specific local hypermedia control is indeed suitable to advance towards their goal; for buying a book, this is true for most users because the online shop has been modelled according to how people shop in the physical world, hence it is *natural* that the book should be "put into the cart" for enabling the user to, eventually, buy it.

From a technical perspective, this process is on the Web enabled by a mechanism that is part of the Uniform Interface constraint of the REST architectural style [1]: *Hypermedia as the Engine of Application State* (HATEOAS) constrains REST systems so that clients only perform hypermedia state transitions on resources using actions that are provided within the hypermedia that the server delivers [2]. With HATEOAS the only implemented capability of a hypermedia client should be the way to select which hypermedia control is useful to achieve the goal. This avoids tight coupling between clients and servers, and permits a hypermedia environment and its clients to evolve independently from one another.

While central to the way humans navigate the Web, the REST HATEOAS constraint is today not emphasized in the design of hypermedia environments that are roamed *by machines*. However, following larger academic emphasis on this topic in the wake of the Web of Things movement [3, 4, 5], industry and standardization groups have been gaining interest in HATEOAS over the past decade, for instance at the World Wide Web Consortium's (W3C) Web of Things (WoT) Working Group.[25]. This rising interest is driven by an increasing requirement on machine clients to more autonomously navigate hypermedia environments, and handle environment dynamics [6].

We argue that not least due to today's wide availability of powerful AI models that might be used by artificial agents to provide guidance in hypermedia environments, revisiting the role of HATEOAS and the matching process of high-level domain goals of agents with low-level hypermedia controls is required. HATEOAS, together with suitable local traversal strategies, could form a cornerstone to permit artificial agents to roam the Web – and use its services – similar to how people do it.

5.4.3 Strategies for Autonomous Navigation

In-line with research in the field of Autonomous Agents and MAS, we suppose that an artificial agent for a hypermedia environment is created by an agent designer according to a (design-time) model that the designer holds of the environment. This creation process involves the design and programming of the agent's logic towards achieving the agent's design objective. At run time, the agent is placed in a hypermedia environment and executes its programmed behavior. In environments such as the Web, it is however likely that the agent will encounter an environment that does not exactly match the expectations of the agent designer. The agent then either needs to *cope with* unexpected violations of the design-time assumptions (e.g., nonexistent hyperlinks or link relations); or it may *optimize* its way to achieve the goal if discovered hypermedia controls suggest better ways to achieve its design objective.

We argue that, to permit this level of adaptation to the environment, the agent should be equipped with strategies of evaluating which of a set of encountered hypermedia controls will enable it to realize its design objective.

We refer to such strategies as *Course Check and Revision Strategies* (CCRS). A CCRS describes what an agent does at run time to verify if the next step of its planned course of action is (still) possible and/or prudent given the available run-time information and the agent's objective. From an application perspective, CCRS can be either reactive or proactive: *Reactive CCRS* are designed to enable agents to cope with unexpected variations of the run-time environment that violate design-time assumptions. The agent hence is required to deviate from the original course of action; for instance, the agent designer might not have known that an agent would need to be logged in to access a specific resource – the agent in this case requires a CCRS that permits it to discover how to log in to the system at run time. *Proactive CCRS* permit agents to discover opportunities that it may choose to exploit to achieve its design objective in a better (faster/cheaper/more privacy-friendly/etc.) way. Following these opportunities, the agent then actively changes its designed course of action to optimize the path towards its design objective.

[25] `https://www.w3.org/WoT/wg/`

Table 4 Categories of CCRS with examples (non-exhaustive) based on the CCRS' main dependencies.

Category	**Depends on**	**Examples**
Independent	–	Fail, Random links, Retry, Backtrack
Environment-Oriented	environment features	Affordance-based, Stigmergic
AI-Oriented	computational resources, data availability, model/environment fit	Prediction, Reinforcement Learning, Planning, Reasoning
Socially-Oriented	presence of other agents / communication	Consultation, Delegation, Human-in-the-Loop

When creating an agent to reach an objective in a hypermedia environment, the agent designer may then equip its agent not only with the primary (sometimes even hard-coded) course of action – e.g., a list of URIs to access; a list of link relations[26] to follow; a list of W3C WoT Interaction Affordances[27] to exploit; etc. – but also with one or several CCRS among the ones presented below.

5.4.4 Course Check and Revision Strategies

In this section, we propose a list of CCRS that an agent designer might select from when creating agents that roam Web hypermedia environments – or other environments where HATEOAS-like feedback is available. We do not claim that this list is exhaustive; rather, it should provide a starting point for the creation of a catalog of CCRS while at the same time further illustrating our approach and connecting it to different areas within artificial intelligence and beyond. The choice of a specific CCRS depends on the amount and type of autonomy and flexibility that an agent designer intends to give to the agent it creates as well as on the specific abstraction of the environment that the agent designer holds at design time.

We group CCRS in four categories: *Independent*, *Environment-Oriented*, *AI-Oriented*, and *Socially-Oriented* which are briefly illustrated in Table 4.

5.4.4.1 Independent CCRS

Independent CCRS do not place any special requirements on the hypermedia environment and do not require specific information about available hypermedia controls.

5.4.4.2 Environment-oriented CCRS

Environment-oriented CCRS depend on specific features of the environment to recover from unexpected situations (when used reactively) or to optimize the hypermedia traversal of an agent (when used proactively).

[26] `https://www.iana.org/assignments/link-relations/link-relations.xhtml`
[27] `https://www.w3.org/TR/wot-thing-description11/`

5.4.4.3 AI-oriented CCRS

In AI-oriented CCRS, the agent is equipped with access to an AI (sub)system that it consults at run time to reactively resolve unexpected situations or to proactively optimize its traversal of the hypermedia environment. For these strategies to remain efficient, the agent designer needs to select a type of AI that is compatible with the resources in the hypermedia environment and that can readily use the information that is provided by these resources towards eluding the appropriate next hypermedia control to use for achieving the agent's design objective.

5.4.4.4 Socially-oriented CCRS

In socially-oriented CCRS, the agent relies on other agents to support its traversal of hypermedia. The supporting agents are considered to be autonomous entities rather than tools that the agent makes use of (we discuss this in more detail in the following Section 5.4.6). In socially-oriented CCRS, the agent designer needs to have sufficient confidence in the ability and willingness of other agents to support the traversing agent at run time, and such agents need to be present, capable to communicate between each other, and perceivable by the created agent.

5.4.5 CCRS in a Single Agent Scenario

We provide a simple yet sufficiently complex scenario to illustrate our proposal and the interplay between the creation of an agent at design time and its execution at run time. For this scenario, suppose that a designer creates an agent that it intends to task with switching on a lamp in a hypermedia environment and then leaving feedback about its usage of the hypermedia environment. At design time, the designer holds expectations about several aspects of the to-be-encountered run-time environment, and each of these expectations is connected to a confidence:

E1: The designer knows with full confidence that the environment's entry point URI is `https://mythings.example.org/room-d-12`.

E2: The designer knows that the entry point URI represents a thing directory, i.e. a list of hyperlinks to Web APIs of physical devices that are contained in a room. The designer is sure that there is only a single lamp in this list and believes that the hyperlink to the lamp's API is located on the second page of the directory. While the designer neither knows the URI of that page nor of the lamp's API, the designer believes that the directory exposes `next` link relations to support pagination.

E3: It is unknown to the designer what specific lamp the agent will encounter at run time, and hence it does not know about the API. The designer furthermore does not know whether the agent will be required to first log in before it can switch the lamp.

E4: The designer has high confidence that all hypermedia resources expose hypermedia controls for leaving textual feedback. The designer knows that the link relation that is exposed to hint at these controls is `leaveFeedback` and they know about the specific hypermedia control required.

The designer creates an agent to achieve the design objective based on this environment model: Based on (E1), it programs this agent to first send a GET request to `https://mythings.example.org/room-d-12`. Based on (E2), it programs the agent to, next, find a `next` link relation and follow it until the returned resource representation contains a hyperlink to the API of a *Lamp*. Based on (E3), it programs the agent to achieve the goal

Lamp On which it specifies declaratively using a suitable language (e.g., RDF[28]). Finally, based on (E4), the designer programs the agent to find a `leaveFeedback` link relation and use it to send a feedback text.

Because of its little knowledge or low confidence with respect to E2 and E3, the agent designer furthermore equips the agent with several CCRS to locally cope with deviations and/or optimize its traversal of the hypermedia environment:

- To cope with a resource not exhibiting the expected link relations in Step 2 and Step 4 of the process (i.e., `next` and `leaveFeedback`), the designer gives the agent an AI-Oriented CCRS like using a large language model for the ability to cope with unexpected vocabularies.
- For achieving the goal of switching on the lamp, the agent designer equips the agent with a CCRS that uses an AI planning system for locally computing a plan. At run time, this CCRS will succeed if suitable metadata is available, where typical planners require the pre- and post-conditions of using a hypermedia control to be specified; these can then be chained to lead the agent to achieve its goal.
- The agent designer knows that the planning strategy, while remaining a possibility, might fail for several reasons. For this reason, the designer includes another CCRS that uses uses Consultation/Delegation. The agent is hence programmed to contact a (hard-coded or locally discovered) resource – e.g., a specialized local lamp-switching agent or a human agent – when it requires help with achieving its goal of switching on the lamp.

When the agent now starts executing its program in the given hypermedia environment, it will engage none, any, or all its CCRS after each usage of a hypermedia control to possibly modify its future course of action. In this example, the CCRS depend on implicit or explicit environment support, e.g. on information being available about the environment for the *Planning CCRS*, and on other agents being present and capable for the *Consulting/Delegation CCRS*.

5.4.6 CCRS in a Multi-Agent Scenario

In this section, we investigate how our proposed type of increased local autonomy through CCRS impacts the behavior and design of a Multi-Agent System (MAS). In a MAS, several agents work towards the achievement of their individual goals [7]. The way these individual goals are related to each other defines whether the agents within the MAS are *cooperative* – in this case, a top-level goal is usually decomposed into smaller problems that are solved by agents that share results and communicate with each other. In a non-cooperative scenario, agents might have competing interests, but usually are expected to work rationally and towards achieving an equilibrium [8].

In our exploration we considered only a *cooperative* MAS expanding the scenario from Section 5.4.5 to one with two *interdependent* agents in two roles:

- The `Lamp Operator` is tasked to use hypermedia to turn on a lamp in a repository and then leave feedback.
- The `Heater Operator` is tasked to use hypermedia to turn on a heater and then leave feedback. It holds a key that, when submitted through HTTP POST to the `https://mythings.example.org/authorizeAll` endpoint, will authorize other agents to access the repository resource.

[28] `https://www.w3.org/RDF/`

In this situation, the MAS designer will create a *multi-agent plan* that requires the `Heater Operator` to first post the key to unlock the repository, then access the repository, and then search for the heater hypermedia control and actuate the heater. The `Lamp Operator` agent will be programmed to wait until it can access the repository and then proceed to find and use the lamp control. In our approach, the MAS designer equips the two agents with suitable CCRS to permit them to recover from unexpected run-time situations and to identify optimization potential at run time.

However, suppose that, at run time, the `Heater Operator` uses a CCRS to discover that there is a *direct access link* to the heater control in the environment that permits it to access that control *without first posting its key.* Although it might be prudent for this individual agent to optimize its course of action, this behavior would not satisfy the goal of the MAS, since the `Lamp Operator` would in this case not be able to complete its design objective.

5.4.6.1 Dealing with Multi-Agent Coordination

The use of CCRS in MAS is hence constrained in multi-agent plans that require coordination. Problems arise whenever an agent discovers a environmental *shortcut* through a CCRS that might be more rewarding for the agent but that leads to the skipping of a *coordination point* that is required for the overall MAS behavior. This effectively limits the freedom of an agent that cannot use its CCRS without considering the effects on the whole MAS, and the direct consequence is that agents either need to *refrain* from using CCRS or they need to *coordinate* with other agents whenever applying a CCRS that might impact other agents.

Several possibilities might be available, however thet will be largely dependent on the concrete scenario, including the abilities of the agents and the characteristics of the environment.

In our MAS scenario, the `Heater Operator` might for exaple follow one of these strategies upon finding a shortcut through a CCRS:

- If the designer *explicitly designed an organization for the MAS*, the agent may first look up the organizational specification for the MAS. This specification will in this case include the obligation on the agent to post the key, and the agent would be in violation of this if it continued following the CCRS.
- If the agent *can communicate with other agents*, it may follow the CCRS-suggested course of action, but first broadcast the key (and how to use it) to other agents so that they can use the key by themselves if needed. If no *direct* communication is possible but the environment permits creating shared resources among the agents in the MAS, the agent might leave the unlock information in the environment for other agents to find.
- If the designer creates an *explicit synchronization point* in the agent plans where the agents are programmed to meet, the agent may choose the CCRS-provided route to reach this point, but – if the other agent is not reaching the synchronization point – it might backtrack and repeat the part of the original plan that it skipped.

5.4.6.2 Dealing with Heterogeneous Agents

The agents might also have *heterogeneous capabilities* – different actions they are able to perform in the environment, different means of communication with other agents, different cognitive capabilities, etc. – and also *heterogeneous knowledge* about their design objectives and the environment – different vocabularies that they may understand, different data they have access to, etc. If it is not possible to capture these differences at design time in a comprehensive model, which may occur in open systems like the Web, agents should be enabled to adapt to such differences at run time to still be able to coordinate in the MAS.

Considering again the heater-and-lamp multi-agent scenario, the `Heater Operator` might decide to delegate the task of using the key to other agents in the environment since it wants to pursue a more direct plan that was discovered using its CCRS. In order for the agent to consider this delegation strategy, the designer must have confidence that other agents can (i) receive such messages, (ii) understand a shared vocabulary to express goals and plans, and (iii) be able to perform a similar subset of actions.

If any of these conditions are violated *at run time*, the delegating agent might realize that no agent is responding to the task it was trying to delegate and use its CCRS to further adapt to this circumstance.

References

1 Fielding, R.T. and Taylor, R.N., 2002. *Principled design of the modern web architecture.* ACM Transactions on Internet Technology (TOIT), 2(2), pp.115-150.

2 Ruby, S. and Richardson, L., 2007. *RESTful web services* (p. 454). Sebastopol, CA: O'Reilly.

3 Guinard, D., Trifa, V. and Wilde, E., 2010, November. *A resource oriented architecture for the web of things.* In 2010 Internet of Things (IOT) (pp. 1-8). IEEE.

4 Mayer, S., Inhelder, N., Verborgh, R., Van de Walle, R. and Mattern, F., 2014, October. *Configuration of smart environments made simple: Combining visual modeling with semantic metadata and reasoning.* In 2014 International Conference on the Internet of Things (IOT) (pp. 61-66). IEEE.

5 Mayer, S., Inhelder, N., Verborgh, R., Van de Walle, R. and Mattern, F., 2014, October. *Configuration of smart environments made simple: Combining visual modeling with semantic metadata and reasoning.* In 2014 International Conference on the Internet of Things (IOT) (pp. 61-66). IEEE.

6 Ciortea, A., Mayer, S., Gandon, F., Boissier, O., Ricci, A. and Zimmermann, A., 2019, May. *A decade in hindsight: the missing bridge between multi-agent systems and the world wide web.* In Proceedings of the International Conference on Autonomous Agents and Multiagent Systems.

7 Shoham, Y. and Leyton-Brown, K., 2008. *Multiagent systems: Algorithmic, game-theoretic, and logical foundations.* Cambridge University Press.

8 Axtell, R.L., 2002, July. *Non-cooperative dynamics of multi-agent teams.* In Proceedings of the first international joint conference on Autonomous agents and multiagent systems: part 3 (pp. 1082-1089).

5.5 Learning and Reasoning with Behavioural Norms

Mehdi Dastani (Utrecht University, NL)
Kemal A. Delic (The Open University – Milton Keynes, GB)
Jérôme Euzenat (INRIA Grenoble Rhône-Alpes, Saint Ismier, FR & Univ. Grenoble-Alpes, FR)
Stefano Mariani (University of Modena, IT)
Dave Raggett (W3C – United Kingdom, GB)
Munindar P. Singh (North Carolina State University – Raleigh, US)

Societies of agents are constantly confronting a changing world, environment, and society. They have to learn how to behave in such a context and to reason on how to exploit what they have learned. This covers learning actions that they can do (including independently or jointly), learning how others behave, and learning what is expected from them. Learning may take its source from the web, including the web of things, the behaviour of other agents, and combinations thereof. It can take advantage of diverse techniques (e.g. neural networks, embedding, reinforcement learning, Bayesian inference and Markov games, clustering, active learning, and inductive logic programming), some of them specific to agents, e.g. multi-agent reinforcement learning (MARL). Learning and reasoning will provide agents with the ability to more flexibly, dynamically, and better perform their tasks. Some challenges in this context come from articulated learning at the agent level and at the organisation level, dealing with implicit objects (only modelled in other agents) and explicit objects (that may be embodied in the web). As a typical example of this, we choose to focus on norms and more precisely behavioural norms.

This is a summary of discussions in the Working Group on Learning and Reasoning with Behavioural Norms. We chose to focus on norms as a topical area of research to focus our discussions, rather than trying to cover learning and reasoning more broadly.

5.5.1 Motivations and Bridges to other Working Groups

Societies of agents are constantly confronting a changing world, environment, and society. They have to learn how to behave in such a context and to reason on how to exploit what they have learned. Learning and reasoning connects well with the other topics discussed during this Seminar, in particular in connection with the following questions:

- How can we incorporate metadata about the data used for learning and reasoning, including metadata to determine its provenance and trustworthiness?
- How can we store and query the data needed for learning and reasoning in a decentralised manner, as in pods for individual agents?
- How can learning and reasoning methods accommodate models of actuators, including through affordances, in the design of an environment through which an agent can observe and control the environment?
- How can agents create and execute collaborative plans?
- How can agents interact with each other felicitously, through learning and reasoning about norms?
- How may flexible governance be accomplished by learning and reasoning about norms, especially to figure out the boundaries between competing norms in a contextual manner?

Because the topic is so wide, we choose to focus on one particular kind of objects: norms.

5.5.2 Learning and Reasoning about Behavioural Norms

Norms are patterns of behaviour for group members in which compliance or violations of the implicit or explicit rules may lead to sanctioning. Norms vary in their extent of explicitness or implicitness and may be prescriptive (including proscriptive) or descriptive. Prescriptive norms may carry the force of legal regulations and laws. Sanctions may be positive or negative and may apply to an individual or a group and be given by an individual or a group [3]. One example is for drivers of vehicles on public roads, self-driving vehicles and humans, who will all need to follow the same norms to get along smoothly. Some of these norms are explicitly described in the highway code, whilst others remaining implicit, or tacit, must be learned through experience. Sanctions may involve honking the horn or flashing the vehicle's headlights – besides actual fines for violating prescriptive norms, not solely descriptive ones.

On the web, where people and software agents meet, they both have to follow social norms. It is therefore interesting to consider groups involving combinations of people and agents. The agents are intelligent provided they can learn and reason as they interact with their environment, which includes other group members. Reasoning involves making decisions about actions with the environment, or changing the agent's beliefs, desires, and intentions. Beliefs may also cover the agent's understanding of people and other agents (as in the Theory of Mind modelling).

Some questions that arise:

- What is the relationship between norms and governance?
- Are we using agents to model people or are we investigating social norms for artificial agents?
- What are suitable metamodels for explicit norms?
- How can we describe behaviours in a symbolic or sub-symbolic form (e.g. using deep neural networks or other embedding techniques)?
- How do we revise or update the rules of individual agents that describe their behaviour?
- How can we model heterogeneous communities of agents with different social values?
- How can we avoid unfair norms that are discriminatory to certain classes of users and agents?
- How can we deal with hostile or malicious agents?
- How are agents supposed to know and learn explicit and tacit norms?
- How can we evaluate the effectiveness of learning social norms?

One approach to learning social norms uses multi-agent reinforcement learning based upon (negative) sanctioning, where an agent, or a group of agents, makes known its disapproval of another agent, based upon the second agent's behaviour. The disapproval could be communicated one-to-one to the agent deemed to have violated the norms, or publicly to the group as a whole.

5.5.3 Use Cases Involving Norms

We chose use cases that make the connection, at least in some respects, with agents and the web but require learning and reasoning. They show the practical use of the following aspects:

- Understanding the benefit of incorporating norms with respect to agents on the Web
- Learning, revising, and reasoning about behavioural norms
- Working with implicit and explicit norms
- Supporting norms in relation to planning tasks
- Supporting norms in relation to causal reasoning – understanding intent on the basis that most people/agents will follow the norm, and recognising violations

The use cases that we considered were as follows:

- Ride sharing across town balancing personal desires and needs, following norms, as well as continual learning of norms and related metadata
- Social media assistant advising you (in a dialogue) whether you would violate the norms should you go ahead and send your draft post, and/or for directing posts to moderators for possible sanctions. This is in the context of federated social networks such as Mastodon.
- Checking compliance with regulations such as GDPR and determining how they should evolve to reduce violations [2]: tighter or looser?

The ride-sharing use case is as follows: volunteers or paid drivers offer to drive people across town, e.g. to shops, medical appointments, and so on [1]. A car may be shared by multiple passengers with different pick-up and drop-off locations. Some people may be limited in how far they can walk. What are the rewards for the drivers to drive and for the passengers to accommodate each other's needs? How can software learn metadata, norms, and sanctions from the journeys undertaken? How can we match drivers to passengers dynamically? How can we integrate with information sources providing the essential live or static knowledge, e.g. about roadwork? The people who want a ride should be able to offer suggestions and express their preferences regarding the origin, destination, and timing of their journey.

What is the contribution of plausible knowledge and statistical models? How can we unite such knowledge with argumentation theory for assessing and balancing conflicting desires and constraints? How can we employ other approaches, e.g. constraint satisfaction algorithms, to facilitate smooth functioning and collaboration?

What kinds of norms arise in this use case? For example, a societal objective would be that the elderly or people with walking sticks should be dropped off close to their destination, and likewise picked up close to their starting points. Journey times should not be unreasonably long for any passenger. Journeys should be navigated to avoid known problems (road works, traffic jams).

How can this use case be implemented in a decentralised manner, where the information belonging to different users stays with them and in their control and where their decision making is likewise local?

5.5.4 Identified Challenges

We identified the following challenges and places for design decisions.

- Should the treatment of norms be data-driven or model-driven? We expect both in various ways but capturing their interplay is non-trivial.
- What are the tradeoffs between tacit and explicit norms?
- What are suitable standards for expressing explicit norms as a basis for reasoning about norms and the decisions to take in reference to the norms?
- How can agents incorporate Theory of Mind reasoning, involving beliefs, desires, and intentions (BDI) and other folk psychological notions?
- How can agents collectively learn norms in a federated manner without revealing confidential information of their users to others?
- How can learning and reasoning about norms be performed at the subsymbolic level?
- What are constraints on agent-to-agent communications, e.g., those that affect the learning performed individually and jointly by the agents?
- What would be the nature of an approach based upon publicly sharing information on norm violations, with such information sharing improving reasoning and acting as a potential deterrent to violators?

- How does learning and reasoning about norms relate to trust and trustworthiness, and provide a foundation for the construction of trustworthy systems? Agents should learn to behave in a trustworthy manner and should learn to calibrate their trust in another agent with its observed trustworthiness.

5.5.5 Conclusion

Enabling agents to learn and reason about behavioural norms will be increasingly important given the rapid pace of advances in artificial intelligence, and the need to ensure that applications enrich human society. This raises many challenging issues that we only had the opportunity to skim through in this report.

Acknowledgements

JE was partially supported by the French Agence Nationale de la Recherche (grants ANR-17-CE23-0007-01 and ANR-19-P3IA-0003). MPS thanks the US National Science Foundation (grant IIS-1908374) for partial support for this research.

References

1 Eleni Bardaka, Leila Hajibabai, and Munindar P. Singh. Reimagining ride sharing: Efficient, equitable, sustainable public microtransit. *IEEE Internet Computing (IC)*, 24(5):38–44, September 2020. doi:10.1109/MIC.2020.3018038.

2 Andreasa Morris Martin, Marina De Vos, Julian Padget and Oliver Ray. Agent-directed runtime norm synthesis. Proc. 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS), London (UK), 2023.

3 Luis G. Nardin, Tina Balke-Visser, Nirav Ajmeri, Anup K. Kalia, Jaime S. Sichman, and Munindar P. Singh. Classifying sanctions and designing a conceptual sanctioning process model for socio-technical systems. *The Knowledge Engineering Review (KER)*, 31(2):142–166, March 2016. doi:10.1017/S0269888916000023.

5.6 Challenges and Opportunities in Governing Agents on the Web

Matteo Baldoni (University of Turin, IT)
Victor Charpenay (Mines Saint-Étienne, FR)
Andrei Ciortea (Universität St. Gallen, CH)
Stephen Cranefield (University of Otago, NZ)
Julian Padget (University of Bath, GB)
Munindar P. Singh (North Carolina State University – Raleigh, US)

This paper summarizes the findings of the working group “Governance” at the Dagstuhl Seminar on *Agents on the Web*, which was held in February 2023. This report seeks to identify research challenges relating to governance and the Web architecture.

5.6.1 Introduction

There are major similarities in the motivations behind multiagent systems (MAS) and the Web. Both disciplines and practices seek to advance decentralization and openness in that ideally there is not a single locus of control and agents can behave and interact broadly autonomously under local control.

This report addresses the interplay of multiagent systems with the Web. Specifically, it concerns how constructs and techniques identified in the study of the governance of MAS can be realized over the Web architecture and how the governance of the Web can be beneficially structured based on constructs and techniques developed for the governance of MAS. In simple terms, it seeks to answer the following two questions:

- *What does the Web offer to support the governance of MAS?*
  We anticipate ways to use the scalability and evolvability of the Web to build easy-to-use, widely deployed MAS. Scalability and evolvability are two of the several non-functional requirements that can easily be met if a system's architecture relies on the Web.
- *What does the governance of MAS offer the Web?*
  We anticipate approaches for governance that provide flexibility and local control with formal models that support correctness and generality going beyond the typically procedural kinds of governance seen on the Web today. A challenge here is to map abstract models for governance in MAS to Web components, in a way that preserves Web architectural constraints and thereby guarantees the associated non-functional requirements of scalability and evolvability.

The main contribution of this report is the identification of crucial challenges pertaining to the interplay of MAS and the Web, and the formulation of some initial research questions that might guide future research on this topic.

5.6.2 Key Concepts and Considerations

We understand the Web as a collection of resources, identified with uniform resource identifiers (URIs) and supporting hyperlinked representations, along with a computational architecture that supports locating and accessing the identified resources. The computational architecture is based on standard protocols for manipulating resources (e.g., HTTP, CoAP[29]). We think here of architectural constraints (such as for caching, layering, and uniform interaction) as captured in the original design rationale for the Web Architecture [18], but see also the W3C Recommendation for the Web Architecture [22]. Some of these constraints, especially uniform interaction, are captured by the Linked Data principles [21], which can be supplemented by an ontology specification [34]. Some extensions to the above architecture, such as through the observer pattern (implemented in CoAP [28]) and local state transfer [30], aim at going beyond the "Web of Documents" [19].

There are numerous studies relating the Web and MAS. These exhibit two general trends: some focus on applying RDF and Linked Data to expose agents to hypermedia-driven environments [4, 10, 16, 25], while others combine a formal declarative model of norms [7, 12, 26, 31] to specify social protocols [8]. Such social protocols provide a more thoroughly decentralized conception of Berners-Lee's [1] notion of social machines. Recent W3C Recommendations for the Social Web, including Linked Data Notifications, ActivityStream, ActivityPub and WebSub [20], offer ways to connect these two trends, by implementing social protocols over Linked Data.

[29] Constrained Application Protocol: `https://datatracker.ietf.org/doc/html/rfc7252`

Dimensions that are common to all MAS architecture, such as the environment, organization and interaction dimensions, are defined at a higher level of abstraction than Web resources and protocols. Constraints such as caching, layering and uniform interaction apply to components, exposing a certain functionality through ports. Web components may only have client or server ports, exchanging messages in a standard protocol. Components with client ports only are called origin clients, those with server ports only are origin servers and a third kind of component, proxies, have an equal number of client ports and server ports, forwarding requests from clients to servers or vice versa [18].

To be able to analyse the interplay between MAS and the Web, a mapping from MAS abstractions to (more concrete) Web components is necessary. Given the complexity of both fields, there is no trivial mapping and most likely not a unique mapping across the two levels of abstraction. In the following, we perform a case study to help identify some preliminary mappings that would transfer effective governance mechanisms developed in MAS research.

5.6.3 Example Scenario: Organ Allocation

Consider a simplified version of the process for allocating donated organs and tissue to potential transplant recipients called Carrel [33]. The distribution of organs and tissues in Carrel, given its Spanish and Catalan context, would be overseen by the Spanish ONT, together with the Catalan Organitzaciò CATalana de Trasplantament (OCATT)[30].

Tissue distribution is essentially demand-driven because tissues can be preserved and stored over extended periods with no significant degradation. Organ distribution is essentially supply-driven because the need is known before a suitable part becomes available. For the purposes of this report, we only consider the second case of organ distribution where the need is known before availability.

To achieve its goal, the requesting agent must negotiate with the agents that represent hospitals with potential donors. In the original version of Carrel, participating agents are in effect regimented by so-called governor agents, to prevent non-compliance, but in general, agents may choose to take non-compliant actions. Thus, we assume that agent actions in this contemporary Carrel are all visible to the regulatory bodies, as is the case in the physical world: hospital Transplant Coordination Unit agents (TCUs) communicate their requests to members of ONT/OCATT, who then contact other hospitals to find a donor and select the best match. To participate in Carrel, each agent must hold an authorisation in the form of a certificate.

5.6.4 Relevant Multiagent Systems Approaches

We briefly highlight some relevant multiagent systems approaches.

5.6.4.1 Norms and Roles

Early approaches used agent-mediated electronic institutions [33] to model Carrel by capturing the structure of the interactions between hospitals, tissue banks, and institutional agents managing the process, as well as the norms that govern these interactions and the match between a donor and recipient [17]. Following one approach from the MAS literature [6], we can capture the sociotechnical system requirements in terms of accountability [9] and only then proceed to identify the information exchanges between the agents and hence their individual actions.

[30] `https://trasplantaments.gencat.cat`

5.6.4.2 Allocation Protocol

Consider a simplified *Organ Donation* protocol for TCU agents. The agent holding a surgeon's request for transplant sends its request to another agent, which responds with a donation offer. The requesting agent then has the choice of confirming or rejecting the offer, presumably depending on the strength of this match against any other offers it may have received from other prospective donor hospitals. To facilitate the modeling of norms, such interactions could be captured declaratively in languages such as the Blindly Simple Protocol Language (BSPL) language [29].

5.6.4.3 Norm Representation and Monitoring

Over the last 30 years, researchers in MAS have proposed many approaches to reasoning about norms as well as computational approaches to monitoring a MAS for norm violations [2, 5, 13]. A crucial aspect of this work is to provide a formal representation of norms (see, e.g., [15] for an overview of approaches). For example, the Expectation Event Calculus (EEC) [14] is applicable to this problem.

5.6.5 Candidate Architectures

On the Web, servers hold the power and may hide information, redirect requests or reject them, thereby reducing the action available to agents. We may think of institutions as components with a server port, receiving requests from agents Unlike a "user agent" (browser) in the Web architecture, an autonomous agent might have both client and server roles simultaneously and can have one-sided elementary interactions, where they are not awaiting a response. The requirements of autonomous agents are closer to those of servients in the lingo of the W3C Web of Things Architecture [24], which are components with both client and server roles that can interact in a peer-to-peer manner. If the agent is visible to other agents, i.e. if its representation is dereferenceable, the representation can point, for instance, to a Linked Data Notification inbox that receives messages from other agents [3].

An institution may not need to materialize as a Web component: if norms are defined at design time, agents may be guaranteed to behave (in general) as per these norms, and certification may occur at design time, such that an agent either uses a single certificate throughout the system's lifetime or periodically renews its certificate. Such an institution would have no sanctioning power, nor any need for sanctioning. It cannot easily change the applicable norms either. Updating a norm would potentially require modifying the behavior of all agents at the same time. This is a basic, but inflexible solution.

5.6.5.1 Institution as Read-Only Server

The behavior of an agent may be decoupled from the norms that regulate it, though: if norms are exposed by a read-only (origin) server, an agent may dereference the norms from time to time and internalize whatever formal specifications the server returns. In this configuration, the institutional component has the power of dictating and changing norms at run time. The ability of a

5.6.5.2 Institution as Read-Write Server

In order for the institution to gain sanctioning power, another component may manage its real-time state. The state of the institution includes the level of compliance of each participating agent, which is directly derived from the confirmations/rejections they generate.

To be able to maintain its state, the institutional component must be able to observe each agent-to-agent interaction. For instance, the certificate may be signed not for an agent but for a pair (agent, donation offer ID), forcing agents to request a new certificate every time they make a decision. If the certification server stores a history of confirmations/rejections, it effectively becomes an institutional component that decides in real time whether agents violate norms and, if they do, to sanction them by rejecting their certification request.

The institutional server would become a stateful read-write component, as agents, through their certification requests, change the state of the overall institution. Yet, it remains a purely reactive component, with a single server port.

5.6.5.3 Institution as Proxy

In the above configuration, the institutional component has no knowledge of how agents negotiate. If the institution is to be omniscient, another kind of component should be used. On the Web, it is common to use proxy servers to monitor activity.

The main architectural constraint over proxies is that they have a client port and a server port, such that incoming requests (on the server port) are either immediately responded to or forwarded to another server, possibly after a rewriting step. In our working example, the institutional server may be replaced by a proxy without modifying in any way the behavior of agents. Agents send requests to the proxy, which can keep track of negotiations and add a certificate on-the-fly if the requesting agent behaves properly. The proxy may turn a confirmation into a rejection, to sanction any misbehaving agent. The requesting agent receives feedback on the sanction through the other agent (which acknowledges the rejection, instead of the initial confirmation).

5.6.5.4 Institution as Servient

An alternative is to capture the institution as an explicit stakeholder supported by an agent on par with the other parties in the system. The institution becoming both reactive and proactive, hence must include independent client and server ports, and becomes a servient.

In ordinary operations, this component may have little to say beyond conveying institutional norms and facts as in the previous approach. However, by identifying this institutional entity, we make it subject to accountability. A party can also question the institution, for example, if they fail to receive an organ in a timely fashion. This process may result in the institutional facts being disputed and adjudicated [32] and the norms potentially revised.

If the institution is embodied by an agent, its monitoring and sanctioning power does not depend on architectural constraints (at the level of Web components) but on the behavior of other agents (at the level of MAS abstractions).

5.6.6 Hypermedia-driven Interaction

Hypermedia-driven interaction can support autonomous agents to interact with Web resources in a uniform way while being decoupled from the underlying components. To illustrate how this works, an HTML page typically provides the user with a number of actions, such as clicking a hyperlink or submitting a form. Performing any such action transitions the user to a new page and exposes a new set of possible actions. In each step, the user's browser retrieves not only an HTML representation of the current page from a server but also the hypermedia controls required to transition to new pages. Hypermedia-driven interaction reduces coupling between components (e.g., browsers, proxies, origin servers) and allows them to evolve independently from one another; a central feature that allowed the Web to scale up to the size of the Internet.

In the present case, the various actions – retrieving formal specifications of norms, requesting a certificate, or sending messages to other agents – can be made available to the agents through hypermedia controls. Such hypermedia controls would encapsulate all the information required by an agent to interact with the institution, but to use the hypermedia controls in a reliable manner the agent would have to operate on an abstract model of the institution. For example, if an agent is required to obtain a certificate at run time to enact an *Organ Donation* protocol, the agent could discover such an action possibility through hypermedia.

The institutional model could evolve throughout the agents' lifetimes: for example, from using norms and certificates defined at design time to a model based on evolving norms [27] and certificates to be obtained at run time. Such an evolution would be reflected in the hypermedia environment through the action possibilities provided to agents at run time and thus, to cope with this evolution, the agents would adapt to a new course of action that meets their objectives. Some related work investigates the design of agents able to plan and adapt to dynamic hypermedia environments (e.g., see [11, 23]).

5.6.7 Discussion: Research Questions and Challenges

The contribution of this report is in identifying some new research questions that can motivate research on the interface of MAS and Web architectures. Specifically, we propose the following:

1. How should we model the presence of a governance layer in a MAS?
2. What aspects of a web-based deployment of a MAS may be subject to governance policies and included in an institutional environment?
3. How do we map the required properties of an institutional environment (monitoring, reasoning, sanctioning power) to constraints and mechanisms of a web-based deployment and more specifically to hypermedia controls?
4. How can we specify and implement composed and stacked agent environments (e.g. institutional environments that overlay physical ones and institutional environments that extend others)?
5. Can and should a governed MAS on the Web be modelled and implemented as a hypermedia application?

5.6.8 Conclusion

Thinking about MAS and the Web together opens up new opportunities in building large-scale sociotechnical systems. Such systems would take advantage of the flexibility derived from MAS and the scalability and familiarity (to most developers) derived from the Web. The possibilities are promising and we invite the research community to join us in investigating them.

Acknowledgments

The working group participants would like to thank Dame Wendy Hall and Mehdi Dastani for the input they provided at Dagstuhl. Victor Charpenay thanks the French Agence nationale de la recherche (grant ANR-19-CE23-0030-01) for partial support for this research. MPS thanks the US National Science Foundation (grant IIS-1908374) and the Department of Defense (Science of Security Lablet) for partial support for this research. Andrei Ciortea thanks the Swiss National Science Foundation (grant 189474) for partial support for this research.

References

1 Tim Berners-Lee. *Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web*. Harper Business, New York, 1999.

2 Guido Boella, Leendert W. N. van der Torre, and Harko Verhagen. Introduction to normative multiagent systems. *Computational & Mathematical Organization Theory*, 12(2-3):71–79, 2006. doi:10.1007/s10588-006-9537-7.

3 Sarven Capadisli and Amy Guy. Linked data notifications, 2017. URL: https://www.w3.org/TR/ldn/.

4 Victor Charpenay, Tobias Käfer, and Andreas Harth. A unifying framework for agency in hypermedia environments. In Natasha Alechina, Matteo Baldoni, and Brian Logan, editors, *Engineering Multi-Agent Systems*, pages 42–61. Springer International Publishing, 2022.

5 Amit Chopra, Leendert van der Torre, Harko Verhagen, and Serena Villata, editors. *Handbook of Normative Multiagent Systems*. College Publications, 2018.

6 Amit K. Chopra, Fabiano Dalpiaz, F. Başak Aydemir, Paolo Giorgini, John Mylopoulos, and Munindar P. Singh. Protos: Foundations for engineering innovative sociotechnical systems. In *Proceedings of the 22nd IEEE International Requirements Engineering Conference (RE)*, pages 53–62, Karlskrona, Sweden, August 2014. IEEE Computer Society. doi:10.1109/RE.2014.6912247.

7 Amit K. Chopra and Munindar P. Singh. Custard: Computing norm states over information stores. In *Proceedings of the 15th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, pages 1096–1105, Singapore, May 2016. IFAAMAS. doi:10.5555/2936924.2937085.

8 Amit K. Chopra and Munindar P. Singh. From social machines to social protocols: Software engineering foundations for sociotechnical systems. In *Proceedings of the 25th International World Wide Web Conference*, pages 903–914, Montréal, April 2016. ACM. doi:10.1145/2872427.2883018.

9 Amit K. Chopra and Munindar P. Singh. Accountability as a foundation for requirements in sociotechnical systems. *IEEE Internet Computing (IC)*, 25(6):33–41, September 2021. doi:10.1109/MIC.2021.3106835.

10 Andrei Ciortea, Olivier Boissier, and Alessandro Ricci. Engineering world-wide multi-agent systems with hypermedia. In Danny Weyns, Viviana Mascardi, and Alessandro Ricci, editors, *Engineering Multi-Agent Systems*, pages 285–301, Cham, 2019. Springer International Publishing.

11 Andrei Ciortea, Simon Mayer, and Florian Michahelles. 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 MultiAgent Systems*, AAMAS '18, page 813–822, Richland, SC, 2018. International Foundation for Autonomous Agents and Multiagent Systems.

12 Owen Cliffe, Marina De Vos, and Julian Padget. Specifying and reasoning about multiple institutions. In *COIN 2006*, pages 63–81, 2007. doi:10.1007/978-3-540-74459-7_5.

13 Rosaria Conte, Rino Falcone, and Giovanni Sartor. Introduction: Agents and norms: How to fill the gap? *Artificial Intelligence and Law*, 7(1):1–15, 1999. doi:10.1023/A:1008397328506.

14 Stephen Cranefield. Agents and expectations. In *International Workshop on Coordination, Organizations, Institutions, and Norms in Agent Systems*. Springer, 2013. doi:10.1007/978-3-319-07314-9_13.

15 Stephen Cranefield, Michael Winikoff, and Wamberto Vasconcelos. Modelling and monitoring interdependent expectations. In Stephen Cranefield, M. Birna van Riemsdijk, Javier Vázquez-Salceda, and Pablo Noriega, editors, *Coordination, Organizations, Institutions, and Norms in Agent System VII*, pages 149–166. Springer Berlin Heidelberg, 2012.

16 Oğuz Dikenelli, Oylum Alatlı, and Rıza Cenk Erdur. Where are all the semantic web agents: Establishing links between agent and linked data web through environment abstraction. In Danny Weyns and Fabien Michel, editors, *Agent Environments for Multi-Agent Systems IV*, pages 41–51. Springer International Publishing, Cham, 2015.

17 M. Esteva, J. Padget, and C. Sierra. Formalizing a language for institutions and norms. In Jean-Jules Meyer and Milind Tambe, editors, *Intelligent Agents VIII*, volume 2333 of *Lecture Notes in Artificial Intelligence*, pages 348–366. Springer Verlag, 2001. ISBN 3-540-43858-0.

18 Roy T. Fielding and Richard N. Taylor. Principled design of the modern web architecture. *ACM Trans. Internet Technol.*, 2(2):115–150, May 2002. `doi:10.1145/514183.514185`.

19 Roy T. Fielding, Richard N. Taylor, Justin R. Erenkrantz, Michael M. Gorlick, Jim Whitehead, Rohit Khare, and Peyman Oreizy. Reflections on the REST architectural style and “principled design of the modern web architecture” (impact paper award). In *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering*, ESEC/FSE 2017, page 4–14, New York, NY, USA, 2017. Association for Computing Machinery. `doi:10.1145/3106237.3121282`.

20 Amy Guy. Social web protocols, 2017. URL: `https://www.w3.org/TR/social-web-protocols/`.

21 Tom Heath and Christian Bizer. Linked Data: Evolving the Web into a Global Data Space. *Synthesis Lectures on the Semantic Web: Theory and Technology*, 1(1):1–136, February 2011. `doi:10.2200/S00334ED1V01Y201102WBE001`.

22 Ian Jacobs and Norman Walsh. Architecture of the world wide web, volume one. Technical report, World Wide Web Consortium, 2004. URL: `https://www.w3.org/TR/webarch/`.

23 Matthias Kovatsch, Yassin N. Hassan, and Simon Mayer. 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)*, pages 54–61, 2015. `doi:10.1109/IOT.2015.7356548`.

24 Michael Lagally, Ryuichi Matsukura, Michael McCool, and Kunihiko Toumura. Web of things (WoT) architecture 1.1. Technical report, World Wide Web Consortium, 2023. URL: `https://www.w3.org/TR/wot-architecture/`.

25 Eoin O’Neill, David Lillis, Gregory M. P. O’Hare, and Rem W. Collier. Delivering multi-agent microservices using cartago. In Cristina Baroglio, Jomi F. Hubner, and Michael Winikoff, editors, *Engineering Multi-Agent Systems*, pages 1–20, Cham, 2020. Springer International Publishing.

26 Julian Padget, Marina De Vos, and Charlie Ann Page. Deontic sensors. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 475–481. International Joint Conferences on Artificial Intelligence Organization, 7 2018. `doi:10.24963/ijcai.2018/66`.

27 Bastin Tony Roy Savarimuthu and Stephen Cranefield. Norm creation, spreading and emergence: A survey of simulation models of norms in multi-agent systems. *Multiagent and Grid Systems*, 7(1):21–54, 2011. `doi:10.3233/MGS-2011-0167`.

28 Zach Shelby, Klaus Hartke, and Carsten Bormann. The Constrained Application Protocol (CoAP). Technical Report RFC 7252, Internet Engineering Task Force (IETF), Fremont, California, June 2014. Proposed standard; `https://tools.ietf.org/html/rfc7252`.

29 Munindar P. Singh. Information-driven interaction-oriented programming: BSPL, the Blindingly Simple Protocol Language. In *AAMAS-11*, pages 491–498, Taipei, May 2011. IFAAMAS. `doi:10.5555/2031678.2031687`.

30 Munindar P. Singh. LoST: Local State Transfer – An architectural style for the distributed enactment of business protocols. In *Proceedings of the 9th IEEE International Conference on Web Services (ICWS)*, pages 57–64, Washington, DC, July 2011. IEEE Computer Society. doi:10.1109/ICWS.2011.48.
31 Munindar P. Singh. Norms as a basis for governing sociotechnical systems. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 5(1):21:1–21:23, December 2013. doi:10.1145/2542182.2542203.
32 Pankaj R. Telang, Anup K. Kalia, John F. Madden, and Munindar P. Singh. Combining practical and dialectical commitments for service engagements. In *Proceedings of the 13th International Conference on Service-Oriented Computing (ICSOC)*, number 9435 in Lecture Notes in Computer Science, pages 3–18, Goa, India, November 2015. Springer. doi:10.1007/978-3-662-48616-0_1.
33 Javier Vázquez-Salceda, Ulises Cortés, Julian Padget, Antonio López-Navidad, and Francisco Caballero. The organ allocation process: A natural extension of the Carrel agent-mediated electronic institution. *AI Communications*, 16(3):153–165, 2003.
34 W3C. OWL 2 Web Ontology Language: Document overview, December 2012. W3C Recommendation. Accessed 2023-03-03. URL: https://www.w3.org/TR/owl2-overview/.

5.7 Agents on the Web: Specifying and Implementing Testbeds

Cleber Jorge Amaral (Federal Institute of Santa Catarina – São José, BR)
Jean-Paul Calbimonte (HES-SO Valais – Sierre, CH)
Jomi Fred Hübner (Federal University of Santa Catarina, BR)
Timotheus Kampik (Umeå University, SE & SAP Signvavio – Berlin, DE)
Tobias Käfer (KIT – Karlsruher Institut für Technologie, DE)

5.7.1 Introduction

Agents are often challenging to test due to their complexity and the unpredictability and dynamism of the environments in which they are situated. It is considered infeasible to test all courses of actions that complex agents can take [1, 2]. Even for one particular goal, agents may have several alternative plans or may apply non-deterministic approaches, for example in the context of data-driven methods such as machine learning. In addition, testing stochastic scenarios and situations that entail interaction with other agents and humans can be challenging to reproduce and control [3, 4]. Ensuring that an agent behaves as desired in a relatively friendly environment is already challenging; assuring this at a scale that includes numerous autonomous software artifacts running relatively freely in the environment is a crucial problem [2].

For the specific case of agents on the Web, it is important to consider the Web as *the* decentralized environment where interactions happen, rendering the governance of agent on the Web a key challenge [5]. Interactions can take places between hetreogenous agents (e.g., using usual agent communication languages) but also between agents and other Web entities, applying various of actuators and sensors. We consider the case where the agent environment is identified by a set of resources that are published and made accessible on the Web. While these resources can be linked to real entities, like sensors, robots, actuators, drones, or smart devices, they can also be linked to virtual entities and services. The *situatedness* of these agents is an integral part of the Web environment, where resources and other agents are identified and located through URIs.

This chapter reports the main results of the working group "Benchmark and Testbeds", as discussed during the Dagstuhl Seminar "Agents on the Web". The different challenges regarding testing and benchmarking of agents on the Web were discussed, and specific scenarios and scenarios types for testing and benchmarking were proposed. Through the proposed set of scenarios and scenario types, we identified potential future work regarding the conception and implementation of testbeds for Web Agents.

5.7.2 Testing and Benchmarking of Agents on the Web

The Web provides numerous opportunities for the use of agents and multi-agent systems that are diverse with respect to use cases and underlying technology stacks. In this section we introduce a list of categories of scenario types that can be used for testing and benchmarking MAS on the Web. We assess the scenario categories in terms of how easy (if possible) it is to apply techniques for testing autonomous software artifacts, such as unit, integration, regression, stress, and beta testing, and how to benchmark different application development approaches. A comparison of the categories considering their strengths and weaknesses is provided in Table 5.

The first scenario categories is based on legacy Web-based applications used in a "production-like" environment, i.e. in an environment that reflects the socio-technical complexity of a system whose operation has critical real-world implications, for example in a business context. The actual environment, which can be virtual or physical, is the "real-world" setting in which applications and devices are operating. Because each system has a distinct set of requirements and constraints, throughout the years several protocols (e.g.: CORBA, OPC, RPC) and architectural styles (e.g.: SOAP, REST, GraphQL) were adopted [6]. Legacy Web-based applications are usually a composition of different technologies that are integrated into a functionally somewhat coherent and cohesive system of high technological heterogeneity. Besides the lack of standardization, legacy web-based applications often have many human-machine interfaces of different maturity levels and frequently systems are hard-wired to each other via system actions on graphical user interfaces (in the case of so-called robotic process automation). Navigating this interface heterogeneity can be a key challenge for agents on the (current) Web. Another characteristic is that such applications have little space for mistakes when deploying new versions since they are usually in day-by-day usage by many users and often perform business-critical transactions. Examples of legacy web-based applications are: ERP systems, industrial automation plants [7] and digital marketplaces. One advantage of adopting this scenario category is that it exposes the system under test to a realistic environment which can facilitate software maturity and resilience, in particular because it reflects real-world requirements and performance indicators, allowing for meaningful benchmarking of the current system as well as of partial or full replacements that are technologically more advanced.

The third scenario category is based on simulated environments. Simulated environments can provide reasonably realistic constraints and intuitive visual interfaces for many scenarios. They are often built using simulation frameworks such as Gazebo[31] and Morse[32] which provide

[31] Gazebo is available at `https://gazebosim.org/`
[32] Morse is available at `http://morse-simulator.github.io/`

Table 5 Overview of scenario categories, with strengths and weaknesses.

Scenario category	Strengths	Weakness
Heterogeneous Web-based applications in production-like environments	• Realistic conditions and use of agents • Easier identification of system performance and compliance • Easier understanding of user interactions impacts over the system	• Systems are often rigid with not much opportunity to apply artificial agents • It is often hard to reproduce some situations that occurs in the actual environment • It can be time-consuming to test applications in such scenarios • It is often challenging to gain access to such scenarios due to privacy and security concerns • Scenarios are often in critical missions with little space for experimentation
Simulated environments	• Usually less complex than production environments and easier to grasp for humans • Testing conditions are easier to control and reproduce • Sets of tests can be performed quickly • Usually more cost-effective than testing than production environments	• May lack important characteristics and interactions that can be found in production environments • Time-consuming to develop a scenario with the necessary accuracy
Programming contest scenarios	• Has well-defined performance indicators that facilitates comparisons • Competitions often provide more realistic testing conditions than traditional simulators • Development and improvement is made easier by the fact that such scenarios frequently call for a limited range of actions	• May lack important characteristics and interactions that can be found in real-world environments • Testing conditions are limited to the contest proposal
Testbeds	• Realistic conditions and use of agents • Easier identification of system performance and compliance • Has well-defined performance indicators that facilitates comparisons	• Complex to define (requires substantial engineering effort) • It can be time-consuming to test applications in such scenarios

tooling for applying physical constraints, and for drawing and animating representations of the environment, thus facilitating human interpretability. Such frameworks also provide tools for monitoring and debugging systems. There are many possible scenarios that can be simulated, such as transportation and search-and-rescue scenarios. As the scenarios can be simpler than the actual environment, simulated environments can facilitate the development of applications. However, the application is not exposed to all situations that can potentially occur in the real world.

Another scenario category are programming competitions, such as MAPC[33] and RoboCup[34]. These competitions provide different scenarios that can run either simulated or real-world environments. A distinguishing characteristic of this scenario category is that in order to facilitate competition, the scenarios have well-defined performance indicators, which enables the comparison of the different approaches.

Finally, there are scenarios provided as testbeds. Testbeds are defined for experimenting and comparing approaches [8]. For instance, a scenario may allow agents to enter buildings and rooms therein in order to interact with appliances such as light switches, which in turn affects the state of the environment. This scenario shares characteristics of simulated and competition scenarios as it is defined for a small set of actions; it is not supposed to be in use in production, and its performance requirements and indicators can be easily defined and monitored. Ideally, testbeds share some of their characteristics with production-like scenarios

[33] More information about the MAPC can be found at `https://multiagentcontest.org/`
[34] More information about the RoboCup can be found at `https://www.robocup.org/`

as they are usually developed using real devices and run in a (potentially cyber-physical) realistic environment. In the context of the WoT, due to the lack of existing simulated and contest scenarios and limited access to WoT scenarios in real-world environments, testbeds can provide ideal conditions for testing and benchmarking MAS on the Web.

5.7.3 Technical Features of Agents on the Web

In this section, we propose a synthesis of technical terms of related to the engineering of agents & MAS and existing standards and practices from the Web.

5.7.4 Interaction Levels for Agents

Envisioned and existing agents on the Web perform tasks on different levels of interaction with the environment and other agents, which can get categorized into:

1. Data Gathering – such agents crawl the web in order to gather information, and may exhibit a querying interface. That is, they operate in read-only mode with safe HTTP requests.
2. Actuation – such agents may also read information, but their characteristic is that they additionally perform writing operations on the environment using unsafe HTTP requests.
3. Communication – such agents engage in conversation with other agents. To this end, they also perform writing operations, but to specific endpoints of other agents, with the goal of communicating.

Note that in the context of testing and benchmarking, in order to provide means for comparing approaches and MAS applications, the focus is "top-level goals rather than approach-specific goals" [9].

5.7.5 Agent Terminology on the Web

In order to combine MAS and Web approaches, a common view on what can constitute agenthood on the Web needs to be established. Depending on the above interaction levels for agents, it is especially on level 3, where the following aspects are required:

- Identity – an agent needs an identifier, i.e., a URI on the Web.
- Endpoint – an agent needs a way to receive communication, e.g., an LDN inbox.
- Entry point – an agent needs a way to expose information, e.g., an FOAF profile document on the Web.

The other levels 1 and 2 require other aspects:

- Situatedness – the agent needs a reference to where it is. On the Web, this could be a (set of) seed URI(s) for a data gathering agent; or an actuating agent could be constrained in its behavior using a query.
- Norms – to formulate desired agent behavior using norms, a notion of norms for the Web needs to be established. This could be implemented as agent behavior descriptions, for which there are no standards and practices yet.
- Signifiers / possible actions – for an agent to find out what it may do in an environment, descriptions of actuable items are required; on the Web there are, e.g. `schema:PotentialAction`, which represents a coarse match to the agent terminology.
- Adaptivity – an adaptive agent can act in different environments. On the Web this may mean that the agent leverages reasoning to deal with heterogeneous descriptions.

5.7.6 Running, Testing, and Benchmarking Agents on the Web

With those proposed translations, the following steps may need to be taken to run, test, and benchmark agents:

- Define very simple norms – with the notion of behavior hardly defined for the web, simple condition-action rules may be used to implement a very simple norm.
- Analyze norm and policy compliance – here, log analysis techniques may come in, corresponding metrics need to be defined.
- Enact norm violation punishment – depending on the scenario and the control over the environment, it may be possible to punish agents.

5.7.7 Open Questions and Future Work

Testing and benchmarking are cornerstones of modern software engineering. Still, both topics receive relatively little attention in the context of the (engineering) multi-agent systems research. As reliable and trustworthy behavior is both particularly important and challenging when it comes to Web-scale systems, the research need around testing and benchmarking is even more pronounced here. Some of the open questions that future work can address are the following:

- Which abstractions are central to testing and benchmarking of MAS on the Web?
- What are good practices and pitfalls of Web-scale benchmarking of MAS?
- Does testing MAS on the Web require dedicated tooling support?

Initial works towards the implementation of a benchmark framework for testing Agents on the Web have started to emerge, as in the BOLD server[35]. Future developments and extensions to this type of framework may be a promising starting point for facilitating norm-based validation, agent behavior testing, assurance of policy compliance, and other aspects as discussed above. This working group has concluded that providing testbed environments such as the ones suggested in this chapter can potentially facilitate the systematic evaluation of approaches to engineering agents on the Web and to place a greater focus on the assurance of quality and reliability.

References

1 Winikoff, Michael. *BDI agent testability revisited.* Auton. Agents Multi Agent Syst., 31(5), 1094–1132, 2017, `https://doi.org/10.1007/s10458-016-9356-2`.

2 Koopman, Philip and Wagner, Michael. *Challenges in autonomous vehicle testing and validation.* SAE International Journal of Transportation Safety, 1(4), 15–24, 2016.

3 Linz, Tilo. *Testing autonomous systems.* The Future of Software Quality Assurance, 61–75, 2020.

4 Timotheus Kampik, Cleber Jorge Amaral and Jomi Fred Hübner. *Developer Operations and Engineering Multi-agent Systems.* Engineering Multi-Agent Systems – 9th International Workshop (EMAS), 2021, `https://doi.org/10.1007/978-3-030-97457-2_10`

5 Timotheus Kampik, Adnane Mansour, Olivier Boissier, Sabrina Kirrane, Julian Padget, Terry R Payne, Munindar P Singh, Valentina Tamma and Antoine Zimmermann. *Governance of Autonomous Agents on the Web: Challenges and Opportunities.* ACM Transactions on Internet Technology, 4, 1–31, 2022.

6 Richardson, Leornard and Ruby, Sam. *RESTful Web Services.* ISBN 9789896540821, 2007.

[35] Bold Benchmark: `https://github.com/bold-benchmark`

7 Otávio Arruda Matoso and Luis P. A. Lampert and Jomi Fred Hübner and Mateus Conceição and Sérgio P. Bernardes and Cleber Jorge Amaral and Maicon R. Zatelli and Marcelo L. de Lima. *Agent Programming for Industrial Applications: Some Advantages and Drawbacks.* arXiv, 2020.

8 Bouron, Thierry and Ferber, Jacques and Samuel, Fabien. *MAGES: A multi-agent testbed for heterogeneous agents.* Decentralized Artificial Intelligence, 2, 195–214, 1991.

9 Victor Lesser, Charles L. Ortiz and Milind Tambe. *Distributed Sensor Networks: A Multiagent Perspective.* isbn 9781461350392, 1-376, Springer US, 2003.

Participants

- Cleber Jorge Amaral
Federal Institute of Santa Catarina – São José, BR
- Matteo Baldoni
University of Turin, IT
- Olivier Boissier
Ecole des Mines – St. Étienne, FR
- Samuele Burattini
University of Bologna, IT
- Jean-Paul Calbimonte
HES-SO Valais – Sierre, CH
- Pierre-Antoine Champin
INRIA – Sophia Antipolis, FR
- Victor Charpenay
Mines Saint-Étienne, FR
- Amit K. Chopra
Lancaster University, GB
- Andrei Ciortea
Universität St. Gallen, CH
- Rem Collier
University College Dublin, IE
- Stephen Cranefield
University of Otago, NZ
- Mehdi Dastani
Utrecht University, NL
- Kemal A. Delic
The Open University – Milton Keynes, GB
- Jérôme Euzenat
INRIA Grenoble Rhône-Alpes, Saint Ismier, FR & Univ. Grenoble-Alpes, FR
- Catherine Faron
Université Côte d’Azur – Sophia Antipolis, FR
- Nicoletta Fornara
University of Lugano, CH
- Fabien Gandon
INRIA – Sophia Antipolis, FR
- Wendy Hall
University of Southampton, GB
- Andreas Harth
Fraunhofer IIS – Nürnberg, DE
- Jomi Fred Hübner
Federal University of Santa Catarina, BR
- Tobias Käfer
KIT – Karlsruher Institut für Technologie, DE
- Timotheus Kampik
Umeå University, SE & SAP Signavio – Berlin, DE
- Matthias Kovatsch
Siemens Schweiz AG – Zug, CH
- Brian Logan
University of Aberdeen, GB & Utrecht University, NL
- Stefano Mariani
University of Modena, IT
- Simon Mayer
Universität St. Gallen, CH
- Mahda Noura
TU Chemnitz, DE
- Eoin O’Neill
University College Dublin, IE
- Julian Padget
University of Bath, GB
- María Poveda-Villalón
Technical University of Madrid, ES
- Dave Raggett
W3C – United Kingdom, GB
- Alessandro Ricci
Università di Bologna, IT
- Sebastian Schmid
Universität Erlangen-Nürnberg, DE
- Daniel Schraudner
Universität Erlangen-Nürnberg, DE
- Munindar P. Singh
North Carolina State University – Raleigh, US
- Ruben Taelman
Ghent University, BE
- Danai Vachtsevanou
Universität St. Gallen, CH
- Jan Van den Bussche
Hasselt University, BE
- Antoine Zimmermann
Ecole des Mines – St. Étienne, FR