Multi-Agent Interactions on the Web through Linked Data Notifications

Jean-Paul Calbimonte[1], Davide Calvaresi[1,2], and Michael Schumacher[1]

[1] University of Applied Sciences and Arts Western Switzerland,
HES-SO Valais-Wallis, Sierre, Switzerland
[2]Scuola Superiore Sant'Anna, Pisa, Italy
{name.surname}@hevs.ch

Abstract. The evolution of the Web towards a semantically-enriched information space has risen several challenges and opportunities concerning the interaction, knowledge representation, and design of multi-agent systems. Many of these have been explored in the past, such as the usage of ontologies for defining agent knowledge bases, the definition of semantic web services, or the usage of reasoning for intelligent agent behavior. Although these efforts have resulted in important research achievements, there is still a need to provide a simple –yet comprehensive– way of interconnecting decentralized intelligent agents through a generic Web-based infrastructure. In this paper we analyze how multi-agent systems can use extensions of the Linked Data Notifications W3C recommendation as the backbone for a Semantic Web-enabled infrastructure for agent communication.

1 Introduction

Multi-agent systems have shown an enormous potential for solving different types of tasks in several domains, such as health-care [7], financial technologies [19], traffic monitoring [10], and e-commerce [8]. Agents are capable, through different paradigms and strategies, to act according to their knowledge, goals, and dynamic environment, using intelligent algorithms, continuous learning, and knowledge management techniques [13]. The decentralized nature of multi-agent systems (MAS) requires them to rely on coordination and communication mechanisms that may require heterogeneous interactions over complex networks. This allows agents to exchange information and cooperate regardless of their physical location. The Web provides a natural environment for such interactions, thanks to the standards and protocols developed in the last decades.

However, the evolution towards a *Semantic Web* [5] has risen several challenges and opportunities concerning the interaction, knowledge representation, and design of MAS. Many of these have been explored in the past, such as the usage of ontologies for defining and exploring agent knowledge bases [18], the definition of Semantic Web services [20] for orchestration and negotiation, or the usage of reasoning for intelligent agent behavior [12]. Although these efforts have resulted in important research milestones, there is still a need to provide a simple and comprehensive way of enabling a generic Web-based communication among decentralized intelligent agents. Even if the initial vision of the Semantic Web explicitly evoked the emergence of these agents, in practice most implementations of the Semantic Web have focused on ontology models, reasoning, Linked Data, or RDF data management and querying.

In this paper we analyze how multi-agent systems can use extensions of existing W3C recommendations to interact on the Web, under a decentralized scheme. We describe a work-in-progress proposal how the W3C Linked Data Notifications (LDN) [9] recommendation can be used as the backbone for a Web-enabled infrastructure for agent data interchange.

As an example of an application for such environment, let us consider the following use case. Roy, a middle-aged trekking enthusiast takes a trailing path near the Alps. He is equipped with an agent-based smart-watch with health monitoring capabilities, which can collect several physiological data on real time. At the same time, as he has recently had episodic breathing difficulties, his smart-watch can coordinate with a health recommendation application, which depending on the sensor readings, history, and current location/-path difficulty/trekking time, etc. is able to propose alternative paths that are better suited for Roy. The health recommendations also take into account different characteristics of the nearby points-of-interest. For example, as Roy has vertigo issues, cliffs and voids in the trekking paths are avoided. Additionally, the local weather service is consulted in order to avoid local strong winds. Finally, depending on Roy's tiredness, sugar levels and stress, point of care and catering services can be proposed, coordinated and booked through his smart-watch agent.

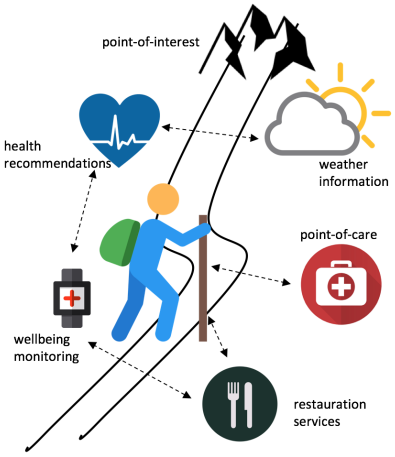

Fig. 1. Use-case: agent interactions on the Web for health recommendations.

To make these interactions possible, agents for the different described instances need a common language and a communication interface. The Web and its foundational standards, along with explicit semantics for data interchange, can pave the way for decentralized agent interactions, as argued in this work. The paper is organized as follows: we introduce LDN in Sec. 2, the main requirements for Web Agent interactions in Sec. 3. Sec. 4 described the use of LDN for agent messaging. Sec. 5 presents related work, before discussion in Sec. 6.

2 Linked Data Notifications

Linked Data Notifications (LDN) [9] is a recently endorsed W3C Recommendation[1] for decentralized data interchange of notifications on the Web. This protocol is designed as a generic and simple mechanism to send and consume data, based on the Linked Data [4] principles and usage of RDF (Resource Description Framework) for data representation. LDN has the potential to be used for virtually any type of notifications, including social media activity, sensor updates, or document updates, to name some examples. Although the adoption of LDN is still to be assessed, its characteristics make it an interesting option for different types of applications on the Web, for which extensions and/or profiles could be defined.

[1] `https://www.w3.org/TR/ldn/`

LDN defines three basic types of actors: *sender*, *receiver*, and *consumer*, and the notifications refer to (or are about) a certain *target*. The target is detached from its *inbox*, which is the endpoint where notifications can be consumed or sent. Senders may send notifications to an inbox, receivers may accept them and make them available, and consumers may retrieve them. Given that a target is not necessarily attached to its inbox, it is possible to separate a Web resource from the endpoint where notifications will be handled. As it can be seen in Figure 2 (top), a discovery process allows senders and consumers to retrieve the inbox location through a simple `GET`/`HEAD` HTTP request. Once the inbox location is known, senders can `POST` notifications to it, and consumers may `GET` the references to notifications contained in the inbox (Figure 2, bottom).

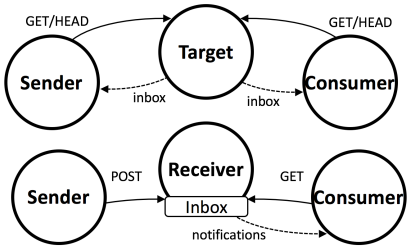

Fig. 2. LDN. Top: Discovery process of a target inbox. Bottom: send and retrieve notifications from an LDN inbox.

3 Requirements for Interactions in the Web of Agents

Agent communication and protocols have been long studied, designed and implemented in the past, as presented in Section 5. Although there have been attempts to standardize these interactions, which could make it possible to integrate agents on the Web, these efforts (e.g. KQML [14], FIPA ACL [1]) have reached little adoption in practice. In the following, we identify a set of requirements for agent interactions on the Web.

R1: Standard and extensible messaging. Agents on the Web should be able to exchange any type of data, in different formats and representation means. **R2: Standard metadata.** Agents may use Web standards for representing metadata in their interactions. Metadata may include information such as participants, time constraints, performatives, conditions, etc. **R3. Asynchronous and distributed communication.** Agents on the Web should be able to send and receive messages, as well as coordinating among them without the need of a central entity that governs their interaction flow. **R4. Standard Web protocols.** Communication among agents should be implemented on top of widely supported Web Standards such as HTTP, but not excluding others. This implies no commitments to a particular agent implementation or framework. **R5. Web identifiers.** Agents and their resources, including message items, should all be named using identifier standards for the Web (i.e. URL/URI/IRIs), which provide unicity and de-referenceability. **R6. Semantic representation.** To allow agents to understand and act accordingly to a given message, semantic representations should be used. These should align with Web standards (e.g. OWL, RDF), and allow extensibility and high expressiveness.

4 LDN for Agents on the Web

This section provides a high-level overview of how decentralized agents can communicate on the Web using the LDN recommendation. In this proposal we take into account the requirements presented in the previous section, while considering the characteristics and principles behind LDN. In the following, we explain the main aspects of this proposal, including technical and design features.

HTTP-based communication. Given that LDN is entirely based on HTTP requests and responses, agents using LDN should also rely on this protocol for most of their interactions. The ubiquity of HTTP on the Web makes it the natural candidate for most types of exchanges, although –as the LDN specification states– other protocols could be used in certain circumstances, e.g. WebSockets for push-subscriptions.

Agent identification. Given that LDN relies on the principles of Linked Data [4], URIs (or IRIs) are used to identify all entities involved. This includes the agents themselves, which should be de-referenceable in order to obtain more information about them. This feature would overcome the agent visibility, which in the traditional framework is limited to their single or federated container[3]. Moreover, introducing proper encoding mechanisms, the perception of agent's environment can be enriched and enhanced, thus fostering wider understanding and exploitation. As an example, an agent can be de-referenced through a `GET` operation over its IRI, e.g.:

```
GET http://example.org/agents/health-agent
```

The response to this request should include metadata about the agent, such as its name, scope, endpoint, ontologies, etc. [1] As prescribed by LDN, each of these agents can provide an endpoint to which messages can be sent, i.e. the inbox. This inbox does not need to be located within the same environment as the agent itself, providing further flexibility.

Endpoint discovery. Each agent may advertise its inbox as indicated by LDN, with the LDP `inbox` predicate. As an example, consider the following JSON-LD message content response for the previous agent request:

```
{ "@context": "http://www.w3.org/ns/ldp",
  "@id": "http://example.org/agents/health-agent",
  "inbox": "http://example.org/agents/health-agent/inbox" }
```

The content indicates the inbox location, and potentially other useful metadata. This discovery phase would indeed be the first interaction between two agents that wish to establish a conversation or initiate a negotiation.

RDF data representation. Agent messages in practice could adopt any representation format and/or model. However, LDN agent implementations may preferably use RDF as a common and standard representation framework. RDF natively integrates the use of URIs for identifiers, allows using extensible vocabularies, and makes it possible to attach explicit semantics to all statements. Metadata annotations should be expressed in RDF, i.e. sender, receiver, performative, protocol, date-time, reply information, conversations, etc. (see FIPA ACL for common metadata information [1]). As an example, the metadata below, represented in RDF (JSON-LD serialization) contains information about an *agree* message, indicating the sender agent, receiver, conversation information, etc.

```
  { "@id": "ex:agree_request1",
    "ag:permormative": "ag:Agree",
    "ag:sender": "ex:agent1",    "ag:receiver": "ex:agent2",
    "ag:reply-to": "ex:agent3",  "ag:protocol": "ag:RequestWhen",
    "ag:conversationId": "ex:conversation3",    "ag:inReplyTo": "ex:conversation1",
    "ag:ontology": "http://example.org/ontology#",
    "ag:content": "..."   },
```

Sending agent notifications An LDN agent may `POST` notifications to an agent inbox endpoint, as it is specified in LDN. Essentially, the `POST` body should contain the agent message (e.g. an RDF graph) that will be fed to the inbox of another agent. As an example consider the JSON-LD representation of a call for proposals agent message:

```
POST /agents/health-agent/inbox HTTP/1.1
Host: example.org
Content-Type: application/ld+json

{"prov:generatedAtTime": "2017-09-14T04:00:00.000Z",
 "@id": "ex:callForProposals1",
 "@graph": [
   { "@id": "ex:cfp1", "ag:permormative": "ag:CallForProposals",
     "ag:sender": "ex:agent1", "ag:protocol": "ag:ContractNet",
     "ag:ontology": "http://example.org/healthOntology#", "ag:content": "..."   }],
 "@context": {
   "prov": "http://www.w3.org/ns/prov#", "ex": "http://example.org#", "ag": "https://w3id.org/rdf-
        agents/msg#"}  }
```

Notice that the call is made against the agent inbox URI, and that the message indicates metadata information such as the sender, identified with its own URI (`eg:agent1`). It also includes the reference to the ontology used to represent the message content, the protocol (e.g. `ContractNet`), the type of message (i.e. *performative*), etc. The message content is not included for space reasons, but one could specify an type of arbitrary message, given the flexibility of RDF.

Interaction protocols. Agent languages have been proposed in the past, even reaching a certain level of standardization. Agreement is not only necessary at format or message level, but also for the type of interactions themselves. For instance, the FIPA ACL standards identify several protocols for agent interactions. As an example, consider the Request Interaction Protocol[2] partially depicted in Figure 3. The sequence diagram shows how an agent performs a request, which can be refused or agreed by a second agent, leading then to an *inform* or *failure* message. The generic nature of such protocols allows implementers to reuse them for different scenarios in practically any domain.

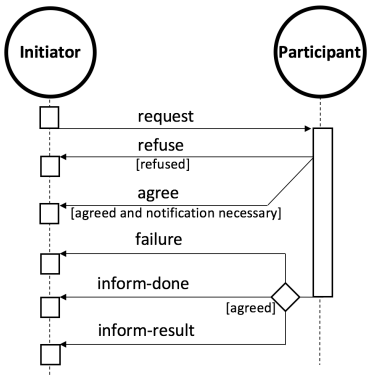

Fig. 3. FIPA ACL request interaction.

An LDN agent implementation should be able to support these interaction protocols, using the technical mechanisms provided by LDN. As an example, consider the diagram in Figure 4. It depicts part of a Contract Net interaction protocol, according to FIPA ACL. Intuitively, it consists of a call for proposals which is made available to an agent inbox. These can later be accessed by the inbox owner (or owners), thus allowing them to respond to it by sending proposals. These proposals can afterwards be accepted by the initiator agent. Using LDN, all these messages should conform to the RDF structure presented above, and would be exchanged preferably through HTTP, with `GET` and `POST` operations as shown in Figure 4.

[2] `http://www.fipa.org/specs/fipa00026/SC00026H.html`

Publishing inbox elements. As indicated by LDN, consumers may access an agent inbox in order to obtain the messages available there. Although LDN does not impose a fixed access control mechanism, it should be noted that different *security*, *privacy* and *ownership* schemes should be enforced at this level. Putting aside the security constraints, LDN specifies that performing a `GET` over an inbox should return the notification URIs listed as objects to the LDP `ldp:contains` predicate.

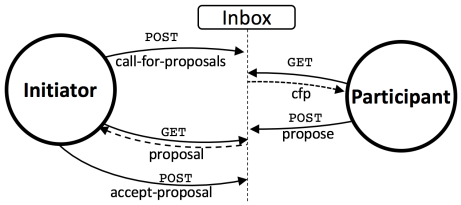

Fig. 4. LDN agent interactions for a CfP protocol.

Agent reasoning. Regardless of any specific architecture (e.g., BDI), the LDN adoption do not affect traditional reasoning engines, since the implementation of a simple data-parser can guarantee the retro-compatibility with already existing mechanisms and foster the development of new ones with increased capabilities due to the semantic expressiveness and the simplified and extended interactions.

5 Related Work

Internet of Things, knowledge-based, and network-based/oriented systems are gaining momentum in the market. Therefore, it is a major challenge to integrate uncountable heterogeneous devices, virtual entities and human end-users. Such fast-paced growing networks wrap or connect several frameworks, mostly exploiting ad-hoc interaction methods, or over-complicating already structured communication standards. However, to achieve a common understanding, the definition of common formats and semantics (preferably standard) are necessary. Under the hypothesis of agents' rationality, the scientific community has defined a number of agent communication languages. For example, KQML [14] is a simple protocol mainly used in the academic world defining the basics of interaction among intelligent entities, in particular in MAS [17]. Later on, the Foundation for Intelligent Physical Agents (FIPA) [2] improved and extended the previous languages, defining the FIPA ACL (Agent Communication Language) in 1997.

Although such protocols provide a relevant support for agent communication, there is still no clear understanding/definition of the semantics of individual speech acts. Moreover, basic concepts required to define the semantics are still missing. Hence, a variety of semantic simplification strategies need to be employed when designing MAS. For example, JADE [3] and JASON [6], two of the most used multi-agent platforms, are compliant with the FIPA standards. Thereby, the agents running on such platforms could interact with any agent (language and platform independent) FIPA compliant. The FIPA ACL message structure is characterized by both mandatory (e.g., message type indicated as performative - *request, inform*) and optional (e.g., recipient, sender, ontology) contents. MAS' messages might strictly adhere to the ACL standard just encoding the messages according to the parameters listed in Table 1 in [1]. Nevertheless, real-scenario applications require handling more complex and articulated contents. Thus, extending such parameters is an obliged path.

Concerning negotiation protocols, these are mainly characterized by 1-to-1 or 1-to-n interactions between *initiators* (who proposes, the task to be performed and its

boundary conditions) and *contractors* (who propose themselves as "solvers" replying to the required conditions with a bid) dynamically [16]. Although argumentation and negotiation in MAS can involve sophisticated, high-level reasoning, the relevance of the information representation and encoding for a common understanding is crucial. For example, in the context of agents operating in and crawling the web, the information can be expressed both in natural language or exploiting a multitude of different formats. Thus, although the agents know how to interact, they need a common knowledge base and/or an ontology to give an actual meaning to their interaction. Currently, markup languages such as XML are heavily used for agent data representation. Nevertheless, there is a growing trend on moving beyond the implicit semantic agreements inherent in languages such as XML and JSON. Standards and mechanisms such as RDF are developed to tie the information on machine-readable objects, finally allowing semantic interoperability [15].

6 Discussion

Multi-agent systems have proven to be useful in a wide range of application domains, especially when autonomous coordination and intelligent behavior is required. When such systems scale to the Web, additional needs arise, regarding heterogeneity, message semantics, mutual understandability, and decentralization. Although there have been important efforts targeting scenarios where Web agents interact for a given task (see Section 5), most of these approaches are either too complex, or have been abandoned in the last years.

While on the Semantic Web community, the idea of Web agents relies at the core of its original vision, in practice there is still not a commonly agreed mechanism for enabling these agents to communicate with each other, using well established standards. The challenges to achieve this vision are still numerous:

- Adoption of semantically rich messaging mechanisms (e.g. RDF-based) among Agents on the Web.
- Usage of ontologies and vocabularies that link existing Web protocols (e.g. LDN) and Agent-communication standards (e.g. FIPA ACL).
- Definition and agreement of system-agnostic Agent communication primitives, based on existing MAS languages.
- Provision of agent discovery, selection and orchestration services, based on existing standards.
- Implementation and adoption of best practices of agent-based mechanisms for Web interactions.

These high-level challenges describe only some of the urgent needs in this scope, but they already show the need for MAS and Semantic Web communities to work on research topics that address these issues. In this paper, we present a vision of how this could be implemented in practice, using existing Web standards, and specifically relaying on the Linked Data Notifications protocol. Although the generic nature of this specification may not be enough to describe interaction protocols among agents, an extension, or a profile for LDN, can fill this gap. This effort is complementary to recent developments

towards MAS deployed as hypermedia applications [11] and Semantic Web services [20]. The examples and scenarios described in this paper provide an initial work-in-progress vision of how this can be done, although a more detailed specification of these extensions needs to be made and implemented, thus stepping towards its validation. In the future, we plan to continue developing this vision, and providing a feasibility evaluation using real use-cases. We believe that this approach may open new research perspectives for designing Web-scale solutions governed by intelligent agent-based systems.

References

1. FIPA ACL Message Structure Spec. `http://www.fipa.org/specs/fipa00061/`.
2. Foundation for Intelligent Physical Agents Standard. `http://www.fipa.org/`.
3. F. Bellifemine, A. Poggi, and G. Rimassa. Jade: a fipa2000 compliant agent development environment. In *Proc. of the 5th Intl. Conf. on Autonomous agents*, pages 216–217, 2001.
4. T. Berners-Lee, C. Bizer, and T. Heath. Linked data-the story so far. *IJSWIS*, 5(3):1–22, 2009.
5. T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific american*, 284(5), 2001.
6. R. H. Bordini, J. F. Hübner, and M. Wooldridge. *Programming multi-agent systems in AgentSpeak using Jason*, volume 8. John Wiley & Sons, 2007.
7. D. Calvaresi, D. Cesarini, P. Sernani, M. Marinoni, A. Dragoni, and A. Sturm. Exploring the ambient assisted living domain: a systematic review. *Journal of Ambient Intelligence and Humanized Computing*, pages 1–19, 2016.
8. D. Calvaresi, M. Marinoni, A. Sturm, M. Schumacher, and G. Buttazzo. The challenge of real-time multi-agent systems for enabling iot and cps. In *WI*, pages 356–364, 2017.
9. S. Capadisli, A. Guy, C. Lange, S. Auer, A. Sambra, and T. Berners-Lee. Linked data notifications: a resource-centric communication protocol. In *ESWC*, pages 537–553, 2017.
10. B. Chen, H. H. Cheng, and J. Palen. Integrating mobile agent technology with multi-agent systems for distributed traffic detection and management systems. *Transportation Research Part C: Emerging Technologies*, 17(1):1–10, 2009.
11. A. Ciortea, O. Boissier, A. Zimmermann, and A. M. Florea. Give agents some rest: A resource-oriented abstraction layer for internet-scale agent environments. In *AAMAS*, pages 1502–1504, 2017.
12. M. d'Inverno et al. The dmars architecture: A specification of the distributed multi-agent reasoning system. *Autonomous Agents and Multi-Agent Systems*, 9(1):5–53, 2004.
13. J. Ferber. *Multi-agent systems: an introduction to distributed artificial intelligence*, volume 1. Addison-Wesley Reading, 1999.
14. T. Finin, R. Fritzson, D. McKay, and R. McEntire. Kqml as an agent communication language. In *ICIKM*, pages 456–463, 1994.
15. J. Hendler. Agents and the semantic web. *IEEE Intelligent systems*, 16(2):30–37, 2001.
16. S. Kraus. Negotiation and cooperation in multi-agent environments. *Artificial intelligence*, 94(1-2):79–97, 1997.
17. Y. Labrou and T. Finin. Semantics for an agent communication language. In *International Workshop on Agent Theories, Architectures, and Languages*, pages 209–214. Springer, 1997.
18. S. Luke, L. Spector, D. Rager, and J. Hendler. Ontology-based web agents. In *Proceedings of the first international conference on Autonomous agents*, pages 59–66. ACM, 1997.
19. Y. Luo, K. Liu, and D. N. Davis. A multi-agent decision support system for stock trading. *IEEE network*, 16(1):20–27, 2002.
20. S. A. McIlraith, T. C. Son, and H. Zeng. Semantic web services. *IEEE intelligent systems*, 16(2):46–53, 2001.