Towards context-based authorizations for interactions in Hypermedia-Driven Agent Environments - The CASHMERE framework

Alexandru Sorici[1] and Adina Magda Florea[1]

University Politehnica of Bucharest, Splaiul Independentei 313, Bucharest, Romania
{alexandru.sorici, adina.florea}@upb.ro

Abstract. Agent-oriented software engineering has recently seen a sustained effort towards the definition of a new class of Multi-Agent System design, called Hypermedia MAS, which promotes an alignment between MAS engineering and the Web architecture to enable development of large, open, dynamic and long-lived interaction systems. A major challenge in these envisioned MAS environments is enabling agents to *discover* the resources whose affordances they require. Hypermedia MAS design principles push for discovery and use of resources by exploiting the link structure of web resources, but little focus has been placed thus far in ensuring *authorized* access to the resources of a large MAS environment. To address this, we propose a framework for *context-based* authorizations for access and discovery of resources in a Hypermedia MAS, inspired by work on Attributed-Based Access Control and RDF Stream Reasoning. We detail the design of the framework functionality and the proposed integration with current Hypermedia MAS platforms, highlighting advantages, challenges and current limitations of the approach.

Keywords: Hypermedia MAS · Web-of-Things · Context · RDF Stream Processing · Context-Based Access Control.

1 Introduction

In recent years agent-oriented software engineering has seen a sustained contribution effort towards a vision that enables the deployment of world-wide hybrid communities of people and artificial agents, making use of the Web. A new class of multi-agent system (MAS) design is being defined, referred to as *Hypermedia MAS* [12], which posits that MAS engineering should be aligned with the web architecture so as to enable large, open, dynamic and long-lived interaction systems. The cornerstone of the approach is the use of semantic hypermedia to enable the interaction among heterogeneous entities in MAS, such as software agents, sensors, devices, services and people.

One leading engineering model within Hypermedia MAS [14] proposes an alignment between the Agent & Artifacts MAS development meta-model [26] and the Web-of-Things (WoT) W3C Thing Description (TD) specification [7].

The Agents & Artifacts model introduces an explicit dimension for programming of the environment of a MAS, which happens in terms of *Artifacts* and their deployment into various *Workspaces*. Artifacts encapsulate the functionality of digital services, sensors or actuators and expose their working in terms of observable properties and events, as well as actions that can be invoked on them. On the other hand, the W3C WoT TD specification describes a formal information model and a common representation for the Web-of-Things, where Things (e.g. web-enabled services, devices, sensors) are characterized by their property, event and action affordances which clients can use by means of RESTful interactions following the HATEOAS principles (Hypermedia As The Engine Of Application State). It is easy to see the similarity of the A&A and TD models which is why Hypermedia MAS platforms such as Yggdrasil [14] build on their integration, creating MAS environments which have an explicit web-resource based representation of the artifacts they contain.

A major challenge in developing application over large, open and dynamic hypermedia MAS environments is enabling agents to *discover* the resources whose affordances they require. While the design principles of Hypermedia MAS promote discovery by navigating the link structure constructed between WoT Things, there currently is no indication on how to search and use Web Things in an *authorized* manner, which would respect the access policies that heterogeneous designers wish to set in place for the Things they deploy in a large hypermedia MAS environment. Furthermore, there is no indication of a *process* by which authorization would be granted or revoked, which is suited to a large, open and dynamic environment.

Running Scenario To give an example of the mentioned challenges, we introduce a simple scenario that is a straightforward adaptation from the use case introduced in [14] where a digital assistant (modeled as a BDI agent) has to notify a person every time a relevant event occurs. The BDI agent is situatated in a hypermedia environment and is able to discover an artifact controlling a smart light bulb. The agent uses the light bulb to implement a blinking pattern that visually notifies the user of new events. Changing the color of the light helps distinguish between positive and negative notifications. Our adaptation of this scenario relies on adding more details to the situation, which quickly give rise to the need for authorized access. The visual notification service is desired by a university which implements a hypermedia MAS environment at the level of the whole campus. The university encourages each lab to be individually responsible for the smart devices it installs in their room, as long as they are made available in the hypermedia environment. However, the university considers that *discovery* of the artifacts wrapping over any smart device is only allowed for employed personnel who are physically present in the rooms where the devices reside, to prevent BDI agents of university visitors from interacting with the devices, as well as any kind of remote control.

Context The running scenario defines a situation where the *context* of a user (e.g. employment status, physical location) has to directly inform the interactions that the BDI agent of the user can execute in the hypermedia MAS environment. We interpret the notion of *context information* according to a general, application-specific definition given in the framework of Ambient Intelligence (AmI): "Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves" [18].

We also argue that the *operational perspective* [32] of logically partitioning context information along dimensions of engagement (e.g. individual, space, time, activity, relational) can be usefully exploited to inform a mechanism for *context-based* authorization of interactions within a hypermedia MAS. This view is further strengthened by the AmI perspective that specific enough *shared context* between two entities acts as a permission and even obligation for information exchange between the entities [22].

In light of the above, our contributions in this work are:

- Describe the need for and the design of a framework for Context-Aware Search and Discovery in Hypermedia MAS Environments (CASHMERE). The center focus of the approach is providing a method for *context-based authorization.*
- Present a development and integration road map, detailing how the CASHMERE framework can be integrated into the working of a Hypermedia MAS Environment. We present advantages, challenges and limitations of the envisioned approach.

The remainder of the article is structured as follows. Section 2 provides background on the Hypermedia MAS environments, frameworks to establish authorization policies in dynamic systems, as well as the use of RDF streaming technologies to implement context-based policy rules. Section 3 details the core functionality of the CASHMERE framework explaining the means for context representation and shared context identification. The design of the integration between CASHMERE and a Hypermedia MAS environment is presented in Section 4, while the challenges and current limitations of the approach are discussed in Section 5. We conclude the paper with the outline of upcoming development work in Section 6.

2 Background

We start by analysing the motivation behind and technologies that support our proposed framework. We submit that: (i) current principles underlying design of Hypermedia MAS Environments are incomplete with respect to authorized access to the resources they enable exploring, (ii) authorization in large scale, dynamic MAS environments must employ an equally dynamic access control mechanism, (iii) events and actions, collectively called *context information*, which are shared by agents and resources in an environment can *count-as* justification for authorizing agent access to a resource, and (iv) modeling the events and actions as RDF information streams is a natural and flexible means to reason about the conditions that count-as *sharing context*.

2.1 Hypermedia Driven Agent Environments

Ciortea et al. introduce three main principles for the design of Hypermedia Multi-Agent Systems [14]. The first principle promotes a uniform representation (e.g. in the form of an RDF graph) of resources and the relations between them in a hypermedia environment. One intended consequence of the uniform representation of relations between entities (e.g. agents, tools, organizations) is the improved ability to crawl and discover entities *of interest*. However, the text in [14] does not make clear how *interest* is defined and no distinction is made between *discoverability* (e.g. through a search engine) and then *use* of a resource by invoking the affordances it provides.

Crawling entities based on their uniform representation to build a *directory* of resources is seen as an effective means to exploit the distributed nature of hypermedia, while also enabling agents to go beyond *locality*. The latter concept seems to be interpreted as a limitation of FIPA-based MAS to only gain access to resources and service that are advertised by the agents themselves in a local network. However, another interpretation of *locality*, not addressed in [14], relates again to the *interest* of the interaction, to the conditions under which a resource or service is accessed. From this perspective of context management, it is desirable to keep information consumption and interactions *localized*, meaning that both provider and consumer of the affordances exposed by a resource engage with each other under an *authorization* granted by the existence of a *common context* (e.g. related to a shared space, a joint activity, a membership in an organizational structure).

Principles 2 and 3 from [14] advocate for the use of a single-entry point into an Hypermedia MAS environment, as well as the *observability* of resources. Taken together, these guidelines affirm that any resource in a Hypermedia MAS that is of potential *interest* to agents should make itself actively observable through its explicit representation and notification of changes to its state or affordances. Furthermore, once an entry-point in the resource representation of a MAS environment has been gained, link relations between entities in the environment should enable the exploration of other resources in the environment. These principles are required to design evolvable and long-lived hypermedia MAS, but they also in need of additional considerations with respect to practical deployments of hypermedia MAS. Uniform representation and observability can enable a machine readable description of security schemes (e.g. token based, OAuth2 based - see also Security Schema of WoT Thing Description [7]), but it does not define the *conditions* under which such a secured access to changes in states and affordances of a resource are obtained.

We claim that *observability* should be amended to consider a *common context* driven *authorization* mechanism that can narrow down what different resource developers consider should be of *interest* to different agents in the hypermedia MAS.

2.2 Dynamic Access Control

The WoT Thing Description [7] specification provides a vocabulary to set *schemes* in place (e.g. API key, Bearer, OAuth2) which *secure the access* to a resource. However, with the exception of OAuth2, no other modeled security scheme defines *authorizations* for the different resource affordances. Furthermore, even in the case of OAuth2, there is no model of a mechanism by which to decide which authorizations to include within the OAuth2 token depending on the *situation* (e.g. the capabilities and intention of the agent, the state of the environment).

The idea of an authorized exploration and use of resources in a hypermedia MAS is partially acknowledged in [15], where the challenges to *autonomy* in the WoT list the notion of *regulation as a first-class abstraction*, citing common practices of ensuring fair resource access, such as the Robots Exclusion Protocol, rate limiting or licensing policies. The authors bring forth normative MAS research [9,20,23] to mention that regulative norms and prescriptions can be used to specify and enforce how agents can interact with each other and their environment, enforced either through social means or a top-down authority manner. However, no concrete mechanism of integrating an authorization method into the workings of existing hypermedia MAS platforms (e.g. Yggdrasil [14]) is advanced.

The core of the dynamic access control problem in the context of hypermedia MAS poses the following question: how can resources signify to an agent the *set of conditions* and the *process of reasoning* about them which determines the granting or revocation of a permission to exploit an affordance of the resource? CASHMERE starts from the premise that application specific agent and environment *context* can *count as* the catalyst by which normative dimensions such as *permission / prohibition* are expressed and implemented at the level of a hypermedia MAS. This view is further supported by work in the domain of Access Control for the Internet-of-Things. The survey of Qiu et al. [24] highlights that in *dynamic* and *open* computing environments traditional access control models such as Role-Based Access Control (RBAC) are not adapted to fit application dynamics. For such cases an alternative model is gaining popularity[1][2], namely Attribute-Based Access Control (ABAC) [13] which proposes that subject requests to perform operations on a resource are granted or denied based on *attributes* of the subjects, resources or the environment and policies that relate to these attributes [27]. Extensions of the ABAC model which use an ontology to define roles and attributes and SWRL rules to infer additional attributes have also been proposed [19]. These include other external context sources for a richer

[1] NextLabs ABAC solution for business-critical data control: `https://www.nextlabs.com/products/technology/abac/`

[2] Styra - authorization as a service at scale: `https://www.styra.com/blog/dynamic-authorization-with-policy-based-access-management/`

attribute space and cases of multiple agents interacting with the same resource have been proposed.

The thought and motivation behind the CASHMERE proposal for context-based authorization is also founded on work in *situated artificial institutions* [16] which defines a framework for expressing and reasoning about *count-as* situations with respect to norms in agent organizations. Specifically, the SAI framework is concerned with relating normative regulation to some *interpretation* of the environment that *counts as* the constitution of role assuming, obligations, permissions or prohibitions.

While SAI is explicit in formalizing constitutive specifications in terms of rules for agent-, environment/event- and state-status functions, the CASHMERE framework is more pragmatic in its use of the *count-as* principle. In a manner to be detailed in Section 3.1, CASHMERE proposes a rule-based mechanism to identify the agent and thing related context information and the conditions under which these *count-as* a *shared context.* The shared context acts as a constitutive function that creates a *permission* of interaction between agents and the resource affordances they seek to use.

2.3 Modeling Context Information with CONSERT

CASHMERE proposes having an explicit model of *context* (agent abilities, environment state, organizational situation) and its *dynamics* (how context changes in time) as the underpinning for the mechanism by which authorized discovery of resources is implemented. To accomplish this, a model for *context representation* is required.

Context information representation relies on the CONSERT meta-model [29] which introduces the *work horse* representations of *ContextAssertions* and *ContextAnnotations*. *ContextAssertions* use the predicate in a subject-predicate-object triple as the main model entity. A statement such as `locatedAt(agent_alex, lab308)` is modeled as a binary *ContextAssertion*, whereby the central element is the fact of being `LocatedAt` and the subject and object entities are `agent_alex` and `lab308`. This form of reification has the advantage that it can naturally support the addition of supplementary information (*ContextAnnotations*) such as timestamp of assertion, temporal validity or provenance of the information. ContextAssertions are also characterized by a *mode of acquisition* which defines an operational attribute signaling how *dynamic* the assertion is. The CONSERT Model distinguishes between *static* (assertions which hold true indefinitely - e.g. the spatial containment of a room in a building), *profiled* (ContextAssertion who have a long-term, but still limited temporal validity - e.g. the employment status of a researcher), *sensed* (event-like ContextAssertions, who are assumed to change frequently in a system - e.g. the physical location of a person in a building) and *derived* (produced by some inference method whose input consists of other ContextAssertions) acquisition modes.

The ability to model annotations and to distinguish between sources / flows of context information is beneficial because it provides a clearer way to identify different sources of context, as well as to reason over its validity in time in a environment that captures dynamic events and actions of agents (see example in Listing 1.1 and Section 3.2.

2.4 RDF Stream Reasoning

The context-aware ABAC model introduced in [19] makes use of ontologies to express attributes and SWRL rules to define policies that implement access control. However, many WoT application scenarios involve conditions that are dynamic in nature (e.g. relate to mobility of agents, are tied to a cycle of activity) which require an interpretation of context information as it changes *in time*. As detailed further in Section 3, CASHMERE expresses rules to identify conditions for *shared context* using RDF Stream Processing [17] techniques (RSP). RDF Stream Processing has emerged in recent years as a collection of approaches (e.g. C-SPARQL [10], CQELS [21]) involving extensions to RDF representation and the SPARQL query language which are meant to address the *continuous processing* requirement of *semantic data streams*. This collection has been later unified under a single query model, RSP-QL [17], which can be interpreted in a prototype engine (Yasper [31]) and for which an API specification (RSP4J [30]) has been defined, that enables the construction of RDF Stream generators, consumers, as well as custom operators and interpretation engines.

RSP-QL defines the semantics of interpreting *time-varying* RDF graphs, it describes means to define the duration and trigger conditions for *evaluation windows* and it defines the functionality of relational-to-relational (equivalent to SPARQL 1.1 operational semantics), relational-to-stream (from solution mappings to RDF streams), stream-to-relational (from a stream to a single graph coalesced from the union of all RDF graphs within ane evaluation window) and stream-to-stream operators. The latter operators distinguish between modes of operation that allow for (i) generating a stream of solution mappings (RSTREAM) and (ii) determining which solution mappings have been newly added (ISTREAM) or removed (DSTREAM) with respect to those obtained from the previous evaluation window.

Since hypermedia MAS environments promote the explicit semantic representation of entities as web resources using RDF, the use of RSP within CASHMERE to reason about the attributes and context of the MAS environment and its actors is an obvious advantage. A further benefit is the ability to factor in reasoning over the temporal dimension of the context streams and extract streams of authorization grants and revocations, as will be detailed in Section 3.2.

3 Shared Context Identification

The core of the CASHMERE vision lies in the idea that the *context of the interaction* between agents and the resources in their environment is a *conditioning space* which can be expressed in sufficiently rich detail that it can leveraged to grant authorization for discovery and use of artifact affordances in a hypermedia-driven agent environment. The extremes of this context-based access range from having no condition whatsoever on the interaction (i.e. public access) to requiring a role that is specifically conditioned to be played by a single entity. For anything in between, it becomes highly relevant to design a method that is both flexible and comprehensive enough to include static and dynamic environment and agent-generated events into conditions that *count as* context *shared* by an agent and a resource. The shared context then warrants the *authorization* of the interaction.

In what follows, we describe our means of identifying *shared context* in terms of (i) how we can partition context information into *domains of interest* and (ii) how we express rules that determine whether two entities share the same *context domain*.

3.1 ContextDomains: Partitioning Context Information

The CONSERT context management deployment specifications [28] introduce two concepts that enable a system to logically partition the context information that it has to provision to consumers. *ContextDimensions* are *ContextAssertions* (from among the ones that a system handles) that define a *privileged* direction (e.g. spatial, activity related, relational) of context provisioning. Along each *ContextDimension* a set of *ContextDomain* (potentially hierarchically aranged) can be defined. In our running example an obvious spatial *ContextDimension* is given by the `locatedAt(Agent, UniversitySpace)` *ContextAssertion*, which gives rise to *ContextDomains* that refer to indoor locations of the university (such as `lab308`). Because indoor locations have a natural spatial inclusion relation (which can be captured by a *static ContextAssertion*, such as `containedIn(UniversitySpace, UniversitySpace)`, a hierarchy of *ContextDomains* becomes possible.

We can now posit that resources (e.g. devices, services) and consumers who are part of the information provisioning setup of the same *ContextDomain* (e.g. agents and devices in `lab308`) are inherently *sharing context*. Therefore, our method of shared context identification can resolve to verifying if two entities are *members* of the same *ContextDomain*. The next section defines the reasoning mechanism which interprets the conditions under which *ContextDomain* membership is granted or revoked.

3.2 Stream Processing for Shared Context Identification

In Section 2.4 we discussed RSP as an approach suited to implement the rules by which one or more entities are considered to share a context. The main advantage of this approach lies in the ability to process streams of RDF information which can encompass the *ContextAssertions* that are considered *sufficient* to denote *membership* in the same *ContextDomain*.

In the simplest case, the sufficiency criterion can limit itself to the observance of a *sensed ContextAssertion* that defines the *ContextDimension* and the instance of the *ContextDomain* which partition the context information. In our running example, the sensed *ContextAssertion* `locatedAt(agent_alex, lab308)` could *count as* sufficient to establish `agent_alex` as a member of the *ContextDomain* associated with Lab 308. However, in some cases (like in our running scenario) it is desirable to have conditions of *ContextDomain* membership which are more restrictive, considering that shared context is used to authorize access to artifacts whose affordances are ascribed to the same *ContextDomain*. In our example, the ability to discover the existence of the smart light bulb from Lab 308 and to control it is limited to agents that represent people who are employees of the university and are physically present in the room.

```
 1  PREFIX consert: <http://pervasive.semanticweb.org/ont/2017/07/consert/core/>
 2  PREFIX ann: <http://pervasive.semanticweb.org/ont/2017/07/consert/annotation/>
 3  PREFIX foaf: <http://xmlns.com/foaf/0.1/>
 4  PREFIX vcard: <http://www.w3.org/2006/vcard/ns#>
 5  PREFIX precis: <http://aimas.cs.pub.ro/consert/ontologies/precis#>
 6
 7  REGISTER STREAM <SharedLab308Context> AS
 8  CONSTRUCT ISTREAM {
 9      precis:lab308group vcard:member ?agent .
10  }
11  FROM NAMED :staticAssertions
12  FROM NAMED :profiledAssertions
13  FROM NAMED WINDOW :pLoc [RANGE PT10S STEP PT10S] ON STREAM :PersonLocated
14  WHERE
15  {
16      GRAPH :staticAssertions { ?agent rdf:type foaf:Person . }
17      GRAPH :profiledAssertions {
18          ?worksAssertion a precis:WorksAt ;
19              consert:assertionSubject ?agent ;
20              consert:assertionObject precis:upb ;
21              ann:hasAnnotation ?validAnn .
22          ?validAnn a ann:TemporalValidityAnnotation ;
23              ann:startTime ?employmentStart ;
24              ann:endTime ?employmentEnd .
25      }
26      WINDOW :pLoc {
27          ?persLocAssertion a precis:LocatedAt;
28              consert:assertionSubject ?agent ;
29              consert:assertionObject precis:lab308 .
30      }
31      BIND (xsd:dateTime(NOW()) AS ?date)
32      FILTER (?date > ?employmentStart && ?date < ?employmentEnd)
33  }
```

Listing 1.1. Demonstrator scenario shared context identification query

Listing 1.1 shows the SPARQL CONSTRUCT query that acts as a typical rule by which membership in a *ContextDomain* is determined. In line 9, the rule conditions already assume the existence of resource (`precis:lab308group`) denoting a *ContextDomain Group* modeled as an instance of `vcard`[3] organization to adhere to Web Access Control specifications [6] (see also Section 4). The CONSTRUCT statement is accompanied by an ISTREAM keyword (an example of a stream-to-stream operator) which signifies that the result of the query is an RDF Stream itself which will trigger with a new event only when the query produces a different output compared to previous time instances (cf. [17] for more detail on RDF stream operators). The body of the query distinguishes three context information input sources (different SPARQL graphs), depending

[3] `https://www.w3.org/TR/vcard-rdf/`

on the *mode* of acquisition: static assertions (which identify the agent - line 16), profiled assertions (the employment status of the agent - lines 17-25) and a *named window* defining the stream of *sensed* `PersonLocated` *ContextAssertion* instances (lines 26-30). RDF stream windows are defined using range (window duration) and step (the temporal slide from one window content evaluation timestamp to the next) parameters (line 13). The value of these parameters is dependent on the application use case and is required to be set in tune with the frequency of `PersonLocated` instances that arrive on the stream. Notice that, since the `precis:WorksAt` *ContextAssertion* instance is a *profiled* one, it is interrogated for its `TemporalValidityAnnotation` (lines 22-24), which is then used to check validity of the employment status (line 32).

In Listing 1.1 we see an example of the `ISTREAM` stream-to-stream operator, which streams new events only if they differ from the previous window evaluation. In a HyperAgent environment deployment, the *ContextDomainGroup* Artifact that manages the *ContextDomain* membership would also have a query registered that has an equivalent body, but a `DSTREAM` operator in the CONSTRUCT head. The `DStream` operator triggers with the events that existed in the previous window evaluation, but not the current one. This effectively enables a *ContextDomainGroup* Artifact to manage both new memberships (`ISTREAM`), as well as expiring ones (`DSTREAM`).

4 Integration in Hypermedia-Driven Agent Environments

As mentioned in Section 2.1, we consider hypermedia-driven agent environments (of which Yggdrasil [14] is an exponent) as a main type of platform benefiting from the CASHMERE framework. The general overview of the information and interaction flow that realizes the integration of CASHMERE into a hypermedia MAS platform is presented in Figure 1. Steps (1) and (2) summarize the functionality described in Section 3. Steps (3) and (4) highlight the fact that the context-aware authorization functionality can be exploited at different levels within a Hypermedia MAS - from adapting the interaction with each individual

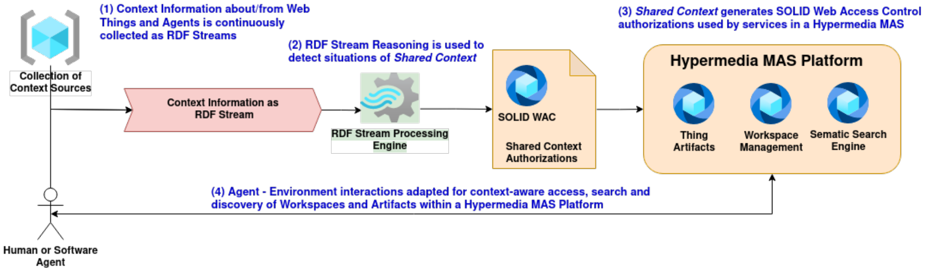
(1) Context Information about/from Web Things and Agents is continuously collected as RDF Streams
Collection of Context Sources
Context Information as RDF Stream
(2) RDF Stream Reasoning is used to detect situations of *Shared Context*
RDF Stream Processing Engine
SOLID WAC
Shared Context Authorizations
(3) *Shared Context* generates SOLID Web Access Control authorizations used by services in a Hypermedia MAS
Hypermedia MAS Platform
Thing Artifacts
Workspace Management
Sematic Search Engine
(4) Agent - Environment interactions adapted for context-aware access, search and discovery of Workspaces and Artifacts within a Hypermedia MAS Platform
Human or Software Agent

Fig. 1. General overview of the information and interaction flow that integrates the CASHMERE context-aware access functionality into a hypermedia-driven multi-agent platform.

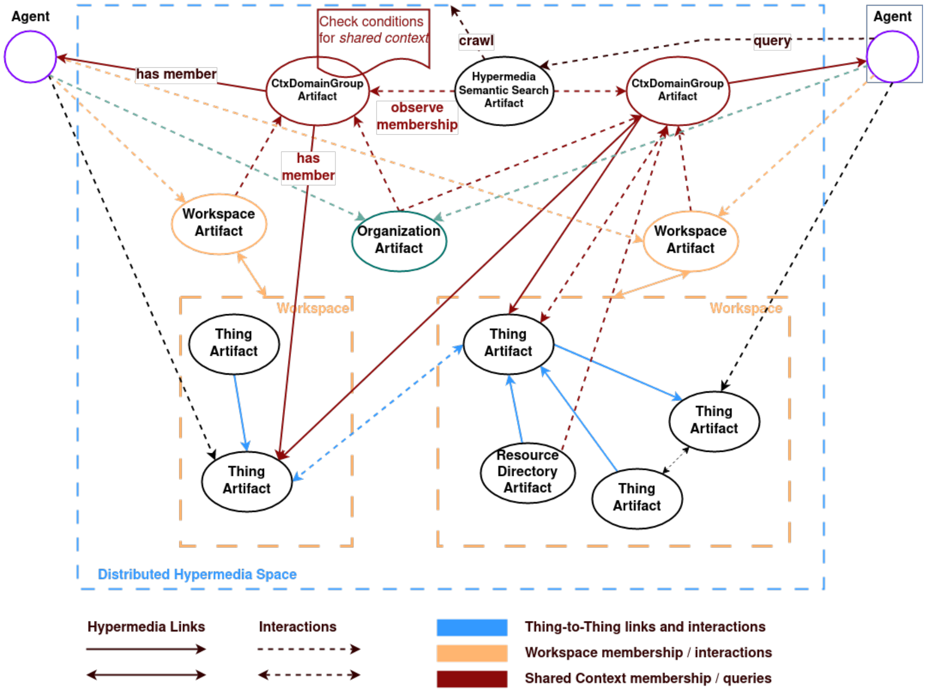

Fig. 2. Diagram showing the integration of ContextDomain Group Artifacts into a typical Agents and Artifacts hypermedia environment deployment

Thing Artifact to enhancing the functionality of a semantic hypermedia search engine.

Step (3) is further detailed integration-wise in the block diagram of Figure 2, which is designed as a reinterpretation of the conceptual overview of hypermedia MAS environments presented in works such as [14,11,25]. The diagram shows the envisioned composition of an Agents & Artifacts Container. Notice the addition of the artifacts managing *ContextDomain Groups*, which expose observable properties that signal the *membership* in the same *ContextDomain*. Things, Workspaces, ResourceDirectories, Organization Artifacts and Semantic Hypermedia Search Engines can subscribe to such observations and justify authorizing the access of an agent to all their affordances based on *sharing the same context*. This means that the effects of context-aware authorized access can reflect at several levels of granularity. For example, Thing and Workspace artifacts may refuse a *focus* request from agents that do not share any context. A Hypermedia Search Engine can omit sending notifications (e.g. through WebSub) about answers to queries of agents who do not share the same *ContextDomain* as the Things that are a response to their queries.

Notice that the *ContextDomains* form a separate logical partitioning than that of Workspaces or Organizations existing in the MAS Hypermedia Environment. Thing Artifacts from several Workspaces can be part of the same ContextDomain. This decouples the design of the *deployment* (Workspaces) means from the design and implementation of the *conditions for access* to functionality, which do not have to be programmed in from the start and thus have the capability to evolve.

To transform the identified shared *ContextDomain* membership into an actionable authorization mechanism the CASHMERE framework makes use of the Web Access Control (WAC) specifications [6] that are part of SOLID [5]. The first requirement in WAC is that all entities for whom an authorization is to be defined have to be identifiable by a WebID [8]. At the current stage, the shared context identification mechanism in CASHMERE only requires a URI that uniquely identifies a an entity (agent, artifact). However, reliance on WebID ensures that the *shared context* based authorization process can be doubled in security by means of *authenticating* agents based on public keys stored in the FOAF [3] profile reference by the WebID.

WAC further specifies that each web resource requiring an authorized access must advertise the *Access Control List* (ACL) resource that contains the authorizations to the protected resource. It must do so by responding to a HTTP request including a Link header with the `rel` value of `acl`. An ACL resource can expose an RDF document which lists instances of `acl:Authorization` (see the ACL Ontology [2]). An `acl:Authorization` will specify: (i) the resource for which it provides an authorization (`acl:accessTo`), (ii) the access mode (e.g. `acl:Read, acl:Write, acl:Control`) and (iii) whom the authorization applies to (e.g. `acl:agent, acl:agentClass, acl:agentGroup`). The `acl:agentGroup` mode of identifying authorization subject is particularly suitable for the CASHMERE setup, because it allows identifying an instance of a `vcard:Group` which can contain individual FOAF profiles as members. This maps directly to the *ContextDomain* membership CONSTRUCT outputs that have been presented in Section 3.2.

An ACL resource representation also includes an `acl:default` predicate which specifies the container resource in a hierarchy of containment, whose Authorization can be applied by default when no custom Authorization is defined for an individual protected resource. In a Hypermedia MAS as defined in [14,25], the *Workspace* hierarchy governing the deployment of Thing Artifacts can be used to manage an Authorization hierarchy. In our running scenario, the Authorization resource which enables access to smart light in Lab 308 to university employees physically present in the room can be attached to the *Workspace* containing the smart light Thing (and, possibly, other Things) instead of the Thing itself.

Figure 3 summarizes the way in which the representation and functionality of entities in a Hypermedia MAS environment have to be complemented to make use of the WAC-based authorized access proposed in CASHMERE. Each artifact that implements the functionality of a Thing or Workspace is additionally tasked with exposing a representation for the ACL resource that contains the authorizations defined for the artifact, which include access permissions granted by the observed memberships in different *ContextDomain* Groups (Lab308 in our

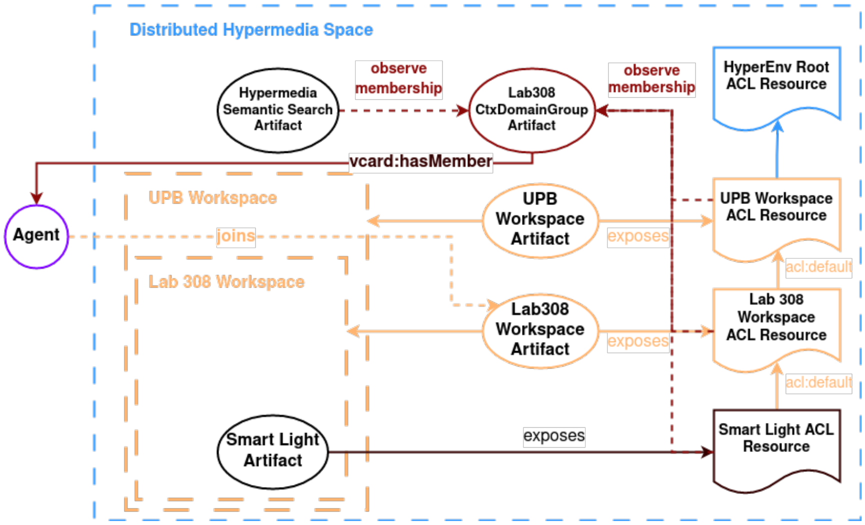
Distributed Hypermedia Space
Hypermedia Semantic Search Artifact
observe membership
Lab308 CtxDomainGroup Artifact
observe membership
HyperEnv Root ACL Resource
vcard:hasMember
Agent
joins
UPB Workspace
Lab 308 Workspace
UPB Workspace Artifact
exposes
UPB Workspace ACL Resource
acl:default
Lab308 Workspace Artifact
exposes
Lab 308 Workspace ACL Resource
acl:default
Smart Light Artifact
exposes
Smart Light ACL Resource

Fig. 3. Summarized view of WAC usage based on shared *ContextDomain* for typical artifact instances - Thing, Workspace, Semantic Search Engine - encountered in a Hypermedia MAS Environment.

running example). The artifacts that want to enable context-aware authorized access need to also implement the *authorization match* procedure as indicated by WAC specifications [6], which involves running a SPARQL ASK query over the RDF graphs containing `acl:Authorization` instances. In particular, for checking memberships produced by the *ContextDomain* membership streaming procedure detailed in Section 3.2, artifacts have two options: (i) use a *federated* SPARQL query for the group membership verification, running it against the RDF graph stored by the *ContextDomainGroup* artifact, (ii) use the stream output of the *ContextDomainGroup* artifact to keep local caches of *ContextDomain* memberships and run the query against the latter. Notice that Workspace hierarchy can be exploited to address default authorizations, where the root most ACL resource is defined at the level of the Hypermedia MAS environment itself.

Semantic Search Engines service, like the one introduced in [11], are meant to enable discovery of Thing Artifacts by *type* or *functionality descriptions*. When observing *ContextDomain* memberships, the functionality of the search engines can be adapted such that the result bindings that are answers to subscribed queries are filtered to contain only Artifact instances that share the same context as the agent making the subscription.

5 Discussion

The described functioning of CASHMERE and its integration into a Hypermedia MAS environment has certain advantages, but it is not without its challenges and limitations, which we discuss in the following.

Advantages Authorized access to a resource based on membership in a *ContextDomain* is a conceptually simple, but effective mechanism for dynamic access control, precisely because *shared context* in WoT applications is commonly decided based on privileged dimensions of differentiation (e.g. a spatial location *ContextDimension* in our running example). Applications have the flexibility to determine what information from the agent, environment and state discourse space best qualifies as *distinguishing* context and, thus, constitute the objects of those *ContextAssertions* as *ContextDomains*. For each established *ContextDomain*, the RSP-based rules can further constrain or loosen the conditions under which membership in the *ContextDomain* is granted. Also noteworthy is the ability to pre-seed the artifact workspaces with default access control policies, which can be based on classical role-based conditions, and leave only the dynamic aspects of an application domain to be managed under the *ContextDomain* membership premise. The SOLID specification for default policy hierarchies and policy resolution ensures that the authorization procedure remains consistent.

From a technical perspective, the proposed working of CASHMERE is a convenient implementation fit to existing hypermedia MAS platforms, such as Yggdrasil. The RSP4J API enables extending artifact functionality to operate as both generators (to feed the context information streams) and consumers (to make use of the stream of *ContextDomain* membership granting or revocations) of RDF streams.

Challenges For the development roadmap, several design and implementation challenges stand before. An initial observation to be made is that we made no assumption about the sources of the context information streams. In particular, we currently place no restriction on whether the source of context information is found only within the artifacts deployed in a hypermedia MAS environment, or whether they can also be external to the environment (but capture information *about* events in the environment). For within environment sources, the main technical challenge lies in developing the interface through which existing artifacts can turn their observable properties and events into RDF streams. For external sources, either direct usage of the RDF4J API or platforms such as OntopStream [4] (which performs streaming semantical data access from heterogeneous sources such as Kafka, Kinesis or JDBC databases), could be used.

An additional implementation challenge relates to the artifact functionality extensions required to evaluate access policies, as well as to perform policy conflict resolution. To address these, the CASHMERE framework proposes following the SOLID Access Control Policies specifications [1].

A design issue currently still under investigation is how different entities of a hypermedia MAS environment react upon authorization revocations. Should an access denial imply that the artifact be not discoverable (in a manner similar to the no-crawl policies used for websites), or should the artifact remain observable (and describable) in the workspace, but unfocusable by an agent? The former option is simpler to manage and safer conceptually. However, it contradicts the *Observability* principle mentioned in [14] and could burden development of use cases where agents wish to use *planning* methods to compose a future functionality, even though the *current* context denies them access to the artifacts required in the plan result. The latter option implies that artifact affordances remain discoverable at all time but that their *use* is conditioned by shared context. In this case, a method for *explaining* denied use is required, such that any planning methods can understand what *context* the agent needs to be a part of to gain access to the artifact functionality. The reified form of *ContextAssertions* and parsable SPARQL syntax of membership rules already makes it feasible to identify the list of conditions and find references to agents (using the `assertionSubject` predicate) that are bound by them. However, further research is required to develop an appropriate method to offer easily consumable access approval or denial explanations to agents.

Limitations One point that is relevant in MAS interactions, but currently not addressed by CASHMERE is making the distinction between an agent and a *client* (e.g. another artifact) acting on behalf of the agent. *ContextDomain* membership is determined with respect to the agent requesting access and the artifacts which it can potentially use. Potential support in this issue is switching from use of the SOLID ACL [2] ontology to the more comprehensive ACP ontology [1] which makes the distinction clear, but the underlying issue of having a means to determine whether an agent intention is behind a *linked* artifact operation invocation remains an open problem, requiring artifacts to explicitly advertise the issuer of their original operation invocation.

On the technical side, the CASHMERE framework currently makes no indication on the way in which to perform periodic evaluation of the RSP queries for *ContextDomain* verification. The default is to use the *step* parameter indications for window definitions. However, depending on the application, an evaluation triggered by the arrival of a new *ContextAssertion* might be more computationally appropriate than periodic re-evaluations. Instrumenting application specific guidelines and configuration options for membership rule trigger conditions remains an aspect of future work.

6 Conclusion

In this paper we presented the current state of design principles for hypermedia multi-agent system platforms which give rise to current instances, such as Yggdrasil. We highlighted that, while these principles encourage development of large-scale and long-lived agent interaction spaces, they do not cover the relevant aspect of managing an authorized access to the resources exposed in hypermedia environments. Building on work and ideas from domains such as Context Management in Ambient Intelligence (Sections 2.3 and 3.1), Dynamic Access Control and Situated Artificial Institutions (2.2), as well as RDF Stream Processing (Sections 2.4 and 3.2) we presented the CASHMERE framework, whose purpose it is to provide a solution for authorized resource access in a hypermedia MAS environments based on the premise of *shared context.* We further presented the design of the integration of CASHMERE into existing hypermedia MAS solutions which adopt the Agents & Artifacts paradigm as their core abstraction (Section 4).

In future work we plan to first focus on the development roadmap of the CASHMERE functionality laid out in Section 3 by leveraging the RSP4J API [30] and the Yasper [31] engine to build an artifact implementing shared *ContextDomain* identification rules. A subsequent development effort targets implementation of the SOLID ACP policy evaluation functionality, which must be available at the level of several key components of a hypermedia MAS environment (e.g. individual artifact, workspace and semantic search engine). In longer term research we plan to provide point wise guidelines and solutions to the identified challenges and limitations of the CASHMERE framework.

Acknowledgement

This work has been supported by funding under grant agreement PN-III-P1-1.1-PD-2021-0756 from the Romanian National Research, Development and Innovation Plan.

References

1. Access control policy specification. `https://solid.github.io/authorization-panel/acp-specification`, accessed: 2023-02-15
2. Acl ontology. `http://www.w3.org/ns/auth/acl`, accessed: 2023-02-15
3. Foaf vocabulary specification. `http://xmlns.com/foaf/0.1/`, accessed: 2023-02-15
4. Ontopstream development repository: streaming semantical data access of relational data streams. `https://github.com/chimera-suite/OntopStream`, accessed: 2023-02-17
5. Solid project. `https://solidproject.org/`, accessed: 2023-02-15
6. Web access control specification. `https://solid.github.io/web-access-control-spec`, accessed: 2023-02-15
7. Web of things (wot) thing description 1.1, w3c candidate recommendation. `https://www.w3.org/TR/wot-thing-description/`, accessed: 2023-02-15
8. Webid specifications. `https://www.w3.org/2005/Incubator/webid/spec/`, accessed: 2023-02-15
9. Balke, T., da Costa Pereira, C., Dignum, F., Lorini, E., Rotolo, A., Vasconcelos, W., Villata, S.: Norms in mas: definitions and related concepts. In: Dagstuhl Follow-Ups. vol. 4. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik (2013)
10. Barbieri, D.F., Braga, D., Ceri, S., VALLE, E.D., Grossniklaus, M.: C-sparql: a continuous query language for rdf data streams. International Journal of Semantic Computing **4**(01), 3–25 (2010)
11. Bienz, S., Ciortea, A., Mayer, S., Gandon, F., Corby, O.: Escaping the streetlight effect: Semantic hypermedia search enhances autonomous behavior in the web of things. In: Proceedings of the 9th International Conference on the Internet of Things. pp. 1–8 (2019)
12. Boissier, O., Ciortea, A., Harth, A., Ricci, A.: Autonomous agents on the web. In: Dagstuhl-Seminar 21072: Autonomous Agents on the Web. p. 100p (2021)

13. Bonatti, P.A., Samarati, P.: A uniform framework for regulating service access and information release on the web. Journal of Computer Security **10**(3), 241–271 (2002)
14. Ciortea, A., Boissier, O., Ricci, A.: Engineering world-wide multi-agent systems with hypermedia. In: Engineering Multi-Agent Systems: 6th International Workshop, EMAS 2018, Stockholm, Sweden, July 14-15, 2018, Revised Selected Papers 6. pp. 285–301. Springer (2019)
15. Ciortea, A., Mayer, S., Boissier, O., Gandon, F.: Exploiting interaction affordances: on engineering autonomous systems for the web of things (2019)
16. De Brito, M., Hübner, J.F., Boissier, O.: Situated artificial institutions: stability, consistency, and flexibility in the regulation of agent societies. Autonomous Agents and Multi-Agent Systems **32**, 219–251 (2018)
17. Dell'Aglio, D., Della Valle, E., Calbimonte, J.P., Corcho, O.: Rsp-ql semantics: A unifying query model to explain heterogeneity of rdf stream processing systems. International Journal on Semantic Web and Information Systems (IJSWIS) **10**(4), 17–44 (2014)
18. Dey, A.K.: Understanding and using context. Personal and ubiquitous computing **5**, 4–7 (2001)
19. Dong, Y., Wan, K., Huang, X., Yue, Y.: Contexts-states-aware access control for internet of things. In: 2018 IEEE 22nd International Conference on Computer Supported Cooperative Work in Design ((CSCWD)). pp. 666–671. IEEE (2018)
20. Hübner, J.F., Boissier, O., Kitio, R., Ricci, A.: Instrumenting multi-agent organisations with organisational artifacts and agents: "giving the organisational power back to the agents". Autonomous agents and multi-agent systems **20**, 369–400 (2010)
21. Le-Phuoc, D., Dao-Tran, M., Xavier Parreira, J., Hauswirth, M.: A native and adaptive approach for unified processing of linked streams and linked data. In: The Semantic Web–ISWC 2011: 10th International Semantic Web Conference, Bonn, Germany, October 23-27, 2011, Proceedings, Part I 10. pp. 370–388. Springer Berlin Heidelberg (2011)
22. Olaru, A., Florea, A.M., El Fallah Seghrouchni, A.: A context-aware multi-agent system as a middleware for ambient intelligence. Mobile Networks and Applications **18**(3), 429–443 (2013)
23. Ossowski, S.: Agreement technologies, vol. 8. Springer Science & Business Media (2012)
24. Qiu, J., Tian, Z., Du, C., Zuo, Q., Su, S., Fang, B.: A survey on access control in the age of internet of things. IEEE Internet of Things Journal **7**(6), 4682–4696 (2020)
25. Ricci, A., Ciortea, A., Mayer, S., Boissier, O., Bordini, R.H., Hübner, J.F.: Engineering scalable distributed environments and organizations for mas. In: Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS), 2019, Canadá. (2019)
26. Ricci, A., Piunti, M., Viroli, M.: Environment programming in multi-agent systems: an artifact-based perspective. Autonomous Agents and Multi-Agent Systems **23**, 158–192 (2011)
27. Servos, D., Osborn, S.L.: Current research and open problems in attribute-based access control. ACM Computing Surveys (CSUR) **49**(4), 1–45 (2017)
28. Sorici, A., Picard, G., Boissier, O., Florea, A.: Multi-agent based flexible deployment of context management in ambient intelligence applications. In: Advances in Practical Applications of Agents, Multi-Agent Systems, and Sustainability:

The PAAMS Collection: 13th International Conference, PAAMS 2015, Salamanca, Spain, June 3-4, 2015, Proceedings 13. pp. 225–239. Springer (2015)

29. Sorici, A., Picard, G., Boissier, O., Zimmermann, A., Florea, A.: Consert: Applying semantic web technologies to context modeling in ambient intelligence. Computers & Electrical Engineering **44**, 280–306 (2015)
30. Tommasini, R., Bonte, P., Ongenae, F., Della Valle, E.: Rsp4j: An api for rdf stream processing. In: The Semantic Web: 18th International Conference, ESWC 2021, Virtual Event, June 6–10, 2021, Proceedings 18. pp. 565–581. Springer (2021)
31. Tommasini, R., Della Valle, E.: Yasper 1.0: Towards an rsp-ql engine. In: ISWC (Posters, Demos & Industry Tracks) (2017)
32. Zimmermann, A., Lorenz, A., Oppermann, R.: An operational definition of context. In: Modeling and Using Context: 6th International and Interdisciplinary Conference, CONTEXT 2007, Roskilde, Denmark, August 20-24, 2007. Proceedings 6. pp. 558–571. Springer (2007)