Semantic Interoperability for Web Services based Smart Home Systems

Hannu Järvinen and Petri Vuorimaa
Department of Media Technology, Aalto University School of Science, Espoo, Finland

Keywords: Ambient Intelligence, Building Automation, FIPA, Interoperability, JADE, Multi-agent System, oBIX, Semantic Web, Smart Home, Web Intelligence, Web of Things, Web Services, WebSocket.

Abstract: One of the key issues in smart home systems is the lack of interoperability between available solutions. While data representation and communication formats need to be unified, the goal is to provide intelligence for the system control. In this paper, we present a solution for Web services based smart home systems to publish semantic data, and to communicate with agent systems. Communication of browser-based agents with a smart home system is demonstrated. Efficiency of the system is measured and the results provided. The presented solution provides a basis for exploiting intelligence from multi-agent systems in smart home system control.

1 INTRODUCTION

The main issue with smart home interoperability is that products from different manufacturers are incompatible with each other, and even products of the same brand are unlikely to really interoperate. A smart refrigerator can, for example, send tweets or communicate its content to a smartphone application, but is unable to share data with another smart device. True interoperability can be achieved when heterogeneous smart components can utilize data and suggest actions from and to each other. If manufacturers would agree on common standards, the problem would be nonexistent as the systems would be interoperable by design. Unfortunately, while the vision of smart buildings is nothing new, agreeing on the standards between larger groups of companies has been unsuccessful, except in entertainment electronics.

Smart home interoperability can be separated into four different layers. At the bottom, syntactic interoperability layer considers the capability of devices to be connected into the same system. Functional layer considers inter-device logic that enables actions to be executed in one component based on the information from another component. On semantic layer, data is presented semantically, categorized using context ontologies, and linked to other semantic data. Intelligence layer provides services for controlling the smart home in more sophisticated fashion, including, e.g., adaptive logic for automatizing the smart home behavior and programming. The work presented in this paper concentrates on realizing the semantic layer in a smart home, and enabling the use of the intelligence layer with standard agent technology support.

We consider two practices for implementing the semantic layer. First, we use a standard extensible smart home data format that allows developers to define their own abstractions. This creates simple semantics for the data by categorizing different smart home concepts. Second, we define an ontology based on this smart home Web language, and transform the complete data model into a corresponding semantic representation. The transformation allows representing smart home data in RDF/OWL. To enable the use of the intelligence layer, we implement an agent to handle the communication between smart home and agent systems. Further, we demonstrate interoperability between the smart home system and browser agents.

To realize the above contributions, we present details of the semantic transformation, and the agent platform integration with the smart home system. Measurements are made to calculate the system delays caused by the RDF transformation and the agent platform integration, and to demonstrate the communication with a browser agent. The smart home system is based on open Building Information eXchange (oBIX) (Considine, 2010) specification and the integrated agent platform is the Java Agent DEvelopment Framework (JADE) (Bellifemine et al., 1999) compliant with the Foundation for Intelligent Physical Agents (FIPA).

Järvinen H. and Vuorimaa P..
Semantic Interoperability for Web Services based Smart Home Systems.
DOI: 10.5220/0005201501410148
In *Proceedings of the International Conference on Agents and Artificial Intelligence* (ICAART-2015), pages 141-148
ISBN: 978-989-758-073-4

2 RELATED WORK

Extensive amount of research has focused on smart home interoperability (Jiang et al., 2004). Many studies have concentrated on connecting heterogeneous devices and subsystems together, and providing a unified interface on top (Perumal et al., 2010; Wang et al., 2007; Tokunaga et al., 2002; Miori et al., 2006). The approach is applicable for limited amount of devices, as protocol converters and integration platforms need to convert data separately for each protocol. Huge number of different protocols makes the approach cumbersome and sometimes impossible due to use of proprietary protocols.

One of the protocols for the unified high-level Web services interface is open Building Information eXchange (oBIX) (Considine, 2010) from the Organization for the Advancement of Structured Information Standards (OASIS). The data format is based on XML and defines, similarly to any common programming language, a small set of primitive data types for describing the data. It also allows developers to define data structures for their own devices and systems.

Recently, the use of semantic technologies in smart environments has been suggested by many authors (Chen et al., 2009; Wang et al., 2004a; Zhang et al., 2005). It provides a way to represent data on a higher semantically meaningful level, and share common understanding of the concepts using ontologies. As opposed to attempts of standardizing smart home protocols, semantic community is cooperating and the standardization of semantic formats has been successful. In addition, different application domains can define their own ontologies. In turn, these ontologies can link concepts and properties to common ontologies, for example, in Linking Open Data[1] (LOD) cloud. Thus, applications can understand data from different domains and interoperate.

Various studies propose ontology-based context models for representing smart home context information semantically (Gu et al., 2004; Kim and Choi, 2006; Xu et al., 2009). Ontologies, such as CobraOnt (Chen et al., 2003) and CONON (Wang et al., 2004b) have been defined to model context information. Such research interest in using ontologies suggests that adding a semantic layer in smart home can be useful for developing intelligent applications.

Use of Multi-Agent System (MAS) for the building control has also been studied by, e.g., (Huberman and Clearwater, 1995). They present a system using market-based MAS to provide solution to the problem of thermal resource distribution. However, to generalize the idea, detailed research has to be done on the integration techniques between the building systems with the MAS, including data transformation and semantic presentation.

Sashima et al. developed an agent-based context-mediated Web service coordination framework in ubiquitous computing (Sashima et al., 2005). In their system, each agent is a proxy between SOAP Web services and FIPA Agent Communication Language (ACL). They suggest solutions to coordination issues such as context-aware matchmaking and representation alignment. Similarly to their work, we transform standard smart home Web services language into FIPA ACL and RDF/OWL. However, in our approach the transformation is handled by one centralized agent. Thus, individual system components only need to follow standards from their own domain.

Coyle et al. suggested a sensor fusion-based middleware for smart homes, called ConStruct (Coyle et al., 2007). Similarly to an integration platform, the solution interconnects heterogeneous devices and presents the information with a unified format, in this case, RDF. In their earlier research (Coyle et al., 2006), they outlined the idea of transforming XML information from oBIX via XSLT to RDF and illustrated it with the example data of a simple thermostat from the oBIX specification. However, the transformation is not presented in detail and generated semantic data seem not to follow any static ontology. In addition, the transformation is lossy. In Section 4.1, we compare this with our ontology-based transformation.

3 SEMANTIC INTEROPERABILITY

Semantic representation of the smart home information can be used to facilitate the interoperability. As stated above, on semantic interoperability layer data is presented semantically, categorized using context ontologies, and linked to other semantic data. In addition to the use of standard Semantic Web technologies, other techniques are also considered. Low-level device protocols and syntactic layer Web based data formats can also provide support for describing semantics.

The semantic support in our smart home system consists of three layers. Figure 1 illustrates how layers build from simple to more complex and interoperable semantics. *Smart Home Web Service Language Layer:* The most basic semantics are defined with primitive oBIX data types. They can be used to categorize data into different object types, e.g., boolean and string value objects. Then, developers can define

[1]The Linking Open Data (LOD) cloud - http://lod-cloud.net/

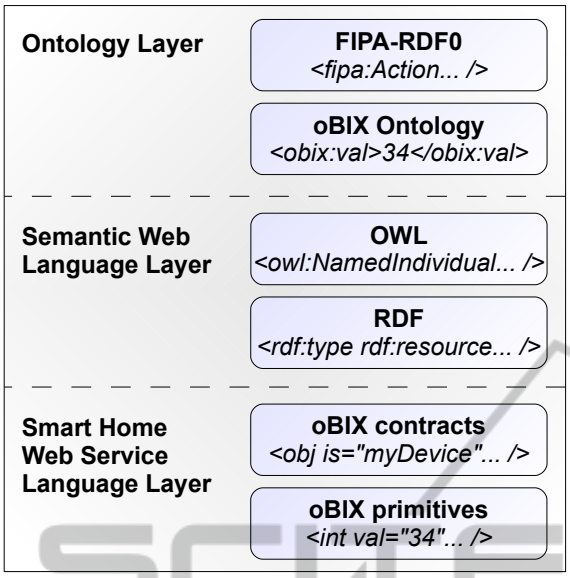

Figure 1: Semantic interoperability layers.

concepts with prototype structures, called oBIX contracts. This extensibility makes oBIX a strong candidate when choosing a smart home Web language. *Semantic Web Language Layer:* On top of the oBIX primitives and contracts, Semantic Web technologies are exploited. With RDF we can represent the data as subject-predicate-object triples, and make statements about the Web resources as smart home information. OWL makes the language more expressive and provides a standard way to define ontologies. *Ontology Layer:* With these tools, we are able to define oBIX ontology for describing the oBIX smart home data in Semantic Web. We are also able to transform the oBIX XML data into RDF/OWL in compliance with the oBIX ontology. Now, we can exploit other ontologies alongside with the oBIX ontology. As we are integrating an agent platform with the smart home system, we can use FIPA-RDF0 ontology for the standard presentation of the ACL message content. We could also define generic smart home interoperability ontology for describing the device attributes and providing device and vendor specific annotations, e.g., links to device manuals, software updates, and product catalogs.

4 IMPLEMENTATION

The objective of the research was to provide semantic interoperability layer for Web based smart home system, and enable intelligence layer with support for standard FIPA compliant agents to communicate with the smart home system. Additionally, we wanted to allow browser-based agents to communicate with the system.

The smart home server used in the implementation is a further development result of our Java based oBIX server. Required features were developed to support semantic technologies and the integration with an agent platform. JADE agent platform was integrated with the oBIX server using a gateway agent that communicates with both systems. JADE is FIPA compliant and one of the most used agent platforms available.

The main components of the system are illustrated in Figure 2. Integration of the oBIX Server with the MAS is realized through the Gateway Agent. RDF Transformer in the oBIX server is used to convert the oBIX XML data into corresponding RDF/OWL representation. The integrated agent platform utilizes WebSocket communication when communicating with external agents. The Gateway Agent functions as a bridge between the two worlds, oBIX and agents. Any agent communicating with the Smart Home System need to have an ontology alignment with FIPA-RDF0 but also with oBIX ontology.

4.1 Ontology and Transformation

We defined oBIX ontology to represent the device information semantically. Now, with the ontology there is an explicit way of describing the oBIX data in RDF/OWL format. When comparing our method with the one in (Coyle et al., 2006), the transformation process and the semantic representation are different. Their example illustrated a transformation of, e.g., oBIX name facet values into RDF predicates. Thus, it is impossible to define an ontology, as every oBIX XML element with a new name would require an update to the ontology. Additionally, their transformation is lossy as only part of the information from oBIX XML format is included in the RDF representation. Our transformation instead generates RDF/OWL that is based on the oBIX ontology. The ontology covers every possible oBIX element. It defines classes for all the oBIX primitives, and properties for their facets. As the ontology covers the whole oBIX object model, the transformation is also lossless. A transformation example is depicted in Figure 3. Each XML element in oBIX is transformed into OWL NamedIndividual element. While oBIX has element hierarchy, RDF/OWL lines up all NamedIndividual elements to the same depth. Hierarchy information is stored using hasChild object property.

In practice, RDF Transformer module was developed for the oBIX server, and two different XSLT transformation files were created. The first transfor-

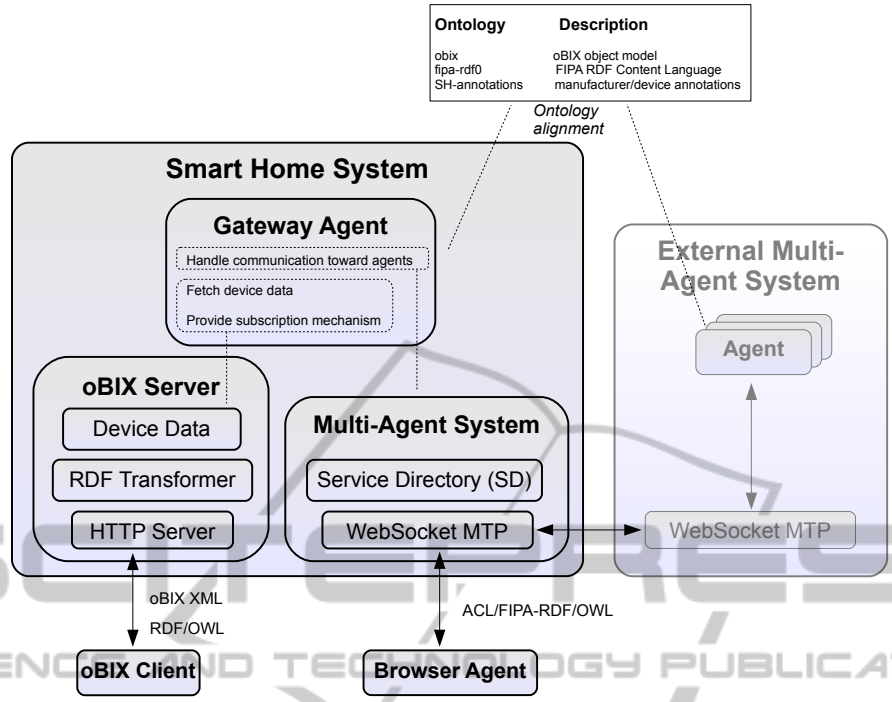

Figure 2: Integration architecture of the smart home system.

```
<obj name="temperatureSensorDevice" href="http://localhost:8080/tempsensor">
   <real name="sensorvalue" href="/tempsensor/value" val="20.809423712614205"
     displayName="Temperature Sensor Value" writable="true"/>
</obj>
```

```
<rdf:RDF xmlns="http://www.obix.org/resource/obix#" xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:owl="http://www.w3.org/2002/07/owl#" xmlns:xs="http://www.w3.org/TR/2008/REC-xml-20081126#"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <owl:NamedIndividual rdf:about="http://obixserver/tempsensor#obj">
      <rdf:type rdf:resource="http://www.obix.org/resource/obix#Obj"/>
      <name>temperatureSensorDevice</name>
      <href>http://obixserver/tempsensor</href>
      <hasChild rdf:resource="http://obixserver/tempsensor/value#real"/>
   </owl:NamedIndividual>
   <owl:NamedIndividual rdf:about="http://obixserver/tempsensor/value#real">
      <rdf:type rdf:resource="http://www.obix.org/resource/obix#Real"/>
      <name>sensorvalue</name>
      <href>/tempsensor/value</href>
      <displayName>Temperature Sensor Value</displayName>
      <writable>true</writable>
      <val>20.809423712614205</val>
   </owl:NamedIndividual>
</rdf:RDF>
```

Figure 3: An example transformation from oBIX XML to RDF/OWL.

mation generates pure RDF/OWL compliant with the oBIX ontology, and the second transformation additionally takes into account the FIPA-RDF0 ontology as it is used to generate data for the agent systems. Contracts in oBIX are not represented as classes in the oBIX ontology because that would require dynamic updates to the ontology itself. However, that could be useful in some application scenarios, and could be implemented as a separate oBIX contract ontology.

The oBIX ontology provides a common vocabulary for the agents to handle the smart home information. Additionally, it can be exploited on the intelligence layer for making inferences from the oBIX data to provide reasoning and find inconsistencies. Thus, while it here functions as a common semantic data representation format with shared vocabulary, it also enables further applications to be built on top.

4.2 Data Interfaces

The implemented system provides three different data interfaces for the communication with external ap-

Request:

```
(SUBSCRIBE
 :sender ( agent-identifier :name BrowserAgent@browser :addresses (sequence ws://browserAgent:7779/acc))
 :receiver (set ( agent-identifier :name obixagent@130.233.46.104:1099/JADE :addresses (sequence ws://130.233.46.104:7778/acc)) )
 :content "<?xml version='1.0'?>
<rdf:RDF xmlns:rdf='http://www.w3.org/1999/02/22-rdf-syntax-ns#'
         xmlns:fipa='http://www.fipa.org/schemas#'
         xmlns:owl='http://www.w3.org/2002/07/owl#'
         xmlns:obix='http://www.obix.org/resource/obix#'>
   <rdf:Description rdf:about='http://130.233.46.104:8080/tempsensor/value#real'/>
</rdf:RDF>")
```

Response:

```
(INFORM
 :sender  ( agent-identifier :name obixagent@130.233.46.104:1099/JADE  :addresses (sequence ws://130.233.46.104:7778/acc ))
 :receiver  (set ( agent-identifier :name BrowserAgent@browser  :addresses (sequence ws://browserAgent:7779/acc )) )
 :content  "<?xml version=\"1.0\" encoding=\"UTF-8\"?>
<rdf:RDF xmlns:rdf=\"http://www.w3.org/1999/02/22-rdf-syntax-ns#\"
         xmlns:fipa=\"http://www.fipa.org/schemas#\"
         xmlns:owl=\"http://www.w3.org/2002/07/owl#\"
         xmlns:obix=\"http://www.obix.org/resource/obix#\"
         xmlns:xs=\"http://www.w3.org/TR/2008/REC-xml-20081126#\"
         xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\">
   <owl:NamedIndividual rdf:about=\"http://130.233.46.104:8080/tempsensor/value#real\">
      <rdf:type rdf:resource=\"http://www.obix.org/resource/obix#Real\"/>
      <obix:name>sensorvalue</obix:name>
      <obix:href>http://130.233.46.104:8080/tempsensor/value</obix:href>
      <obix:displayName>Temperature Sensor Value</obix:displayName>
      <obix:writable>true</obix:writable>
      <obix:val>20.985796380082657</obix:val>
   </owl:NamedIndividual>
</rdf:RDF>"
 :reply-with  BrowserAgent@browser1407161249881 )
```

Figure 4: A subscription request from a browser agent to the gateway agent and a response sent by the gateway agent.

Table 1: Characteristics of the smart home data interfaces.

	oBIX XML	oBIX RDF	JADE RDF
Protocol	HTTP	HTTP	WebSocket
Language	oBIX	oBIX	ACL+oBIX
Data Model	oBIX	RDF	RDF
Serialization	XML	XML	String + XML

plications. Main characteristics of the interfaces are summarized in Table 1.

The oBIX specification (Considine, 2010) defines both Simple Object Access Protocol (SOAP) and Hypertext Transfer Protocol (HTTP) bindings for the communication. In our oBIX server, the HTTP interface is supported. The interface is simple with direct mappings from read, write, and invoke operations to corresponding HTTP GET, PUT, and POST request methods. Testing the GET method is thus as easy as writing the URI of the object, e.g., *http://obixserver/temperaturesensor*, to the navigation bar of any browser.

The system also supports requesting RDF/OWL data in XML serialization over HTTP. This is implemented by adding a parameter support for the URIs. For example, the example URI mentioned above can be requested in RDF format using the URI *http://obixserver/temperaturesensor?format=rdf*. The additional delay caused by the transformation process is discussed in Section 5.

The integrated agent platform provides a standard interface for FIPA compliant agents to communicate with the gateway agent. The gateway agent in turn provides access to the oBIX information in the smart home system. Now, intelligent agents can communicate with the system using ACL and understand the content described in FIPA-RDF content language.

4.3 Gateway Agent

Smart home information is provided to the agents through the integrated JADE agent platform. A special gateway agent in the platform was developed to handle the communication with the oBIX server. External agents can contact the gateway agent to access the information on the oBIX server.

The gateway agent interprets the FIPA-RDF messages coming from other agents and forwards the requests to the oBIX server. At the moment, query (FIPA, 2002b), request (FIPA, 2002c), and subscribe (FIPA, 2002d) interaction protocols are supported. Query-ref communicative act equals to oBIX read request (HTTP GET) and is used for querying for an identified object. Request communicative act equals to oBIX write request (HTTP PUT) and is used to request to perform a write action. Subscribe communicative act is used to request a subscription on an object and subsequent updates when the referenced object changes. Figure 4 presents a subscribe request message sent to the gateway agent and one of the responses from the gateway agent.

4.4 Real-time Data Update

The oBIX specification describes the oBIX data model, XML presentation and the HTTP REST and SOAP communication interfaces. However, it does not define any details on how the devices are actually connected to the server. One possibility is to integrate oBIX server into device or subsystem controllers that have direct access to the device data. The idea of the specification is that enterprise applications can access the smart home information through the defined oBIX

interface. Unfortunately, there is no support for real-time data updates. In practice, while oBIX provides a feature called watches for reducing the traffic load when polling for changes in the device data, it does not offer any true real-time notification method to implement server-push communication.

The agent platform integration provides a definite improvement to the real-time communication abilities of the smart home system. The subscribe interaction protocol (FIPA, 2002d) can be used to receive updates on specified data from the server immediately when available. It provides the server-push method that the oBIX specification lacks. Automatic notifications followed by the subscriptions also prevent the possibility of occasionally not registering some frequently changing data values.

4.5 WebSocket and Browser Support

FIPA standards define a reference model for agent Message Transport Service (MTS) (FIPA, 2002a). The model defines the modules in agent platforms that are responsible for delivering messages. Specifically, Message Transport Protocols (MTPs) are protocol level implementations used for message transport between platforms. A platform can have any number of MTPs for inter-platform communication.

Web browsers are traditionally HTTP clients. This has complicated the development of web applications for the Web 2.0. WebSocket is a fairly new technology that provides full-duplex communication channels between web browsers and web servers. Once the client (browser) sends a handshake, the HTTP server interprets it as an upgrade request and establishes a WebSocket. Now, the client can receive data from the server at any time without specifically requesting it.

We run the integrated agent platform with the WebSocket MTP (Järvinen et al., 2014). While WebSocket communication is more appropriate for the agent communication paradigm than, e.g., HTTP, there is another benefit that motivated us to utilize it. When using a WebSocket, client always initiates the communication. However, after the bi-directional pipe has once been established, both sides can send and receive data at any time. This provides a highly useful server-push feature for the systems that do not want to implement server side functionalities.

Using WebSocket MTP in the integrated agent platform allows browsers to communicate with the agents. This provides a standard way for desktop and mobile device browsers to communicate with the smart home system and receive updates in real-time. Mobile devices can now take part in distributed computing of a smart home. This can result in interesting applications in the future as mobile devices have multiple functionalities that can benefit the smart home system. In addition to the possible sensor and actuator services, smart phones can provide different kind of context information, e.g., location and user availability, and function as a remote user interface and computational entity for several purposes.

To facilitate the development of browser-based agents, we have implemented an ACL JavaScript library. It handles the creation of the FIPA message envelope and the ACL representation. The client only needs to define the agent name, address, and the actual message content. The library was used in the measurements described in the Section 5.

5 PERFORMANCE

Real-time requirements of smart home systems set limits for system delays. We evaluated the presented system by conducting performance measurements. The objective of the measurements was to find out additional system delays caused by the data conversion and the agent platform integration. Additionally, we measured the network traffic load when using different data interfaces. As we also wanted to demonstrate the interoperability with browser agents, the tests were conducted with JavaScript client programs running in a browser. Further, we developed a browser agent for visualizing temperature data.

To measure the network traffic, and to calculate the additional delays caused by the data conversion and the use of an agent platform, we developed four JavaScript test programs. In each test, the client program and the server program were run in separate computers connected with a hub. Tcpdump packet analyzer[2] program was used to record the network traffic for the post test analysis. Also, for the three first test programs the Round-Trip Time (RTT) of each request-response cycle was recorded. Wireshark packet analyzer software[3] was used for the network traffic analysis.

Each test program fetched a piece of smart home information repeatedly 10 000 times from the server. First test program sent HTTP GET requests to the oBIX server fetching an element in oBIX XML format. Second test program fetched the same information in RDF format over HTTP. Both third and fourth test programs again fetched the same information, but used WebSocket communication, ACL, and the integrated JADE agent platform. The difference in

[2]Tcpdump packet analyzer - http://www.tcpdump.org/

[3]Wireshark packet analyzer - http://www.wireshark.org/

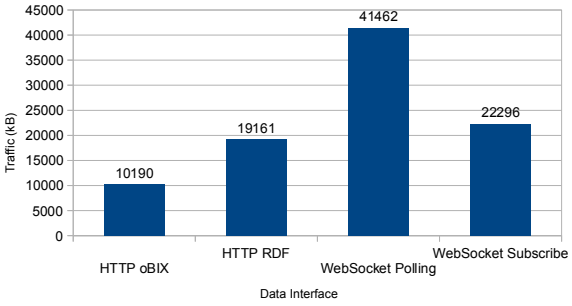

Figure 5: Comparison of traffic load produced using different data interfaces.

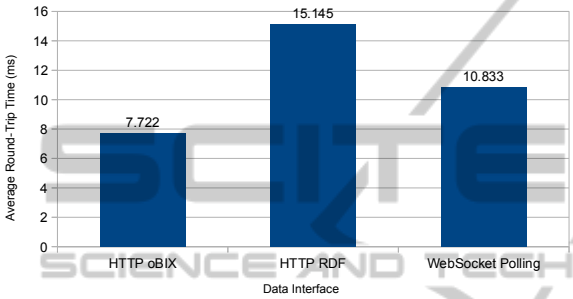

Figure 6: Comparison of RTT (ms) using different data interfaces.

these tests is that the third program was polling the information with Query-ref messages, while the fourth program subscribed to the information once and got an update every time the information changed on the server. The sample data on the server was updated in 100 ms time intervals and the clients were polling that information in 300 ms time intervals.

Comparison of traffic load produced using different data interfaces is depicted in Figure 5. More traffic is produced using the integrated agent platform with WebSocket interface compared to the oBIX HTTP interface. This can be explained with the differences in the message format. While the overhead of the HTTP protocol frame is many times larger compared to the WebSocket protocol frame (Järvinen et al., 2014), the message body in our application overturns that advantage. As seen in Figure 3, the RDF/OWL XML format is clearly more space-consuming than the corresponding oBIX XML format. In addition, agent communication requires a comparatively large message envelope and ACL message body, in addition to the actual FIPA-RDF content. It is notable that with no need to poll for changes in the data, agent subscription method gains significantly in the comparison and ends up with almost equal traffic load with HTTP RDF method. Larger packet size is compensated with the reduced number of packets.

Comparison of the round-trip times is depicted in Figure 6. It can be seen that the additional delay caused by the RDF transformation is only approximately 7 ms, which is acceptable in most use cases. Surprisingly, using the integrated agent platform interface results in less delay than using HTTP RDF interface. From the results we can conclude that the additional delays are not significant. The subscription test was not included in these results, as it does not have separate request for each update, and is thus not applicable. However, it is notable that the subscribe method is naturally the best option when implementing applications with strict real-time requirements. Subscription method also ensures that every update is actually communicated to the client, while in oBIX polling it is possible to miss rapid changes that happen between poll cycles.

In addition to the measurements, we developed a temperature view browser agent. The application subscribed to the temperature information on the smart home server and received sensor value updates every 100 ms. The application visualized the received information in real-time with HTML5 Canvas 2D API.

6 CONCLUSIONS

We presented a solution for realizing the semantic interoperability layer in Web services based smart home systems. The system provides smart home information semantically in RDF/OWL format using both an HTTP interface in oBIX server and a WebSocket interface in JADE agent platform. The system enables the use of MASs to provide intelligence for smart home control.

To represent oBIX information semantically, we defined oBIX ontology. The ontology consists of classes for all the oBIX primitives, and properties for their facets. Every possible oBIX element can be aligned with the ontology. Additionally, FIPA-RDF0 ontology was used for the agent platform integration to support standard FIPA agent communication.

We conducted performance measurements to find out the additional delays caused by the data transformation and the agent platform integration, and to measure the traffic load when using different interfaces. Standard agent communication envelope and ACL format cause more traffic load when using the agent platform interface. However, the use of subscription method can even out the difference significantly. The additional delays are not significant.

In addition to semantic interoperability, subscription support enabled real-time data updates that do not normally exist in oBIX solutions. Real-time communication abilities can be of high importance in vari-

ous smart home scenarios. Automatic updates also prevent the possibility of occasionally not registering some frequently changing data values.

We developed an ACL JavaScript library, and demonstrated interoperability between browser agents and the smart home system. The use of WebSocket MTP enabled the communication. Using the default HTTP MTP in JADE it would be impossible for browser agents to receive messages from the smart home system. Providing a standard Web interface for mobile devices, e.g., smart phones can result in interesting applications in the future. We consider browser agents as important technology to participate in realizing the ambient intelligence.

Both the idea of transforming smart home Web services data into RDF and to represent it in standard FIPA-RDF agent language have been proposed by earlier research in the field. However, detailed and evaluated generalizable approach similar to ours to map Web services based smart home information from oBIX into RDF has not been presented before.

Future work includes better data mapping with known ontologies, implementation of real world use cases, and development of intelligent agent services.

REFERENCES

Bellifemine, F., Poggi, A., and Rimassa, G. (1999). Jade–a fipa-compliant agent framework. In *Proceedings of PAAM*, volume 99, page 33. London.

Chen, H., Finin, T., and Joshi, A. (2003). An ontology for context-aware pervasive computing environments. *The Knowledge Engineering Review*, 18(3):197–207.

Chen, L., Nugent, C., Mulvenna, M., Finlay, D., and Hong, X. (2009). Semantic smart homes: towards knowledge rich assisted living environments. In *Intelligent Patient Management*, pages 279–296. Springer.

Considine, T. (2010). Open Building Information Exchange (oBIX) version 1.1. OASIS Working Draft 06, June 2010.

Coyle, L., Neely, S., Rey, G., Stevenson, G., Sullivan, M., Dobson, S., and Nixon, P. (2006). Sensor fusion-based middleware for assisted living. In *Smart Homes and Beyond: ICOST 2006: 4th International Conference on Smart Homes and Health Telematics*, volume 19, pages 281–288. IOS Press.

Coyle, L., Neely, S., Stevenson, G., Sullivan, M., Dobson, S., and Nixon, P. (2007). Sensor fusion-based middleware for smart homes. *International Journal of Assistive Robotics and Mechatronics*, 8(2):53–60.

FIPA (2002a). *FIPA Agent Message Transport Service Specification*. Foundation for Intelligent Physical Agents.

FIPA (2002b). *FIPA Query Interaction Protocol Specification*. Foundation for Intelligent Physical Agents.

FIPA (2002c). *FIPA Request Interaction Protocol Specification*. Foundation for Intelligent Physical Agents.

FIPA (2002d). *FIPA Subscribe Interaction Protocol Specification*. Foundation for Intelligent Physical Agents.

Gu, T., Wang, X. H., Pung, H. K., and Zhang, D. Q. (2004). An ontology-based context model in intelligent environments. In *Proceedings of communication networks and distributed systems modeling and simulation conference*, volume 2004, pages 270–275.

Huberman, B. A. and Clearwater, S. H. (1995). A multi-agent system for controlling building environments. In *ICMAS*, pages 171–176.

Järvinen, H., Garcia-Gasulla, D., and Cortés, U. (2014). A push-based agent communication model empowering assistive technologies. *International Journal on Artificial Intelligence Tools*, 23(1).

Jiang, L., Liu, D.-Y., Yang, B., et al. (2004). Smart home research. In *Proceedings of the Third Conference on Machine Learning and Cybernetics*, pages 659–664.

Kim, E. and Choi, J. (2006). An ontology-based context model in a smart home. In *Computational Science and Its Applications-ICCSA 2006*, pages 11–20. Springer.

Miori, V., Tarrini, L., Manca, M., and Tolomei, G. (2006). An open standard solution for domotic interoperability. 52(1):97–103.

Perumal, T., Ramli, A. R., Leong, C. Y., Samsudin, K., and Mansor, S. (2010). Middleware for heterogeneous subsystems interoperability in intelligent buildings. *Automation in Construction*, 19(2):160 – 168.

Sashima, A., Izumi, N., and Kurumatani, K. (2005). Agents that coordinate web services in ubiquitous computing. In *Ubiquitous Computing Systems*, pages 131–145. Springer.

Tokunaga, E., Ishikawa, H., Kurahashi, M., Morimoto, Y., and Nakajima, T. (2002). A framework for connecting home computing middleware. In *ICDCSW*, pages 765–770, Washington, DC, USA. IEEE Computer Society.

Wang, S., Xu, Z., Cao, J., and Zhang, J. (2007). A middleware for web service-enabled integration and interoperation of intelligent building systems. *Automation in Construction*, 16(1):112 – 121. CAAD Futures, 2005.

Wang, X., Dong, J. S., Chin, C., Hettiarachchi, S. R., and Zhang, D. (2004a). Semantic space: an infrastructure for smart spaces. *Pervasive Computing, IEEE*, 3(3):32–39.

Wang, X. H., Zhang, D. Q., Gu, T., and Pung, H. K. (2004b). Ontology based context modeling and reasoning using owl. In *Pervasive Computing and Communications Workshops, 2004. Proceedings of the Second IEEE Annual Conference on*, pages 18–22. Ieee.

Xu, J., Lee, Y.-H., Tsai, W.-T., Li, W., Son, Y.-S., Park, J.-H., and Moon, K.-D. (2009). Ontology-based smart home solution and service composition. In *Embedded Software and Systems, 2009. ICESS'09. International Conference on*, pages 297–304. IEEE.

Zhang, D., Gu, T., and Wang, X. (2005). Enabling context-aware smart home with semantic web technologies. *International Journal of Human-friendly Welfare Robotic Systems*, 6(4):12–20.