Are our homes ready for services? A domotic infrastructure based on the Web service stack

Aiello, Marco; Dustdar, Schahram

Published in:
Pervasive and Mobile Computing

Publication date:
2008

Available online at www.sciencedirect.com
ScienceDirect
Pervasive and Mobile Computing 4 (2008) 506–525
pervasive and mobile computing
www.elsevier.com/locate/pmc

Are our homes ready for services? A domotic infrastructure based on the Web service stack

Marco Aiello*, Schahram Dustdar

Department of Mathematics and Computing Science, University of Groningen, Nijenborgh 9, 9747AG Groningen, The Netherlands
Distributed Systems Group, TUWien, Argentinierstrasse 8, 1040 Wien, Austria

Received 22 March 2006; received in revised form 8 January 2008; accepted 8 January 2008
Available online 19 January 2008

Abstract

The increase in computational power and the networking abilities of home appliances are revolutionizing the way we interact with our homes. This trend is growing stronger and opening a number of technological challenges. From the point of view of distributed systems, there is a need to design architectures for enhancing the comfort and safety of the home, which deals with issues of heterogeneity, scalability and openness. By considering the evolution of domotic research and projects, we advocate a role for Web services in the domestic network. We ground our claim by proposing a concrete architecture for a home in which the health of an elder is monitored. The architecture is implemented on a heterogeneous set of devices, which allows us to evaluate it and draw conclusions on the feasibility of using service-oriented approaches in ubiquitous computing.

Keywords: Domotics; Web service stack; WS-Notification; Publish/subscribe

1. Introduction

Homes are no longer the places where a number of devices carry on computations in isolation to perform simple repetitive tasks, such as washing dishes, but a distributed system of potentially collaborating hosts. For instance, deciding on what appliance is operating at a given moment in order to avoid exceeding a given power consumption limit could be a task of the home network achieved by internal communication, rather than a concern of the human user. Scenarios of this kind are possible today thanks to ever lowering prices and sizes of controllers which are increasing in computational power and networking capabilities. The trend is not stopping: it is easy to imagine that the smart dust [45] technology will make our homes a Babylon of thousands of sensors and actuators constantly exchanging data to achieve global control and monitoring.

Domotics is the field where housing (domus) meets technology in its various forms (informatics, but also robotics, mechanics, ergonomics, and communication) to provide better homes from the point of view of safety and comfort. Traditionally, domotic solutions were provided by one vendor, using one standard for communication, most often

* Corresponding author at: Department of Computing Sciences, University of Groningen, Nijenborgh 9, 9747AG Groningen, The Netherlands.
E-mail addresses: aiellom@cs.rug.nl (M. Aiello), dustdar@infosys.tuwien.ac.at (S. Dustdar).
URLs: http://www.cs.rug.nl/~aiellom (M. Aiello), http://www.infosys.tuwien.ac.at/Staff/sd/ (S. Dustdar).

1574-1192/$ - see front matter
doi:10.1016/j.pmcj.2008.01.002

closed, and were expensive. This is no longer true. Domotic elements are heterogeneous in all aspects. Devices come from different vendors, have different hardware, network interfaces, and operating systems, but still need to interoperate. Users want to be able to have a unique view on all the hardware entering their home. Today's challenge for the home is *total interoperability*.

On the push of this excitement, a number of research and industrial projects are underway with the goal of studying domotic solutions. For instance, at Georgia Tech a domotic home has been built for the elder adult with the goals of compensating physical decline and memory loss, and supporting communication with relatives [33]. This study also considers issues of acceptability of domotics identifying key issues for the adoption of the technology by the end user. Acceptability, dangers and opportunities are surveyed in [38].

At Carnegie Mellon people's behavior is studied by automatic analysis of video images [25]. This is fundamental in detecting anomalies and pathologies in a nursing home where many patients live. Pervading the environment with active landmarks, called Cyber Crumbs, aims at guiding the blind by equipping him/her with a smart badge [39]. A number of projects to give virtual companions to people, to monitor people's health and behavioral patterns, and to help Alzheimer patients are presented in [29]. The social dimension is considered in [13], where social networks are used to model the personal relationship sphere of the user. This network is used for providing information or issuing alarms related to the home. The Gator Tech Smart House [26] is a programmable approach to smart homes targeting the elder citizen. The idea is to have a service layer based on OGSi [43] in order to enable service discovery and composition. All the aforementioned projects play a fundamental role in showing what is feasible today and in testing futuristic approaches, but the goal of total interoperability still remains a challenge.

Web services are a family of XML-based standards designed for the communication of loosely coupled, dynamically bound applications. Web services can be seen as the evolution of the Web where in place of having human beings interacting with applications via HTML forms, applications can directly interact with one another. Alternatively, Web services can be seen as a generalization of distributed object middleware, such as CORBA [44]. In whichever way one considers Web services, there is a strong trend and wide adoption of the technology as means for resolving interoperability issues.

Web services are not only a recent technological hype, but are also defining a new way for designing applications and information systems. They are the most known incarnation of the programming paradigm known as Service-Oriented Computing [37], which has the goal of producing Service-Oriented Architectures (SOA, e.g., see [12]). Of course, interoperability comes at a price. Most notably, the use of XML for the communication is computationally expensive and bandwidth consuming. Furthermore, using XML does not solve per se the ontological problem of having independent applications interoperate.

Nevertheless, we claim that Web services are ready to penetrate our homes, end up in the mobile devices we use, and solve the interoperability problem of domotics. To ground our claim, we propose a Web-service-based architecture for the home. This results in the proposal of a stack of protocols that go from transport up to coordination to be implemented on the devices. In particular, at the coordination level, we choose to use a publish/subscribe protocol which best suites the nature of home communication, that is, asynchronous one-to-many, possibly topic-based, communication. WS-Notification is the most expressive Web service protocol for publish/subscribe, as it defines a tree-based structure for topics and a broker role [24,36], and thus our choice in the presented approach.

We implement the proposed framework in a concrete case study (named ITEA after the Trentino Institute for Public Housing, who commissioned the study) for monitoring elder adults. A home equipped with heterogeneous wireless networking (Bluetooth, WiFi, dedicated radio frequencies), a number of fixed and wearable sensors is transformed into a distributed system using Web services to coordinate with the goal of monitoring the fall of the home inhabitant. The case study shows the feasibility of the approach based on Web services, and how the challenge of heterogeneity, among others, is well addressed. Furthermore, we evaluate the application in a number of circumstances for testing the feasibility of the approach with respect to Web service overhead and reliability.

The work presented here is based and extends the workshop presentations [1,2], while commented code snippets have been presented in the popular programmer's magazine DDJ [3]. In addition to the previous publications, here we give a complete overview of the use of Web services in domotics, provide examples of domotic scenarios, offer a survey of the state of the art in the field, give details of the architecture and present an evaluation of it in the context of healthcare. The remainder of the paper is organized as follows. In Section 2, we overview the evolution of domotic standards and identify a number of prototypical scenarios. Section 3 presents a number of Web service standards for publish/subscribe messaging. The domotic architecture we propose is explained in Section 4, while its application to

the case study of monitoring the health of an elder citizen is discussed in Section 5. Related work and projects are systematized in Section 6. Finally, Section 7 draws conclusions on the proposed architecture and the feasibility of using Web services in the home.

2. The evolution of domotics: Scenarios

One of the main reasons for the flourishing of domotic projects is because today we have cheap, small and potentially interoperable devices which can be used to improve our homes. We came a long way from the initial proposals. Think of the X-10 protocol: proposed in the early 70s, it uses the power-line to send 16 messages to a maximum of 256 devices. It is still widely adopted for its simplicity and the availability of many devices and interfaces implementing it.

Many standards have been proposed since the X-10, such as Open Services Gateway Initiative (OSGi), European Home System (EHS), European Installation Bus (EIB), Home Audio Video Interoperability (HAVi), Universal Plug and Play (UPnP), Konnex-KNX, LonWorks, Jini, No New Wires, to name a few. A natural question then arises. How can we classify them? How can we judge their characteristics?

We argue that the key properties to judge a domotic technology and related standards are:

- *openness:* the publicity of the protocol and the possibility of implementing it on any home appliance;
- *scalability:* the possibility of adding and removing devices to and from a home network without affecting its functionalities and its performances. We also talk about dynamic scalability meaning the possibility of adding/removing elements during normal operation of the domotic infrastructure;
- *heterogeneity:* the differences in hardware, network, operating systems, programming languages which are accepted by the infrastructure;
- *topology:* the way in which devices are connected to one another (e.g., using a bus or point-to-point channels) and the relation among distinct computational elements (client-server vs. peer-to-peer).

Based on the degree of openness, scalability, heterogeneity and on the type of topology, we can classify a domotic system. Furthermore, some combinations have been historically more frequent than others. We identify four of these main combinations which we refer to as scenarios, and illustrate them next.

2.1. Scenario 1 (S1): Simple bus

The initial proposals for domotics consider having a bus for communication onto which all appliances are connected (Fig. 1). The protocol is closed, not extendable and usually scalability is limited to a prefixed number of devices and actuators. Heterogeneity is low as all elements have to connect to one kind of network bus, with one type of hardware. The typical example of this scenario is the X-10 protocol. The bus is the power-line. The bandwidth is very modest and the set of signals which can be sent is limited to 16. The X-10 technology is mainly used to control simple appliance such as lights or window shades. The noise of the channel is high and the system suffers, among others, from omission failures. More evolved proposals which fall into this category are the EIB and its following evolutions such as Konnex. These later protocols deal better with scalability and heterogeneity, as it is possible to use different technologies for the bus (phone line, power-line, ethernet, radio frequencies, etc.), and to address thousands of devices. In general, (S1) architectures are particularly suited for small domotic applications where basic coordination among devices is needed. When structured and rich information needs to be sent around, heterogeneous devices need to be deployed, and systems need to be scaled, this type of architectures shows all their limits.

2.2. Scenario 2 (S2): Closed centralized

The need to have a remote control over the home, or a remote service provider for monitoring the home has introduced a gateway in the architecture (Fig. 2). This gateway is the controller of the home which interacts with the central service provider. The topology of the home network is usually client-server, but can also be based on a bus. The protocol used to exchange information between the devices and the controller is closed. This is the typical architecture proposed by companies which want to have the monopoly over the home infrastructure in order to sell all devices, actuators and appliance's interfaces (Echelon's LonWorks, BTicino's MyHome, Sistema Casa, and many more). (S2) architectures address fine the needs of basic home automation and have been reasonably sold by vendors,

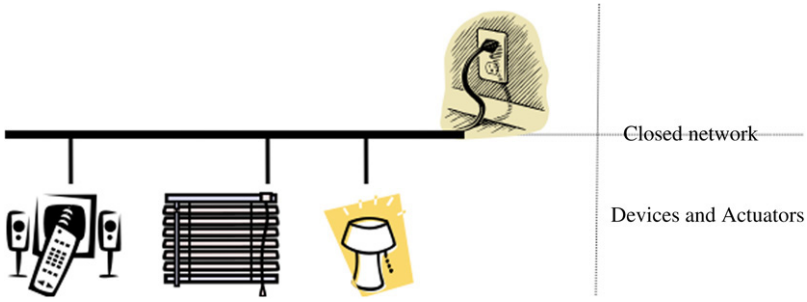

Fig. 1. S1: Simple bus scenario.

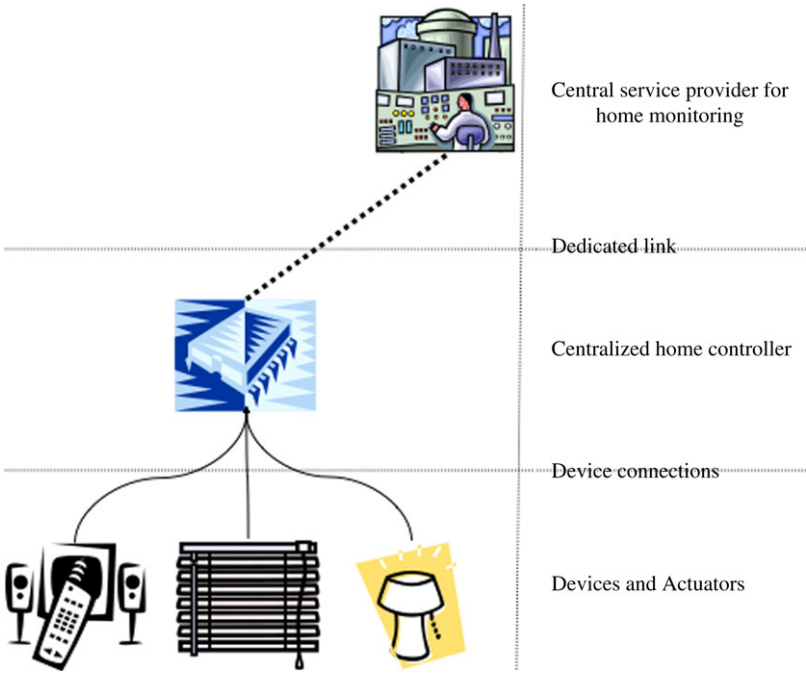

Fig. 2. S2: Closed centralized.

though they show all their limits when scalability is necessary and even worse when there is the need of introducing new components. In summary, solutions of this kind are closed, relatively scalable, and handle heterogeneity poorly.

2.3. Scenario 3 (S3): Open server-based hierarchy

The current trend is to open up the home network and to use public standards for the communication. Since the implementation of the standard may be computationally expensive, not all sensors and actuators are able to implement it. Therefore, an intermediate layer is introduced forming a hierarchy of domotic elements. Groups of sensors and actuators communicate with a controller, which in turn communicates with the centralized monitor of the home (usually a PC). Other devices communicate directly with the centralized monitor. To provide for remote control and monitoring of the home, the centralized controller may be connected to the Internet or other appropriate network. Fig. 3 shows an example of the open server-based hierarchy using Web service as the standard for communication between the centralized controller and its clients [31,2]. (S3) architectures are open and deal very well with heterogeneity. Scalability is good in terms of adding and removing devices, but the central server may become a bottleneck and jeopardize reliability, being the single point of failure affecting the whole domotic infrastructure.

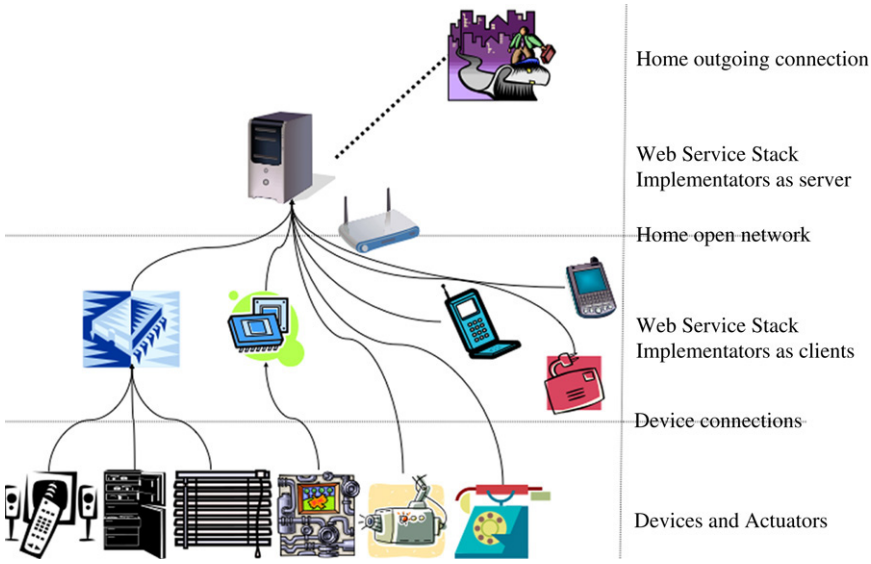

Fig. 3. S2: Open server-based hierarchy using Web services.

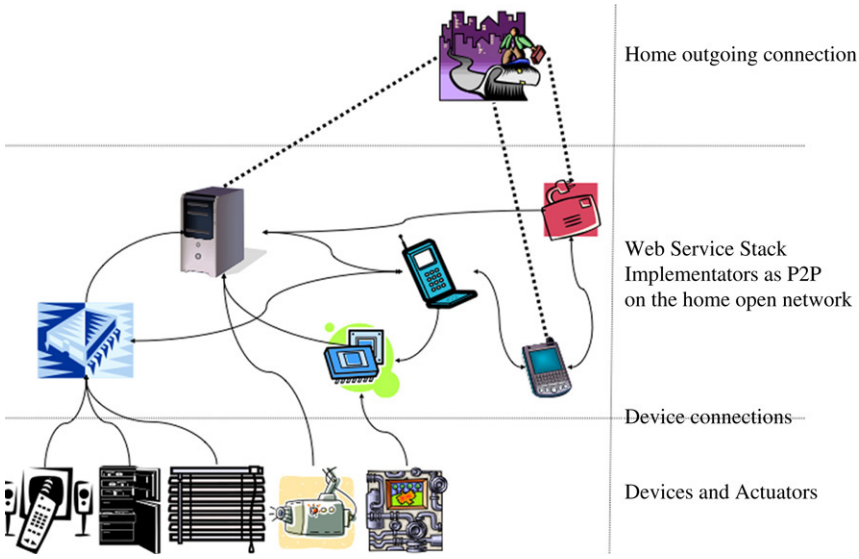

Fig. 4. S4: Web service P2P architecture.

2.4. *Scenario 4 (S4): Web service P2P architecture*

The need of freeing the home from a unique controller, thus improving reliability and scalability of the domotic infrastructure, yields the evolution of the topology of the scenario (S3), from centralized to peer-to-peer. In this way, openness and heterogeneity are untouched, while scalability is greatly enhanced. The Web-service-based solution is shown in Fig. 4. Notice that now the PC is not the only gateway with the external world, as other devices may connect to the outside possibly using different networks. The P2P scenario, without the use of Web services, is also adopted by Jini and HAVi. (S4) type architectures handle very well the problems of scalability and can be very effective when performance becomes an issue. In addition, nomadic users may much more easily interact in this sort of architectures. Additional concerns have to be though taken care of, such as inconsistencies in data and ontologies, security, replication and routing mechanisms.

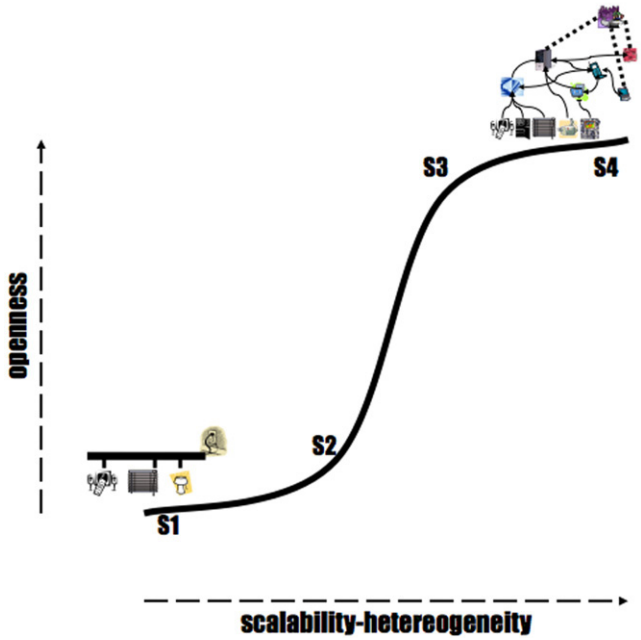

Fig. 5. The evolution of domotic technology.

2.5. Discussion

The scenarios just introduced represent the temporal evolution of domotic technology and standards. From closed non-scalable systems unable to handle heterogeneity, we are moving towards peer-to-peer networks totally heterogeneous (at the hardware, network, OS, manufacturer levels) of home devices. This evolution is depicted in Fig. 5, where increasing degrees of openness are on the vertical axis and increasing degrees of scalability and heterogeneity are on the horizontal axis. The scenario (S1) is at the origin of the evolution then moving towards more scalable but still closed systems (S2). The closed architecture then uses open standards (S3). The next trend of which we are currently witnessing the dawn is (S4): Open, scalable, and heterogeneous infrastructures based on peer-to-peer topologies.

As mentioned above, OSGi, Jini and HAVi are instances of (S4), though, it is important to notice that there are some limitations. OSGi and Jini require an implementation of the Java Virtual Machine (or a subset of it, the KVM) on all devices of the network. This is a strong computational requirement which cannot be met by sensors and basic actuators. HAVi limits the scope of the communication between devices to multimedia.

3. Publish/subscribe and Web services

From the discussion of the trends in domotic middleware, it is apparent that the communication taking place among decoupled home devices is most often asynchronous, goes beyond the simple one-to-one communication and needs to take into account scalability. The contents of messages are usually notifications of events that interest a set of other devices that may, or may not, react to them independently. The paradigm for this kind of communication is known as publish/subscribe [20].

In the publish/subscribe model there are entities which generate messages (*producers*) via a publish operation and others which consume messages (*subscribers*). There may also be intermediaries in the routing of the messages (*brokers*). The messages (*events*) are often organized according to some hierarchy of topics (*types*) [6]. If, on the one hand, publish/subscribe takes care of scalability and loose coupling, we also need to consider the heterogeneity problem, on the other hand. To account for the latter, we resort to Web services.

3.1. Web services

Web services are a family of XML-based protocols that enable the creation of massively distributed loosely coupled applications [4]. The fundamental paradigm behind Web services is *publish, discover, invoke*. That is, services publish

	WS-Notification	*WS-Events*	*WS-Eventing*
topics	yes	no	no
topic advertising	yes	no	no
brokers	yes	no	no
send modes	push	push/pull	push/pull
sub. delegation	yes	no	yes

Fig. 6. Web service publish/subscribe specifications: a comparison.

information regarding their functionalities which are dynamically discovered by other services and then invoked. Notice that in general no assumption is made on how the functionalities are implemented or which services will be available at run-time.

Web services are systematized by considering the protocol stack, in the spirit of the ISO/OSI networking stack. The stack (see for instance [15]) is based on the transport and encoding layers which are covered by the SOAP protocol and various transport means (HTTP, SMTP, GPRS, etc.). On top there is the Quality of Service layer in which various non-functional properties of service invocation are specified, e.g., WS-Security, WS-Transaction, WS-Agreement. Further above are then the protocols for describing and finding individual services: WSDL and UDDI live in this level. At the topmost level, one finds protocols for process coordination, for managing choreographies, for event notification and so on [46].

This model of computation provides for a number of key features which include openness (all standards are public, XML-based, and extensible), scalability (given the loosely coupled and asynchronous nature of Web services scalable architectures can always be built using them), and handling of heterogeneity (no assumptions are made on hardware, operating system, network, or even on the transport layer). Therefore, Web service are a natural choice for satisfying the openness and scalability/heterogeneity requirement of future domotic infrastructures. From the point of view of topology, Web services do not commit to one choice. It is possible to build both client-server and peer-to-peer architectures.

The role of Web service is thus that of guaranteeing the total interoperability of home appliances. It provides the communication and coordination infrastructure for all home appliances which have a minimum of computational capability. Sensors and actuators need not speak XML and Web services, but their controllers do, as well as all peer devices populating and entering the home. Furthermore, gateways for communicating outside the home and coordinating with external services need to speak Web services both inside and outside the home.

3.2. Publish/subscribe standards: WS-Notification

A number of Web service specifications exists in the arena of publish/subscribe communication: WS-Events [11], WS-Eventing [8], and WS-Notification [24]. Following [36], in Fig. 6 we compare the main characteristics of the three specifications. In particular, one notices the superiority of WS-Notification as it allows the presence of brokers and, most importantly, because it has a specification for event types.

WS-Notification is a family of related specifications that define a standard Web service approach to notification using a topic-based publish/subscribe pattern. In particular, it includes three normative specifications:

- WS-BaseNotification defines the Web service interfaces for NotificationProducers and NotificationConsumers. It includes standard message exchanges to be implemented by service providers that wish to act in these roles, along with operational requirements expected of them. This is the base specification on which the other WS-Notification specification documents depend;
- WS-BrokeredNotification defines the Web service interface for the NotificationBroker. A NotificationBroker is an intermediary which, among other things, allows publication of messages from entities that are not themselves service providers. It includes standard message exchanges to be implemented by NotificationBroker service providers along with operational requirements expected of service providers and requestors that participate in brokered notifications;
- WS-Topics defines a mechanism to organize and categorize items of interest for subscription known as “topics”. These are used in conjunction with the notification mechanisms defined in WS-BaseNotification. WS-Topics

defines three topic expression dialects that can be used as subscription expressions in subscribe request messages and other parts of the WS-Notification system. It further specifies an XML model for describing metadata associated with topics.

3.3. The open grid service infrastructure (OGSI)

The Open Grid Service Infrastructure (OGSI) is also based on Web services for interconnecting computational resources and defines the way in which elements of a network of computers can interact [43]. The infrastructure was originally defined to create heterogeneous grids for executing computationally intensive tasks. In this context, the stronger requirements are: interoperability and quality of service. To address the first issue, the OGSI resorts to Web services and extends the open standards with additional constructs necessary for grid management. This includes information regarding quality of service attributes. Thus, the core of OGSI is a definition of a component model which includes stateful Web services, extensions of WS interfaces such as WSDL, asynchronous communication, and instance references.

OGSI, thanks to its resorting to Web services, has been used also in contexts different from purely computationally intensive parallel processing. In fact, OGSI may be deployed in pervasive computing environments. For instance as the gateway for a home, possibly deploying internally a UPnP network (see e.g., [32]). The relation between OGSI and the approach we propose next is very strong, most notably because both rely on the same Web service technologies. Though there are important differences.

The most notable one is that the publish/subscribe protocol of OGSI is not well suited for domotics. In fact, it is possible to provide subscription expressions, but there is no notion of topics. The latter is very important as typically there are rich ontologies tied to the home environment. In fact we shall build on the notion of a typed event subscription since it is essential in homes with many different devices and possible events.

Furthermore, we remark that the approach based on WS-Notification does not modify the WS standards, unlike OGSI, and is thus more portable. Not only that, but OGSI is a heavier protocol and will generate even more overheads than the approach we propose, especially regarding XML parsing and inter-node communication. In fact, the architectures for using OGSI in domotics have seen it deployed only as the gateway for the home, and not as the *internal* infrastructure. The performance issues of OGSI are widely discussed in the literature. For instance, advantages and disadvantages of using OGSI with respect to Jini and JXTA are reported in [23]. Relevant here is the reported overhead of using OGSI standards and forcing all communication to rely on the HTTP transfer protocol. Further experimental results that test the scalability of OGSI are reported in [47].

4. Proposed architecture

Any computational element taking part in the domotic environment must thus be able to communicate with the others using the Web service dialect WS-Notification. This means the ability of networking and having an implementation of the Web service stack. In particular, XML processing abilities are necessary.

4.1. The Web service stack implementation for domotics

Web services are used to allow dynamic information exchange among loosely coupled and independent home devices following a publish/subscribe messaging pattern. Fig. 7 shows the Web service stack that we propose to use for domotics. In particular, in the implementation realized for the case study, we have various options at the network level such as Bluetooth, WiFi, and GPRS. Invocation is performed using the Simple Object Access Protocol (SOAP), while no assumption is made on transport: HTTP or other device dependent mechanisms are used.

The main difference with respect to the stack usually produced in SOA resides at the coordination level. Instead of using a state-based standard requiring some form of centralized and synchronous communication between the service orchestrator and individual services participating in the application, we resort the WS-Notification. This choice is driven by the nature of domotics: devices need a form of one-to-many asynchronous communication, possibly, topic-based.

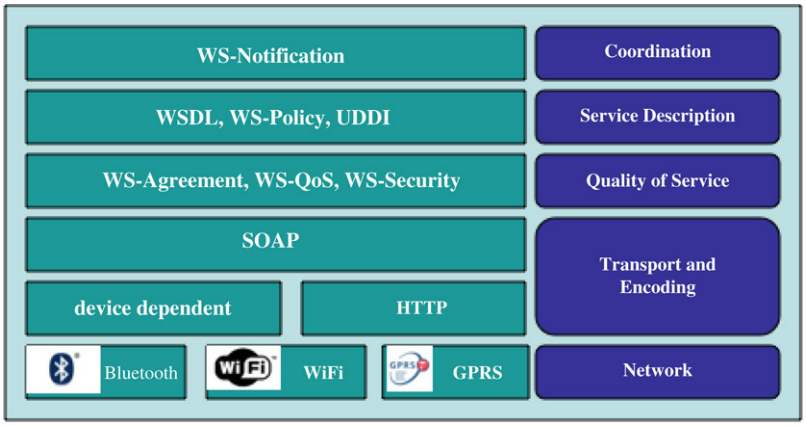

Fig. 7. The Web service stack implementation for domotics.

4.2. System's topology

The proposal is thus that of equipping all devices with sufficient computational power with an implementation of the Web service stack. In addition, there must be a server acting as global coordinator for the notification of events. As for the devices or sensors with low computational power, these are basic and implement simple communication protocols. They refer to a master device which fuses the sensor data and communicates with the rest of the home by implementing itself the Web service stack.

The system's topology is thus a centralized one with a number of devices asynchronously interacting via publish/subscribe pattern with it. An instance of the (S3) scenario described in Section 2. This topology suits well the needs of homes requiring high reliability of communication and for applications that are not data and bandwidth intensive, as it is the case of the case study of this work.

In Fig. 8, we depict a deployed case study, which will be illustrated in Section 5. In this case, we have a room equipped with a camera and a user wearing an accelerometer sensor. These communicate via a dedicated radio link with a computer implementing the Web service stack. Such computer also acts as centralized event server handling all notifications and registrations and being the unique controller of the home. The server is connected to the outside world via the Internet and GPRS links, while local wireless connectivity is guaranteed by Bluetooth and WiFi interfaces. There is no a priori fixed number of elements taking part in this system, as devices can come to the network, discover the WS-Notification server, get the topic tree of that environment and register for any available event.

4.3. The event server

In the domotic architecture, there must be at least one element acting as event server and broker. This element comprises several interacting components. First, events generated by devices, sensors in particular, may be fused in order to generate higher-level events (for instance, the fact that someone is laying down should not trigger a fall event, unless there is also an acceleration event). To fuse sensor data, a rule engine is used. Notice that a sensor can directly generate an event by simply having a rule which does not require any additional condition for event propagation. The rule engine is depicted on the left of Fig. 9 running on the server. The rule engine is connected to the actual WS-Notification interface. The hierarchy of events (event tree) that can be generated and that one can be registered to is managed in a separated component. Notice that the event tree is a dynamic structure that can be modified at run time by the event server, but also by clients via appropriate WSDL calls. Client registration to events are stored in an appropriate database in the event server. Registrations to atomic or complex events can be made anytime. Registration may expire or be cancelled.

The WS-Notification protocol includes the possibility of having *preconditions* and *selectors* connected with the registration of an event and controlling its generation and delivery. For instance, a client could register to the event of the home user fall, only under the condition that it is night.

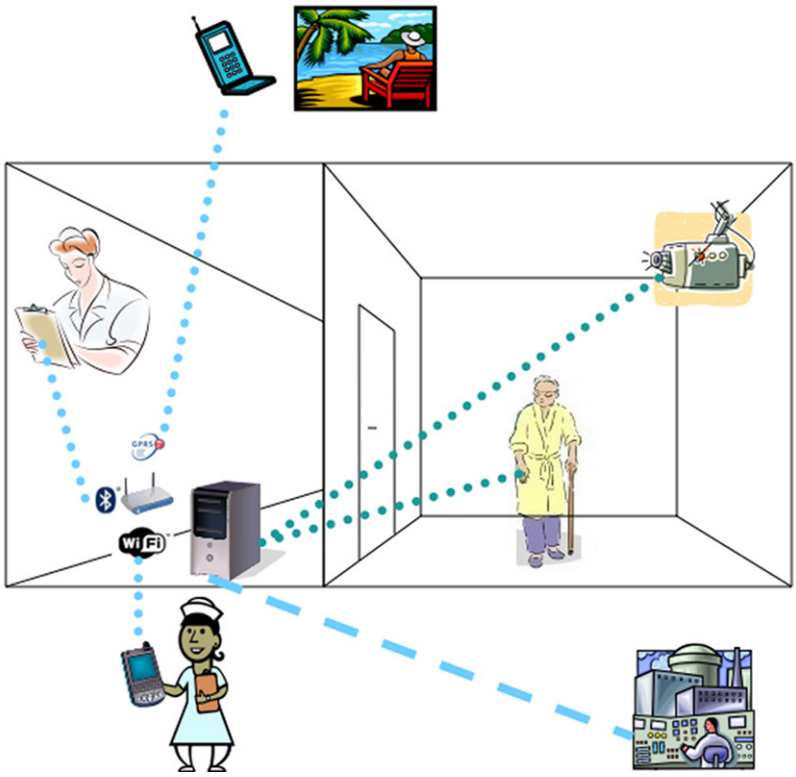

Fig. 8. The proposed architecture in the ITEA scenario.

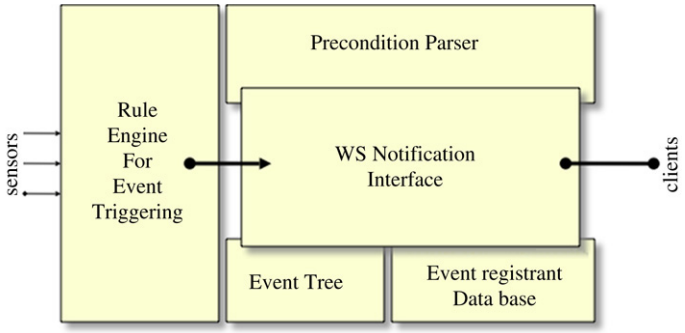

Fig. 9. The event server.

4.4. Element interaction

Let us consider an example of fall to understand what are the interactions among the various elements of the architecture. Referring to Fig. 10, at time 0 the user begins to fall and at time 0.5 seconds he reaches the ground and lies down (top of the figure). A possible behavior of the accelerometer is represented below in the figure: at regular intervals the accelerometer sends a liveliness message to inform of his correct functioning, then it notifies when it detects an acceleration due to the fall and an even stronger, negative acceleration due to the impact with ground. Finally, no acceleration is detected when the user is lying on the floor. The images coming from the camera are analyzed and the position of the user is categorized in {upright, bent, lying}. The software currently in use is very accurate as long as there are no other people in the range of the camera, otherwise tracking of the user may result in mistakes. Finally, a rule engine (called ‘expert system’ in the figure) aggregates the notifications from the accelerometer and from the camera and generates an attention event followed by a fall detected event.

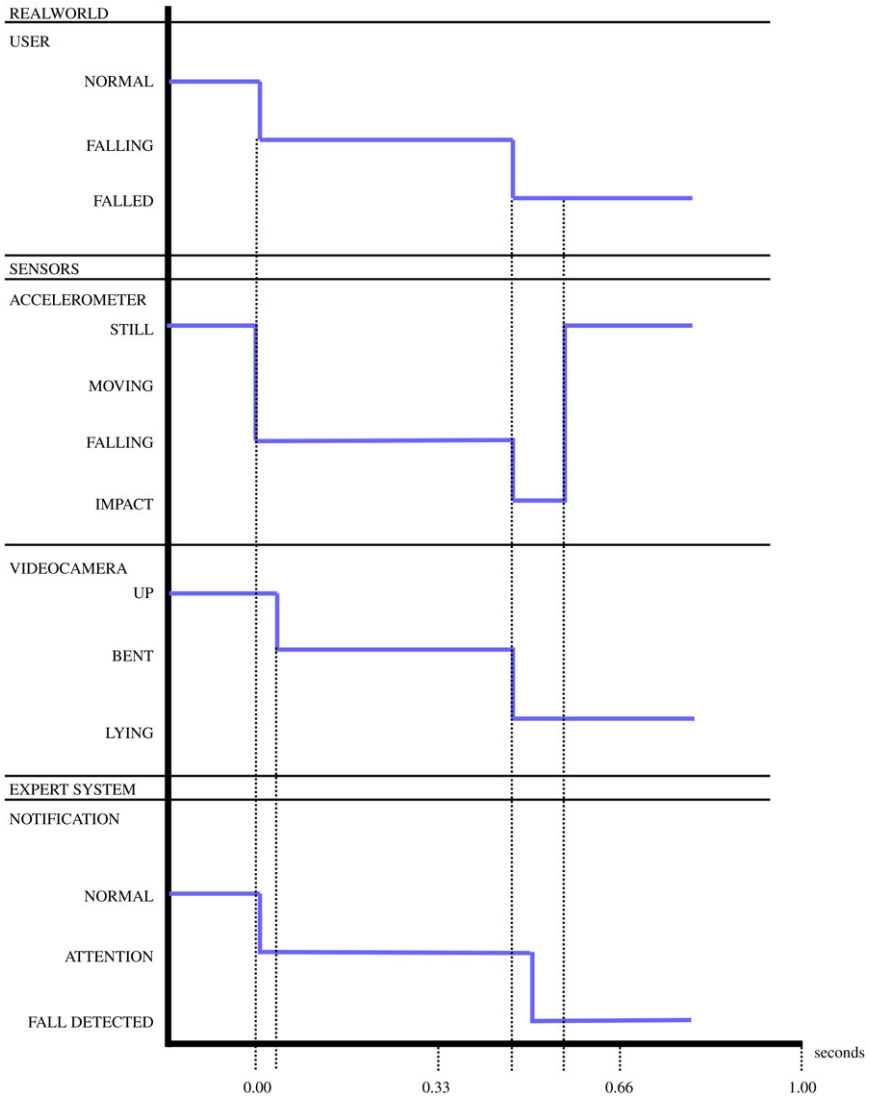

Fig. 10. A possible interaction of sensors and devices over time.

4.5. *WS-Notification in practice*

The three main entities in WS-Notifications are, as seen in Section 3.2 the producer, the broker and the consumers. In the proposed architecture, the server acts as broker (containing also the rule engine, the event tree and registrant database), while the producers are the sensors (possibly filtered by the rule engine) and finally the consumers are the mobile devices.

Let us consider the most relevant elements of the interface of WS-Notification during a typical interaction sequence, shown in Fig. 11. First the consumer is interested in discovering the available topics for which it is possible to subscribe or to add to (`GetResourceProperty(Topics)`). Then the broker replies with the list. At this point the consumer can `Subscribe(Topic, id)` to a topic or a set of topics. When a publisher generates an event, it uses the `Notify(message)` method of the interface to inform the broker. The consumer can also ask the broker to be informed of the last notified message for a given topic via the `GetCurrentMessage(topic)`.

The broker adheres to the following process to decide what to do upon notification of an event. First it checks whether it is a known type of event. If so, it checks in its subscription databases if there are subscribers and then it climbs the topic hierarchy to find if there are subscribers to parent topics. Whenever a subscriber is found the broker

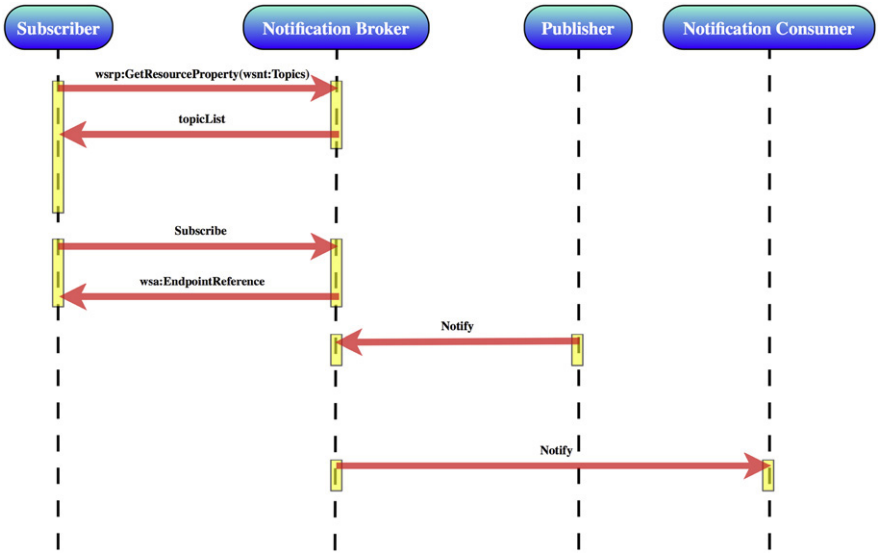

Fig. 11. Interaction diagram for notification publishers, brokers and consumers.

creates a list of the consumers and notifies them as in the list. When notifying, the broker also checks for paused consumers, in which case it drops the notification.

4.6. WS-Notification messaging

The event server keeps, among other things a dynamic structure representing the topics available for publication and registration in the home. Such data structure is hierarchically organized in the form of a tree. The root of the tree is called event and represents the entire set of the events that can occur. An example of such a WS-Topic is the following:

```
<?xml version="1.0" encoding="UTF-8"?>

<wstop:topicSpace name="Event"
 targetNamespace="event broker namespace URI"
 xmlns:esns=" event broker namespace URI "
 xmlns:wstop="http://www.ibm.com/xmlns/stdwip/web-services/WS-Topics">
    <wstop:topic name="Topics" final="true" />
    <wstop:topic name="MySubscribedEvents" final="true" />
    <wstop:topic name="Health" >
        <wstop:topic name="Fall" />
        <wstop:topic name="High pressure" />
    </wstop:topic>
</wstop:topicSpace>
```

where we notice a `Health` event having two children events, that are `Fall` and `High pressure`, representing respectively possible states of the home inhabitant.

Now suppose that a mobile user wants to subscribe to a specific event. Initially, s/he discovers the topic tree as described above. Then it must specify a number of parameters. First, the address of the client which will be receiving the event notifications. This is achieved by sending a message that adheres to the following structure:

```
<wsnt:ConsumerReference>
  <wsa:Address> hostName:port </wsa:Address>
</wsnt:ConsumerReference>
```

where `hostName` is the client's host name, and `port` is the default port on which the client is waiting for message notification. Second, one must choose the event to subscribe to. The way to express the event name is specified, in the topic expression dialect exposed by the broker. In our case, we have to insert the tag:

```
<wsnt:TopicExpression dialect=myDialect
 xmlns:esns="eventServerNamespaceURI">
    esns:eventPath
</wsnt:TopicExpression>
```

where `myDialect` is the reference to where the topic expression dialect is, and `eventPath` is the name of the event expressed following that dialect. Third, one has to specify an expiration time for the subscription. The expiration time is not a mandatory parameter to provide. If it is not specified the default value is infinite, that is, the subscription will not expire. The format for expiration is

```
<wsnt:InitialTerminationTime>
  xmlDate
</wsnt:InitialTerminationTime>
```

where `xmlDate` is the date and time (expressed as standard XML Schema type). Finally, one may also specify precondition and selector parameters. These determine the conditions under which the notification to the event consumer will take place.

When the event broker receives a subscription for a particular event, it has to store it into an appropriate repository. This will be queried when an event occurs. If there are devices subscribed to that event, and the rules of the precondition parser are satisfied, a notification will be sent to all appropriate subscribers. The notification has the following form

```
<wsnt:Notify>
  <wsnt:NotificationMessage>
    <wsnt:Topic dialect="myDialect"
     xmlns:esns="eventServerNamespaceURI">
        esns:eventPath
      </wsnt:Topic>
      <wsnt:Message>
        message
      </wsnt:Message>
    </wsnt:NotificationMessage>
</wsnt:Notify></wsnt:InitialTerminationTime>
```

where for brevity we omitted the SOAP headings.

4.7. Discussion

The salient characteristics of the proposed architecture are its openness and its scalability. These features are obtained through the use of standard XML-based Web service protocols and the publish and subscribe communication pattern.

If Web services are becoming one of the most effective and popular tools for interoperation, they come at the price of using verbose and computationally intensive XML-based messages. Thus, we do not expect all home devices to "talk" Web services, but rather we envision a home where moderately to very powerful devices implement the Web service stack, while the little devices such as sensors connect to some interface device implementing the Web service stack. The Web service infrastructure becomes then a sort of Enterprise Service Bus [12] for the home interconnecting a possibly infinite set of homogeneous sensor networks and ultimately guaranteeing the interoperation of the various home busses.

These busses can then be implementing any other domotic standard such as OSGi, HAVi, UPnP, Konnex, and the Web service infrastructure becomes the "glue". An additional advantage is the backward compatibility. One can build

on top of existing domotic installations by simply wrapping the existing elements behind new bridges implementing the Web service stack. Finally, installations requiring the presence of the Java Virtual Machine, such as OGSi and Jini, can seamlessly interoperate with installations not requiring Java, such as HAVi devices.

The relation with OGSi [19] is particularly relevant as our proposed approach and OGSi address the same issue, i.e., that of interoperation of embedded devices. However, we remark a deep difference in approaching the subject. The OGSi initiative proposes the addition of a middle layer on all devices based on Java technology and assumes the availability of some computational resources to additionally operate this layer, while we propose a message-based approach in which the information is passed on using standardly formatted messages. This said, we remark again that the two approaches, though different in philosophy, are still compatible. In fact, OGSi could be governing a set of devices, and then be interconnected to other sets of devices via the WS-Notification infrastructure we propose here.

5. Case study

It has been shown that life expectancy increases if the elder adult is left to live in her/his own home. Furthermore, avoiding hospitalization, unless strictly necessary, reduces social costs. Thus the goal is to let the person live in her/his own home as long as possible without jeopardizing her/his safety and comfort. Our case study is based on the requirements of a pilot project of the Trentino Institute for Public Housing (ITEA) to be deployed and tested in the complex of Piazza Garzetti in Trento, Italy.

5.1. The ITEA case study

ITEA is promoting the study of off-the-shelf and state-of-the art technology with the goal of aiding the aging population. Recent statistics show that in Trentino 17.8% of the total population is above 65, 8.7% is above 75, and 2.4% above 85 [27]. As in most wealthy regions of the world, these percentages are steadily increasing. The requirements of the project can be summarized by the following points:

- *support without interference:* daily behavior of the user should not be modified or affected by the introduction of domotic infrastructure;
- *deploy everywhere:* the domotic infrastructure must be deployable in any existing home with the minimum possible intervention/modification of the apartment;
- *safety:* the safety of the home inhabitant should be always guaranteed;
- *health monitoring:* the health of the user should be constantly monitored to detect when external intervention is needed.

One of the most common accident affecting the elder adult in her/his own home is the fall. In Great Britain, for example, one third of home accidents of people above 65 are falls. The goal of the case study is thus the realization a fall detector satisfying the above requirements.

The case study is best understood referring to Fig. 8. An elder citizen lives in his home. The home is equipped with a wireless camera in every room. The user also wears a belt accelerometer which transmits data to a mini personal computer in the home. The personal computer is connected to the Internet and is equipped with a number of wireless interfaces: Bluetooth, WiFi, and GPRS. A nurse, or relative living in the same apartment of the user has a Bluetooth equipped device. It could also be the case that there is one nurse for the entire building. In this case, s/he would be equipped with a WiFi device. Relatives not living near the building and not having access to the local networks can be reached on their mobile phones via the GPRS connection. Finally, a health or emergency center can be connected via a wired Internet connection.

5.2. Sample interaction

Suppose a nurse is entering a domotic home, such as the one of Piazza Garzetti in Trento and she has her mobile phone with her. She might then wonder what services are available in the house she has just entered. If she already has the client implementing the Web service stack she can directly browse through the topic tree of the home, if not, she can simply download it using a Bluetooth, a WiFi or GPRS connection.

Then she is presented with the currently available topic tree. For instance, there might be a category of health events. In Fig. 12a, we show a screenshot from the Symbian OS implementation for the Nokia 6000 family. One can

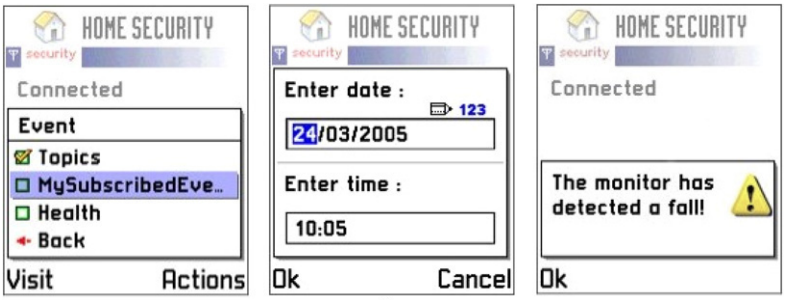

Fig. 12. Viewing the topic tree (a), registering for an event (b), and receiving a notification (c).

then browse the topic tree, view all available events and decide to register for one event, for instance, the fall event in the health category. The topic tree can also be modified by a client. The subscription is finalized by specifying an ending time for the subscription (Fig. 12b).

Suppose a new device which detects fires is introduced in the home. This could create a new topic branch for alarms and have a leaf event called "fire" (not shown).

Once the client has registered for the fall event and the server's rule engine deduces a fall event has occurred (by cross-relating data coming from the camera and the accelerometer), then all clients registered to the event are notified. This in turn is transformed in the client into an appropriate message for the user, as shown in Fig. 12c.

5.3. Implementation

The case study has been implemented using the following equipment. *Web service stack implementation as server:* a standard personal computer,[1] video-processing software realized by CNR-IMM, Lecce, a WS-Notification server, and the JESS rule engine [21]. *Web service stack implementation as clients:* a mobile phone Nokia 6600 running Symbian OS and a C++ implementation of the Web service stack; a PDA Palm Tungsten C running Palm OS running the J2ME[2]; a Windows mobile C# implementation (run in emulation); and a java applet for interaction via the web. *Devices and actuators:* a wireless camera; a custom wearable MEMS accelerometer with a radio transmitter, developed by ITC-irst, Trento.

In Fig. 13, the Palm Tungsten and Nokia 6600 are depicted in front of the emulated version of the Windows Mobile PDA. The picture is taken when a notification of a `Fall` event is received by these registered devices.

5.4. Evaluation

The proposed architecture is deployed in an actual home located in Trento–Piazza Garzetti. The objective of the installation is to exhibit the potential of state-of-the-art technology and to test the proposed framework. The main goal of our evaluation is to establish the feasibility of the Web service approach and its reliability. To this end, we have used devices of very different nature, not to compare their relative performances, but rather to show that indeed the handling of the heterogeneity is easily achievable. To show that backward compatibility is possible we have used a relatively 'old' Palm pilot, while a more recent Symbian phone (notice, not a smart phone) illustrates where we currently are.

We have run a number of experiments to show feasibility and reliability of the proposed approach. Here we report on the two most relevant ones. The first one deals with the time necessary to consume a message. The evaluation measures the time needed by a device to process a message of an event and present it to the user. This time does not include network latency time, but it does include the time necessary for interacting with the on-board network interface, to parse the XML messages and for displaying information to the GUI.

[1] AMD Athlon 2000, 768MB RAM DDR, Windows XP Service Pack 2, J2SE 1.4.2-10, Ethernet 10/100, WiFi access Point: DWL-1000AP+ D-Link.

[2] Palm Tungsten C, Palm OS, Java: J2ME CLDC 1.1 MIDP 2.0.

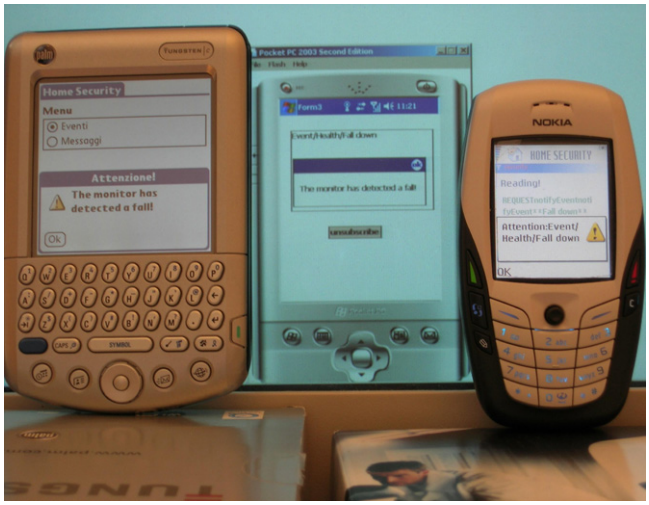

Fig. 13. Heterogeneous devices running the domotic application.

Measure	Average	Deviation	Precision
Nokia, Symbian OS, C++	2,322.10	645.00	0.12
Palm, Palm OS, Java	8,584.00	120.37	10.00
PDA emulator, XP, .Net/C++	636.00	491.00	1.00
Applet, XP, Java	41.52	28.74	1.00

Fig. 14. Message processing times in milliseconds.

Notification frequency [seconds^{-1}]	1/7	1/6	1/5	1/0.65	1/0.07
Nokia, Symbian OS, C++	100%	100%	100%	—	—
Palm, Palm OS, Java	100%	91%	89%	—	—
PDA emulator, XP, .Net/C++	100%	100%	100%	100%	—
Applet, XP, Java	100%	100%	100%	100%	100%

Fig. 15. Percentage of received notifications for increasing event frequencies.

In Fig. 14, the results of the first test are summarized. For each device, the type of device, the operating system, and the programming language used to develop the application are shown. For each of these, we computed the time necessary to process a notification. In figure, the average processing time over a number of notifications in varying load conditions is reported together with the standard deviation and granularity of the measurements. For the two physical devices the processing time is in the order of a few seconds, while for the device running on virtual machines on a PC the processing times are in the order of the milliseconds.

The second test, is to see how many notifications can be handled in the unit of time by each device. This is not so relevant for the presented healthcare case study, where one expects to have very few events per year, but it does give an indication of how to approach can scale to other situations. The actual goal of the test is thus to test the number of missed notification by each single device while increasing the number of notifications sent per unit of time.

An increasing number of notification is sent, in order to see what is the maximum rate tolerated by the device (Fig. 15). Down to one message every seven seconds, all devices capture all notifications. The first device to miss notifications is the Palm. What happens is that the notification processing time is higher than the time of arrival of subsequent notifications. A similar situation happens with the mobile phone when one message every 0.65 seconds

is sent. Even though the processing time is much higher, the device is able to store the arriving messages. The experimentation with the falling event notification has shown the limit to be at 66 messages. After this number is reached, there is an overflow and the application throws a panic exception. With the virtual environments (PDA emulator and Java Virtual Machine/Applet), when going below the rate of a notification every 0.07 seconds, the results get less meaningful as other factors intervene in the generation and loss of notifications.

5.5. Discussion

The evaluation shows the feasibility and reliability of the approach. One notes that standard devices with limited resources, such as a standard Nokia mobile phone, can process notifications in few seconds and can tolerate message rates up to 1 every half second. Furthermore, the presented evaluation should be taken as a picture on the current state of technology, not as the limit that can be reached with Web services in domotics. Currently we are assisting at mobile devices which are more and more powerful computationally. Not only that, implementation of better performing XML libraries is well underway. In our implementation, we have used standard XML libraries and, in two cases, the implementations are based on the Java Virtual Machine which is known to be an additional performance overhead.

Strictly related is the porting of the architecture to different case studies. Clearly the one for monitoring the fall is one where very few events are generated, but the current processing times and especially the current trend of technology, lets us foresee that the architecture can be used in contexts where many more notifications per unit of time need to be handled. For instance, in an edutainment environment real time requirements can be much stronger and notifications of user interactions may occur many time per second.

While performing the main test, we have also observed the battery life of the devices. In particular, for the Nokia phone we have remarked very little additional power consumption. The 150 hours of declared battery life go down to more than a half of that (around 90 hours) with the application constantly running.

Incidentally, we remark that the requirements of ITEA presented in Section 5.1 are fully met by the proposed architecture. First, support without interference is achieved as the system assumes nothing from the user. The only requirement is to wear the accelerometer. Deploy everywhere is obtained by using wireless technology for the communication among all elements of the infrastructure. Nothing peculiar in the topology or facilities of the home is assumed. Finally, safety and health monitoring are guaranteed by the specific application which reliably notifies of hazardous situation in which it is likely that the user has fallen. Furthermore, the system has been built with off-the-shelf products the cost of which is affordable by the average household.

6. Related work

Domotics is an application area where many different research fields have an impact. Technological issues meet sociological and medical concerns. Of particular interest is the case of the elder population [41]. The physical inabilities which inevitably come with aging, together with the need of health monitoring, demand special attention which used to be possible only in healthcare structures. With appropriate domotic technology, it is possible to give the same level of comfort and safety letting the elder citizen live in her/his own home. The goal is not that of offering a technological home to the end-user, which is unlikely to go through a learning curve to use the new potential of her/his home, but rather to transparently pervade the home of intelligence aiding the user, possibly without her/his awareness. Of course, awareness and control can always be returned to the user, if so desired.

A common problem affecting the elder adult in his/her home is that of falling. This dangerous event has an incidence of at least 30% amongst persons aged over 65 [40], therefore, this type of accident is attracting the attention of domotic research. In [34] the monitoring of the user is achieved via a camera which detects posture, while in [40] the monitoring is achieved using an array of infrared sensors. Other work on fall detection has been reported in [17, 35,18].

Other researchers have proposed an approach based on Service-Oriented Computing for domotics, as we do here. Tarrini and Miori advocate in [42], as we do, the use of Service-Oriented Architectures for domotic applications. In particular, they concentrate on the layers above the operating system for the devices and propose the use of the DomoML markup language [22] for solving the interoperability problem. On the negative side, there is no reference to any implementation, evaluation, or case study. The Gator Tech Smart House [26] is a programmable approach to smart homes targeting the elder citizen. The idea is to have a service layer based on OGSi [43] in order to

enable service discovery and composition. This work is close to ours as for the SOA spirit, though it does not commit to any open standard or XML-based technology hindering openness and dynamic scalability of the approach. Furthermore, no references is made to the communication model adopted in the home. A similar approach is taken in the MAVHome [14] project, which utilizes a centralized approach using CORBA and a relational database for storing sensor information. In this work no indication is given on the efficacy and efficiency of the enabling middleware. Furthermore, the degree of openness appears to be limited, given the specific choices made.

The problem of interconnection and configuration of devices in the context of pervasive computing has been highlighted in [10] bringing to the adoption of an ad hoc solution in the context of edutainment scenarios [9]. Jammes et al. [28] report on preliminary work on the usage of SOA for solving the interoperability problem with special focus on legacy issues. The work involves the implementation of a Web service stack on devices following a form of device profiling. The stack proposed is based on WS-Eventing rather than WS-Notification and implemented on Linux, Windows, Windows CE, ThreadX, and QuadrOS. QoS levels of the stack, such as WS-Security, are not implemented. Preliminary experimental results on response times of the devices are encouraging, but no further evaluation is provided.

7. Conclusions

The home is calling for our attention. The availability of appliances with increasing computational power and networking abilities, together with the availability of low cost computational elements (ranging from mobile phones hardware for few dollars to smart dust technology) is creating a wide spectrum of opportunities for industry and research activities. The research issues involve many fields including medicine, sociology and technology. In particular, in the field of distributed systems, there are open challenges that have to do with openness, scalability, and heterogeneity.

We claim that Web services have a central role in the home of the future as the infrastructure for total interoperability. The vision is that of a hierarchy of home sensors, actuators and devices with different computational and networking abilities. Some of these implement the Web service stack, while the simplest ones will simply connect to a controller implementing the Web service stack. The communication pattern is the publish/subscribe one.

We showed the feasibility of the proposed approach in a concrete case study by monitoring the falls of the elder adult in a domestic environment using a number of heterogeneous sensors and clients. Communication and coordination is achieved using Web service technology and WS-Notification in particular. The resulting information system is scalable as devices can join and leave the architecture dynamically. The hierarchy of published and subscribed events is dynamic as well and may be modified at runtime.

There are a number of issues regarding the proposed Web service-based infrastructure which are opened by the present proposal. First, the move from a scenario of type (S3) to an (S4) one is interesting and possible. Such a move may prove to be crucial in different application scenarios. It implies getting rid of the centralized WS-Notification server and implementing a purely peer-to-peer topology. Second, one can consider the issue of context awareness [5] of the elements which are mobile in the domotic environment. In the current setting, all computational elements are assumed to have the same context, but one can generalize and consider different contexts, for instance, different topic trees depending on the localization of the device. Third, we have not considered security issues in the proposed architecture, but these are surely relevant given the sensitivity of the data involved and the potential threats to the health of the home inhabitant. Having built the approach on the Web service stack, we consider protocols belonging to this stack to be the natural candidates for addressing these issues, e.g., WS-Security [30]. Fourth, one can consider the monitoring of other diseases and hazardous events regarding the home inhabitants. Fifth, one can consider other application scenarios in which network partitions and churns are the norm, further challenging the message delivery guarantees and mechanisms (e.g., a Manet [7,16]).

Acknowledgments

We thank the colleagues involved in the ITEA project, the students and programmers who contributed to the research and implementation for the case study, most notably Manuel Zanoni, and also Alessandro Zolet, Michele Iannotta, Mirnes Osmic, Paola Santoni, and Adriana Zampa. We are also grateful to the anonymous reviewers for their constructive comments which improved the contents and presentation of the paper. We thank Eirini Kaldeli and

Elie El-Khoury for proofreading the article. The research reported in this paper is supported under University of Trento Special Project Grant CRS no. 401000876 on Domotics.

References

[1] M. Aiello, The role of Web services at home, in: IEEE AICT-ICIW '06: International Conference on Internet and Web Applications and Services, WEBSA, 2006, pages 164–ff.
[2] M. Aiello, M. Marchese, P. Busetta, G. Calabrese, Opening the Home: A Web service approach to domotics, in: Applied Computing 2005, vol. I, 2005, pp. 271–279.
[3] M. Aiello, M. Zanoni, A. Zolet, Exploring WS-Notification: Building a scalable domotic infrastructure, DrDobbs Journal: Software Tools for the Professional Developer 371 (2005) 48–51.
[4] G. Alonso, F. Casati, H Kuno, V. Machiraju, Web Services, Springer-Verlag, 2004.
[5] M. Baldauf, S. Dustdar, F. Rosenberg, A survey on context aware systems, International Journal of Ad Hoc and Ubiquitous Computing 2 (4) (2007) 263–277.
[6] R. Baldoni, A. Virgillito, Distributed event routing in publish/subscribe communication systems: A survey, Technical Report 15-05, Dipartimento di Informatica e Sistemistica, Università di Roma La Sapienza, 2005.
[7] P. Bellavista, A. Corradi, E. Magistretti, REDMAN: An optimistic replication middleware for read-only resources in dense MANETs, Pervasive and Mobile Computing 1 (2005) 279–310.
[8] Box et al. Ws-eventing specification, 2004. ftp://www6.software.ibm.com/software/developer/library/ws-eventing/WS-Eventing.pdf.
[9] J. Burke, J. Friedman, E. Mendelowitz, H. Park, M. Srivastava, Embedding expression: Pervasive computing architecture for art and entertainment, Pervasive and Mobile Computing 2 (2006) 1–36.
[10] J. Burke, E. Mendelowitz, J. Kim, R. Lorenzo, Networking with knobs and knats: Towards ubiquitous computing for artists, in: Ubiquitous Computing 2002, Concepts and Models Workshop, 2002.
[11] Catania et al. Ws-events specification, 2003. http://devresource.hp.com/drc/specifications/wsmf/WS-Events.pdf.
[12] D. Chappell, Enterprise Service Bus, O'Reilly, 2004.
[13] S. Consolvo, P. Roessler, B. Shelton, A. LaMarca, B. Schilit, S. Bly, Technology for care networks of elders, IEEE Pervasive Computing 3 (2) (2004) 22–29.
[14] D. Cook, Health monitoring and assistance to support aging in place, Journal of Universal Computer Science 12 (1) (2006) 15–29.
[15] F. Curbera, R. Khalaf, N. Mukhi, S. Tai, S. Weerawarana, The next step in Web services, Communications of the ACM, [37], 29–34.
[16] F. De Rosa, A. Malizia, M. Mecella, Disconnection prediction in mobile ad hoc networks for supporting cooperative work, IEEE Pervasive Computing 4 (3) (2005) 62–70.
[17] T. Degen, H. Jaeckel, M. Rufer, S. Wyss, SPEEDY: A fall detector in a wrist watch, in: Proceedings. Seventh IEEE International Symposium on Wearable Computing, 2003.
[18] J. Demongeot, G. Virone, F. Duchêne, G. Benchetrit, T. Hervé, N. Noury, V. Rialle, Multi-sensors acquisition, data fusion, knowledge mining and alarm triggering in health smart homes for elderly people, Editions scientifiques et médicales 325 (2002) 673–682.
[19] P. Dobrev, D. Famolari, C. Kurzke, B.A. Miller, Device and service discovery in home networks with OSGi, IEEE Communications Magazine 40 (8) (2002) 86–92.
[20] P.Th. Eugster, P.A. Felber, R. Guerraoui, A.-M. Kermarrec, The many faces of publish/subscribe, ACM Computers Survey 35 (2) (2003) 114–131.
[21] E. Friedman-Hill, Jess in Action : Java Rule-Based Systems, Manning Publications, 2003.
[22] F. Furfari, L. Sommaruga, C. Soria, R. Fresco, DomoML: The definition of a standard markup for interoperability of human home interactions, in: EUSAI '04: Proceedings of the 2nd European Union symposium on Ambient intelligence, ACM Press, 2004, pp. 41–44.
[23] N. Furmento, J. Hau, W. Lee, S. Newhouse, J. Darlington, Implementations of a Service-Oriented Architecture on top of Jini, JXTA and OGSI, in: Second European AcrossGrids Conference, AxGrids 2004, in: LNCS, vol. 3165, Springer, 2004, pp. 90–99.
[24] Graham et al., WS-Notification: Publish-subscribe notification for Web services, http://www-106.ibm.com/developerworks/library/specification/ws-notification/, 2004.
[25] A. Hauptmann, J. Gao, R. Yan, Y. Qi, J. Yang, H. Wactlar, Automated analysis of nursing home observations, IEEE Pervasive Computing 3 (2) (2004) 15–21.
[26] S. Helal, W. Mann, H. El-Zabadani, J. King, Y. Kaddoura, E. Jansen, The gator tech smart house: A programmable pervasive space, Computer 38 (3) (2005).
[27] ISTAT, Population Census 01, ISTAT Editions, 2001.
[28] F. Jammes, A. Mensch, H. Smit, Service-Oriented device communications using the devices profile for Web services, in: MPAC '05: Proceedings of the 3rd International Workshop on Middleware for Pervasive and ad-hoc Computing, ACM Press, 2005, pp. 1–8.
[29] A. Joseph, Successful aging, IEEE Pervasive Computing 3 (2) (2004) 48–50.
[30] L. Kagal, T. Finin, A. Joshi, Declarative policies for describing Web service capabilities and constraints, in: Proceedings of W3C Workshop on Constraints and Capabilities, 2005.
[31] P. Leoni, A Web service based domotic architecture, Bachelor Thesis, Univ. of Trento, http://dit.unitn.it/~aiellom/tesi/leoni.doc, 2004 (in Italian). Winner of the ANIA-Confindustria prize.
[32] Liang-Yen Lin, Ming-Chun Cheng, Shyan-Ming Yuan, Standards-based user interface technology for universal home domination, in: Int. Conf. on Hybrid Information Technology, IEEE, 2006, pp. 298–307.
[33] E. Mynatt, A. Melenhorst, A. Fisk, W. Rogers, Understand user needs and attitudes, IEEE Pervasive Computing 3 (2) (2004) 36–41.

[34] H. Nait Charif, S.J. McKenna, Activity summarisation and fall detection in a supportive home environment, in: Int. Conf. on Pattern Recognition, ICPR, 2004.
[35] N. Noury, P. Barralon, G. Virone, P. Boissy, M. Hamel, P. Rumeau, A smart sensor based on rules and its evaluation in daily routines, in: IEEE Engineering in Medicine and Biology Society, 2003, pp. 3286–3289.
[36] M. Osmic, Publish/subscribe in the context of Web services, Bachelor Thesis, Univ. of Trento, 2006 (in Italian).
[37] M.P. Papazoglou, D. Georgakopoulos, Service-oriented computing, Communications of the ACM 46 (10) (2003) 24–28.
[38] J. Roberts, Pervasive health management and health management utilizing pervasive technologies: Synergy and issues, Journal of Universal Computer Science 12 (1) (2006) 6–14.
[39] D. Ross, Cyber Crumbs for successful aging with vision loss, IEEE Pervasive Computing 3 (2) (2004) 30–35.
[40] A. Sixsmith, N. Johnson, A smart sensor to detect the falls of the elderly, IEEE Pervasive Computing 3 (2) (2004) 42–47.
[41] V. Stanford, Using pervasive computing to deliver elder care, IEEE Pervasive Computing 1 (1) (2002) 10–13.
[42] L. Tarrini, V. Miori, LIGHT: XML-Innovative Generation for Home Networking Technologies, ERCIM News, 62, 2005.
[43] Tuecke et al., Open Service Grid Infrastructure, 2003. http://www-unix.globus.org/toolkit/draft-ggf-ogsi-gridservice-33_2003-06-27.pdf.
[44] S. Vinoski, CORBA: Integrating diverse applications within distributed heterogeneous environments, IEEE Communications Magazine 14 (2) (1997).
[45] B. Warneke, M. Last, B. Liebowitz, K. Pister, Smart dust: Communicating with a cubic-millimeter computer, IEEE Computer 34 (1) (2001) 44–51.
[46] S. Weerawarana, F. Curbera, F. Leymann, T. Storey, D. Ferguson, Web Services Platform Architecture: SOAP, WSDL, WS-Policy, WS-Addressing, WS-BPEL, WS-Reliable Messaging, and more, Prentice Hall, 2005.
[47] Tianyi Zang, Wei Jie, T. Hung, Zhou Lei, S.J. Turner, Wentong Cai, Ming Zhu, C. Katsinis, An OGSI-compliant grid information service — its architecture and performance study, in: IEEE 7th Int. Conf. on High Performance Computing and Grid, 2004, pp. 63–71.

Marco Aiello is Associate Professor of Distributed Information Systems at the University of Groningen where he leads the Distributed Systems and Software Engineering group. In 2006 he was Lise Meitner Fellow at the Vienna University of Technology, while from 2002 to 2006 he was assistant professor at the University of Trento. He received a Ph.D. from the University of Amsterdam in 2002 that won the 2003 AI*IA Best Dissertation award. He holds a degree in Engineering and Computer Science from the University of Rome La Sapienza, awarded cum laude in 1997. His research interests lie in the fields of Service-Oriented Computing, Document Image Understanding and Spatial Reasoning.

Schahram Dustdar is Full Professor for Internet Technologies at the Distributed Systems Group, Information Systems Institute, Vienna University of Technology (TU Wien) where he is director of the Vita Lab and Honorary Professor of Information Systems at the Department of Computing Science at the University of Groningen (RuG), The Netherlands.

He received his M.Sc. (1990) and Ph.D. degrees (1992) in Business Informatics (Wirtschaftsinformatik) from the University of Linz, Austria. In April 2003 he received his Habilitation degree (Venia Docendi in Angewandte Informatik) for his work on Process-aware Collaboration Systems - Architectures and Coordination Models for Virtual Teams. His work experience includes several years as the founding head of the Center for Informatics (ZID) at the University of Art and Industrial Design in Linz (1991–1999), Austrian project manager of the MICE EU-project (1993–97), and director of Coordination Technologies at the Design Transfer Center in Linz (1999–2000). While on sabbatical leave he was a post-doctoral research scholar (Erwin-Schrödinger scholarship) at the London School of Economics (Information Systems Department) (1993 and 1994), and a visiting research scientist at NTT Multimedia Communications Labs in Palo Alto, USA during 1998.