INSIGHT: Interoperability and Service Management for the Digital Home

Charbel El Kaed[1,2] Loïc Petit[1,2] Maxime Louvel[1,3]
Antonin Chazalet[1] Yves Denneulin[2] François Gaël Ottogalli[1]

[1]France Telecom (R&D, NRS) [2]LIG Grenoble University [3]LISyC UE Bretagne
first.last@orange.com first.last@imag.fr first.last@univ-brest.fr

ABSTRACT

The emergence of plug-n-play protocols, multimedia and ubiquitous applications is shaping the human habitat into a digital one. The actual plug-n-play device proliferation encourages the development of ubiquitous applications providing the user with a wide set of services to accomplish his everyday tasks. Moreover, the user requires a smooth and transparent interaction with each of his devices and a high quality multimedia experience.

However, the digital home is more than ever an heterogeneous and complex environment, devices differ in terms of resources and networking protocols turning the user requirements into an almost impossible task.

This paper presents the INSIGHT middleware involving four modules to simplify the digital home complexity. *DOXEN* provides a cross-device interaction by automatically generating adequate proxies. The *DomVision* scans the digital home, processes and stores valuable information for management and troubleshooting. The *Service Level Checking* analyzes the *DomVision*'s information to provide a personalized service offer for a specific home. And finally, the *Resource Manager* guarantees the end-user QoS.

We also show, through a real experimentation, the ability of INSIGHT to tackle the digital home's complexity.

Keywords

Digital Home, Monitoring, QoS, Interoperability, SLC

1. INTRODUCTION

Advances in embedded systems, plug-n-play protocols, multimedia and software architectures bring the ubiquitous computing vision to the near future. In such environments, ubiquitous applications, deployed in the human habitat act as orchestrators and control plug and play devices to accomplish a specific task. In fact, each device supporting a plug and play protocol is able to announce its capabilities on the network. Thus, the ubiquitous applications rely on such advertisements to identify and interact with the required device. Clearly, the aim of each application is to hide the devices interaction complexity from end-users. For example, a deployed application can be used to select a photo from a camera, then, render it on the living room television.

Furthermore, the actual device plethora allows a proliferation in the development of ubiquitous applications and provide a wide set of services like multimedia user experience, monitoring, maintenance or even surveillance and intrusion detection. Such applications create a new market perspective which can be shared between three major actors. Telecoms operators provide Internet and telephony access through already deployed devices like home gateways. Manufacturers offer certified plug and play devices. And the third parties propose personalized ubiquitous applications to the user based on his actual home devices.

However, the home is more than ever an heterogeneous and complex environment for those three actors and the user. Devices differ in terms of **resources** and **networking protocols**. A wide variety of devices exist in the home and range from scare resource devices like sensors to more abundant ones like set-top-boxes. Furthermore, the ubiquitous applications are installed on home devices and share their underlying physical resources. Thus, the telecoms operators and third party application vendors must guarantee a minimum level of Quality of Service (QoS) when providing their applications. In other words, they must provide a certain amount of resources to the proposed applications such as CPU, memory or network bandwidth. Indeed, if an application lacks some resources, its QoS will be degraded.

As for the networking heterogeneity, multiple "*Plug and Play*" protocols have emerged like the Universal Plug aNd Play (UPnP)[17], the Intelligent Grouping and Resource Sharing (IGRS)[9] and the Device Profile for Web Services (DPWS) [13]. The Plug and Play protocols share three layers of heterogeneity which retain the cross-protocol interoperability [6]. First, each protocol relies on different networking stacks to publish its description and interact with other applications and devices. Second, each protocol announces its description in its own format. The UPnP uses an XML defined template while the DPWS and IGRS use the Web Service Description Language. The third and final heterogeneity layer consists in the description content. In fact, the three protocols target similar domains, however, each protocol defines its own content. A UPnP light, for example, hosts a `SwitchPower` service with a `Switch(true/false)` action to control the light while a DPWS light[15] uses the semantically equivalent action `SetTarget (On/OFF)`.

Middleware 2011 Industry Track, December 12th, 2011, Lisbon, Portugal.

This protocol diversity prevents applications to use any available equivalent device in the home to accomplish a specific task. For example, an application searching to interact with a UPnP light cannot use an available DPWS light. Moreover, designing applications to support multiple protocols is time consuming from the application vendors perspective, since they must implement the interaction with each device. For the telecoms operators, such diversity makes the diagnostic and the troubleshooting a more complex operation and adds a burden cost on their infrastructure and tools to target additional protocols. Furthermore, the lack of cross-device interoperability can frustrate users since they are restrained to a certain protocol and technology.

Additionally, telecoms operators, manufacturers and application vendors provide a maintenance service along with their proposed applications. Therefore, devices and applications of the home network must be monitored to provide sufficient information during diagnostic and troubleshooting operations. Even aside from the protocol issue, those devices are providers of a large quantity of heterogeneous data streams. From the monitoring at service-level status to the core metrics like CPU and network usage, all kind of data is useful to establish diagnostics and troubleshooting.

Based on the monitoring data, the analyzing phase can take place using for instance the Service Level Agreements (SLA) which specifies an expected state of a system like a service, a device or a network. Analyzing results can be used in the home for multiple purposes, such as to inform the hot-line about an event, propose and deploy new services at runtime or trigger troubleshooting operations.

Hence, we focus in this paper on the following challenges: **Interoperability** between plug and play protocols. The **Resource management** to ensure a QoS. And finally, the Home devices **monitoring**.

In such a sophisticated home environment, we believe that a middleware is essential to simplify the complexity for the actors and the user. Therefore, we propose the *INSIGHT* middleware which provides interoperability and service management. It includes four modules: *DOXEN*, a **D**ynamic **O**ntology-based pr**O**xy g**EN**erator that resolves the protocols heterogeneity and provides a cross-device interaction by generating proxies. *DomVision* scans the digital home, processes and stores valuable information for monitoring, diagnostic and troubleshooting. The Service Level Checking for the Digital Home *(SLC4DH)* interacts with the *DomVision* to propose applications based on the existing devices and already deployed applications. And finally, the *Resource Manager* guarantees the QoS in the home.

The rest of the paper is organized as follows, the next section details INSIGHT. Sections 3 and 4 present the experimentation carried out in the home and outlines some performance evaluation. In section 5 related work is discussed. Finally, section 6 concludes and presents future work.

2. THE INSIGHT MIDDLEWARE

The INSIGHT middleware provides an interoperability and service management in the home network. INSIGHT as shown in Figure 1 is a modular middleware with four components distributed on different devices. The cooperation of these distributed modules enables INSIGHT to handle the previously depicted challenges arising of the digital home. We detail in the following each module of INSIGHT.

DOXEN, shown in Figure 1, resolves the protocols heterogeneity by generating adequate proxies to expose non-UPnP devices as standard UPnP ones. This transparent adaptation provides interoperability to the applications and other devices searching to interact with UPnP devices only. The **DomVision** supports device ***Monitoring*** operations which consists in collecting data from the digital home. Such information is used for management operations as defined in [1] which is: administration, configuration and diagnostic. The **Resource Manager** (RM) module guarantees a ***QoS*** to meet the user expectations during the execution of an application. It requests collected data from *DomVision* such as the maximum amount of resources (CPU, Memory, Network Bandwidth) previously consumed by the application in order to reserve the required resources. The **SLC4DH** module allows ***Application proposal*** offering users applications meeting the current state of their digital home. It interacts with the *DomVision* to analyze collected data and proposes applications based on the existing devices and already deployed applications. We detail next each proposed module.

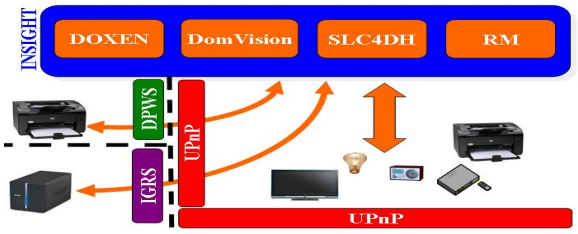

Figure 1: The INSIGHT Modular Architecture

2.1 DOXEN

We propose to use the UPnP protocol as a common pivot. We chose UPnP due to its wide acceptance among device manufacturers and telecoms operators. However, another pivot could be chosen instead. *DOXEN*[6] takes a previously validated alignment containing correspondences between two equivalent devices, then, generates a specific proxy. The proxy announces itself as a standard UPnP device which enables UPnP applications to discover and interact with the non-UPnP devices. The invocations received on the proxy are adapted and transfered to the equivalent non-UPnP device. The adaptation is possible since equivalent devices usually provide similar functions. A printer is always expected to print independently from the service description and protocol. Therefore, a DPWS or IGRS device can be represented using its equivalent UPnP description.

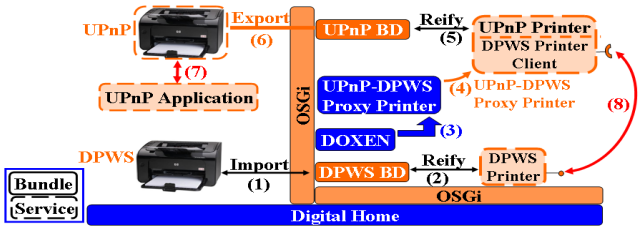

Figure 2: DOXEN's Proxy Generation

Figure 2 overviews *DOXEN* operating in the home. When a DPWS Printer appears on the network, the DPWS Base

Driver [15] (BD) which is a specific network converter, discovers (1) and reifies (2) the DPWS device as a local *OSGi* DPWS Printer Service. Then, *DOXEN* is notified about the arrival of the DPWS device. Therefore, *DOXEN* checks its configuration and since the DPWS printer has an equivalent UPnP printer, then *DOXEN* automatically generates (3) a proxy based on the two devices correspondences contained in an ontology alignment [6]. Once, the generated proxy is installed, it publishes (4) a UPnP printer device which is reified (5) and exported (6) by the UPnP BD [1] as a real device on the network. Thus, the UPnP application interacts (7) with the exported virtual device and the invocations are transfered to the proxy by the UPnP BD. Furthermore, the proxy translates and adapts (8) the invocations to the reified DPWS printer service. Then, the DPWS BD transfers the invocations to the real DPWS printer on the network.

2.2 DomVision

Home monitoring is essential to collect information from devices and keep a historical trace of events and measures. To collect this valuable information, *DomVision* scans the digital home searching for UPnP and UPnP-DM [18] devices and collects different kinds of data ranging from general description to specific device and network information.

The collected information includes the unique device identifier, the hardware model, friendly name and the supported services. Moreover, specific device data like the actual resources measures (CPU, Memory and Network Bandwidth) [18] partitioned by applications are also collected along with other networking information like the device addresses and the network topology. *DomVision* also subscribes to device notifications upon a state change, or an accomplished task. For instance, it attributes a time stamp for each collected data upon the device arrival/departure. Since the cross device interaction has been covered by *DOXEN*, it is also possible to gather data streams from non-UPnP devices.

The internal architecture of DomVision [14], see Fig. 3, is composed of two engines: a relational database which handles the persistence of any data that could be requested later; and a data stream management system [7, 14] which handles the process of data collection. In this usage, we have developed components to couple the two engines together, further reading on this topic is available in [14].

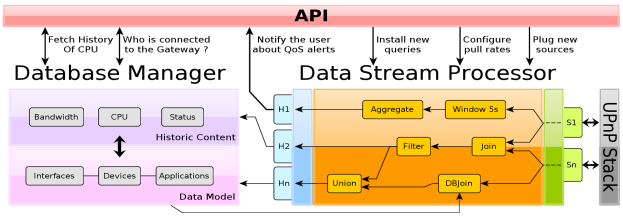

Figure 3: The DomVision internal Architecture

2.3 SLC4DH

The *SLC4DH* [5] module analyzes the information collected from *DomVision* and acts upon pre-configured Service Level Agreement (SLA) between users and services or inter-services. For instance, after analysis and detection of a UPnP digital camera, a UPnP printer and a UPnP TV, the SLC4DH proposes applications to the user conforming with its devices and services versions. The *SLC4DH* takes as input monitoring snapshots of the digital home produced by *DomVision* and analyzes the compliance of the digital home services with the defined SLA. Each SLA specifies one or more Service Level Objective (SLO) related to devices, network links, and services. For example, a SLO expresses a device type detection "Device.type=UPnP TV" or a change in resources "CPU usage $\geq$ 90%". The results produced by the *SLC4DH*, i.e. the verification of SLAs, are then made available for consumption by other modules for multiple purposes like to propose applications, notify the administrator, trigger troubleshooting operations.

SLC4DH has several benefits. It enables each business to express SLA from its own point of view and to associate SLA and services. It enables an analysis at the service level: *SLC4DH* automates the production of SLC results/reports that sum up the compliance of the targeted services with regard to their associated SLA. The SLC4DH is designed as an enabler in order to be easily (re-)usable by mechanisms that need analyzing functions.

2.4 Resource Manager

To guaranty the resources required by applications, we have designed a resource manager (*RM*) [10] module. It consists of two main components: the Global Resource Manager (GRM) and the Local Resource Manger (*LRM*). There is only one instance of the *GRM* in the home which is in charge of managing the network links and coordinating the reservation on each device via the *LRMs*. When the user starts a distributed application, requiring resource reservation, the *LRM* asks the *GRM* to check if there is enough resources. The *GRM* requests each involved *LRM* if it can provide enough resources. If all these requirements are met, resource reservations are guaranteed for the application.

When an application is started, if it has been executed before, with the exact same conditions (stream, devices and software), the *RM* module uses the same reservation values stored in the *DomVision* database. If the application has never been executed, no other alternative is possible than to make the biggest possible reservations. Afterward the *DomVision* module will compute the accurate resources requirements, based on the execution's monitoring.

These CPU and memory reservation are currenlty applied with the Linux control groups (*cgroups*) [11]. However, the LRM is generic and can rely on any other OS capable of providing isolation and reservation mechanisms.

3. EXPERIMENTATION

We distributed the INSIGHT middleware on different devices in the digital home connected to the *LiveBox* Internet gateway device in a star like topology as shown in figure 4. We placed *DOXEN* and the *SLC4DH* modules on a Felix [1] OSGi framework running on a *SodaVille* Set-Top-Box (STB) with an Intel Atom 1.2 GHz, 384 MB of RAM and an open-JDK 6 implementation. We put along with *DOXEN* on the STB, three UPnP-DPWS ontology alignments (lights, clocks, printers) [6], *DOXEN* can request or receive additional ontologies from the operator or the application vendor. The *DomVision* module and an Apache Derby database are deployed on an Felix [1] running on a *GuruPlug* with a 1.2 GHz *Marvell* CPU and 512 MB of RAM. A *GRM* and a *LRM* modules are deployed on a *BeagleBoard* with 700 MHz ARM Cortex-A8 CPU, 256 MB of

RAM and a standard *10.10 Ubuntu* distribution. We also deployed a UPnP standard media renderer [17] profile on the *BeagleBoard* which is connected to the TV screen through a high-definition multimedia interface. We deployed a *LRM* module along with a UPnP standard Media Server [17] profile on a 2 GHz CPU and 1 GB of RAM *DELL* laptop. We also installed a UPnP and a DPWS Light [1], [15] on a Felix framework. We used a DPWS HP 4515x printer implementing a DPWS standard profile. We detail next the three experiments carried out in the digital home.

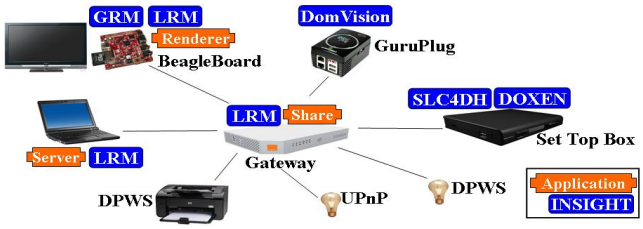

Figure 4: The Media Sharing Use case

3.1 Media Sharing Application

We implemented, the UPnP "Media Share" application which allows users to select any media (video, audio, photo) from a UPnP Media Server and render it on a UPnP Media Renderer. The application also interacts with a UPnP printer to print a photo from a UPnP Media Server. Additionally, the application offers personalized "Multimedia" profiles, for instance, the "Home Cinema" profile dims the selected surrounding lights when a movie starts. The "Media Share" application requires four mandatory UPnP devices: a Media Server, a Media Renderer, a printer and a light.

We specified a *SLA* with four mandatory *SLO*s requiring the four UPnP device types. We installed the SLA on the SLC4DH. *DOXEN* detects the DPWS Printer and Light then triggers the proxy generation. It installs on the STB two UPnP-DPWS proxies to expose the DPWS HP 4515x Printer and the DPWS Light as standard UPnP devices. The *DomVision* collects data from the new detected UPnP Printer and Light. The *SLC4DH* fetches data from the *DomVision* and checks the compliance with his SLA. Since the SLOs are satisfied, the SLA is compliant. The *SLC4DH* proposes the "Media Sharing" application to the user which downloads it on the gateway via the CWMP protocol[4].

Once the "Media Share" application is started by the user, it first scans the network searching for the four UPnP devices and asks the user to validate the lights, the media server and renderer to interact with. Once the user selects the "Home Cinema" profile and the movie, the application dims the DPWS and UPnP lights. The interaction with the DPWS light is transparent for the application, since the DPWS light is exposed as a UPnP standard light through the proxy. The *LRM* on the *Gateway* informs the *GRM* about the specific selected movie. The *GRM* queries the *DomVision* for the resource usage by the selected movie. If the answer to the query is not found in the local database, the *DomVision* requests the resource usage from the operator's database which stores an estimation from multiple digital homes. Once obtained, the *GRM* asks each involved *LRM* device (*BeagleBoard*, *Gateway*, Media Server/Laptop) for resource reservation. Once the requirements are met, the application can be started with a guaranteed QoS. The data stream is sent from the Media Server through the gateway and rendered on the TV by the *BeagleBoard*. The resources are freed when the movie ends.

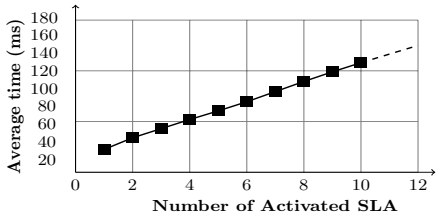

Figure 5: SLC4DH performance results

3.2 Android-UPnP Home Controller

We implemented a UPnP "Home Controller" application for Android on top of a UPnP *Cybergarage*[1] stack connected to the *Wifi* home network. The application interacts **only** with **UPnP** Lights, Clocks, and Printers. The "Home Controller" is capable of printing a file/photo, canceling a job and retrieving the printer status on the DPWS HP 4515x Printer. The Lights can also be turned on/off and dimmed.

3.3 Lights Power consumption

We installed a new query on the *DomVision* to periodically fetch the status and the intensity level of UPnP lights through standard UPnP calls. The aim of this query is to estimate the lights power consumption. The DPWS lights are also queried through the generated proxy. Based on the manufacturer/model description announced by each light, we can deduce each device consumption. Moreover, the *DomVision* can communicate such information to an energy manager service to help the user reducing his energy consumption. For example, this service will not recommend installing a new service if the energy consumption of the digital home is already high enough.

4. EVALUATION

We outline in this section the evaluation of the four modules. Table 1 resumes the time spent by DOXEN to generate and install a proxy, the lines of code of the generated Java files and the Jar bundle size. We evaluated DOXEN on the *Sodaville* and the *Dell* laptop. *DOXEN* spends less than 2 seconds on the laptop and 10 sec on the STB to parse the alignment and generate the proxies. The invocation redirection overhead reaches 20 ms for the printers and less than 5 ms for the lights compared to a direct client invocation.

Table 1: Generated Proxies

Proxy	Time (sec) STB	Time (sec) PC	Jar(KB)
Printer	10.1	1.7	67
Light	4.7	0.94	37
Clock	3.7	0.85	22

To evaluate *DomVision*, we investigated two aspects. First, the flexibility and the non-intrusivity: information

[1] `www.cybergarage.org`

retrieval from a new device must be fairly easy and no implementation must be done on the device. Second, it must be fairly efficient in terms of process delay. For the first point, in 3.3, we used the *DomVision* to monitor lights consumption. The implementation of such use-case consists in creating a new source and writing a query. As the source must only call a UPnP Action and get its results, the source is then an instantiation of a generic UPnP Source. The query fits into a 20 lines XML file containing the declaration of the source, the expression of the query, and the handler declaration to store the stream in the database. Since *DOXEN* exposes equivalent DPWS and IGRS devices as UPnP, collecting information from other devices is also supported. The *DomVision* is non intrusive since it listens to the UPnP Network. For the process delay, we measured on the *DELL* laptop the performance from the data reception to its storage in the database. On the minimal usage, i.e. the query only gets the stream content, we have an average of 5ms delay. Using a coupling with the database to fetch identifiers as well as some simple operations like filtering and removing useless attributes, we have an average of 10ms delay. Finally, with a complex query involving the database as well as a heavy computation using window aggregations and joins, we have an average of 30ms delay. For example, a query may be a request for an average data rate stream over a period of time from a network interface given the number of packet sent/received. Those measures represent the global delay expected from such solution. This delay is less than a query on a UPnP-DM device which is around 50ms to 500ms.

The *SLC4DH* performance results focus on the time spent in order to retrieve and analyze the monitoring snapshots with regard to a growing number of SLAs. Note that all the SLAs used here are different, but with a constant number of SLO. These results are an average of 10 executions of *SLC4DH* on top of a Felix on the *DELL* laptop. The first experimentation involved only one activated SLA, the second involved two different activated SLAs. Finally, the tenth one involved ten different SLAs. Each experiment has lead *SLC4DH* to analyze a hundred home monitoring snapshots, and to the production of SLA compliances and SLA violation results. In Figure 5, all the values of the average time spent are low. Moreover, the average time spent grows almost linearly with the number of activated SLAs. To conclude, the results obtained validate both the Service Level Checking approach in the Digital Home, the *SLC4DH* module developed, and the interaction with the *DomVision*.

To evaluate the resource manager, a multimedia application is started, the server is executed on the *DELL* laptop whereas the client on the *BeagleBoard*. A stress program is launched on the device decoding and playing the multimedia stream, executed with the *mplayer*[2] software. Fig. 6 shows the CPU used by *mplayer* with and without stress. Fig. 6(a) shows the evaluation when the RM is not used, fig. 6(b) shows the evaluation when the framework is used and the required reservation is made for the application. The gray (respectively black) curve shows the execution with a stress (respectively without stress). When the RM is not used the gray curve is lower than the black one. This indicates that *mplayer* uses less CPU when there is a stress. From the QoS point of view, *mplayer* does not have enough CPU to decode all the frames and the user-level QoS is really poor. On the contrary, when a reservation is made, the two curves almost merge, i.e. *mplayer* is not bothered by the stress. From the QoS point of view, there is no difference between the execution with stress and the execution without stress. To show the efficiency of quantity of resource reservation we have conducted the same kind of experiment for network [10] and memory resources. Hence, the *RM* is able to guarantee the applications' QoS, by reserving the required resources.

[2] `www.mplayerhq.hu`

5. RELATED WORK

Different works have been proposed in the literature to handle challenges of the digital home. We overview in this section those supporting one of the following characteristics: *Interoperability*(Int.), *Quality of Service* (QoS), *Monitoring* (Monit.) and *Application Proposal* (App.Prop.).

Table 2: Ubiquitous Middlewares

Middleware	Int.	QoS	Monit.	App. Prop.
Z2Z/Janus [2]	≈	x	x	x
WComp [16]	✓	x	x	✓
Atlas [8]	✓	x	x	✓
Vergara [19]	x	x	✓	✓
EASY [12]	✓	≈	x	x
HomeSOA [3]	✓	x	x	x
INSIGHT	✓	✓	✓	✓

(✓: supported, ≈: partial support, x: not supported)

Table 2 summarizes the middlewares and their supported features. The Z2Z project [2] proposes to resolve protocol heterogeneity by generating protocol gateways and wrappers to expose applications as web services. Their approach resolves the protocol layers heterogeneity but does not handle the service description heterogeneity which is essential for service discovery and interaction. In our solution the Base Drivers [1] and *DOXEN* resolves the protocol heterogeneity.

The *WComp* [16] middleware allows users and experts to compose and propose new services from existing devices and web services. The interoperability is covered by manually specifying proxies to wrap heterogeneous services as *WComp* components with a high-level representation. Even though, the manual proxy specification suits relatively simple devices, it becomes a difficult task when dealing with complex devices like printers. Their application proposal takes another aspect since *WComp* allows users to compose services visually or through commands. This feature makes offering a guaranteed QoS and a troubleshooting service impossible since the user can compose services even incompatible ones.

The Gator Tech project [8] proposes the ATLAS middleware which allows service composition and provides a web-based interface for users to compose or activate/deactivate services. The interoperability between devices is supported by representing each device using the Device Description Language. The description for simple devices such as temperature sensors is acceptable but it becomes a hard task when dealing with complex devices.

Vergara *et al.* in [19] details an autonomic solution similar to our *SLC4DH*. It uses an ontology and a set of rules to provide applications to the end-users based on their existing home devices and services. Their approach captures devices information for applications proposal only without providing administration, diagnostic and troubleshooting capabilities.

The *EASY* middleware proposed in [12] provides service

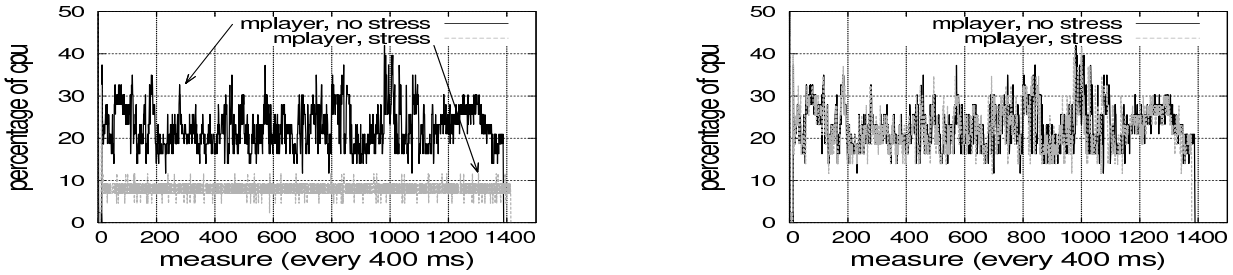

Figure 6: (a) CPU usage without reservation (b) CPU usage with reservation

interoperability and substitution by taking the QoS into account. The main drawback in *EASY* as mentioned in [6] is that it uses a global ontology which is too hard to build for a large set of devices and protocols. EASY exposes the resource requirement and QoS for a lot of metrics such as availability or network performances. Compared to INSIGHT, they do not consider the reservation on actual devices. Moreover, they assume that the resource requirements and estimation are available. However, for multimedia application this issue is not trivial, thanks to the *DomVision* module we were able to estimate the resource usage to guaranty a QoS level.

The *HomeSOA* middleware proposed by *Bottaro et al.* in [3] resolves the protocol and description heterogeneity of devices and services. It uses *OSGi* base drivers to reify devices locally as services then another layer of *refined drivers* abstract service interfaces per device types as a unified smart device. The interoperability is dependent on the manual annotated mapping between the different descriptions. However, there is no support for the QoS and no application proposal for the end-users based on their existing devices. The troubleshooting operations in the *HomeSOA* are not assisted with an event historical trace.

6. CONCLUSION

In this paper we introduced INSIGHT middleware to simplify the digital home complexity. INSIGHT consists of four distributed modules cooperating to handle dynamicity, resource reservation and protocol heterogeneity along with basic monitoring operations. The *DOXEN* module represents DPWS and IGRS devices as standard UPnP devices by automatically generating adequate proxies. The *DomVision* module scans the digital home network and collects information from UPnP devices. Such information is used by the *Resource Manager* to guaranty a QoS for users and applications. The *(SLC4DH)* interacts with the *DomVision* module to propose applications based on existing devices and already deployed applications. We deployed INSIGHT on real devices and carried out experimentations which proved that our middleware is able to handle different challenges in the digital home. Our future work will focus on the energy saving to reduce the digital home power consumption. We will also work on the application deployment from third parties. We will investigate package verification mechanisms to ensure security and prevent malicious functioning.

7. REFERENCES

[1] Apache, F.: Upnp base driver. http://felix.apache.org/site/apache-felix-upnp.html

[2] Bissyandé, T.F., Réveillère, L., Bromberg, Y.D., Muller, G.: Bridging the gap between legacy services and web services. In: Middleware (2010)

[3] Bottaro, A., Gérodolle, A.: Home soa - facing protocol heterogeneity in pervasive applications. In: In the 5th international conference on Pervasive services (2008)

[4] Broadband-Forum: Mr-230: Tr-069 deployment scenarios issue: 1 (August 2010)

[5] Chazalet, A., Bolle, S., Martin, S.: Analyzing the digital homes quality of service. The 4th Int' Workshop on Service Science and Systems (2011)

[6] El Kaed, C., Denneulin, Y., Ottogalli, F.G.: Dynamic service adaptation for plug and play device interoperability. In: CNSM'07 (2011)

[7] Golab, L., Özsu, M.T.: Issues in data stream management. SIGMOD on Management of data (2003)

[8] Helal, S.: The gator tech. www.icta.ufl.edu/gt.htm

[9] IGRS: http://www.igrs.org/

[10] Louvel, M., Bonhomme, P., Babau, J.P., Plantec, A.: A network resource management framework for multimedia applications distributed in heterogeneous home networks. In: AINA (2011)

[11] Menage, P.: Adding generic process containers to the linux kernel. In: Proc. of the Linux Symposium (2007)

[12] Mokhtar, S.: Semantic Middleware for Service-Oriented Pervasive Computing. Ph.D. thesis, University of Paris 6 (2007)

[13] OASIS: Devices profile for web services version 1.1, 2009. http://docs.oasis-open.org/ws-dd/dpws/1.1/os/wsdd-dpws-1.1-spec-os.html

[14] Petit, L., Roncancio, C.L., Labbé, C.: Astral: An algebraic approach for sensor data stream queryingṘesearch report, LIG (2011)

[15] SOA4D: Soa 4 devices. forge.soa4d.org

[16] Tigli, J.Y., Lavirotte, S., Rey, G., Hourdin, V., Cheung-Foo-Wo, D., Callegari, E., Riveill, M.: WComp Middleware for Ubiquitous Computing: Aspects and Composite Event-based Web Services. Annals of Telecommunications 64 (2009)

[17] UPnP: www.upnp.org

[18] UPnP: Device management- simplify the administration of your devices (2011)

[19] López de Vergara, J.E., Villagrá, V.A., Fadón, C., González, J.M., Lozano, J.A., Álvarez Campana, M.: An autonomic approach to offer services in osgi-based home gateways. Computer Communication (2008)