

# The Amigo Interoperable Middleware for the Networked Home Environment<sup>1</sup>

Daniele Sacchetti<sup>2</sup>, Yérom-David Bromberg, Nikolaos Georgantas, Valérie Issarny  
ARLES Research Team<sup>3</sup>, UR Rocquencourt, INRIA, France

Jorge Parra  
Ikerlan, Spain

Remco Poortinga  
Telematica Instituut, The Netherlands

## 1. Introduction

The Amigo interoperable middleware aims at enabling ambient intelligence within the networked home environment by addressing the seamless integration of networked devices and related application services within the home system (i.e., devices from the Consumer Electronics (CE), home automation, mobile and PC domains). The Amigo interoperable middleware architecture is specifically designed to realize an open networked home system that dynamically integrates heterogeneous devices as they join the network (§2). Our demonstrator (§3) will show interoperability among heterogeneous networked services from the CE, home automation, mobile and PC domains, using the Amigo middleware (§4).

## 2. The Amigo Interoperable Middleware

Key functions of the Amigo interoperable middleware core are depicted in Figure 1, which illustrates interoperability between a SLP-RMI and a UPnP service. Functions of the Amigo interoperable middleware core lie in: (i) *SDP Detection* and *Interoperability*<sup>4</sup> (*SDI*) for service discovery independent of the specific service discovery protocols used by networked services for advertising and requesting services (e.g., SLP and SSDP in our example), and (ii) *Service interaction interoperability* (*SII*) for enabling interaction between services, independent of the specific brokers (interaction protocols) used by networked services for being accessed and/or accessing remote services (e.g., RMI and SOAP in our example).

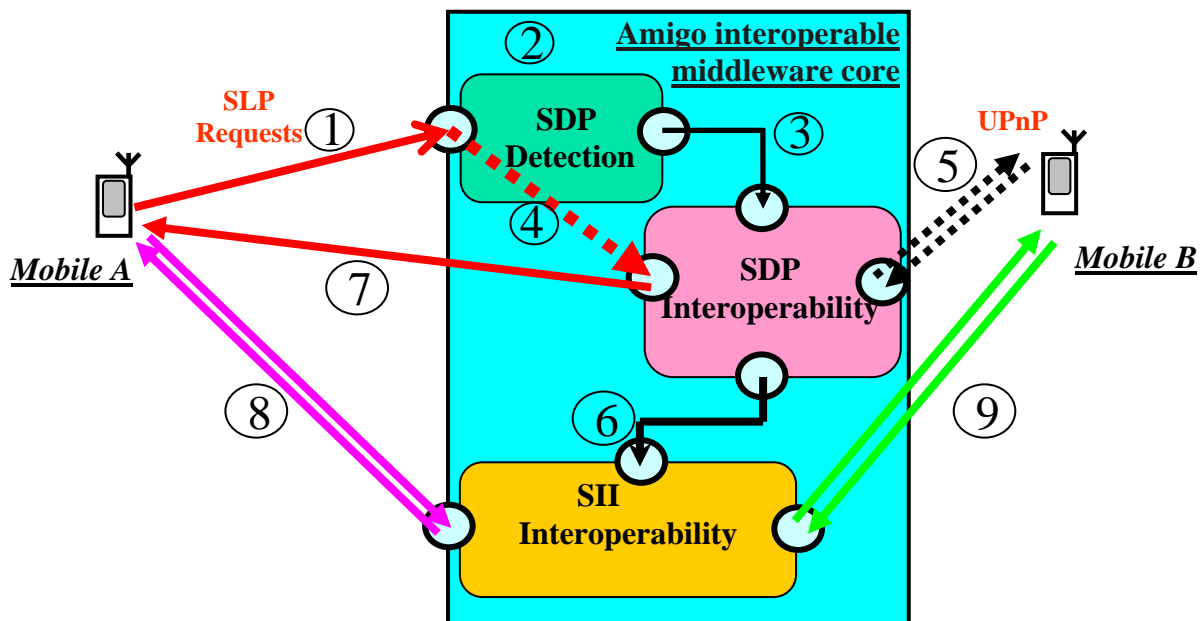


Figure 1: The Amigo interoperable middleware core

The Amigo interoperable middleware core may be deployed on one of the following three types of nodes of the networked home environment: service client, service provider or gateway, as most convenient according to the specific architecture of the networked home environment, and capabilities and usage of the networked devices. One key feature of the Amigo interoperable middleware core is that it is transparent to applications. Services that

<sup>1</sup> This work is done as part of the FP6 IST IP Amigo project; <http://www.hitech-projects.com/euprojects/amigo>

<sup>2</sup> Contact author: [Daniele.Sacchetti@inria.fr](mailto:Daniele.Sacchetti@inria.fr)

<sup>3</sup> <http://www-rocq.inria.fr/arles/>

<sup>4</sup> Y-D. Bromberg and V. Issarny. INDISS: Interoperable Discovery System for Networked Services. In Proc. of Middleware'2005.

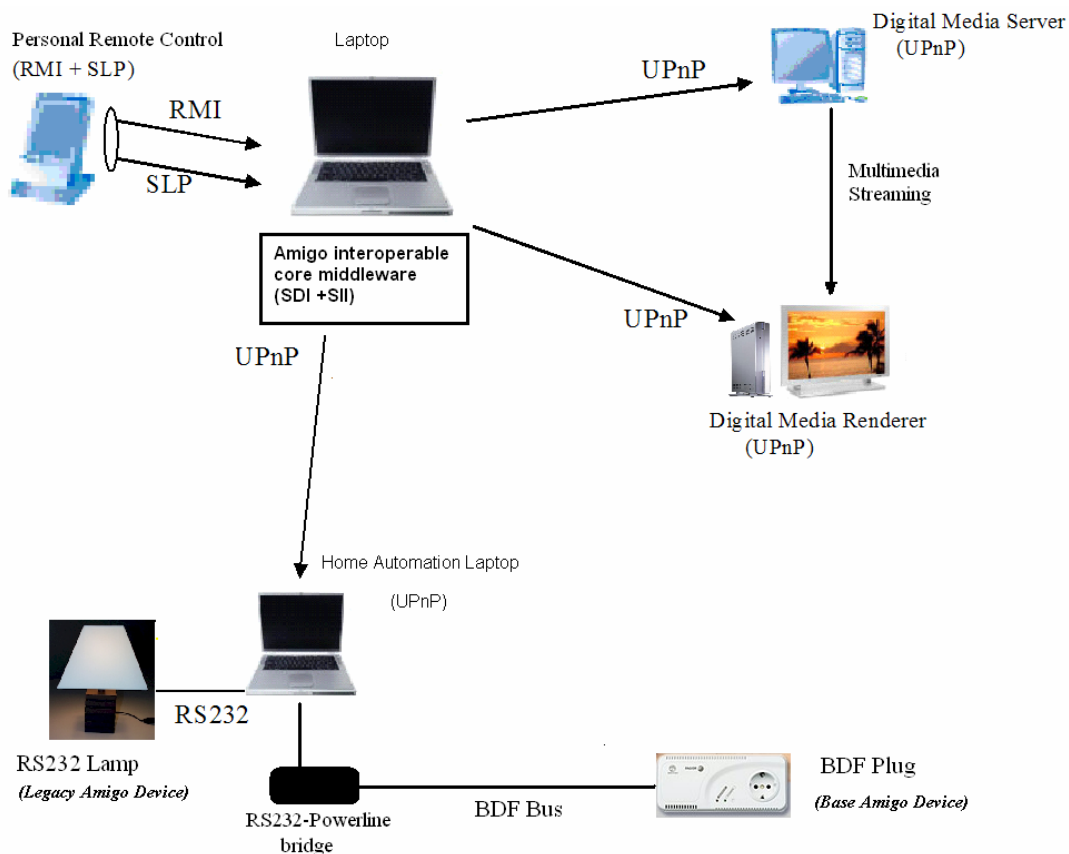
are networked in the Amigo home environment use legacy middleware core (e.g., SLP-RMI in our example) to interact with the environment, and in particular with remote networked services. The Amigo interoperable middleware core then interposes at the network layer to ensure interoperability with services based on distinct middleware technologies (e.g., interaction with a UPnP service in our example).

### 3. Scenario and Prototype Infrastructure

The demonstrator will illustrate the following scenario:

*It's a Tuesday evening and John is returning home after a day at work. He opens the door, takes his Personal Remote Control (PRC) out of his pocket and, as there is not enough light in the living room, turns on the lamp using the PRC's domotic GUI. Then, sitting on the sofa, he switches the TV set on, and using the PRC's GUI for Consumer Electronics, he browses the multimedia content directory available on the Digital Media Server (DMS) of the Amigo home environment via the wireless network. After having listened to some music, he finally decides to watch a movie. Before playing the movie, he turns on the coffee machine (using the PRC's domotic GUI) to warm up some coffee. And when coffee is ready, he finally turns off the lamp using the PRC's domotic GUI and sits on the sofa to watch the movie.*

Figure 2 shows the infrastructure of the prototype that implements the above scenario. The figure also shows the interactions among the different networked devices and components, in terms of service discovery and interaction, with a reference to the underlying middleware technology.



**Figure 2: Prototype infrastructure**

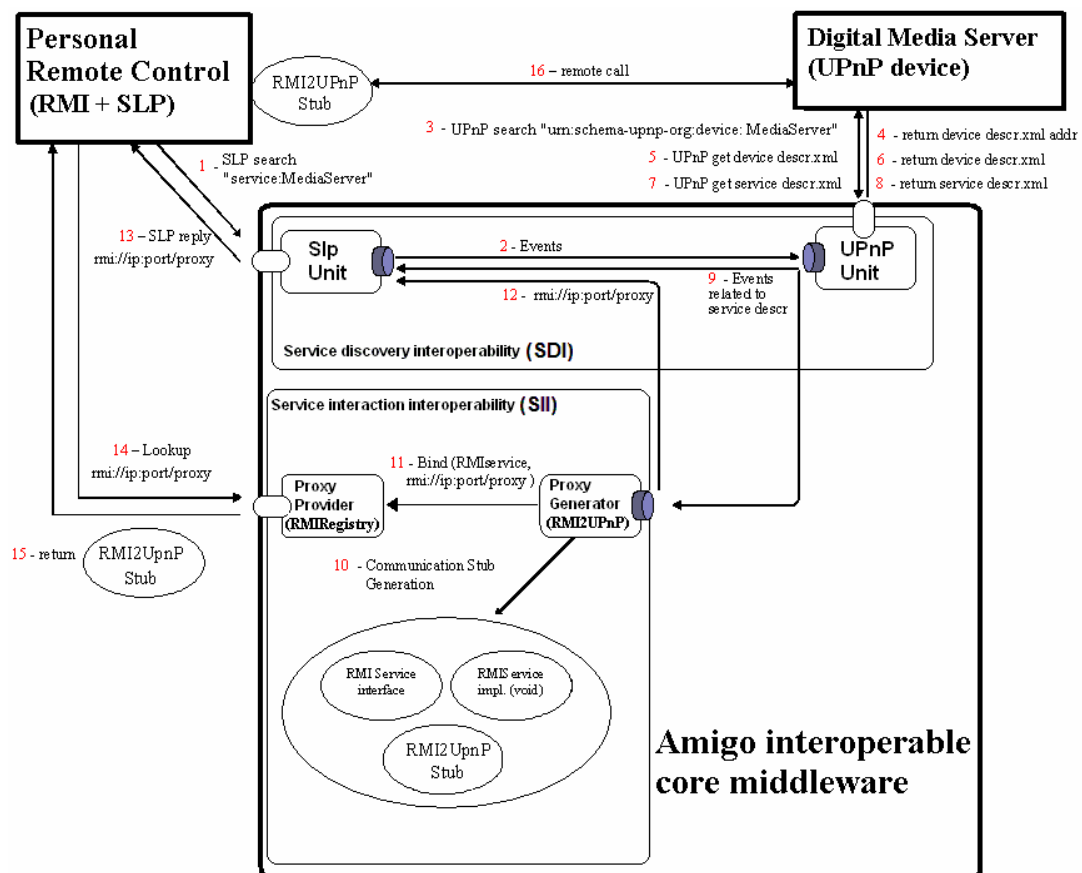
In our prototype, the *Personal Remote Control* is realized by a PDA (Personal Digital Assistant) that runs client applications providing the functionalities to control the CE and home automation devices available in the Amigo home environment. The *Digital Media Server* (DMS) provides the multimedia content. The *Digital Media Renderer* (DMR), which visualizes multimedia content, consists of a Philips Streamium<sup>5</sup> connected to a TV set through a SCART cable. The *Laptop* hosts the Amigo interoperable middleware core. The *Home Automation*

<sup>5</sup> <http://www.streamium.com/products/sl300i>

*Laptop* implements a UPnP proxy that controls the home automation devices installed in the home environment, i.e., the *lamp* and the *coffee machine*, connected to the laptop through a *plug*. The *PDA*, the *laptops*, the *DMS* and *DMR* are all connected through a Wireless WiFi Network in the infrastructure mode. As the *Laptop* runs the Amigo interoperable middleware core and further acts as an interoperability gateway, the services and clients running on the different networked devices can be discovered and interact, independent of the service discovery protocols and service interaction protocols upon which they are implemented. In the specific prototype infrastructure that we consider, the services related to the CE and home automation domains are based on the UPnP middleware technology, while the PDA runs SLP as service discovery protocol and RMI as service interaction protocol.

#### 4. Prototype Realization using the Amigo Interoperable Middleware

Figure 3 shows the sequence of messages and events exchanged among the different components involved in our prototype to enable service discovery and interaction among software components based on different middleware technologies, where we focus on the interaction between the Personal Remote Control, the Digital Media Server (DMS) and the Amigo interoperable middleware core (the SDI comprising SDP Detection & Interoperability, and the SII).



**Figure 3:** Service discovery and interaction interoperability integrated into the prototype

Interoperability is specifically achieved through the following steps:

1. The client application issues an SLP search that is handled by the SDI component.
2. The SLP request message is translated into UPnP events by the SDI component.
3. The SDI component issues a UPnP search .
4. The UPnP device matching the search request replies with a message containing the URI of the document describing the device and all the services supported by the device. The message is received by the SDI component.
5. The SDI components sends a UPnP message to request the device description.
6. The SDI component receives the UPnP description (XML document) of the device.

7. For each service supported by the device, the SDI component sends a UPnP message to get the XML description of the service.
8. The SDI components receives the UPnP description of the services.
9. Data obtained from Steps 4, 6 and 8 are translated into SLP events.
10. The Proxy Generator dynamically creates the RMI proxy of the UPnP remote service that has been discovered, as the client is based on the RMI technology<sup>6</sup>.
11. Once the proxy is generated, it is registered by the Proxy Generator on the RMI Registry.
12. The address bound to the proxy in Step 11 (*rmi://ip:port/proxy*) is notified to the SDI component.
13. The SDI component embeds the service address in an SLP reply, which it sends to the client.
14. The client receives the service address (*rmi://ip:port/proxy*). Since the client expects the remote service to be an RMI service, it looks up on the RMI Registry at the address received in the SLP reply.
15. The RMI Registry on the Amigo interoperable middleware core returns the service proxy to the client application.
16. The client finally invokes a method on the remote service using the proxy. The message sent over the network and the reply from the service are SOAP messages and the proxy is in charge of the translation from SOAP to Java for the client.

The above process enabling service discovery and interaction interoperability, and detailed for the special case of DMR, is the same for access to any other UPnP device integrated in our prototype. The various steps of the process will further be highlighted in the demonstration, using a viewer application (see Figure 4), which is written in the Python language<sup>7</sup> and utilizes a gaming library called PyGame<sup>8</sup> for displaying and moving images.

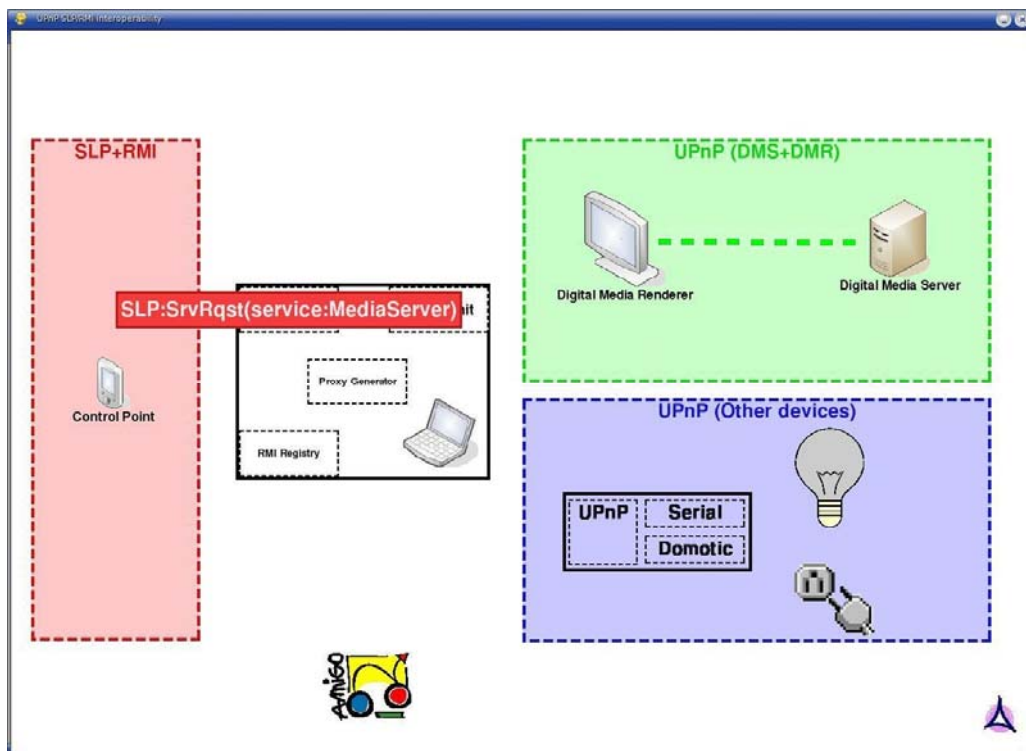


Figure 4: Viewing Amigo interoperability

## 5. Conclusion

Our demonstrator primarily builds upon the Amigo middleware, which allows integrating services based on heterogeneous middleware technologies, in the networked home environment. Such a feature is enabled by the Amigo interoperable middleware core that implements technology-independent middleware-layer interoperability methods so that services of the networked home environment may be discovered and accessed by the other networked services, and conversely, independent of the service-oriented middleware technology the various networked services are implemented upon.

<sup>6</sup> <http://java.sun.com/products/jdk/rmi/>

<sup>7</sup> <http://www.python.org>

<sup>8</sup> <http://www.pygame.org>