Medical Informatics in a United and Healthy Europe
K.-P. Adlassnig et al. (Eds.)
IOS Press, 2009
© 2009 European Federation for Medical Informatics. All rights reserved.
doi:10.3233/978-1-60750-044-5-366

Interoperability Services in the MPOWER Ambient Assisted Living Platform

Marius MIKALSEN [a,1], Sten HANKE [b], Thomas FUXREITER [b],
Ståle WALDERHAUG [b,c], Leendert WIENHOFEN [a]
[a] *SINTEF ICT, Trondheim, Norway*
[b] *Austrian Research Centers GmbH – ARC, Vienna, Austria*
[c] *Department of Computer Science, University of Tromsø, Norway*

Abstract. Ambient Assisted Living systems for the ageing and cognitively disabled do not exist in isolation. What characterizes such systems is the cooperation of several different stakeholders in the care process and the service platforms need to address this. This paper reports on our work in the EU IST MPOWER project where we have designed and implemented interoperability services based on patterns, service-oriented architectures, web services and XSDL transformations. The services we present are freely available as open source under the MIT license.

Keywords. information systems, systems integration, ambient assisted living, home care, continuity of care

1. Introduction

During the requirement process in the EU IST project MPOWER[2], we found the need for interoperability services particularly applicable to the field of ambient assisted living and e-inclusion [1, 2]. This is a domain where the goal is to enable elderly to live independent and active longer, and the interoperability services differ from traditional B2B services, or even inter-EHR system interoperability. We found three categories of Ambient Assisted Living (AAL) interoperability services; 1) notification and alarming services, 2) health services, and 3) voice and video communication services. These services are deemed to be interoperable as they involve other (external) services than MPOWER core services.

Our solution is based on *i*) SOA and Web Services, *ii*) Interoperability Patterns, and *iii*) Message transformations using Extensible Stylesheet Language Transformations (XSLT). This paper introduces the methodological foundations for the implementation of these services, explains the service implementations, and discusses our experiences in implementing and using them.

[1] Corresponding Author: Marius Mikalsen, SINTEF ICT, S.P. Andersens vei 15B, Trondheim, NO-7465; E-mail: marius.mikalsen@sintef.no.
[2] MPOWER project, http://www.mpower-project.eu (IST, Grant Agreement Number 034707).

2. Methods

Our approach to SOA is based on Web Services. We chose web services as they "… provide a standard means of interoperating between different software applications, running on a variety of platforms and/or frameworks" [3]. This means that any AAL application (be that .NET or Java) can communicate with MPOWER services using HTTP and SOAP. MPOWER service interfaces are defined using WSDLs. Likewise, MPOWER services use SOAP messages internally, and can easily exchange an internal service with another external web service that are found more suitable. This provides developers with choice and flexibility. MPOWER further implemented an UDDI service and a message bus to achieve the fundamental SOA concept of loose coupling (see Figure 1).

Although the interoperability services in MPOWER differ from traditional services, we applied software patterns to find general and reusable interoperability designs. "Patterns describe common ways of doing things and are collected by people who spot repeating themes in design" [4]. Pattern resources are [5–9]. Hohpe and Woolf provide the message translator pattern that we applied in the Google health service (the service is described below) [4]. The pattern works by putting a special filter between applications, to translate one data format into another. The translator pattern contains sub-patterns that solve different things with regard to transformation. These are the envelope wrapper, the content enricher, the content filter, and the normaliser pattern.

We applied the XSLT to implement the Message Translator pattern. XSLT is an XML-based language used for the transformation of XML documents into other XML or "human-readable" documents. This makes it very well suited for use with web services and SOAP messages.

3. Results

The services presented in this section are available as open source under the MIT license. They can be downloaded from http://www.mpower-project.eu.

3.1. Notification, External Notification and Alarming Services

This is a group of services which offers functionality to send messages internally in an MPOWER installation as well as SMSs and emails from MPOWER to external systems and users by using external and public email and SMS gateways. We have grouped notifications together with alarms as an alarm typically is sent through the notification and external notification mechanisms.

The Notification service provides functionality for applications registering and un-registering for notifications. This subscription is stored on the server available to the MPOWER service bus (see Figure 1). The notification service has methods for registering and un-registering, send notifications to subscribers, send alarms and resend alarms when they changed status.

The ExternalNotification service has methods for sending SMS messages and for sending emails. Internally the MPOWER SMS service uses "pics" (http://www.pics.co.at) as SMS gateway. The email service uses a server from one of the MPOWER partners but can easily be configured to any other email server. The notification services are implemented as a BPEL working with the message bus.

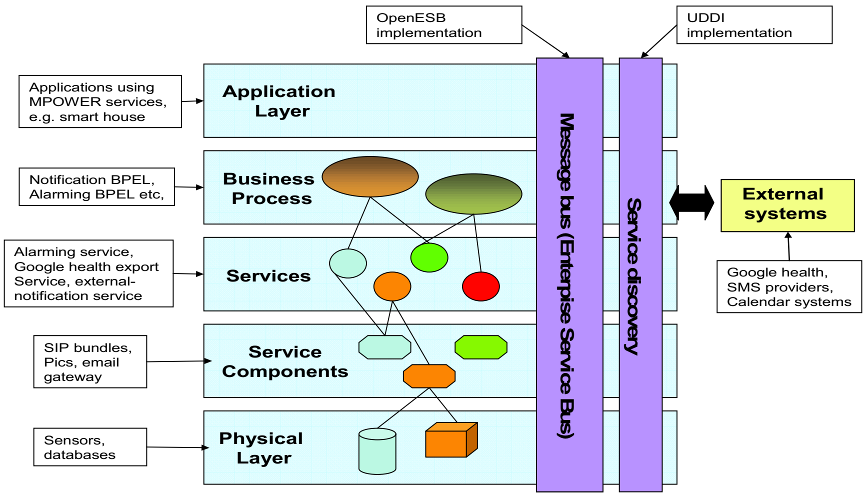
OpenESB implementation
UDDI implementation
Applications using MPOWER services, e.g. smart house
Application Layer
Notification BPEL, Alarming BPEL etc,
Business Process
Alarming service, Google health export Service, external-notification service
Services
SIP bundles, Pics, email gateway
Service Components
Sensors, databases
Physical Layer
Message bus (Enterpise Service Bus)
Service discovery
External systems
Google health, SMS providers, Calendar systems

Figure 1. MPOWER SOA architecture and services (notification, health and communication)

3.2. Health Services

MPOWER Google health export: The export to Google Health is realized through the MedicationPlanSynchronizer serice. This web service offers two methods. The first exports the MPOWER internal data structure to the Continuity of Care Record (CCR) used by Google. The second display the CCR as pure xml format or html format (for debugging purposes). The service is designed so that message transformations to European standards such as UNE 13606 [10] are enabled through the provision of an appropriate XSLT.

MPOWER calendar export: In the AAL domain, one needs to coordinate the scheduling of appointments for healthcare services and/or for the use of resource slots between stakeholders. The service can also be used for scheduling appointments with relatives, friends and other members of patient social network as well as for any kind of activity including social activities or medication activities. This is done using the iCal format, which is applicable across many calendar systems (see Figure 2).

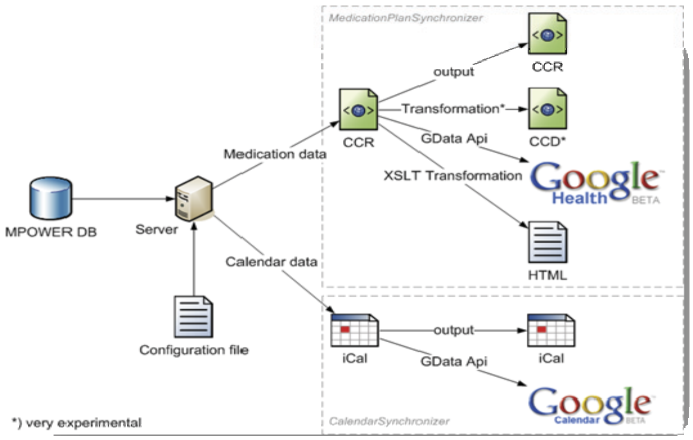

Figure 2. MPOWER Google export

3.3. Voice and Video Communication Services

The MPOWER Communicator (voice and video communication) services provide the possibility to call other users (audio live stream) as well as to see them (IP-camera live stream). To accomplish this, we have implemented the Session Initiation Protocol (SIP) which is a signaling protocol widely used for setting up and tearing down multimedia communication sessions over the internet. Several SIP applications exist today, some of them for free and/or open source, but they are typically tightly coupled to a graphical user interface. To this end, we have stripped down SIP services so that can be used with particular AAL domain user interface needs (see Figure 3).

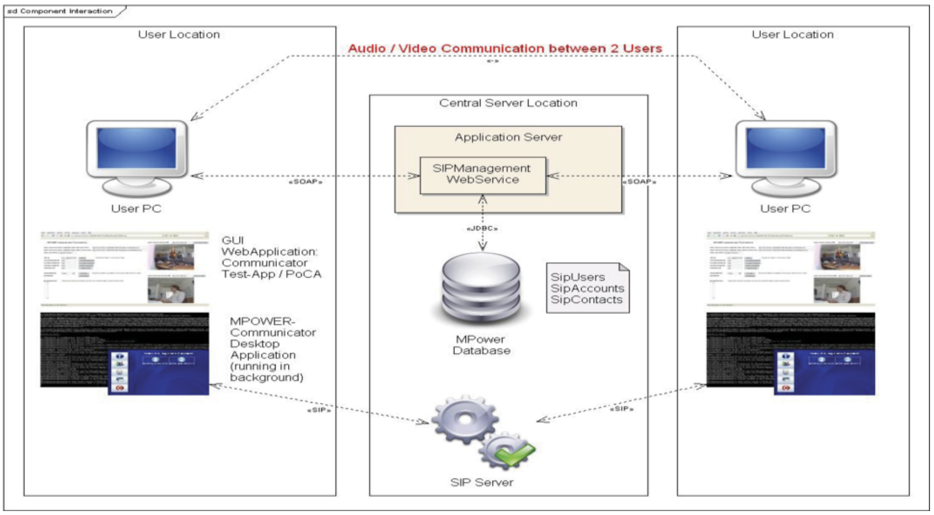
sd Component Interaction
User Location
Audio / Video Communication between 2 Users
User Location
Central Server Location
Application Server
SIPManagement WebService
«SOAP»
«SOAP»
User PC
User PC
«JDBC»
GUI WebApplication: Communicator Test-App / PoCA
SipUsers SipAccounts SipContacts
MPower Database
MPOWER-Communicator Desktop Application (running in background)
«SIP»
«SIP»
SIP Server

Figure 3. MPOWER communication services

4. Discussion

The notification and alarming services is typically top priority in the field of AAL as safety is paramount and care givers want to be alarmed in cases of irregularities (e.g., fire). We used the OpenESB implementation to achieve as loose coupling as possible between the components. We found it challenging to achieve sufficient level of quality of service using the currently available OpenESB implementation on the Glassfish application server. This combined with no guarantee of IP QoS is a drawback for web services use in the AAL domain. Commercially available service buses and application server implementations could prove more stable.

Originally we set out to show MPOWER interoperability with a Norwegian core journal project that based on a Norwegian version of CEN standards. This system did not become available in the projects lifetime. Therefore, to demonstrate interoperability features of the MPOWER export services, we shared the MPOWER data with the Google health system over an available standardized interface. Google Health uses the CCR standard, but given the XSLT conversion capability, it is a matter of providing the coding rules. Such functionality is available on COTS products like Microsoft BizTalk Mapper (commercial) or JSimpleX Visual XSL (open source).

A challenge we discovered was to maintain a health record synchronized with Google Health as we had no common identifiers across MPOWER and Google. This is a common problem and is amongst others addressed by the Healthcare Service Specification Project in their record location and update service (RLUS) which is becoming an OMG standard.

5. Conclusion and Future Work

The services we have developed should be seen as a start for interoperable services within ambient assisted living in Europe. Following from this beginning we would suggest comparing the performance between commercially available platforms for application servers and for enterprise service buses. Another point to investigate is how well the services in the MPOWER framework harmonizes with needs found in other AAL projects and at what points the interoperability mechanism need to be extended or changed. Finally, important work remains to be done on the effects of interoperable services. We need to investigate how AAL interoperability services influences aspects such as increased independence, more effective care, cost savings and increased safety in Europe.

References

[1] Walderhaug, S. et al. (2008) Reusing models of actors and services in smart homecare to improve sustainability. *Studies in Health Technology and Informatics* 136:107–112.

[2] Walderhaug, S., Stav, E., Mikalsen, M. (2008) Experiences from model-driven development of homecare services: UML profiles and domain models. In Chaudron, M.R.V. (Ed.) *Models in Software Engineering, Workshops and Symposia at MODELS 2008, Toulouse, France. Reports and Revised Selected Papers*, Springer, 199–212.

[3] World Wide Web Consortium (W3C), *Web Services Architecture,* 2004, http://www.w3.org/TR/ws-arch/wsa.pdf.

[4] Hohpe, G., Woolf, B. (2004) *Enterprise Integration Patterns*. Vol. 1, Addison-Wesley, Boston.

[5] Fowler, M. (2004) *UML Distilled – A Brief Guide to the Standard Object Modeling Language*. Vol. 3, Third edition, Addison-Wesley, Boston.

[6] Gamma, E., Helm, R., Johnson, R., Vlissides, J. (1995) *Design Patterns: Elements of Reusable Object-Oriented Software.* Addison-Wesley, Reading, MA.

[7] Buschmann, F., Meunier, R., Rohnert, H., Sommerlad, P., Stal, M. (1996) *Pattern Oriented Software Architectures: A System of Patterns*. John Wiley & Sons, Chichester.

[8] Schmidt, D., Stal, M., Rohnert, H., Buschmann, F. (2000) *Pattern Oriented Software Architecture. Volume 2: Patterns for Concurrent and Networked Objects*. John Wiley & Sons, Chichester.

[9] Fowler, M. (2002) *Patterns of Enterprise Application Architecture.* Addison-Wesley, Boston.

[10] ISO 13606-1:2008, http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=40784.