BANip: enabling remote healthcare monitoring with Body Area Networks[1]

Nikolay Dokovsky, Aart van Halteren, Ing Widya

University of Twente
PO Box 217
7500 AE Enschede, The Netherlands
{dokovsky,halteren,widya}@cs.utwente.nl

Abstract – This paper presents a Java service platform for mobile healthcare that enables remote health monitoring using 2.5/3G public wireless networks. The platform complies with today's healthcare delivery models, in particular it incorporates some functionality of a healthcare call center, a healthportal for patients that aggregates healthcare services from collaborative care centers. Our service platform simplifies the development of mobile healthcare applications, because application developers are shielded from the complexity of communicating with mobile devices over 2.5/3G networks. In this paper we focus on the BAN interconnect protocol (BANip) which is our solution to extend the local services offered by a Body Area Network (BAN) to remote healthcare center.

Introduction

Information and Communication Technology (ICT) plays an important role in cost reduction and efficiency improvement in healthcare delivery processes. ICT in emergency services for example enables trauma teams at hospitals to diagnose remotely injured patients at accident scenes and therefore improves the time to treatment [1]. The introduction of 2.5/3G wireless technology even takes healthcare further to mobile healthcare (m-health) which enables ambulatory patients to lead a normal daily life. For example, the pregnant women m-health trial in [2] investigates mobile and remote monitoring of the maternal and fetal vital signs whilst the pregnant women can proceed with their daily life activities rather than be hospitalised for monitoring. ICT also plays an important role in the introduction of new healthcare stakeholders in healthcare delivery such as healthcare call centres, healthportals which provide health services around the clock to patients located everywhere [3]. These healthcare call centres typically collaborate closely with secondary care centres, e.g. hospitals, laboratories and clinics. In the delivery of healthcare, they may be viewed as intermediaries between patients (e.g. at home) and healthcare practitioners (e.g. at hospitals).

[1] This work has been sponsored by the European Commission within the IST project MobiHealth (IST-2001-36006) [2].

This paper presents a Java service platform for m-health. The platform can accommodate call centres, which may reside in hospitals as a front-office or are located remotely from a secondary care centre. In the latter case, call centres act as an added value stakeholder in healthcare delivery that aggregates healthcare services from collaborative care centres. In this context, the platform provides end-to-end m-health services to the end-users (i.e. patients and healthcare practitioners) using health intermediaries which are transparent to these users. In particular, the implemented platform provides remote monitoring services of measured vital signs of patients (e.g. ECGs, oxygen saturation in patients blood) in a store-and-forward as well as in a (near) real-time mode [4].

Besides the healthcare delivery stakeholders, the platform spans over the domains of several connectivity stakeholders, in particular ISPs and 2.5/3G wireless network operators. The development of end-to-end healthcare services spanning over multi stakeholders' domains is also a challenge addressed in this paper.

Moreover, the embedded physiological monitoring system in the platform is based on wearable Body Area Networks (BANs) [5], [6], which nodes are vital sign sensors and a gateway to communicate with nodes in the administrative domains of the healthcare stakeholders using 2.5/3G wireless infrastructures.

In this paper, we present the platform design using Java, Jini and Jini Surrogate technologies. We address how we apply these technologies to provide end-to-end m-health services in a wireless environment with intermediaries, while these intermediaries are in a way transparent for the end-users. We also present how these technologies have been used to discover, authenticate and register BANs which can be switched on anytime and anywhere in a 2.5/3G wireless network coverage area.

MobiHealth Service Platform

The MobiHealth service platform enables remote monitoring of patients using 2.5/3G public wireless infrastructures. Patient data is collected using a Body Area Network (BAN). A healthcare practitioner can view and analyse the patient data from a remote location. In this setting, the BAN acts as a provider of patient data and the healthcare practitioner acts as a user of that data.

This section presents the BAN, the service platform and identifies the technical challenges that have been addressed during its development.

Body Area Network

The healthcare BAN consists of sensors, actuators, communication and processing facilities. Depending on what type of patient data must be collected, different medical sensors can be integrated into the BAN. For example, to measure pulse rate and oxygen saturation, an oximeter sensor is attached to the patients' finger. In the case of an ECG measurement, electrodes are attached on the patient's arms and chest.

Communication between entities within a BAN is called intra-BAN communication. Our current prototype uses Bluetooth for intra-BAN communication. To use the BAN for remote monitoring external communication is required which is

called extra-BAN communication. The gateway that facilitates extra-BAN communication is called the Mobile Base Unit (MBU). Our current prototype employs an HP iPAQ H3870, that runs Familiar Linux and a J2ME [7] compliant Java virtual machine, as an MBU. Other mobile devices, such as a SmartPhone or a J2ME enabled PDA could also act as an MBU.

Fig. 1 shows the architecture of a BAN. Sensors and actuators establish an ad-hoc network and use the MBU to communicate outside the BAN.

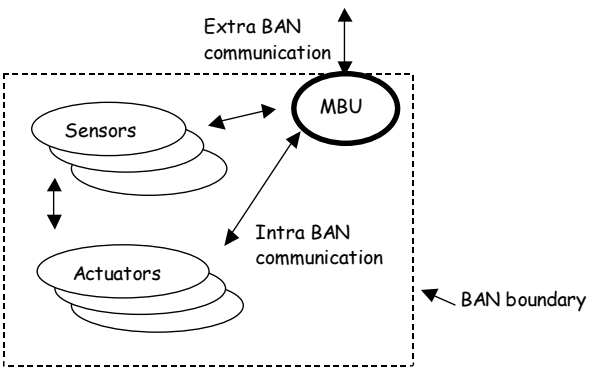

Fig. 1. BAN architecture

Platform Functional Architecture

Fig. 2 shows the functional architecture of the service platform. The dotted square boxes indicate the physical location where parts of the service platform will be executing. The rounded boxes represent the functional entities of the architecture, while the black oval shaped dots represent interaction points between the functional entities. The M-health service platform consists of sensor and actuator services, intra-BAN and extra-BAN communication providers and an M-health service layer. The intra-BAN and extra-BAN communication providers represent the communication services offered by intra-BAN communication networks (e.g. Bluetooth) and extra-BAN communication networks (e.g. UMTS), respectively. The M-health service layer integrates and adds value to the intra-BAN and extra-BAN communication providers. The M-health service layer masks applications from specific characteristics of the underlying communication providers.

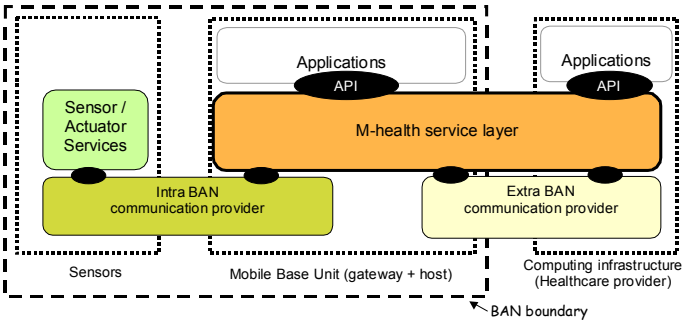

Fig. 2. Service platform functional architecture

The service platform supports two types of applications. The first type are targeted at direct interaction with m-health service users and can range from simple viewer applications that provide a graphical display of the BAN data, to complicated applications that analyze the data. They can be deployed on the MBU (for on-site use e.g. by a visiting nurse) or on the servers or workstations of the healthcare provider (for use e.g. by a healthcare practitioner). The second type of applications add functionality to the core m-health service, such as permanent data storage or on-demand real-time data streaming. These applications are deployed at the healthcare servers providing offline or online (with respect to the MBU status) data processing.

The API interaction points in Fig. 2 represent the two possible locations where application components can interact with the m-health service platform.

Technical Challenges

In this paper, we focus on the platform parts related to the extra-BAN communications. The description of the MobiHealth service platform in the previous section exposes the following technical issues which we address in the next sections. In particular these issues are imposed by the characteristics of m-health delivery, e.g. ambulatory patients, privacy sensitive and trustability of medical data and the multi stakeholder involvement in today's healthcare delivery

- How does the platform discover and register BANs of ambulatory end-users?

A BAN and in particular the MBU will typically be switched on/off anytime at the convenience or the necessity of the end-user (e.g. patient) located anywhere in the coverage area of a 2.5/3G wireless network. A switch-off needs to be identified by the platform to release the service and its resources. On the other hand, the platform needs to discover a BAN which is switched on to enable a peer end-user at a healthcare centre to find the BAN data. This process of mobile BAN discovery is further complicated by the policy of the 2.5/3G wireless network operators, which typically assign an IP address to a terminal (i.e. MBU) dynamically from the private address space [8] at the establishment of the wireless connection (e.g. GPRS).

Therefore, a BAN/MBU discovery and a release mechanism has to be developed for the MobiHealth service platform.

- How to deal with the limitation of the resources of BANs used for monitoring services?

In the MobiHealth service platform, the supplier of data for remote monitoring is the BAN, which should be wireless and wearable. Traditionally, providers of data (such as web servers) are deployed on a computing infrastructure with sufficient network and processing capacity. Consumers of data (such as web browsers) assume that providers are available most of the time (except for maintenance) and with sufficient bandwidth to serve a reasonable amount of consumers. For the service platform, the producer and consumer roles are inverted because the provider of data is deployed on a mobile device (i.e. the MBU) while the consumer of data is deployed on a fixed host with sufficient processing and communication capacity. It is a technical challenge to deal with this inverted-producer-consumer problem in the design of the platform.

- How to provide reliable and secure end-to-end healthcare services in multi healthcare delivery stakeholders domains?

It is generally required that for privacy reasons medical data of patients has to be transferred in a secure way. The transfer of the data needs also to be reliable to enable correct interpretation by healthcare practitioners. It is a technical challenge to develop an architecture and the associated mechanisms for an m-health service platform involving multi healthcare delivery stakeholders that highlight the added value of the involved stakeholders. For example, the MobiHealth call centre which addresses the reliability and security issues of m-health delivery in such a way that it releases other stakeholders from the burden of addressing these issues in their full complexity themselves. The scarce bandwidth resources of GPRS offered today by the operators and their policy to prioritize voice above data complicate the data transfer mechanisms further. Application protocols designed for use in local area networks, such as RMI and IIOP, are in this context not applicable for the GPRS links.

- How to provide a scalable end-to-end m-health service?

The Mobihealth service platform must support a many-to-many relation between the BAN and the MobiHealth secondary care centre. Potentially more than 100.000 simultaneously operating BANs should be supported. The medical data produced by those BANs could be of interest to various healthcare practitioners. Therefore the BAN must be able to produce medical data from as many locations as possible (thus supporting full patient mobility) and the medical data must be accessible from as many as healthcare centres. In summary [9],

- the service platform must scale to a size that it can support 100.000+ BANs (numerical scalability)
- the platform must support multiple (secondary) healthcare centres (i.e. multiple organizational domains) (organizational scalability)
- the platform must scale to a large geographical area (i.e. European or world scale) (geographical scalability).

JINI and MobiHealth

The internals of the MobiHealth service platform depend on JINI technology [10], [11]. This section summarizes the key features of JINI and identifies how these features contribute to the services offered by our platform.

JINI

One of the problems that the MobiHealth service platform solves is the dynamic discovery of the services offered by a BAN. A BAN is configured to the needs of end-users and can typically be switched on/off anytime at the convenience or the necessity of the end-user (e.g. patient). A key design decision is to represent the services offered by a BAN as a JINI service. JINI provides mechanisms whereby the service lifetime, service location and service implementation details become irrelevant to the service user.

A JINI service provider can dynamically attach or detach itself from a given JINI network community, a so-called djinn. A special core service, called LookUp Service (LUS), supports the registration of services. A service provider (e.g. a BAN) registers a Service Proxy into LUS together with a set of predefined service attributes. Before a service provider can register a service it must contact the LUS, which is the discovery process (i.e. interaction 1 and 3 in Fig. 3). The figure shows the interactions between the service provider, service user and the LUS, the first 4 interactions belong to the set-up phase and are preliminary to the service execution phase, i.e. the data transfer phase in monitoring applications. An essential element of the discovery and registration process is the exchange of a ServiceProxy. This proxy encapsulates the client part of the service logic and communication protocols needed for service execution.

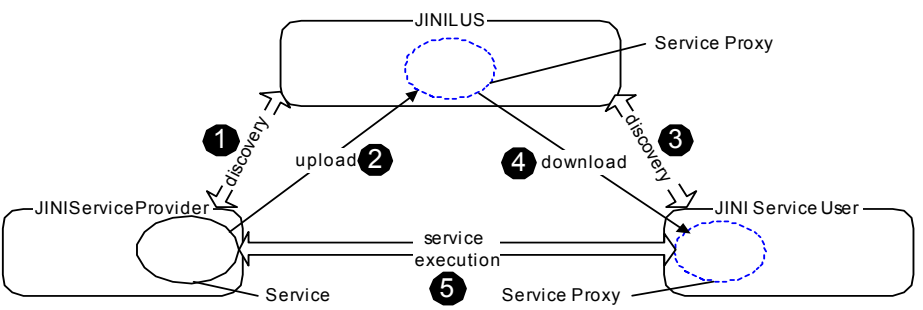

Fig. 3. Service discovery and registration

A service user only needs to know the interface to the service. In some cases the service proxy includes the whole service implementation, moving the service execution entirely into the service users virtual machine (VM). In other cases the service proxy contains networking functionality, which will allow the service user to connect to the remote service provider. The Mobihealth platform uses the latter

scheme, which allows for adaptation of the protocol used for service execution, without changes in the service user (e.g. the peer client of the monitoring system at a healthcare center).

When a service user discovers a LUS, it can either browse it for a particular service or it can register itself for service availability notifications based on event messages.

Leasing

Service resources in a JINI network are acquired for a certain period based on a service lease. A service user must renew a lease within certain time period. If that lease cannot be renewed, for whatever reason (network failure, system crash, etc.), the client can conclude that the service is not available anymore. Similarly, the service provider can free specific resources just because a service user has not renewed a granted lease.

JINI Surrogate Architecture

Although service execution can be based on any protocol, JINI requires that RMI is available for LUS discovery. Consequently a service provider must execute in a VM that supports RMI. Limited resources inhibit the use of RMI on the MBU, therefore making the MBU unfit as a host for a JINI service provider. We employ the JINI Surrogate Architecture [12], [13] (Fig. 4) to enable a BAN to still offer services in a JINI network.

According to the Surrogate Architecture, a device which initially cannot join a JINI network because it does not meet the connectivity and functional requirements, can still join if it is represented by a surrogate object. The surrogate object acts as a service provider on behalf of the device and shields service users from the specific means to communicate with the device. At application level, this yields that a healthcare practitioner at a healthcare centre transparently retrieve BAN data from the surrogate host as if he retrieves the data from the BAN. In a multi stakeholder m-health delivery model, surrogate hosts typically reside at the domains of the call centers.

Furthermore, surrogates rely on a Surrogate Host for life cycle management and a runtime environment. Device specific communication between the surrogate and the device is called the Interconnect Protocol.

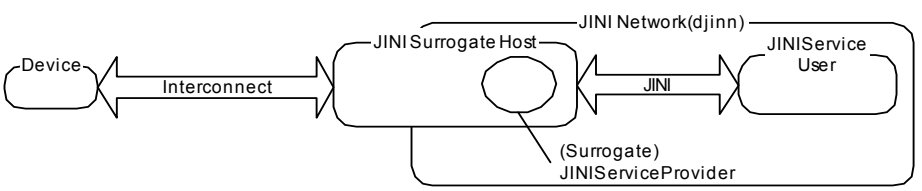

Fig. 4. Elements of Surrogate Architecture

The surrogate architecture specification requires the interconnect protocol to fulfill at least three mechanisms: discovery, surrogate upload and keep-alive.

The purpose of the discovery mechanism is to make a surrogate host aware of the device existence and vice versa. Implementation depends on the device communication capabilities.

Once a device and the surrogate host discovered each other, the device can join the JINI network. To do so, the surrogate host must be provided with the surrogate object that will act in the JINI network on the behalf of the device. The device itself can upload the surrogate object or it can send to the surrogate host the location point from where the surrogate can be downloaded.

After a surrogate has been instantiated and activated by the surrogate host, the device must guarantee that it is able to perform the exported service. Consequently, the interconnect protocol must implement a keep-alive mechanism to inform the surrogate host if the device is still active and connected. As soon as the device cannot confirm its online status, the surrogate host can deactivate the JINI service and its correspondent surrogate object.

Design and Implementation of the BANip

The MobiHealth service platform uses a surrogate object to act on behalf of the MBU and thus allows a BAN to offer its services in a JINI network despite its resource limitations. The BAN interconnect protocol (BANip) is our protocol for interaction between the MBU surrogate and the MBU device. The BANip fulfils the three basic requirements of the surrogate architecture (i.e. discovery, surrogate upload and keep-alive).

Internal Structure

Fig. 5 shows a refined view of the M-health service layer in the data transfer phase. The arrows in the figure show the flow of the BAN data. The BANip entity is a protocol entity for the BAN interconnect protocol. Peer entities can be found on the MBU and near the surrogate component. The BANip entities communicate through a proxy, that has authenticated and authorized the BAN in the set-up phase. The surrogate component acts as a regular JINI service provider which in our case uses Remote Method Invocation (RMI) for service execution.

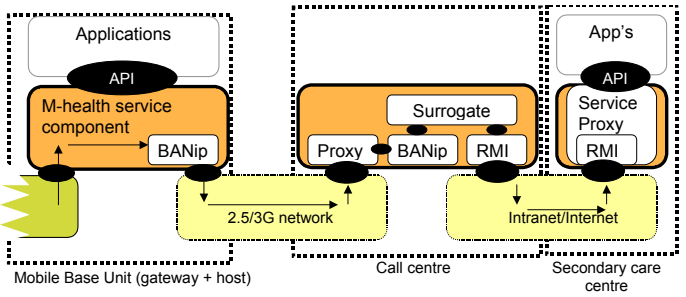

Fig. 5. Refined view of the service platform in de data transfer phase

Applications, e.g., situated in a secondary care center, requiring access to the MBU services download a service proxy in the set-up phase and use that proxy to transparently communicate with the surrogate in the data transfer phase.

Interconnect Protocol Design

The BANip defines a set of messages plus a specific programming model, following closely, in that respect, the implementation of the IP interconnect [14].

Since the MBU device can only support a Java 2 ME virtual machine (CDC or CLDC) and since the communication is over wireless networks, we chose to encapsulate the BANip into HTTP. HTTP client support is mandatory for all J2ME (both CDC and CLDC configurations) virtual machines, which makes our platform portable across various mobile devices. An additional benefit of using HTTP is that we can use standard HTTP proxy mechanisms for BAN authorization and we can use HTTPS for secure delivery of BAN data.

As identified earlier, the MBU obtains its IP address from a telecom operator network therefore it is often not possible to initiate a connection from the Internet to the MBU. This problem is solved when the communication between the MBU device and the MBU surrogate is encapsulated in HTTP requests that are initiated from the MBU device.

The BANip has three message categories: surrogate lifecycle messages, private interconnect messages and externally initiated messages. Fig. 6 shows an example for each of these message categories, labeled 1,2 and 3 respectively.

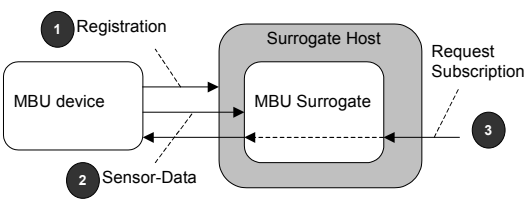

Fig. 6. Examples of BANip messages

The `Registration` message is a surrogate lifecycle message that an MBU device sends to the surrogate host. The message includes initialization data for the surrogate and a URL that points to the location of the JAR file that contains the MBU surrogate implementation. This is sufficient information for the surrogate host to obtain and instantiate the appropriate surrogate object.

Once activated the surrogate host expects at regular intervals a "keep-alive" message, which is another surrogate lifecycle message. If such is not received, the surrogate can be deactivated and its JINI service offer removed from the LUS.

The `Sensor-Data` message is a private interconnect message, pushed from the MBU to the surrogate. The purpose of this message is to deliver sensor data to the surrogate object. We chose for a push mechanism as sensor data arrives at the MBU at regular intervals and we want to cache this data at the surrogate.

The `RequestSubscription` message is an externally initiated message, since it is initiated at the service proxy and propagates through the surrogate to the MBU device. However, the surrogate has no means to actively post HTTP requests to the MBU device because this device cannot act as an HTTP server. Therefore, we encapsulate the `RequestSubscription` message into the HTTP response of the keep-alive messages that arrive regularly at the surrogate.

Because the low bandwidth and high latency are very common characteristic of a GPRS link, we focused our attention in eliminating the overhead of the HTTP interconnect as much as possible. For example, instead of sending each BAN measurement in a single HTTP post request, a number of "sensor data" messages is first collected and then sent together. Additional improvement was achieved with HTTP chunking, where BANip messages are conveyed as chunks of one long-term HTTP request. The third optimisation is the implementation of a "deflate" compression algorithm.

Conclusions

The MobiHealth service platform resolves a number of technical challenges when communicating with mobile devices over a 2.5/3G public wireless infrastructure.

A key design decision for the platform is to represent the services offered by a BAN as a JINI service. Consequently, the discovery and registration problem of BANs has been solved. We solved the inverted producer-consumer problem using the JINI Surrogate Architecture and have implemented an HTTP based interconnect protocol called BANip. Our platform simplifies the development of remote healthcare applications, deployed in a call-center or in a secondary care center. The platform shields these organizations from the complexity of securely communicating with wireless mobile devices over 2.5/3G networks. We apply webserver technology for our BANip that can be extended with well-know techniques (e.g. load-balancing) to support at least 100.000 BANs. Our choice for HTTP provides improved flexibility for example for porting the BANip to other mobile devices. For this reason we believe our platform to be scalable in the numerical, organizational and geographical dimensions.

The platform will be deployed in nine healthcare field trials in four European countries to evaluate its functionality and to assess the feasibility of 2.5/3G wireless technologies in combination with Java and Internet technologies for m-health service delivery.

Acknowedgement

The authors would like to thank George Koprinkov for his diligent contribution to the implementation of the BANip and the MobiHealth service platform.

References

1. Gagliano, D.M. and Xiao, Y., "Mobile Telemedicine Testbed" in Proc. 1997 American Medical Informatics association (AMIA) Annual Fall Symposium, pp.383- 387, National Library of Medicine Project N0-1-LM-6-3541, 1997;
2. MobiHealth, "MobiHealth project IST-2001-36006", EC programme IST, 2002, http://www.mobihealth.org;
3. N. Maglaveras, et al, "Home care delivery through the mobile telecommunications platform: the Citizen Health System (CHS) perspective", International Journal of Medical Informatics 68 (2002), pp. 99-111.
4. Hung, K. and Zhang, Y-T., "Implementation of a WAP-Based Telemedicine System for Patient Monitoring", in IEEE transactions on Information Technology in Biomedicine, vol. 7, no. 2, June 2003, pp. 101 – 107;
5. Val Jones, Richard Bults, Dimitri Konstantas, Pieter AM Vierhout, Healthcare PANs: Personal Area Networks for trauma care and home care, Proceedings Fourth International Symposium on Wireless Personal Multimedia Communications (WPMC), Sept. 9-12, 2001, Aalborg, Denmark;
6. E. Jovanov, et al, "Stress Monitoring Using a Distributed Wireless Intelligent Sensor System", in IEEE Engineering in Medicine and Biology Magazine, May/June 2003
7. Sun Microsystems, "CDC: An Application Framework for Personal Mobile Devices", June 2003, http://java.sun.com/j2me
8. Y. Rekhter et al., "Address Allocation for Private Internets", RFC 1918, February 1996.
9. B. Clifford Neuman, Scale in Distributed Systems, Readings in Distributed Computing Systems, IEEE Computer Society Press, ISBN 0818630329, 1994
10. Sun Microsystems, "Jini Architecture Specification", http://www.sun.com/software/jini/specs/jini1_2.pdf
11. Sun Microsystems, "Jini Technology Core Platform Specification", http://www.sun.com/software/jini/specs/core1_2.pdf
12. Sun Microsystems, "Jini Technology Surrogate Architecture Specification", July 2001, http://surrogate.jini.org/sa.pdf
13. Sun Microsystems, "Jini Technology Surrogate Architecture Overview", http://surrogate.jini.org/overview.pdf
14. Sun Microsystems, "Jini Technology IP Interconnect Specification", http://ipsurrogate.jini.org/sa-ip.pdf