A framework for the development of ubiquitous patient support systems

Jelena Mirkovic
Center for Shared Decision Making and Collaborative Care Research, University Hospital
Oslo, Norway
jelena.mirkovic@medisin.uio.no

Haakon Bryhni
Department of Informatics
University of Oslo
Oslo, Norway
haakon@bryhni.com

Cornelia M. Ruland
Center for Shared Decision Making and Collaborative Care Research, University Hospital
Oslo, Norway
cornelia.ruland@rr-research.no

Abstract —**Ubiquitous healthcare systems can provide advantages to patients, enabling them to access medical information and support systems independent of their current place and time. However, due to specific requirements regarding security and usability standard mechanisms for enabling terminal and application mobility are not acceptable for healthcare information systems. We propose a service architecture framework (the CONNECT framework) that enables content adaptation and session management for ubiquitous patient support systems and addresses requirements specific for healthcare systems. The CONNECT framework provides support for terminal and application mobility, and enables easier implementation, maintenance and adaptation of patient support systems for different types of terminals, networks and services. Additionally, it addresses security and usability requirements that are of high importance for healthcare systems. Based on the proposed framework the testing environment is implemented; and as a result, we conclude that the system's scalability and performance is not significantly affected.**

Keywords- mobile devicse; healthcare system; mobility; session management; security; usability; content adaptation.

I. INTRODUCTION

New terminals such as smartphones and tablet PCs have enabled ubiquitous information services. Users can access services independent of time and place, and content can be dynamically adjusted to the current context and terminal type. The potential of these services is increasingly recognized in application areas such as learning and business, but ubiquitous and seamless information systems can also be leveraged in healthcare. Mobile devices, home computers and embedded patient terminals can be utilized in healthcare services to provide delivery of information to patients at the point of need. In this way, patients can be equipped with powerful tools and support systems that can help them in their everyday health management, and patients can get more involved in decision-making regarding their own health [1][2]. Problems related to increasing healthcare costs and the higher demand for healthcare personnel and services can be addressed and reduced [3].

However, due to security and usability requirements development of ubiquitous healthcare systems introduces new challenges and unresolved issues. Utilization of standard mobility management mechanisms raises a question of who is the responsible entity in charge for implementing and enforcing security mechanisms during migration of user´s session to new terminal/network, and who ensures that patient privacy is managed in accordance with laws and regulations. Policy for protection of patient information differs between countries, and healthcare systems must adhere to national requirements. In the USA, Health Insurance Portability and Accountability Act (HIPAA) compliance is required, but only general requirements are set. Implementation rules and strength of the required security mechanisms are not defined by HIPPAA [4]. In Norway, a PKI system is mandated and certificates must be used for authentication and encryption [5]. PKI-based system is used for certain public web-based solutions with external authentication devices. For mobile applications the requirements are so high that no services are offered today. In the CONNECT (Care Online: Novel Networks to Enhance Communication and Treatment) project we have developed a prototype of a secure mobile access system managing security issues at the application level [6]. The proposed security architecture addresses all requirements set for healthcare information systems and have been submitted for approval to Norwegian local authorities. Due to various legislation requirements defined in different countries managing security of healthcare information systems have to be tailored to specific characteristics of a system and context of use.

Additionally, for healthcare applications and services usability, user-friendliness, and usefulness of the system are very important due to great spectrum of potential future users and variety of their needs and expectations from system's functionalities. Adaptation of interface elements and user interaction to different contexts of use (e.g., types of devices and their characteristics, OSs, and communication network types) is primary requirement. The way in which mobility issues (e.g., session transfer, handoff between networks) are managed can also greatly influence usability of the system due to additional requirements for user interaction and adaptation of interface for a new context of use.

This work was supported by The Research Council of Norway under
ıtion Sharing

PervasiveHealth 2012, May 21-24, San Diego, United States

DOI 10.4108/icst.pervasivehealth.2012.248594

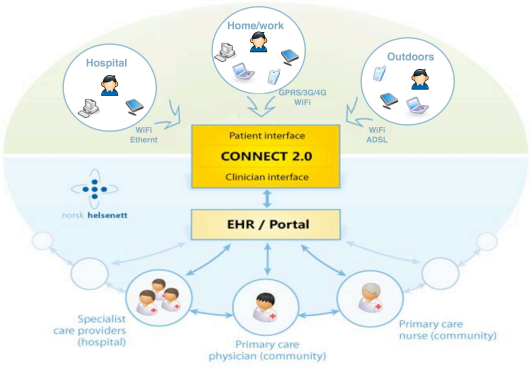

Figure 1. Possible usage scenarios.

The work presented in this paper is part of the CONNECT project, which has as its main goal to develop and test a suite of context-sensitive communication and information tools for accessing a patient-clinician shared Personal Healthcare Record (PHR) [7]. The CONNECT system allows patients to monitor and record their symptoms in their PHR and lets them stay connected with their care providers between clinical encounters. Providing patients mobile and seamless access to the CONNECT system is one of the main requirements in our project. Patients are enabled to receive information at the point of need regardless of space and time, utilizing in this manner the full potential and purpose of developed patient support tools.

In this paper, we present a design and architecture framework for the development of ubiquitous patient support systems with session management and content adaptation functionalities (the CONNECT framework). Through the proposed framework, we describe how mobility, session management, and content management and adaptation can be addressed in mobile healthcare information services. Additionally, we describe how security and usability issues are managed as part of the proposed framework, and how they are adjusted to specific requirements set for healthcare services. We describe a generic framework for developing patient support tools that is not dependent on a specific developer platform, access network or device type.

This paper is structured as follows. In Section 2, we describe ubiquitous patient support system usage scenarios and discuss challenges that must be addressed during development. Section 3 discusses related work. In Section 4, we describe the CONNECT framework and its modules for server and client applications. Section 5 gives a description of an implemented prototype and presents results from testing. Finally, in section 6, we present final conclusions and describe planned future work.

II. UBIQUITOUS PATIENT SUPPORT SYSTEM SCENARIO

For enabling patient mobility and seamless access to patient support systems, there are numerous requirements that must be addressed, such as providing security during session management and migration and providing usability of the application and adaptation of application content to different devices and access networks. The possible usage scenarios for a ubiquitous patient support system are presented in Figure 1. The patients have their user accounts on the patient support system and can access the service from hospitals or their homes, cars, or offices using different types of networks (e.g., Ethernet, WiFi, GPRS, 3G, 4G, and ADSL). The patient support system is usually deployed as a part of a hospital information system and is connected to the EHR system. Using the system, patients can contribute to their everyday health and symptom management. Information submitted by patients can also be made available for primary care physicians, primary care nurses, specialist care providers and other healthcare personnel, when approved by the patient.

Application mobility as defined in [8] enables a user to start interacting with the service using one device and transfers the session to another device automatically when a new communication channel is opened. While the session is transferred, an additional mechanism to warrant session consistency must be provided to protect storage and to avoid transfer of invalid session data. Application mobility must also be provided within an acceptable time frame. Due to the different network types that can be utilized, it is highly possible that communication can be interrupted unexpectedly and that a system must be able to save current session data and enable a user to continue a previously started session without information loss. When the application is resumed on the same terminal over a different network or on a new terminal, a new authentication process must protect security. Because this may happen frequently, it is necessary that the renewed authentication is both user-friendly and secure.

One example of application mobility is described in the following scenario. A patient in a hospital uses a hospital tablet PC to start a session with the patient support system and starts writing a message to a doctor with a question regarding a current health problem. In the meantime, a nurse comes with discharge papers and tells the patient to go home. The patient simply turns off the application, ending communication with the patient support system, and leaves the hospital. Before the communication session ended, the session state was stored on the server so that the patient can seamlessly resume the session. On the way home in a bus, the patient starts the mobile application, chooses the option to resume the session, and the exact state is retrieved, enabling the patient to continue writing the message from where he/she left. Assuming patchy mobile coverage, the session may lose connectivity or the user may be interrupted again, but this is no problem. At home, the patient is given the option to retrieve the pending session from the mobile phone or to use another communication device (e.g., a PC, laptop, or a personal tablet PC) to finish and send the message to the doctor.

To provide this functionality, there are numerous challenges that must be addressed. There are many types of devices and user interfaces that must provide access to a patient support system. Each device and device type has different properties and capabilities (e.g., screen size, input capabilities, operating system, computational power, and type of network interfaces). To enable a user-friendly service, content presented to a user on one terminal should be adjusted to a terminal's characteristics. Additionally, different features should be

presented on different devices, enabling the user to perform actions suitable for the specific terminal and context of use. As a result, not all session states should be transferred to all terminals, thus requiring adaptation of the session state depending on the context of use.

Terminal mobility provides support for accessing the same system over different network types. The communication channel and roaming interruptions must be taken into consideration because multiple devices and networks are supported. As an example, roaming between multiple WLAN Access Points in a hospital is very different from roaming from a WLAN connection in the hospital to a 3G network available when the patient moves.

It is necessary to assure that all communication is protected from unauthorized users as patient data are transferred through different communication networks and that the architecture conforms to health service security requirements. This question is addressed in more detail in [6].

III. Related Work

In the literature, we have found very few projects that address mobility and session management in healthcare services for patients. The projects addressing these issues usually describe systems and services accessible to healthcare providers and not to patients.

Project Ubidoctor [9] presents a middleware-based service infrastructure for the support of ubiquitous medical applications enabling doctors to use the system independent of their current place and time. The proposed middleware architecture supports session management and content adaptation for healthcare applications for physicians. Although we find this work to be the closest to ours, it still does not address the issues of terminal mobility (e.g., it does not describe what happens when network handoff is performed). The Ubidoctor project does not describe what requirements are needed from client applications and how those requirements can be implemented, and does not say if and how security is managed in the proposed middleware.

One other project is Activity Based Computing [8], which describes an architecture supporting local mobility within hospitals. This project analyzes mobile medical work in one hospital environment and proposes architecture for supporting the mobility of medical personnel. This project is focused more on managing application mobility across different terminals (application roaming), without terminal mobility and support of roaming through different network types (use of the application is limited just to hospital environment). Additionally, the proposed services can be provided solely through a web browser that can limit usability and user-friendliness of a user-interface. Using just web browser for accessing service on all devices does not offer possibilities for utilization of terminal specific characteristics and adaptation to specific context of use.

Generic support for terminal mobility is known from the literature using technologies such as Mobile IP, Mobile VPN, and SIP [10][11]. Mobile IP provides seamless connectivity across heterogeneous networks, but it requires operator or enterprise support mechanisms such as the hosting of a Mobile IP Home Agent and the provisioning of mobile user credentials. Unfortunately, Mobile IP has not reached adoption in popular mobile terminals, and few operators and enterprises use the technology. Mobile VPN is popular in enterprise environments but is not well suited for operation and use in the general public, as it adds an additional level of encryption, authentication and user provisioning, which adds to the communication overhead. Additionally, for both Mobile IP and Mobile VPN solutions, additional software must be installed on the mobile device for the service to be available to end users. SIP is well suited to register a mobile client [11] and has terminal support, but SIP does not provide seamless connectivity for the application as the terminal moves. SIP would also duplicate some of the application-specific mechanisms we need in the CONNECT application. SIP is primarily designed as a registration service, where the mobile terminal can be reached from the network (using SIP URIs), but for a patient service, our requirement is only connectivity from the patient to the service, and the reverse is not needed.

In the paper [12] it is described a unified middleware that isolate mobile healthcare applications from mobility management, client discovery, and transport of multimedia traffic. The proposed all IP-based framework describes how SIP can be improved and used for performing mobility management, session establishment and handover functionalities. The tests of the proposed solution provides improved handling of handovers over heterogeneous networks with better QoS and low packet loss during transitions. However, in the presented work the issues of security and protection of patients sensitive data during handover is not addressed, and security provided only by the SIP protocol is not acceptable for healthcare applications and protection of private patients medical data. For example, authentication of the user in SIP protocol session initiation process can be done using HTTP digest mechanisms or S/MIME mechanisms [13] and utilization of these mechanisms for protection for patients medical data is not always acceptable.

One approach on how application mobility can be handled by adding proxy server in system architecture that recognize session migration request and forwards connection from one client to other is given in [14]. For maintaining sessions in application migration scenarios SOCKETv5 [15] proxy is used. The paper presents only how session migration is handled, and do not describe how and if terminal mobility is supported. When session is transferred authentication of the user is performed only using the application and device IDs. This means that any person with the device that is previously registered for this specific service can get access to the system, and this type of security implementation is not acceptable for mostly healthcare applications.

From previous work, we can see that the number of projects addressing mobility and session management issues for healthcare services is very low and security mechanisms implemented are not always acceptable for protection of patients' private data. Additionally, due to the diversity of possible terminal types, access networks and users requirements usability issues and adaptation to specific terminal must be addressed in more depth. We have also

considered standards-based mobility solutions and found that the traditional methods of Mobile IP and Mobile VPN are not well suited for general public use in healthcare services; the use of SIP would duplicate services that we must have in the application framework. We find that research work in this area is lacking, and our work presented in this paper addresses the main issues regarding the development of a ubiquitous patient support system, which allows patient mobility and seamless access.

IV. CONNECT Framework Description

To overcome the identified challenges, we propose and implement a generic architecture framework called the CONNECT framework for developing context sensitive patient support systems with support for handling security, session management, and content adaptation. Utilizing the proposed framework could facilitate implementation, maintenance and adaptation of healthcare services for different types of client terminals, operating systems and access networks available to patients by saving development time.

We develop and describe a generic and modular architecture and design framework. The main reasons for choosing a modular architecture are the ease of integration with other systems, its functionalities, and the possibility of utilizing the same modules in different types of services and applications on the same types of terminals. Additionally, adding new features can be accomplished by including new modules, without changes to other parts of the system. Note that the proposed framework is not dependent on any specific developer platform and can be applied to any type of service and development environment.

Session management is performed as part of the framework. Information about the current patient session is stored in a database repository, and each user has one item in the session database referred to by the user identifier. During the time that a user is using the service, changes in session state are stored locally, in the application cache. The changes to the database repository are made only when the communication with the client side is finished or interrupted. Every session could have one of the following states: active, finished, or interrupted. A session is active when it is used frequently. A session is finished when the user logs out and shuts down the client application. When the session is finished, data about the session are removed from the database. A session is in an interrupted state when the user loses the connection with the service. When the session has an active or interrupted status, the user will be given the option to continue a previously started session or to start a new session when he/she logs in.

Different types of terminals have different device characteristics, different software and OSs, and support different communication networks. For this reason, it is difficult to find a unique solution to develop a client application that is supported and adjusted for most access terminals. One approach to have a single solution for all devices is using a web application. In this way, service could be accessed using a web browser that is available on all devices with an Internet connection. However, due to low usability and poor client acceptance of web browsing applications on all devices (especially mobile terminals) [16][17], we propose the development of applications that are adjusted to each specific terminal type and can be optimized for each device with regard to a specific user interface and design guidelines. To facilitate this process, we propose a client framework for the development of applications that are not dependent on device type and operating system and can be easily adjusted and used on different developer platforms. Following the same framework during the development of different client applications, developers can save time and provide a more consistent and easily managed application that can adapt user interfaces according to terminal characteristics. Additionally, when proposed modules are developed for one client application, they can be easily reused in other applications developed for the same platform.

Separate modules in the proposed framework are in charge for providing security and protection of the user private information. We do not specify which security mechanisms should be implemented in the security modules. The implementations of the security mechanisms in the real-life system depend on many different factors (e.g., type of service and privacy of information that is provided, legislation requirements, and type of device and communication network). Putting security management as part of the proposed framework enables more flexible security management and adjustments of used security mechanisms for specific context of use (for example, different authentication mechanisms can be used for different types of devices and they can be adapted to device's specific characteristics). In the CONNECT framework, we address security issues related to handling terminal mobility. When user changes access network he/she must be re-authenticated over the new communication channel, and intermittent connectivity is handled only by retransmission capabilities in Transmission Control Protocol (TCP). This choice avoids the requirement of a specific terminal mobility technology (such as Mobile IP), and most importantly enables managing security issues at the application level.

The framework supports both application mobility and terminal mobility at the application level. Placing session management in the application layer introduces more development work compared to using services from lower layers; however, this solution allows a more generic framework applicable for any type of client device and communication network. Additional reasons for implementing terminal mobility within the framework (and for not using related methods such as Mobile IP and Mobile VPN) are:

1) Security. When a network changes, we want to perform a re-authentication at the correct level (towards the PHR system) and not only to the mobility provider (such as the mobile operator). We claim that it is necessary for users to be asked to re-authenticate when networks change, ensuring that the session migration is performed for the right user.

2) User friendliness. Session management and terminal mobility can result in additional requirements for user interaction. If additional user interaction and user interface is not implemented as part of the application, this can lead to inconsistent application interface and extra load on the users which is not acceptable for healthcare services that must be

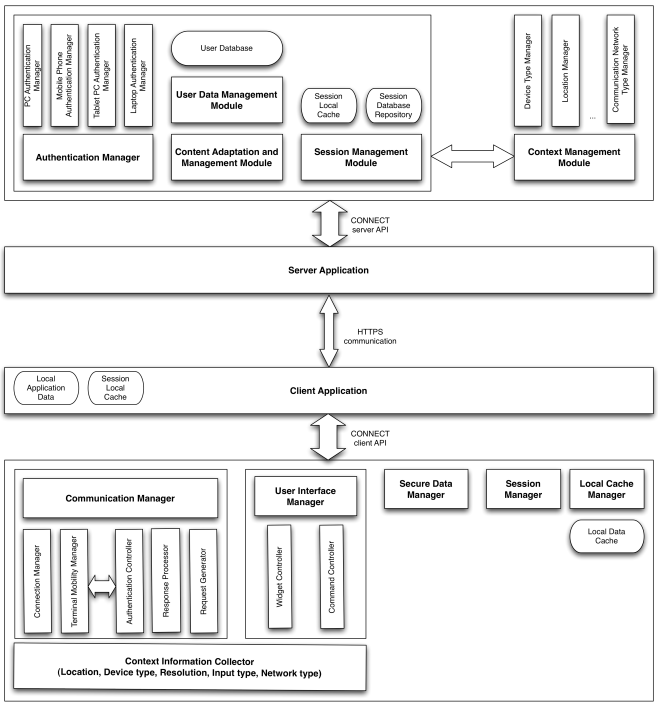

Figure 2. The CONNECT mobility framework.

useful, easy to use, and adapted to different types of users (e.g., elderly and people with specific needs). In our framework the interface for managing terminal mobility is implemented on the application level, and the interface elements is consistent with the rest of the application. Additionally, requirements from user during mobility management must be minimal (e.g., no additional installations/settings of third party software should be needed).

The proposed framework describes both server and client-side system architecture (Figure 2). In the following, we will describe details about the framework and its modules. Each module will provide an Application Programming Interface (API) for the simple reuse of the mobility features across applications and operating systems.

A. Server architecture framework

The server architecture framework provides support for the implementation of security mechanisms, application logic, and session management and the generation of application data adjusted to the user terminal, the network characteristics and the session state. The proposed server architecture framework consists of five main modules:

- *Context Management Module* process information received from a user regarding the context in which a service is used (e.g., device type, location, and network type), and provide this information to other modules for selection of content adaptation, security, and session management policies.
- *Session Management Module* performs storage, maintenance and retrieval of session data.
- *Content Adaptation and Management Module* is responsible for managing user data and for generating responses for a client application adapted for current context of use.
- *User Data Management Module* implements communication with the User Database where all users' private data are stored.
- *Authentication Manager* makes sure that the user is correctly authenticated before access to the service is enabled. Different sub modules can be used for implementation of different security mechanisms applicable for different contexts of use (e.g., type of service, characteristics of client terminal, communication network, users, and laws and state policies regarding the protection of healthcare data).
- *Server Application Controller* is responsible for managing the communication process with the client and for implementing application logic.

B. Client architecture framework

In addition to the server architecture framework, we propose an architecture framework for developing client applications. The client architecture framework consists of the six main modules:

- *Context Information Collector* gathers and provides context information (e.g., user location, device type, screen resolution, input method, and communication network type) to other application modules. This information can be used to determine the context in which the application is used and to detect changes in the user environment.
- *Communication Manager* is in charge of performing communication with the server side.
- *Terminal Mobility Manager* enables user roaming over different types of networks, and, when it detects that the network is changed, it initiates a new authentication process, so the user can continue the current session without exiting the application.
- *User Interface Manager* is responsible for the organization of widgets and interface elements, additional adaptation of content on the screen and user interactions based on specific device characteristics, and for the implementation of events and actions related to a user's commands.
- *Secure Data Manager* is in charge for application specific secure data management (e.g., secure data deletion and ensures that all data are deleted and nothing is stored in permanent memory for security reasons).
- *Session Manager* performs storage, maintenance and retrieval of session data.
- *Local Cache Manager* is responsible for storing application related data that do not endanger a user's

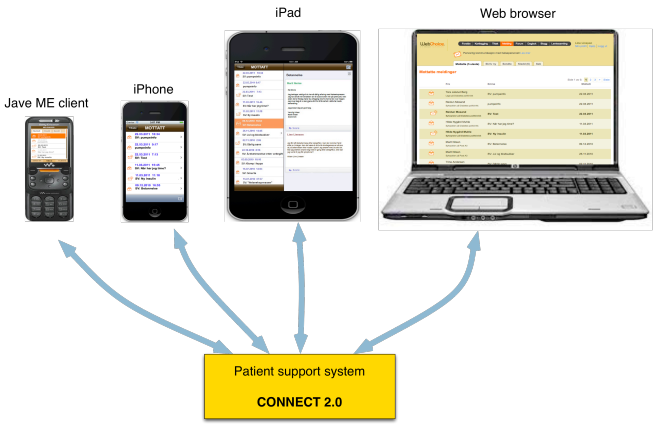

Figure 3. Different client types.

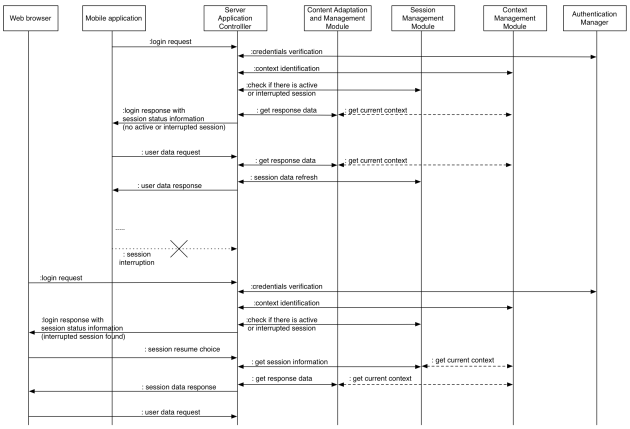

Figure 4. Mobility scenario – message flow.

privacy in permanent device storage (e.g., log data, interface preferences).

- *Client Application Controller* is the main application controller used for implementing application logic.

V. Prototype Implementation And Testing Results

For testing purposes, we developed a server application and three types of client applications: a mobile phone Java client application, an iOS client application and a web client application.

For development of the server application, we used a Java Standard Edition platform. Each module is implemented through a separate package, and the server application is deployed on an Apache Tomcat server. The Tomcat server is running on the CentOS 5.5 with 512 MB of RAM. We used a MySQL 5.5.11 database for implementing the session database repository. The client application for mobile phones was developed using the Java Micro Edition (Java ME). For development and testing of the mobile application, we used the Sun Java Wireless Toolkit. Due to the high popularity and wide acceptance of iOS devices, we also decided to develop mobile applications for the iPhone and iPad. For the development and testing of the applications, we utilized Xcode (the toolset for the development of software for iOS). Using the Java platform, we developed a client web application to enable user access to the service using web browsers.

In the original CONNECT system, there are numerous functionalities offered by the patient support tools (e.g., exchange of messages with health personnel, registration of problems and difficulties, access to databases of advice and help information, and a forum for communication with other patients). For testing purposes, we developed applications that enable access to the messaging module. Functionalities currently implemented are: login to the service, preview of all messages (categorized into three groups: inbox, sent and draft), preview of one message with the message text, sending new and reply messages, and saving a message in drafts. Images of different user interfaces are shown in Figure 3.

The proposed framework is focused on mobility and session persistence, and is compatible with several authentication methods. Secure authentication for mobile healthcare applications are proposed and evaluated in [6].

A. Mobility handling scenario

Session and mobility management in our testing environment are implemented in accordance with a proposed framework, and one possible mobility scenario is described in Figure 4.

First, the user starts using a service on his/hers mobile phone, and after inputting a credentials, a login request is sent to the server. When the Server Application Controller receives the request, it verifies the user's credentials utilizing functionalities of the Authentication Manager. If the credentials are correct, the server identifies and stores the current user's context using the Context Management Module. In our test service, a user's context information is the type of client application that accesses the service, and this information is transferred in an HTTP request header. Afterward, the information is checked for whether there is an active or an interrupted session for this specific user stored in the Session Database Repository. If there is no session data stored, the login response is returned to the user, and he/she can start using the application. When the client application requires a user's data, it sends a request to the server. The request is forwarded to Content Adaptation and Management Module, which gather data regarding the current context from the Context Management Module and generate a response for the user. The Server Application Controller performs a session refresh operation and sends the response to the client application. If there is an interruption in the session, and the user tries to login from his/her computer after a delay, a login procedure will start in the same manner. However, because an interrupted session is identified, a response from the login request will notify the client application about the current session, and the user will be asked if he/she wants to continue the previous session. If the response is yes, the session information will be gathered from the Session Database Repository, and the response containing the session data will be sent to the client application. When it receives session data, the client application can load the last saved session state.

TABLE I. TIME OVERHEAD FOR SESSION MANAGEMENT FUNCTIONALITY

Application type	Session management overhead [seconds]
J2ME	1.05
iOS	1.1
Web browser	0.57

B. Session management time overhead

To determine how much time is needed to perform session management tasks, we tested two usage scenarios and compared results. In both scenarios, we measured the time needed for a user to log in to the application. In the first scenario, session management functionality is disabled, while in the second scenario, session management functionality is enabled. We identified login functionality as the most time-consuming operation in the proposed session management process, and we wanted to find out how system performance is affected when session management functionalities are supported. We performed testing on three types of client applications developed for different client device platforms: J2ME, iOS and a standard web browser. Each user scenario is repeated 10 times, and the difference in the mean values for two user scenarios for three client applications are shown in Table 1. If we include time to perform a secure authentication for J2ME and iOS terminals, we have measured overhead for two-factor authentication including SMS push over WiFi network to be around 25 seconds [6]. Thus, a secure login process require only 4% additional overhead to include session management support. Our conclusion is that this overhead does not present a problem, and time used for session management is dominated by time needed for an authentication process.

C. Scalability

Next, we measured how the session management functionality affects the scalability of the system. For simulating an extra load on the server, we used Pylot software, a open source tool for testing the performance and scalability of web services [18]. We measured scalability using only the web application. Pylot software is used to generate requests for 10, 20, 50, 70 and 100 users simultaneously. Simulated user actions consist of the following steps: logging into the application, going to the messages menu, going to the list of inbox messages, going back to the main menu and logging off the application. We tested this scenario on the client application where session management is supported and on the client application without session management functionality implemented. For the purpose of analysis, we used the average value of the response times for 10 users for all of the performed tests. The results of testing that shows measured times when session management is enabled and when session management is disabled are shown in Table 2. The time difference of the mean values for all of the testing scenarios is shown in Figure 5.

From the graph, we see that the extra time needed for session management varies from 0.1 to 0.6 seconds, depending on the number of simultaneous users. We can conclude that the time overhead measured is acceptable and that the scalability of the system is not substantially affected by adding session and content management functionality based on the described framework.

TABLE II. RESULTS OF SERVICE SCALABILITY TESTING

Number of users	Mean (SD)	
	Without session management	*With session management*
10	0.356 (0.022)	0.418 (0.027)
20	0.665 (0.057)	0.744 (0.05)
50	1.612 (0.114)	2.027 (0.125)
70	2.37 (0.158)	2.792 (0.373)
100	3.76 (0.529)	4.3597 (1.43)

From the results, we calculated that session management functionality takes an average 15% of a server's response time. Assuming that the maximum acceptable time for a server response (keeping the user's attention focused on the dialogue) is 10 seconds [19][20], we set a maximum acceptable session management time to 1.5 seconds. Based on the results presented and the trend analysis, we can say that the proposed test environment is able to support approximately 300 simultaneous users per server. Assuming intermittent use of the servers while using the mobile application, the number of users per server can be up to a thousand active users. From the results obtained, we can conclude that general scalability of the testing environment is acceptable for testing purposes and for a limited number of users. For implementation in a real hospital environment, where the number of patients is higher (e.g. thousands or millions of potential users) additional capacity could be easily added with multiple servers with higher performance characteristics.

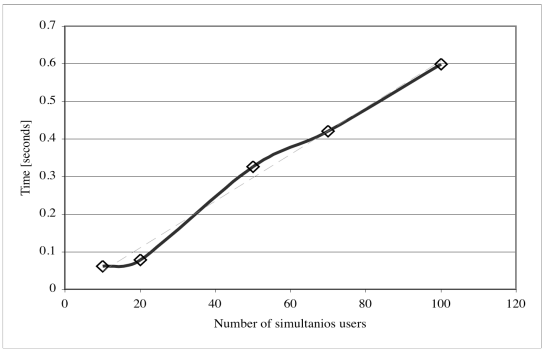

Figure 5. Session management time overhead.

VI. CONCLUSION

In this paper, we describe a framework for the implementation of content adaptation and session and mobility management in ubiquitous patient support systems. We describe a generic and modular architecture and a design framework that provides support for seamless application mobility across any terminals. The framework also provides support for handling terminal mobility between wired connections, WLAN, 4G, 3G, and 2G networks on secure and user friendly manner. Handling security issues as part of a framework enable easy integration of different security mechanisms and solutions that comply with security regulations and laws. The CONNECT mobility framework is easily applicable on different terminals and operating systems and can be used by multiple services on the same terminal. The framework enables the development of user friendly and terminal-specific interfaces with advanced session, content, and mobility management functionalities.

Through the implementation of the framework and several demonstration applications, we showed that the framework could be used for different client platforms and that performance and scalability are not severely affected by adding mobility support. Support for terminal mobility was also proposed and implemented, requiring users to re-authenticate when network access changes, protecting user communication from unauthorized use. Additional tests of the terminal mobility functionalities and security handling are planned as the future fork.

We have proposed placing session, security, and content management functionality into a common framework that can be used by multiple applications. The CONNECT framework has been implemented on relevant mobile terminals such as J2ME, iOS and web. We claim that the use of this framework simplifies the development of mobile healthcare applications with similar security and mobility requirements, but we conclude that the user interface of the applications must be written specifically for each platform to fully utilize the potential of the mobile terminal. With this architecture, it is a challenge that mobility services must be available for different platforms. Thus, we propose to standardize features similar to the CONNECT framework for application mobility in both client and server operating systems.

REFERENCES

[1] E. Murray, J. Burns, T.S. See, R. Lai, I. Nazareth. Interactive Health Communication Applications for people with chronic disease. Cochrane Database Syst Rev. 2005 Oct 19;(4):CD004274.

[2] D. McGeady, J. Kujala, K. Ilvonen. The impact of patient-physician web messaging on healthcare service provision. International Journal of Medical Informatics 2008 Jan;77(1):17-23.

[3] J. Mirkovic, H. Bryhni, and C. Ruland, "Review of projects using mobile devices and mobile applications in healthcare", Proc. Scandinavian conference on Health Informatics, 2009.

[4] Appari, A., Anthony, D. L., Johnson, E. M.. "HIPAA Compliance: An Examination of Institutional and Market Forces", Paper presented at the The 8th Workshop on Economics of Information Security, London, 2009.

[5] Ministry Of Government Administration, Reform and Church affairs,, Requirements specification for PKI for the public sector, January 2005. Available: http://www.regjeringen.no/en/dep/fad/Documents/Acts-and-regulations/retningslinjer/2010/requirements-specification-for-pki-in-th.html?id=611085

[6] J. Mirkovic, H. Bryhni, and C. Ruland, "Secure Solution for Mobile Access to Patient's Health Care Record", Proc. 13th International Conference on E-health Networking, Application & Services, Columbia, MO, 2011.

[7] C. Ruland, A. Jeneson, T. Andersen, R. Andersen, L. Slaughter, B. Schjødt-Osmo, et al, "Designing Tailored Internet Support to Assist Cancer Patients in Illness Management", AMIA Annual Symposium. Chicago, IL, pp. 635-639, 2007.

[8] J. Bardram, T.A.K. Kjær, and C. Nielsen. Supporting Local Mobility in Healthcare by Application Roaming Among Heterogeneous Devices. Mobile HCI2003, Udine , Italy, 2003, pp.161-176.

[9] J. R.B. Diniz, C.A.G. Ferraz, and H. Melo. An architecture of services for session management and contents adaptation in ubiquitous medical environments. In Proceedings of the 2008 ACM symposium on Applied computing (SAC '08). ACM, New York, NY, USA, pp. 1353-1357.

[10] F. Steuer, M. Elkotob, H. Bryhni, and T. Lunde. Seamless mobility over broadband wireless networks. Proc. IST Mobile and Wireless Summit, Dresden, Germany, 2005.

[11] F. Panken, H. Bryhni, P.E. Engelstad, L. Hansson, G. Hoekstra, M. Gilje Jaatun, and T. H. Johannessen. Open Access Networks - Architecture for Sharing Residential Access with Roaming WLAN Users, Special issue on Open Access Networks, Telektronikk Vol. 3, No.4, 2006.

[12] Soomro, A., Schmitt, R. "A framework for mobile healthcare applications over heterogeneous networks", Paper presented at the 13th IEEE International Conference on e-Health Networking Applications and Services (Healthcom), 2011.

[13] Tat, C., Sengodan, S. "On applying SIP security to networked appliances", Paper presented at the IEEE 4th International Workshop on Networked Appliances, 2002.

[14] Højgaard-Hansen, K., Huan Cong, N., Schwefel, H. "Session mobility solution for client-based application migration scenarios." Paper presented at the Eighth International Conference on Wireless On-Demand Network Systems and Services (WONS), 2011.

[15] M. Leech, M. Ganis, Y. Lee, R. Kuris, D. Koblas, and L. Jones, "SOCKS Protocol Version 5," RFC 1928 (Proposed Standard), Mar. 1996. [Online]. Available: http://www.ietf.org/rfc/rfc1928.txt

[16] J. Mirkovic, H. Bryhni, and C. Ruland, "Designing User Friendly Mobile Application to Assist Cancer Patients in Illness Management", Proc. The Third International Conference on eHealth, Telemedicine, and Social Medicine, Gosier, France, 2011 Feb, pp. 64-71.

[17] Jakob Nielsen's Alertbox. iPhone Apps Need Low Starting Hurdles. Retrieved May 30, 2011, from http://www.useit.com/alertbox/mobile-apps-initial-use.html.

[18] PYLOT: Performance & Scalability Testing – Web Services. Retrieved May 30, 2011, from http://www.pylot.org/.

[19] J. Nielsen, Usability Engineering, Chapter 5: Response Times: The 3 Important Limits (Morgan Kaufmann, San Francisco, 1993).

[20] Java Look and Feel Design Guidelines: Advanced Topics. 2001. Retrieved May 30, 2011, http://java.sun.com/products/jlf/at/book/Responsiveness5.html.