Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 42 (2014) 168 – 174

Procedia Computer Science

International Conference on Robot PRIDE 2013-2014 - Medical and Rehabilitation Robotics and Instrumentation, ConfPRIDE 2013-2014

Healthcare Telematics Service Implementation Using OSGi Framework

Veera Ragavan. S[a], Kumail Haider[b], Madhavan Shanmugavel[c]

[a]*School of Engineering, Monash University, Malaysia campus, Malaysia*
[b]*Student, Monash University, Malaysia campus, Malaysia.*
[c]*School of Engineering, Monash University, Malaysia campus, Malaysia.*

Abstract

This paper outlines the rapid application development of a Healthcare Telematics System application for use in emergency first response vehicles on the Service Oriented Architecture model using an OSGi framework. OSGi facilitates the development of application modules at the knowledge level thus encouraging quick creation and prototyping of services. In this paper, we have built a prototype application for use in Medical Telematics to provide remotely placed doctors at hospitals with patient FIRs to enable supervised emergency medical care by paramedics even before the patients reach the hospitals.

Peer-review under responsibility of the Center for Humanoid Robots and Bio-Sensing (HuRoBs)
Keywords: Healthcare Telematics, Service Oriented Architecture, OSGi framework, Emergency First Response

1. INTRODUCTION

The evolution of both mobile and Local Area Network (LAN) technologies offers better opportunities for supporting the inter-operation between mobile devices and devices residing in a LAN. In this paper, we focus on wireless enabled medical equipment and LAN devices in medical environments. By inter-operation, we mean the ability for mobile medical devices, such as pacemakers and gastric stimulator's, to discover and connect with available medical LANs to provide first hand data to doctors and physicians to help them monitor patients health and issue emergency responses. The past few years have produced innovative health oriented networking and wireless communication technologies, ranging from low power medical radios that harvest body energy to wireless sensor networks(WSNs) for in home monitoring and diagnosis. Systems built to facilitate these medical environments have the capabilities to semantically discover the desired medical devices and sensors based on their physical and functional characteristics, and use their services for exchanging the measurement results over the network and produce a higher level of interpre-

* Corresponding author. S.Veera Ragavan.
E-mail address: veera.ragavan@monash.edu

Peer-review under responsibility of the Center for Humanoid Robots and Bio-Sensing (HuRoBs)
doi:10.1016/j.procs.2014.11.048

tation of these results. Every year emergency medical situations cause hundreds of injuries and life-long disabilities. Precious lives are lost due to the delay to transfer patients and vital medical information to the hospital. The usual scenario in a first response emergency vehicle involves frantic communication between the medics and the emergency response initiators/good Samaritans. Vital information such as patient symptoms or patient location can be lost leading to wrong case assessment. The time the patient spends in the vehicle is critical time lost by the medical professionals at the hospital. Another major concern with first response systems is that a patients vitals cant be monitored by the medical professionals who will be treating the patient as soon as he/she will arrive at the hospital. The first look they get at the patient is when they are wheeled in.

From the ODRA flood 1997, the high speed train crash in ESCHEDE 1998, the DANUBE flood 1999 and the ELBE flood in 2002 experience reports were collected. They were analyzed with emphasis on data and work flow in the medical treatment and its command system [1]. The results showed that the use of spoken radio communication causes transmission mistakes or misunderstandings as well as time to setup the connection for each call leads to loss of precious time. Manual distribution of the data also leads to a time shift in the up-to-date-information. These results demonstrate that continuous and reliable data transmission as well as up-to-date transportation is necessary for medical response teams.

2. LITERATURE REVIEW

Telematics is the integrated use of telecommunication and informatics technology. Telematics is considered the next-wave of mobile computing with the development of embedded computing platforms capable of running general purpose applications [5]. Earlier reports have predicted the telematics industry to evolve into a $41 Billion industry within the current decade [6]. Using telematics, information can be collected from the vehicles and can be transmitted to the control center where the data can be used for various applications as well as sending back various commands to the vehicle [7]. In - vehicular Telematics Systems combine mobile computing and telecommunication services using mobile computers, palm devices and smart phones. Information with centralized command is accessed through wireless links such as GSM and WAP methodologies [8]. Some Middleware [10] methodologies are COBRA, DCOM and Jini. Though early middleware used to be transaction oriented (IBMs CICS) the current technology consists of object-oriented middleware with higher level service oriented abstraction (Jini) [8]. The Java based Open Source Gateway initiative (*OSGi*). *OSGi* is known to be a dynamic module system that allows:

- Modularisation (through dynamic modules called as Bundles)
- Visibility of bundle content. (Public, Private API)
- Dependencies between bundles
- Version control of modules
- Simple deployment, Composition and Software Management. [12]

Using *OSGi* has several benefits. Due to small component sizes, it makes it easier to build. Having independency (not coupled) between components gives the option of reusability. Through Eclipse Equinox we are open to a number of services such as Event admin, HTTP services, Log services, User/Wire admin etc. [12]. Advantages of *OSGi* are:

- Incremental update of server function
- Ability to run multiple versions simultaneously
- Class loading performance
- Sharing of components across client and server [13]

The *OSGi* Alliance which governs the *OSGi* specifications and standards introduces a layered model.

- Bundles - Bundles are the *OSGi* components made by the developers.
- Services - The services layer connects bundles in a dynamic way by offering a publish-find-bind model for plain old Java objects.
- Life-Cycle - The API to install, start, stop, update, and uninstall bundles.

- Modules - The layer that defines how a bundle can import and export code.
- Security - The layer that handles the security aspects.
- Execution Environment - Defines what methods and classes are available in a specific platform. [14]

Many *OSGi* implementations exist in the market. Equinox is the Eclipse (IDEs) *OSGi* open source implementation whereas *ProSyst*^{TM} is a commercial version .

3. METHODOLOGY

Service Orientation and Services aggregation form the core of the methodology that has been used in development of this Health care Telematic Application. A distributed system was implemented using an SOA based prototype application which utilizes an *OSGi* container as the framework. The *OSGi* framework is well suited for the dynamic capabilities required for the distributed system utilizing the JAVA class loader architecture for deploying the services at runtime. Figure 1 gives a graphical overview of the distributed nature of the serves required by the application.

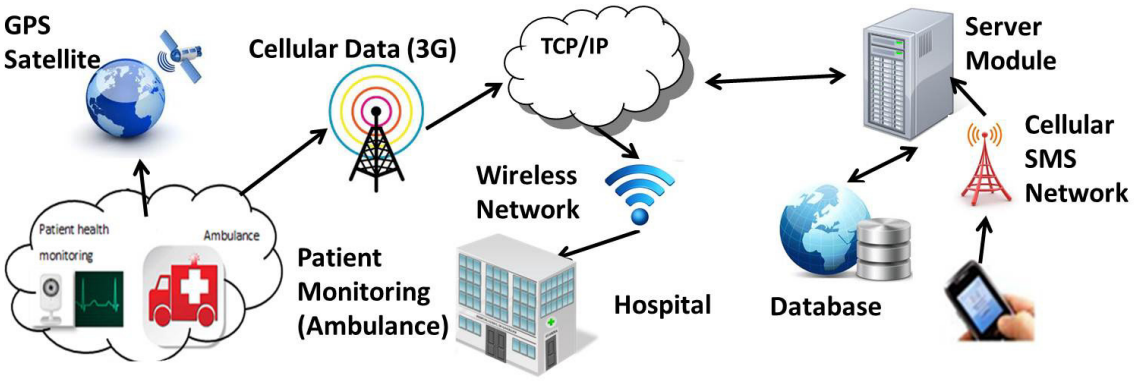


Fig. 1. Overview of Patient Monitoring system Application

The system is initiated by the emergency caller who provides the location of the distress call using a cellular network through his/her mobile device. This location is saved in the database connected to the server and the first response team is dispatched over the internet, through TCP/IP, to the distress location. The ambulance picks up the patient at this location. The ambulance then begins the Patient Monitoring Services(PMS). The services included in the PMS are video streaming, patient health monitoring and audio communication through VOIP. The GPS navigation service created earlier in [4] has been aggregated to provide a navigation system to the ambulance as well. These services utilize an ad-hoc network, built to provide the services with connectivity to the internet. The data path (TCP/IP) for the ad-hoc network to access the internet is provided by the cellular data (3G). These services are published as remote services. The IP cameras fitted in the ambulance provide the streaming functionality. The patient monitoring is implemented through network enabled medical capture devices. Microphones either built in the cameras or standalone hardware can be used for the VOIP communication. For the navigation system, the ambulance uses location data it attains from a GPS satellite via a GPS receiver. These services are accessed by the medical professional/s at the hospital through the internet over TCP/IP using wired or wireless networks. The embedded security features in the *OSGi* framework is utilized to prevent access by unauthorized personnel. These services are presented in the form of a Graphic User Interface (GUI).

4. APPLICATION SCENARIO

To demonstrate the use of the prototype application, we have defined a particular use-case scenario where, a good Samaritan/relative initiates an emergency call on witnessing a person going into cardiac arrest. The healthcare providers receive this call and immediately dispatch a first response team to rush to the location of the call, place the

patient in an ambulance and transport him/her to the hospital for treatment. In an ideal scenario, the ambulance would reach the hospital within a short period of time where the patient would be rushed into the OT for treatment. But in a real world scenario an ambulance, due to traffic and unforeseen hurdles, takes time to reach the hospital. This leads to the patient spending more time in the ambulance and that much less time in the hands of the doctor. We propose to build a prototype system application to help doctors monitor the vitals of the patient and begin their analysis/treatment remotely to help give the patient as much aid as possible while he/she is in the ambulance. The services we propose to build for our application will help the remote medical professional/s receive the vital information they need in a continuous and timely manner without delay in the transmission of data. The information will be available in two separate GUIs. The Doctor Specific GUI will contain the services for patient monitoring while the TCM specific GUI will contain the navigation services.

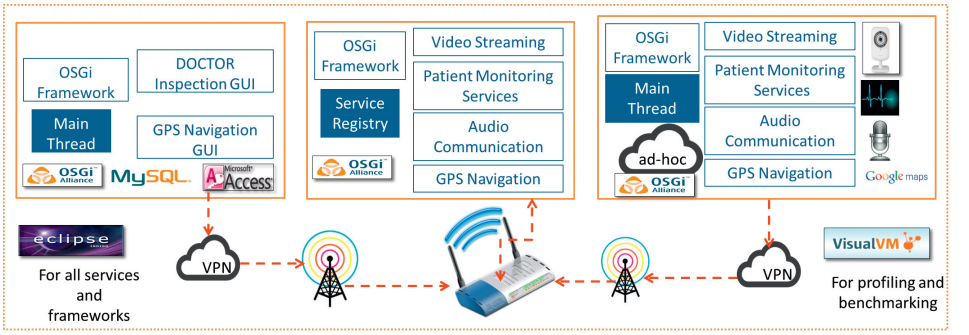


Fig. 2. Service Diagram for Patient Monitoring System (PMS)

5. APPLICATION SERVICES

In emergency situations involving cardiac arrests it is essential for the medical professional/s to continuously monitor and evaluate the vitals of the patient. If the doctor at the hospital does not receive these vitals in a timely manner which could render the actions performed thereof too little, too late. We propose to use video streaming over IP cameras to get a visual feed on the patient to analyze symptoms that cannot be communicated via graphs or data as well as directly communicate with the first response medics to get an accurate first hand report of the situation. This will also be a good method of analyzing the way, first response teams carry out medical protocols.

To help the emergency response vehicle drivers take the most convenient and traffic free route to the hospital, the transport control center will be fitted with video streaming services to provide up to date traffic and route information to the driver. A GPS system will provide the current position data to the control center to help navigate the ambulance. Also a visual indication will be implemented to show the staff at the emergency ward the Expected Time of Arrival (ETA) of the ambulance to help them prepare the stretchers and other required medical equipment to transport the patient into the operation theatre. The services are detailed below,

Secure Login: The secure login service will ensure that only the medical professionals with authorization can gain access to the system through the usernames and passwords that they have been issued by the hospital management. This will prevent unwanted individuals to access the system.

Video Streaming: This is a core *OSGi* service that has been built from the ground up. No proprietary software has been used for the development of this service. It has been specifically built for applications that are intended to be lightweight and portable. The safety of the video stream from hacking is ensured by the embedded security features in the *OSGi* framework. This service will put the Doctor directly in touch with the first response medic and will provide a visual feed to the patient. IP cameras will be used with filtering to ensure good quality video streaming with as close to a 1:1 streaming ratio possible. We have decided not to use any proprietary software to prevent infringing on any copyright or EULA laws. Also proprietary software bundled with hardware usually contains features which, for this system would be considered as bloat-ware.

Patient Medical Monitoring: To provide the doctor with up to date, continuous feedback of the patients vitals, the *OSGi* framework will be pre-configured to provide a common platform for any wirelessly enabled medical monitoring device to send its data over the ad-hoc network and display the captured information in the form of graphs or gauges in as useful a manner as possible. The beauty of systems built on Service Oriented Architecture comes to light in applications such as this, where the underlying structure of the *OSGi* framework makes the system interoperable. Any network enabled medical capture device, built by any manufacturer, just needs to register itself with the framework of the system and the system will recognize the device, slot it into its array of services and provide the device with a platform to record and display its data. Whatever be the platform that the medical device has been built on does not pose a hurdle, since the cross platform exchange of messages by the services in SOA based systems ensures that the services built on different platforms work together in harmony. This is one of the major advantages provided by the *OSGi* framework to provide interoperability with different networks [2]. For our prototype application, due to the unavailability of real time medical data monitoring we have used open source medical data to simulate the services the doctor would view.

Patient Specific Devices: This service will be used for specific patients who have been retro fitted with any medical monitoring device such as insulin pumps, cardiac defibrillators etc. On receiving permission from these devices, they can be slotted into the framework and be provided as a service. The doctor will have the option of viewing data from this source as well. In our test scenario we have provided the option of enabling this service but an actual connection to a device could not be made due to unavailability. Testing was done using Simulation Data as shown in Figure 3(a).

Audio Communication: The IP cameras can utilize their microphones to provide voice over IP (VOIP) audio communication. The use of cell phones is not recommended near the medical equipment because the radio waves used for the communication through cell phones causes interference in these devices [3]. Using a similar method as video streaming, the groundwork for setting up audio communication over an IP address has been carried out for the framework. IP cameras enabled with microphones or standalone microphones can be used to achieve the functionality of this service.

GPS Navigation: The core services provided was built by Dinuka Perrera [4]. Utilizing the ability of systems based on SOA to rapidly aggregate services, this GPS service was introduced into our Patient Monitoring System Application to provide navigational support to the ambulance.

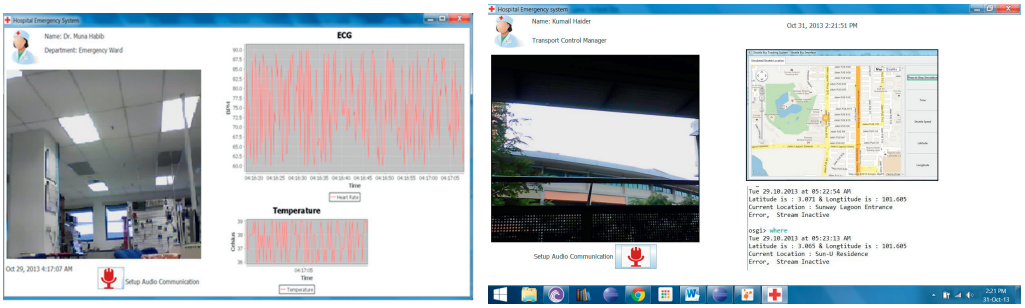
Table 1. SUMMARY OF SERVICES

Service Interface	Modules Located in	Services Offered
IGuiDR	Hospital Module	Start initialization of services for the Doctor GUI
IGuiTCM	TCM Module	Start initialization of services for the TCM GUI
IVideoStream	Ambulance Module (Service Provider) Hospital Module (Service Consumer)	Establish video stream connection via ad-hoc network
IPatientMonitor	Ambulance Module (Service Provider) Hospital Module (Service Consumer)	Capture data from network enabled medical devices
IPatientSpecMonitor	Ambulance Module (Service Provider) Hospital Module (Service Consumer)	Create connection with medical device implanted/on patient for data capture
IAudio	Ambulance Module (Service Provider) Hospital Module (Service Consumer)	Enable audio communication with VOIP
IServiceFactory	Server Module	Handles and tracks the dynamism of the services
IGpsService	Ambulance Module	Collects location data for navigation

The aggregation of the services in the Table 1 to form Health Telematic Applications is shown in Figure 3. Figure 3(a) shows the aggregation of services to obtain the *Doctor GUI*. Figure 3(b) shows the aggregation of services to obtain *Navigation GUI*.

6. PERFORMANCE BENCHMARKING RESULTS

The *VisualVM* profiling tool was used to record the performance of the system. With the help of *VisualVM*, data on the utilization of memory, CPU usage, memory heap and, peak and active thread counts was recorded for further



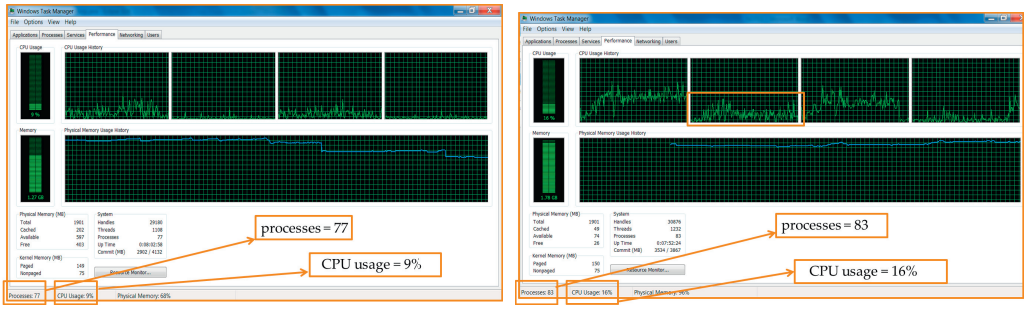
(a) Doctor GUI at the Hospital showing the realtime data acqusition from onsite medical instruments (ECG), Voice over Internet and Live camera feed

(b) Navigation GUI at the Ambulance showing the navigation info, realtime GPS tracking , Voice over Internet and Live camera feed from the Ambulance

Fig. 3. Graphical user Interface of Health Care Telematics Application created using services described in Table 1

analysis. Along with the data recorded with *VisualVM*, the Windows Task Manager was used to corroborate the findings and give a deeper look into the active processes count to analyze the load, the Patient Monitoring System has on the CPU.

In (Figure 4(a)), we see that prior to the initialization of the system, the CPU usage is 9% and the live processes count is 77. After running the system application for over 4 minutes, we see in Appendix (Figure 4(b)) , that the CPU usage is 16% and the live processes count is 83. From the system performance and the graphs provided by the Windows Task Manager, it can be clearly seen that the Patient Monitoring System increases the CPU usage by a significantly low percentage of 7 and the live process count jumps only by 6. *VisualVM* profiler was also used to obtain performance data of application. The memory used is very low (around 18 MB) and the CPU utilization and process count shown by Windows Task Manager is corroborated in these results.



(a) Windows Perfomance Metrics- before Service Initiation

(b) Windows Perfomance Metrics- After Services are running

Fig. 4. Performance of the Application showing minimal resource usage - Application uses only 6 threads and 7% of CPU usage

7. ADVANTAGES

The Patient Monitoring System application uses all the features that make the *OSGi* platform so appealing to application developers. The system is created to allow the services to be dynamic, interoperable and hardware independent. The security features of the *OSGi* framework have been utilized to protect the system against hackers. The system incorporates the use of a profiler which helps the application creators to collect performance and usage statistics, which can be analyzed to improve the systems function and consumption of resources. From the graphs and statistics obtained from the Windows Task Manager and *VisualVM* we can determine that,

- The *Patient Monitoring System* application uses significantly low memory (around 18MB) which allows the system to be portable since the system will even run on connections with low bandwidth. Seeing the low memory consumption, it can also be said that poor/weak connectivity will not prove a present a big hurdle.
- The low CPU utilization shows that the application does not slow down the PC and can be run in the background without hindering other applications.
- The low utilization of resources makes this system application an ideal fit for porting onto small embedded devices.
- The profiling results show that the system application takes up a significantly low thread and process count which coupled with the systems resource friendliness, further corroborates that the system application can be easily ported onto embedded devices.
- The services are running on a JAVA Virtual Machine (*JVM*), which makes the services inter operable and platform independent.

8. CONCLUSIONS

The objective to develop a Service Oriented Application to provide patient monitoring was successfully created and implemented in the Patient Monitoring System application using an *OSGi* framework. Core *OSGi* services were successfully built and published as Open Source Software. These services are video streaming, patient vitality monitoring and audio communication through VOIP. The services are built to be flexible and modular to promote reusability. These services were incorporated into the *Patient Monitoring System* application to successfully simulate a Cyber Physical System. The *OSGi* framework was successfully used to rapidly aggregate previously built services into a proof-of-concept application. The application built is platform and hardware independent by running the *OSGi* framework on a *JVM*. The application architecture allows for dynamism and interoperability to promote system scalability. The security features of the *OSGi* framework were incorporated to make the services secure against hacking and unauthorized access. The system application was subjected to performance benchmarking tests which proved the system to be lightweight and portable. The Conclusion was drawn that this system was a good fit to be ported onto small embedded systems.

References

1. T. Benner, U. Schaechinger, and M. Nerlich, Medical telematics in disaster response, International Center for Telemedicine Regensburg, Studies in health technology and informatics, 2003, pp. 97:15-23
2. S. Moyer, D. Maples, S. Tsang, A. Ghosh, Service portability of networked appliances, Communication Magazine, IEEE Communication Society, New York, pp. 116-121, January 2002.
3. Williams, R.D, Keeping Medical Devices Safe from Electromagnetic Interference, FDA Consumer, Vol. 29, No.4, May 1995, pp. 12-16.
4. R. P. Dinuka, Service Oriented Application Framework For Vehicular Telematics, Final Year Project, Monash University, 2012.
5. B. Chatchsik, B. Isaac, C. Paul, M. Archan, R. Jim, V. Nicolas, Y. Danny, R. Vladimir, H. Henry, S. Craig, Intelligent pervasive middleware for context-based and localized telematics services, in Proceedings of Workshop on Mobile Commerce, 2002.
6. J. Maples, Will Telematics Drive the Industry to Distraction?, Red Herring, April 1, 2001.
7. Ali Karimi, Johan Olsson and Johan Rydell, "A Software Architecture Approach to Remote Vehicle Diagnostics," IT University Of Gteborg, Gteborg, Sweden, 2004.
8. D. Reilly, A. Taleb-Bendiab, A Service-Based Architecture for In-Vehicle Telematics Systems, in Proceedings of 22nd International Conference on Distributed Computing Systems Workshops, 2002.
9. Hughes Systique Corporation. (2006) Automotive Telematics. [Online]. http://www.hughessystique.com/Portals/0/Uploads/Articles/WhitePaperTelematics633833564780651074.pdf
10. Minjung Kim, Yeonjune Choi, YoungBag Moon, Sunjoong Kim, and Oh-Cheon Kwon, "Design and implementation of status based application manager for telematics," in Advanced Communication Technology, 2006. ICACT 2006. The 8th International Conference, 2006.
11. Jeff McAffer, Tom Watson. Programming with Equinox (Powerpoint Presentation), 2006.
12. Martin Lippert. OSGi and Eclipse Equinox explained (Power point presentation), April, 2007.
13. The Eclipse Foundation. (2012) www.wiki.eclipse.org. [Online]. http://wiki.eclipse.org/Riena_Project
14. The Eclipse Foundation. (2012) wiki.eclipse.org. [Online]. http://wiki.eclipse.org/Riena/Getting_started_with_Remote_Services
15. ProSyst Software GmbH. (2010) www.prosyst.com. [Online]. http://prosyst.com/index.php/de/html/news/details/18/smallest-OSGi/
16. Massachusetts Institute of Technology, (2011). [Online]. http://groups.csail.mit.edu/netmit/IMDShield/