International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:3, No:3, 2009

Software Architecture and Support for Patient Tracking Systems in Critical Scenarios

Gianluca Cornetta, Abdellah Touhafi, David J. Santos, and José Manuel Vázquez

Abstract—In this work a new platform for mobile-health systems is presented. System target application is providing decision support to rescue corps or military medical personnel in combat areas. Software architecture relies on a distributed client-server system that manages a wireless ad-hoc networks hierarchy in which several different types of client operate. Each client is characterized for different hardware and software requirements. Lower hierarchy levels rely in a network of completely custom devices that store clinical information and patient status and are designed to form an ad-hoc network operating in the 2.4 GHz ISM band and complying with the IEEE 802.15.4 standard (ZigBee). Medical personnel may interact with such devices, that are called MICs (Medical Information Carriers), by means of a PDA (Personal Digital Assistant) or a MDA (Medical Digital Assistant), and transmit the information stored in their local databases as well as issue a service request to the upper hierarchy levels by using IEEE 802.11 a/b/g standard (WiFi). The server acts as a repository that stores both medical evacuation forms and associated events (e.g., a teleconsulting request). All the actors participating in the diagnostic or evacuation process may access asynchronously to such repository and update its content or generate new events. The designed system pretends to optimise and improve information spreading and flow among all the system components with the aim of improving both diagnostic quality and evacuation process.

Keywords—IEEE 802.15.4 (ZigBee), IEEE 802.11 a/b/g (WiFi), distributed client-server systems, embedded databases, issue trackers, ad-hoc networks.

I. Introduction

IN the actual scenario of telecommunication systems, mobility is one of the main pillars for the development of new technologies and applications. In this context, wireless networks are experiencing amazing technological improvements, prompted by the need to eliminate cables in many applications such as computer and peripherals connections, network and internet access, etc.

Wireless networks may be divided into two categories: infrastructured and infrastructureless (ad-hoc networks). In infrastructured networks, the remote terminals communicate using fixed access points. An access point may be connected to a wired network (e.g. a LAN), and mobile terminals may access the resources of the wired network through this access point. This is the case of wireless networks deployed in office, hotels and airport facilities as well as wireless networks for domestic applications. Nevertheless, the operation of such infrastructured networks, relies on the existence of a previous infrastructure, namely, access points to cover the area where the service has to be offered, a wired network with shared resources and access to other networks. This means that an infrastructured network must be planned, installed, configured and maintained in order to offer a service. All of these requirements shrink the number of applications this kind of networks may support.

On the other hand, situations exist in which it is not possible or preferable to install a network infrastructure. Consider, for example, all those situations in which time constraints, hostile environment, or transitory needs do not justify the investment necessary to build a fixed infrastructure. These kinds of situations are typical in rescue scenarios, natural catastrophes, military operations, etc. Information interchange among several mobile terminals in these circumstances requires the automatic formation of a wireless network that relies exclusively on the available mobile terminals, without the need of any previous installation or configuration. These are the so-called wireless ad-hoc networks [16], [15]. This kind of networks supplies enough flexibility that allows the implementation of mobile applications for on-the-field rescue missions. Hence, this is a field in which technology development and the subsequent applications development, disclose a broad range of promising applications that will make even more effective rescue tasks in the future.

Under the technological innovation view point, the design of this kind of network is still challenging due to several hard-to-tackle problems such as: power consumption, routing and security, still object of intensive research. The field of civil applications of the ad-hoc network is as broad as the military one, limited only by the creativity of the designer. Among the possible applications are worth mentioning: remote monitoring and civil protection (fire, flood, chemical contamination and radiation monitoring, etc.). Nonetheless the use of such networks is not only limited to remote monitoring applications; there are several other application fields such as: domotic and ambient intelligence, asset tracking, RFID, industrial monitoring and control, etc.

So far, the broad application potential of wireless ad-hoc networks has been explored. However, this work will deal with the application of such networks to the design of an efficient medical information system. Today, the main goal in the design of a medical information system is the implementation and the deployment of solutions relying on telemedicine platforms [10], [14]. Such platforms would allow remote monitoring and diagnostics of patients with chronic pathologies who need a constant medical care, but that for different reasons cannot regularly attend medical facilities. Such a system must necessarily rely on wireless technologies

Gianluca Cornetta, David J. Santos and José Manuel Vázquez are with the Escuela Politécnica Superior, Universidad CEU-San Pablo, Madrid, email: gcornetta.eps@ceu.es, dsantos@ceu.es, jmvazquez.eps@ceu.es. Abdellah Touhafi is with the Erasmus Hogeschool, Brussels, email: abdellah.touhafi@docent.ehb.be.

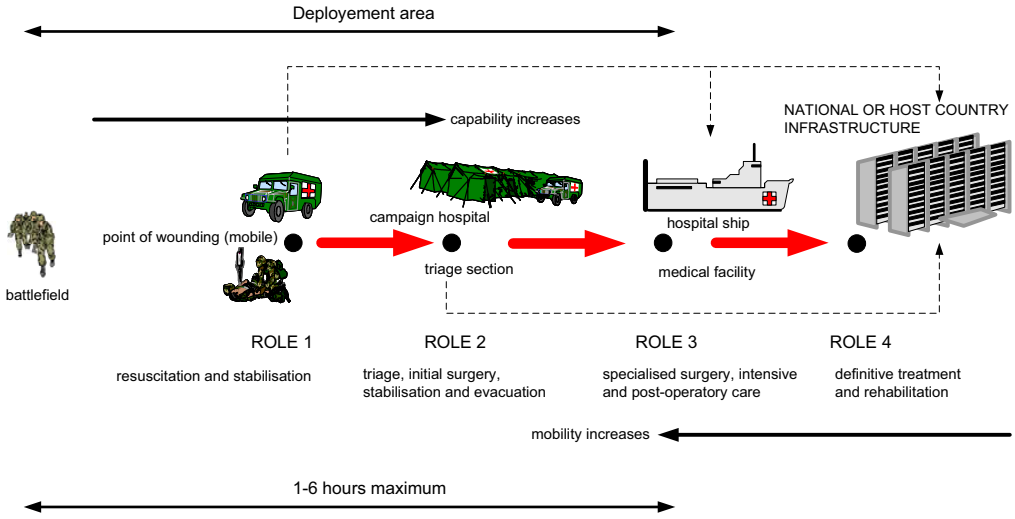

Fig. 1. NATO Campaign Logistics and Evacuation Process.

and mobile devices with reduced weight, power consumption and costs. Furthermore, all the devices that participate in the monitoring task must be able to form a local network capable to connect to the global network. The design of a system with these characteristics is not easy and implies the integration of different microelectronic, communication and information technologies, without loosing the sight of the specific application the system is intended for. Consequently, the information type (text, voice, digitized images, and so on) put constraints on the network bandwidth and the internet connection, the characteristics of the system repository and the local database, etc.

This work describes the architecture of a mobile health (m-health) telemedicine system relying on a wireless network hierarchy of ad-hoc devices with a special emphasis on the design and operation of the graphical user interface and system software. The network hierarchy allows the system integration with the pre-existing e-health infrastructure, as well as the capability to provide remote medical assistance and a reliable framework for tracking patients and victims. Such system relies on the possibility to access, with a PDA or MDA running a specialized software, through the wireless network, to the patient clinical information that is stored on an ad-hoc device called a MIC (Medical Information Carrier). A MIC is a radio-frequency device operating in the ISM 2.4 GHz band and complying with the IEEE 802.15.4 (ZigBee) specifications for MAC and PHY layers. The MIC has the capability to form dinamically an automonous mesh network either with similar devices or with the upper levels of the network hierarchy. Wireless ad-hoc networks make possible this kind of applications thanks to their flexibility and their capability to work without any previous infrastructure. In addition, these networks have also the advantage to be easily complemented with other telecommunication services to allow the connection to a stable infrastructure. In the architecture described in this work, for example, the ad-hoc network deployed on the field can communicate with the campaign hospital or other support means using a WiFi (IEEE 802.11 a/b/g) access point.

The rest of the paper is strucutred as follows. Section II describes the application context of the proposed system, namely, the NATO evacuation chain. A good understanding of the problems related to campaign logistics, troops and resources deployment in the operation area, as well as operational conditions and restrictions concerning means and people, is of paramount importance in order to define system functional hardware and software specifications. Section III deals with client and server software architecture. System wide hardware requirements as well as network architecture and the actors involved in the evacuation process, are used as a starting point to refine software tasks and requirements. In addition, all the design choices concerning system and network architecture are discussed and justified. Moreover, functionals requirements of the applications that run either at the client-side or at the server-side are discussed as well. Due to the particular nature of the problem, a specific server software is not necessary, since an issue tracking system [13] has all the capabilities for the server to operate correctly and efficiently support the demands coming from the different clients. For the aforementioned reason, an off-the-shelf application suitably modified to be adapted to the target scenario has been chosen. The modifications have been necessary to adapt the underlying database to the problem of patient evacuation. In Section IV, possible drawbacks of the proposed solution are analysed and discussed thoroughly. In addition, improvements for future implementations are proposed. Finally, Section V, summarizes the main achivements and reports concluding remarks.

II. The Application Context: NATO Evacuation Chain

The main goal of this development is solving a concrete problem within a framework of broader extension and complexity, namely campaign logistic and telemedicine applied to

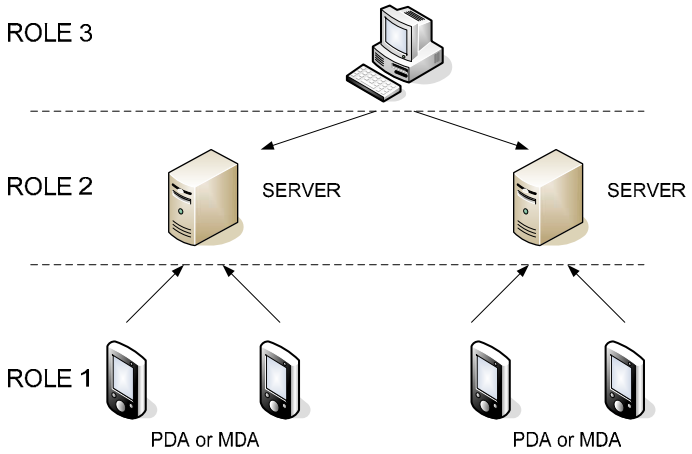

Fig. 2. Distributed Client-Server Architecture with Servers Located at Role Two.

supporting troops and rescue corps deployed in combat areas. In its essence, the principal aim is the exploitation of the last recent advances of the information and communication technologies, in order to provide medical care to troops and personnel deployed in hostile or hard-to-reach areas. On the other hand, the specific aim is the improvement of the information flow among the various agents that may take part or collaborate during the whole evacuation process, as stated in the NATO different protocols and joint doctrines in ratification phase [1], [2], [3].

The NATO campaign medical care process relies on a hierarchy, where each level represents a concrete role. As shown in Fig. 1, NATO directives consider the care capability of the different roles, starting from the basic care and stabilization capabilities of the lower levels (e.g. those closer to the operation area), till the intensive care and rehabilitation capabilities available only at he upper levels of the roles hierarchy (e.g. those farther from the combat area). In other words, the lower roles are characterized by a high mobility, that is sacrificed, as the role level increases, for a larger and more sofisticated operation capability.

At the first role of the evacuation chain, casualties triage and classification is carried out, as well as resuscitation and stabilization. Depending on the urgency level, patient care may be provided either at the same role, or at a higher one. Urgencies may be divided into extreme, absolute, or relative. NATO evacuation protocols consider which role in the evacuation chain is enabled to treat these different kinds of urgencies and up to which level evacuation should be carried out. Preventive evacuation is taken into account as well; for example, an extreme urgency will be managed only by a surgery team at role two, whereas an absolute urgency will only be managed by a campaign hospital at role 3. NATO protocols also consider the cases in which a patient must be moved to the national or host country infrastructure to proceed, for example, to his rehabilitation.

The system described in this work pretends to provide information support to the medical personnel that operates at the lower levels of the roles hierarchy, namely roles 1 and 2; since at the moment a widely-accepted solution or technological platform that can comply with the quality, security, and functionality standards demanded by the extreme conditions in which medical personnel must operate at those roles has not been developed yet. Nevertheless, it is important to point out that, although the system proposed in this paper has been designed to satisfy the needs of a specific community, e.g. the military forces that operate within the NATO, its hardware and software architecture may be easily adapted to a broad range of applications, from civil telemedicine, natural catastrophes and terrorist attack victims tracking, till logistic, asset tracking, and remote monitoring without requiring significant modification, development and industrialization costs.

A. Medical Information

NATO standards require to keep a record for every casualty by opening a MNFMC (Multi National Field Medical Card). This card must be opened by the medical personnel deployed in the point of wounding or in a triage section and contains the information relative to the first aids provided to a patient as well as his actual state and the care required. The medical personnel deployed on the field uses his MDA to send the card with his diagnostics to the upper hierarchy roles. The different modifications to the card informations, as well as the changes of patient status should reside on a system that may control this information. Furthermore, this information should be always available either to the recipient roles, or to the medical personnel on the field who generated the card, as well as to all the higher roles that may need it.

B. Events

An event is defined as every change notification of the situation among roles. For example, an event may be a demand of asistance, an evacuation order, a demand for assesment, etc. These events are directed to all the agents involved in a diagnostics or an evacuation process. Event forwarding and dissemination is managed by an assignment system, that can be accessible by the higher hierarchy levels if needed.

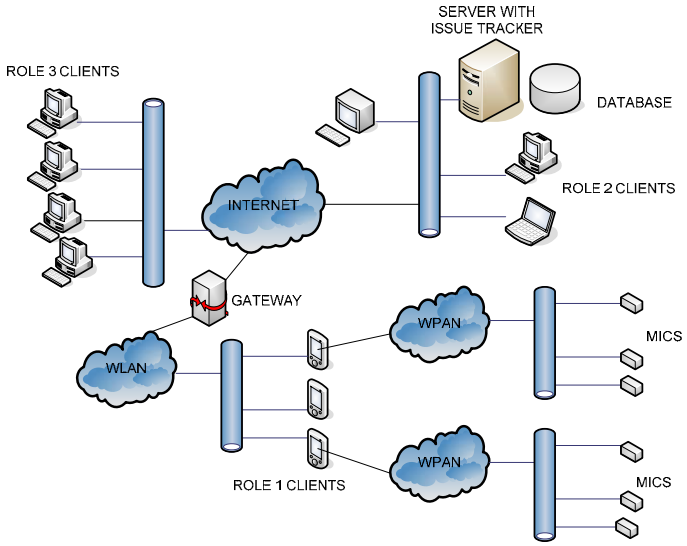

Fig. 3. System Architecture.

C. Basic System Requirements

The software solution is not a simple communication or messaging system, since the problem to tackle presents some peculiarities that requires very specific solutions. For example, an history for each event should be made available to all the units at role 3, to allow them to keep statistical analysis. Moreover, the system should allow different physicians to participate simultaneously on a same case.

The proposed system relies on the client-server architecture depicted in Fig. 2 with a plurality of light-weight clients, namely, clients that incorporate little or no logic and that rely on the server services for all their processing tasks.

In the architecture depicted in Fig. 2, an independent server is located in each element belonging to role two, and each server has its own local clients. An element belonging to role 3 may access as a client to all the role 2 servers. With this design choice, every element at role 1 can access only to the server installed in the role 2 element they depend on, avoiding information and event duplication. However, there is an increased complexity in the following aspects:

1) it is necessary an independent installation, administration and maintenance for all the servers located in the differents role two elements;
2) the role 3 should act as a client of as many servers as the number of role 2 elements.

Another drawback may be the increased possibility of service losses, since the servers are located in areas potentially less safe than at role 3. Nevertheless, the security level of a role 2 is sufficiently high so as to guarantee a very low risk of service loss. In addition, this system shortcoming is counterbalanced by the fact that an hypotetical and very unprobable service loss of a server, will not affect to the services available to all the other units depending on different servers.

Finally, the system should provide the capability to store the information and the events in a repository easily accesible by all the system users. Users may add information to an event and dispose of it whenever they want. Messages among actors are not directly issued, but they are stored in the repository that is also in charge to manage their coherence and order using the issue tracking system.

III. System Architecture

Fig. 3 depicts the architecture of the proposed system. The client-server system relies on a networks hierarchy formed by heterogeneous devices operating with different wireless standards (basically IEEE 802.15.4 and IEEE 802.11 a/b/g) that may coexist in a same area. Medical information is stored in ad-hoc devices called MICs (Medical Information Carriers) operating with IEEE 803.15.4 standard (ZigBee). A MIC is a low-power light-weight device operating with coin-size litium batteries; for this reason the minimization of analog components power consumption is crucial [5], [6]. In addition, particular care must be devoted to the simplification of software and firmware on the MIC side in order to simplify hardware architecture and to reduce embedded memory requirements. MICs can communicate with a PDA or MDA by means of a specialized communication software. Such devices work as relays among MICS and the upper levels of the network hierarchy of which MDAs and PDAs are clients. This work will basically deal with the client and server software architecture.

Situations may exist in which it is necessary to establish a client-server relationship under very special and unconventional conditions. One of these situations may occur with casualties or patients in conflicting areas. In this situation, on one side it is necessary to have patient clinical information readily available and, on the other side, is necessary to process and transmit it in secure conditions to a higher level (in this specific case a server), where specialized personnel has to take relevant decisions. In such conditions it is of paramount

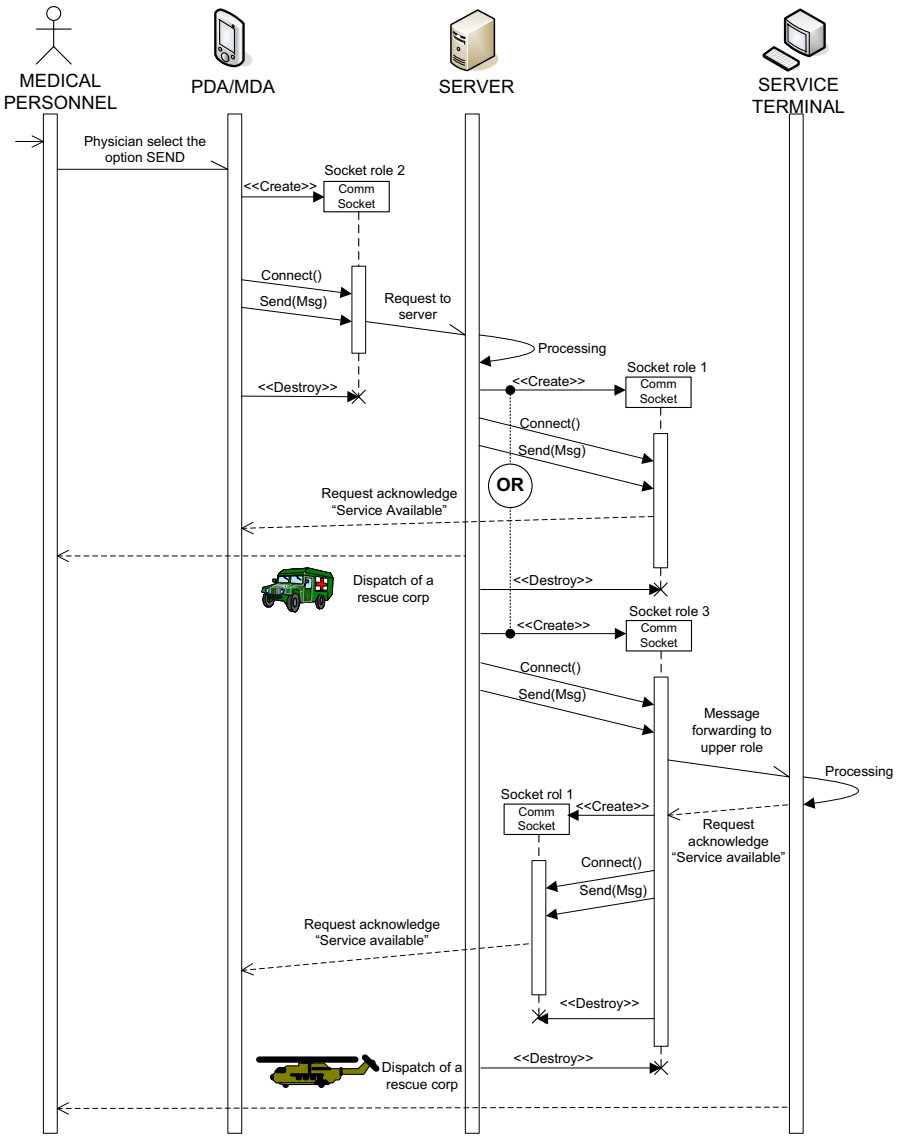

Fig. 4. Simplified Sequence Diagram Depicting System Operation.

importance to know the scope of the system functionalities to be implemented either at the server-side or the client-side, such as, for example, the possibility to establish a secure communication channel not accessible to unauthorized personnel. Fig. 4 shows the simplified sequence diagram that describes system functionality as well as the actors involved in the evacuation process at different roles. The evacuation process starts with a request sent through a mobile terminal (PDA or MDA) by the medical personnel operating at role 1. Once the sending option has been selected, a communication socket with the upper level is created. The server (located at role 2) analyses the service request and depending on service availability decides whether opening a communication socket with role 1 or role 3. In the case the personnel deployed at role 2 is unable to attend the service request, the demand is forwarded to role 3 that will proceed to communicate to the underlying level whether it is able to satisfy the request or not. If there is service availability at role 2, a communication socket with role 1 is opened and the service request will be acknowledged.

A. Client Interface

The light weight client is designed to run under a Linux-based OS and relies on a three-layer architecture: a presentation layer, an application layer, and a data layer. The presentation layer is implemented using Trollech (now Qt-Software) Qt-embedded libraries [11], whereas the data layer relies on SQLite embedded database [9]. The application layer implements a reduced set of functionalities to manage database and communications with the server. SQLite engine is suitable for the proposed application since the local database is very

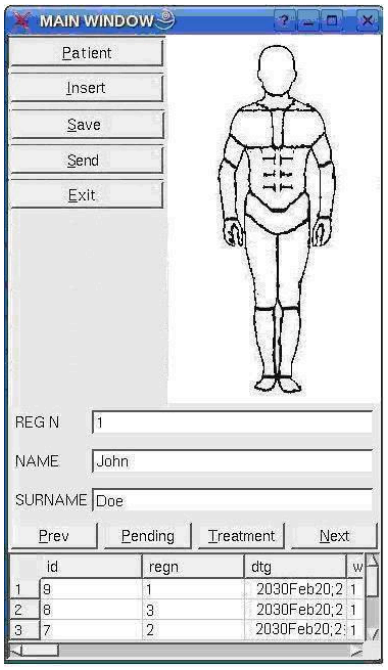

Fig. 5. Main Window of the Graphical User Interface.

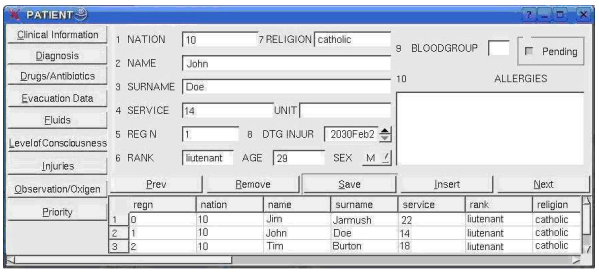

Fig. 6. Patient Management Window.

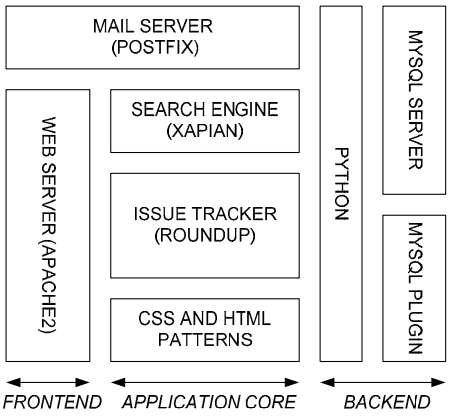

Fig. 7. Server Software Architecture.

small. In addition, SQLite database management system is very compact (500 KByte approximately) making it ideal for embedded applications with little hardware resources.

When the application is started, the main window (depicted in Fig. 5) appears. The window is diveded into three parts. The database navigation interface is located at the bottom and offers the possibility to browse the database entries by using the navigation buttons or the scroll system. The upper field are used to register a new patient in the database. The system also offers the possibility to show the list of evacuation-pending patients, the prescribed treatments and when they where administered the last time.

The command buttons are located at the upper left of the window. In its present version, the client software allows five options: the input of patient clinical data, the insertion of a new database entry, the closing of a database transaction, the data sending through the WiFi network, and the application exit option. In the future the system will be also able to automatically fill and update its local database by reading the clinical information stored in a MIC through the ZigBee network. Finally, the upper right part of the main window depicts a human body silouhette in which it is possible to mark the wounded points.

When the option “Patient” is selected, the form depicted in Fig. 6 appears. This interface allows the insertion into the database of the basic patient information. The database browser is located in the lower part of the window. The interface allows to insert or remove an item from the database. A transaction ends by selecting the option “Save”. In the left part of the window other buttons are located that allow the opening of new forms to complete the patient medical card (diagnosis, evacuation priority, injuries details, and so on).

B. Server Interface

Server software has been designed to run under a Linux OS and has the same three-layer architecture of the client software, namely a presentation layer, an application layer and a data layer. The presentation layer relies on a ligth weight web interface, whereas the data layer is based on MySQL database [7]. Finally, the application layer is mainly formed by the issue tracker engine. The system described in this works relies on Roundup [8] as the issue tracker. In order to work correctly, Roundup needs the system to provide certain basic services as also depicted in Fig. 7. Xapian search engine [12] is an optional component and it is worth using it when the number of database entries exceeds 5000 [8].

An issue tracking system is a software specifically conceived to manage lists of reports and events (tickets) by means of state assignments. A ticket represent an issue to be controlled or solved. In addition, each ticket carries a content associated to the related issue, a state and an assignment. The information may be available at anytime to all the users that participate in the solution of a particular issue. Each authorized user may change ticket contents, state and assignment. The system allows to keep track of the history of each ticket including all its change of state, assignment and content. All these features make an issue tracking system ideal to manage the problem of patient tracking and monitoring along the evacuation chain, where several actors with different roles are involved.

Among all the issue trackers actually available, Roundup

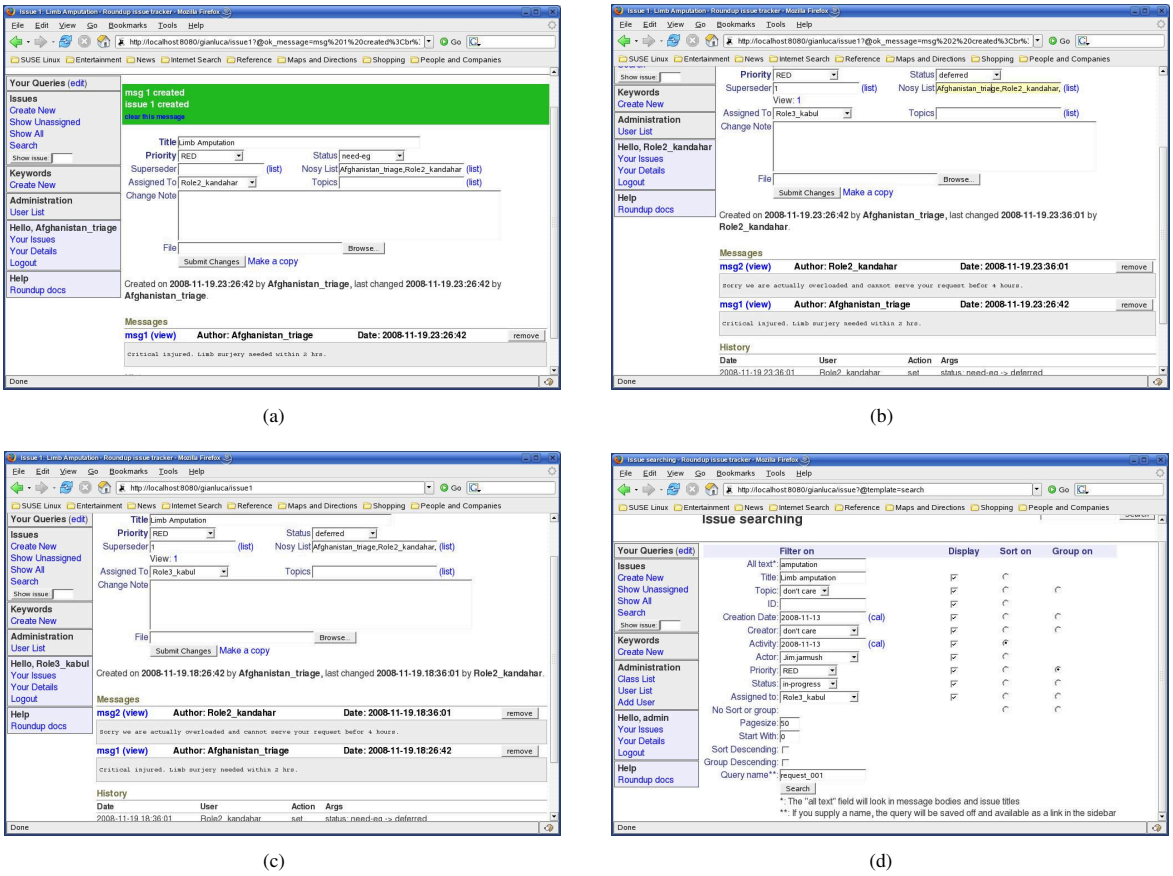

Fig. 8. An Example of Issues Flow:(a) Ticket Creation at Role 1; (b) Reply from Role 2; (c) Notification to Role 3; (d) Issue Search in Local Database.

is particularly appealing since it supports several types of client (web, e-mail, command-line interface) and it is easy to modify since the application core is decoupled from the interface and database management systems. Any type of modification basically implies to operate on a set of scripts that allow changing the configuration of the user, database and system interfaces. Finally, the use of Python as the programming language makes Roundup virtually independent of the hardware platform and operating system.

In its basic configuration, Roundup allows three different kinds of users: the system administrator, the registered user, and the anonymous user. The administrator has all the privileges of a normal user, and, in addition, the possibility to add new user in the system, assign privileges, and manage the local database through the web interface.

Finally, Fig. 8(a)–(d) show how the system keeps track of a possible issue flow during the evacuation process:

1) A system user generates a ticket and associates it to a given event, then fills the associated form with relevant information and assigns to it the state “to be solved” and a target user. For example, a user located at a role one, generates a ticket and submits it to a role two user (see Fig. 8(a)).
2) The target user receives the ticket, can change its content and add additional status information. He also also can forward the ticket to other users in the case he is unable to serve the request. In the case depicted in Fig. 8(b), role 2 cannot serve the request issued by role 1, for this reason changes ticket state to “delayed” and reissues the ticket to other roles.
3) The updated ticket is received both by the user that first generated it and by all the new users that are specified in the message destination field. System notifies the reception of a new ticket to all the users involved in the evacuation process, specifying time and date of creation, message object, users involved and so on (Fig. 8(c)). As soon as the request can be satisfied by one of the users involved in the process, the ticket owner in that moment mark ticket state as “solved”.

Even if a given issue has been solved, the associated ticket does not disappear from the system. In fact, the system keeps a record with the history of each ticket issued. Every new contribution, assignment or change of state is permanentely stored in the local database, so that the role three user on which all the units in the underlying roles depend, could be able to perform a statistical analysis, draw up information or even send orders to the lower hierarchy levels. For this reason, the system allows to perform any kind of query to the local database as depicted in Fig. 8(d).

IV. Discussion and Future Work

The main goal of this research has been finding and evaluating a model and its software implementation to manage and optimise the campaign logistics and the evacuation process of troops deployed in the operation area. All the possible changes and improvements related to the future deployment of the application basically depend on system workload either at the client-side or at the server-side.

The application on the client side has been developed under a Linux OS with a kernel that embeds only a limited number of services in order to reduce memory requirements. The user interface has been programmed in C++ and using the functionalities provided by the Qt-embedded libraries. The result is an efficient and appealing light-weight client. Nonetheless, in the future it could be necessary to increase the number of supported platforms. For this reason, the developed interface is going to be migrated to Android [4]. This implies using Java as the development language. The advantages of such a choice are evident, since on one side it is possible to decouple the application from the underlying operating system, and on the other side it is possible to exploit the APIs and development libraries provided by Android to improve system functionality. In fact, Android supports a huge number of platforms (PDAs, Smart phones, mobile telephones), hardware devices (screens, keyboards, cameras, sound devices, flash memories, etc.), communication standards (WiFi), and development libraries (SQL, OpenGL, SSL, WebKit, SQLite, and so on).

On the server side, the tests have been carried out using a PC and a back-end based on MySQL, which may impose some future limitations in the case of large system workloads. The use of Python as the programming language does not impose serious limitations if the migration to another hardware platform and operating systems is necessary; however, it should be necessary to program a plug-in to allow the interaction with a different database that could help to overcome both the security and workload limitations of MySQL. In addition, in the actual development the use of a logging system was not considered. This, indeed, is not a serious problem, because Roundup already includes this functionality and the only additional step consists in suitably modifying its configuration file. It could be also desirable to improve the user interface on the server side. In the actual development, the design choice was to use the web interface that Roundup provides by default. User interface improvements may be directed toward different directions: improvements of the web interface modifying the Cascade Style Sheets (CSS) and the HTML patterns provided by the application, or embed full custom solutions based on Roundup command-line output.

V. Conclusion

This work has been focused toward the development of a software solution of rapid deployment and installation at the lower roles of the NATO evacuation chain. Within these levels, two kinds of abstractions have been used: the former devoted to the management of the medical cards that NATO standards mandate, the latter devoted to the management of the events occurred during the processing of a medical urgency.

Since all the problems related to the upper levels of the roles hierarchy have already been solved, this work has never dealt in any moment with design issues concerning such kind of problems. The main goal of all the developments described here has been fundamentally the creation of a model that could represent with sufficient accuracy real operation scenarios, and find a software solution capable to satisfactorily resolve the main design issues. The proposed solution has been carefully chosen within all the possible implementations (from the applications to be used, till the choice of the programming languages, operating system, and so on), and, in addition, a particular relevance has been given to the tests that have been automated. It is important to remark once again that the proposed system can be used without too many modifications also for civil telemedicine. The adaptation of the system architecture to civil scope would uniquely require a redesign of the deployment model, since the operation would be still the same. In general, any system whose state changes are event-driven is suitable to be managed with a software solution based on the proposed architecure.

References

[1] NATO STANAG, *AJP 4.10(a): Allied Joint Medical Doctrine*, NATO, 2006.
[2] NATO STANAG, *AJP 4.10.1: Medical Planning*, NATO, 2006.
[3] NATO STANAG, *AJP 4.10.2: Medical Evacuation*, NATO, 2006.
[4] Google Inc., *Android. An Open Handset Alliance Project*, available at `http://code.google.com/android`, 2008.
[5] Cornetta, G., Santos, D. J., and Godara, B., *A Sub-mW Low Noise Amplifier for Wireless Sensor Networks*, Proceedings of the World Academy of Science Engineering and Technology, 34(10), pp. 835-838, 2008.
[6] Cornetta, G., and Santos, D. J., *Low-power Multistage Low Noise Amplifiers for Wireless Sensor Networks*, International Journal of Electronics, 96(1), pp. 63-77, 2009.
[7] MySQL AB, *MySQL. Open Source Database*, available at `http://www.mysql.com`, 2008.
[8] Jones, R., *Roundup Issue Tracker, Project Hompage*, available at `http://roundup.sourceforge.net`, 2008.
[9] Hipp, Wyrick & Company Inc., *SQLite, Project Hompage*, available at `http://www.sqlite.org`, 2008.
[10] Prabaharan, K., Lavanya, J., Goh, K.W., Kim, Y., and Soh, C.B., "Distributed Architecture Toward Telediagnosis", in Proceedings of *1st Transdisciplinary Conference on Distributed Diagnosis and Home Healthcare*, pp. 105-108, 2006.
[11] Qt Software ASA. *Qt for Open Source C++ development on Embedded Linux*, available at `http://trolltech.com`, 2008.
[12] Xapian, *Xapian, Open Source Search Engine Library. Project Homepage*, available at `http://www.xapian.org`, 2008.
[13] Yee, K.-P., *Roundup. An Issue-Tracking System for Knowledge Workers*, retrieved November 21st, 2008, from `http://zesty.ca/sc-roundup.html`, 2000.
[14] White, C.C., Fang, D., Eung-Hun K., Loeber, W., and Yongmin, K., "Improving Healthcare quality through Distributed Diagnosis and Home Healthcare", in Proceedings of *1st Transdisciplinary Conference on Distributed Diagnosis and Home Healthcare*, pp. 168-172, 2006.
[15] Bulusu, N., and Jha, S. (Eds.), *Wireless Sensor Networks: A System Perspective*, Norwood, MA: Artech House, 2005.
[16] Zhao, F., and Guibas, L., *Wireless Sensor Networks. An Information Processing Approach*, San Francisco, CA: Morgan Kauffman, 2004.