Context-aware mobile agents for decision-making support in healthcare emergency applications

Burstein, F.[a], Zaslavsky, A.[b], Arora, N.[b]

[a] School of Information Management & Systems

[b] School of Computer Science and Software Engineering
Monash University, Australia

{frada.burstein, arkady.zaslavsky}@infotech.monash.edu.au

Abstract

This paper applies agents to the highly dynamic and variable context of healthcare emergency decision-support domain. More specifically we advocate the use of mobile agents to support the deployment of an ambulance service in real-time. We presented an implementation of the proposed agent based architecture, which was based on the specific functional and non-functional application requirements set out based on thorough analysis of literature. We also created an illustrative emergency scenario in order to demonstrate the validity and feasibility of our proposed model. From the evaluation of the implementation we were able to identify some of the major technical advantages it has to offer as well as challenges one needs to address in similar attempts. We choose mobile agents as the key enabling technology because they offer a single, general framework in which large-scale distributed real-time decision support applications can be implemented more efficiently. Healthcare was chosen as the target application domain to emphasize both the benefits derived from the exploitation of mobile agent paradigm in this domain as well as to demonstrate the benefits of the proposed approach in the highly uncertain context.

1. Introduction and background

The role of context for better decision-making has been a topic of significant research effort over the years [BRE99, BRE98]. The more ambiguous decision situation, the bigger advantage is there to provide the decision-maker with support in modeling and learning the context for better understanding of decision parameters and implications of selecting particular decision alternatives. Most of real-time decisions are mainly based on access to real-time data or information that can support the decision-making process. In critical situations such as healthcare management or national defense, there is a risk of making incorrect decision based on incomplete or outdated information. In particular, assisting the decision makers in gathering and analyzing the data on damage assessment, meteorology, demography, etc. and providing healthcare decision support in the case of disaster and recovery represents a very important and innovative area for research from the context modeling perspective.

Context-awareness refers to knowledge and understanding of the surrounding environment within which the decision support system has to operate. For instance, available bandwidth, device profile, and last completed transaction are all context attributes [PAD04]. While notion of context is interpreted by computer science researchers somewhat differently, we subscribe here to the definition of context as adopted by the pervasive computing research community. Namely, context is defined as "the set of environmental states and settings that either determines an application’s behaviour or in which an application event occurs and is interesting to the user”[CHK00]. Within the context of real-time decision support, it is imperative to equip the decision maker with full information based on best possible context model, eg., [Bre98] or [PAD04]. For instance, awareness of user confidence in making a decision (context attribute), currency of available information measured on the basis of the last syncing of a mobile device, or approximate time to et a patient to a hospital where he/she will get best care, are all context attributes. Time-critical decision making problems require establishing system architectures that allow support infrastructure for wireless connectivity [PAD04], network security [ALW02] and parallel processing of multiple sources of information.

This paper brings together context awareness, mobile software agents and decision support concepts by proposing a real-time DSS for the highly dynamic and variable healthcare emergency domain. More specifically we demonstrate the use of mobile agents to support the deployment of an ambulance support services in real-time. We presented an implementation of the proposed agent based architecture, which was based on the specific functional and non-functional application requirements set out based on thorough analysis of literature. We describe an emergency scenario to demonstrate the validity and feasibility of our proposed model. We were able to identify some of the major technical advantages as well as challenges one needs to be aware of in similar attempts. We choose mobile agents as the key enabling technology because they offer a single, general framework in which large-scale distributed real-time decision support applications can be implemented more efficiently.

2. Agent technology for context-aware decision support in healthcare emergency context

Mobile agent technology provides an attractive and important technique for building large-scale distributed applications in heterogenous computing environments. A mobile agent can be viewed as an autonomous program that has the ability to transport itself between the nodes of a network entirely under its own control, carrying with it the data and execution state required to resume execution at the destination host from the point it ceased on the original host [Lan98]. Therefore, it is the agent that decides 'when to move', 'where to move', 'what to execute' and 'how to execute it'.

Due to their autonomous and active nature, mobile agents offer several benefits over such traditional technologies as client-server, to enable a mechanism for context-aware decision support. As they are able to encapsulate protocols needed to perform a given task, agents tend to be relatively easy to customize and can rapidly adapt to changing user requirements [DiS01] and run-time context. Apart from providing mobility, agents posses the unique characteristic of adapting to changes in their execution environment and hence have a higher chance of survival and achieving the application objectives over a large, distributed and heterogeneous network when compared against more traditional techniques [LaO99]. Some of current pervasive computing research projects combine agent technology with context awareness, eg, [PAD04]. Through this convergence mobile software agents are capable of embedding light-weight context reasoning engine and do preliminary processing where context originates.

The healthcare sector is not only widely distributed and fragmented but it also exhibits a high degree of heterogeneity with strong local autonomy [GSJ+01]. The current lack of standards across different institutions prevents from using a single software solution to support a co-operative working environment [BBD+03]. It is also important to realize that the domain is very information-intensive [GGH00] and mission-critical. The large volume of medical data with its complexity and variety makes it impractical to be transported between 'point-of-source' and 'point-of-delivery' as in the traditional client-server case. In addition, the hospital environment by default is 'highly mobile' [AMM03] with caregivers constantly on the move. To meet the varying information and resource needs of these personnel and yet be able to support their physical mobility requirements, the mobile computing technique seems to provide an ideal fit. The flexibility offered by this paradigm, however, comes with some major challenges that are uncommon is static counterparts (for example the key issue of low bandwidth and highly unreliable connections). Thus, the healthcare domain is quite unique in that it brings with it several requirements and constraints, which have great practical relevance for any application to be successfully developed in such environments. We advocate the pro-active, goal driven and autonomous nature of mobile agent technology as a mechanism for the highly mobile, dynamic and variable context in real-time decision support for ambulatory services environment. We have used the term 'ambulatory' to refer to the highly mobile, dynamic and variable context of medical front-line support areas such as ambulance and paramedic services [MoB97]. This differs from traditional definition of the term, which generally means 'able to walk' or 'moveable' [MW96]. Henceforth, when we use the term 'ambulatory' we are actually using it in the context of healthcare delivered by ambulance services.

Tung and Sankaran [TuS97] proposed an Internet-based framework for supporting collaborative work across a geographically dispersed set of cooperating mobile software agents. Agents acted as tele-workers and were capable of distributed decision making by collaborating and negotiating with each other. By adopting a proactive approach rather than a reactive one, these agents were able to execute tasks on behalf of their users in an intelligent and autonomous manner. Due to their adaptive nature they were also able to continuously search for conflicting problems during their execution and could automatically intervene when necessary in order to achieve a consensual solution.

We advocate the use of mobile software agents to support the deployment of an ambulance service in a real-time. Agents are not only used to process and retrieve the required information from various hospital information systems (HIS) but are also involved in coordination activities and decision support. The prototype we developed based on the proposed architecture did not intend to solve all the problems associated with the use of agents. Instead some of the fundamental properties exhibited by agents and their ability to overcome the static nature of current distributed networks, makes them very well suited for developing applications to maintain front-line-support areas such as paramedics and ambulance services.

The review of agent-based application showed that although agents have been deployed in a wide variety of scenarios ranging from information filtering/retrieval [LMP+01] to resource coordination/deployment [MSE+03] to supporting mobile users [WMB03], but to be best of our knowledge, nobody thus far has utilized mobile agents for ambulatory support applications. Although, Morton and Bukres [MoB97] did describe an architecture to support ambulatory services for the Wishard Memorial Hospital, this architecture however, was based on standard mobile computing model and needed the additional support of a base station agent (BSA) which acted as an intermediatory to support disconnected entities. On the other hand, by utilizing mobile agents, our proposed prototype can handle any intermediatory entities but also caters to a host of other requirements inherent to such environments. It is also important to note that our application does not intend to replace current legacy systems but rather integrate and complement them. It provides a means for exchanging information in real-time between the various parties involved, in order to support decision making by integrating the factors of mobility and time.

The reason for choosing ambulance services as a target context was two fold. Firstly, the review showed that although mobile agents have been deployed in a wide variety of application scenarios, nobody thus far has utilized them to maintain medical real-time decision-support area such as paramedics and ambulatory services. Secondly, the very nature of the domain, with its mobility and variability, provided an ideal situation where the use of agent technology could be a superior alternative.

3. Proposed Architecture Requirements and Design Principles

In this section we summarise key requirements for any application used for front-line-support medical personnel. These requirements have been formulated with a software engineering perspective and are crucial for our application's success.

3.1 Support for Heterogeneity

The health care sector is widely distributed and fragmented and exhibits a high degree of heterogeneity with strong local autonomy [GSJ+01]. The lack of standards across organizations means that every legacy HIS is unique in itself and is tailored to suit the contexts and demands of the given institution. Individual medical records however, are highly fragmented and can be scattered across multiple institutions. As we have earlier identified that providing medical care is essentially a group activity in which multiple parties need to coordinate their actions (which could be long standing) and share information not only between departments but also across organizations. Therefore, the ability to integrate the heterogeneous data sources and provide the user with a single, unified data view in a transparent manner is essential.

3.2 Disconnection Support

Point-of-care delivery of information is vital for any application in the healthcare environment [AMM03]. In order to meet the varying information and resource needs of mobile personnel and yet be able to support their physical mobility requirements making the mobile computing paradigm an obvious choice. An increasing number of medical practitioners are using personal digital assistants (palm-tops, tablet PC's etc.) with wireless connectivity in order to have access to vital medical information on the move [BBD+03]. This flexibility however comes with some major challenges. One of the key issues introduced by such environments is that, unlike the static counterparts, mobile connections tend having low bandwidth and are unreliable in nature. This implies that the uses are subjected to frequent and unplanned disconnections. This becomes especially true when an ambulance frequently moves between various points across the city. In such dynamic environments it becomes necessary to be able to asynchronously and autonomously carry on with the designated tasks even while the user is disconnected. Therefore, it is an important requirement for our application to be able to support temporary disconnections from the network and adapt its context accordingly.

3.3 Thin Client Support

As the mobile computing paradigm proliferates rapidly into the medical field alongside the existing legacy systems, it is important to realise the changes in the landscape of the computing devices it brings. Unlike stationary desktop oriented machines (PC's), mobile devices (such as phones, PDA's) are constrained by their shape, size and weight. Due to their limited size, these devices tend to be extremely resource constrained in terms of their processing power, available memory, battery capacity and screen size among others. Traditional applications however, are not designed to cater to such issues. In order to achieve an acceptable level of performance, application which target such 'thin clients' need not only be able to conserve resources but should also be lightweight enough [AMM03]. The aim of our proposed prototype is to be able to support real-time 'intelligent' deployment of ambulances based on the latest available hospital resource information. It is envisaged that our application will primarily run on a terminal onboard the ambulance. We assume that due to the nature of such an environment the on-board computer is either a PDA, carried by the paramedic or a dashboard mounted computer, which is similar in capability and resources to a PDA. Hence, the proposed application should be designed such that it is not only feature-packed enough to support the demands of the paramedical personnel but is also capable of running on resource constrained devices.

3.4 Fault Tolerance

Reliability is an essential requirement in critical application domains such as healthcare [BBD+03]. An efficient and effective level of service requires the coordination of a number of entities (some mobile, some static), physically separated by a varying degree of geographical distances. The variability of the conditions and context of this environment make it susceptible to frequent and mostly unplanned failure across the multiple heterogenous nodes.

Applications therefore should be designed in such a way that they must still remain functional even if some individual nodes or specific network connections become unavailable for an unspecified time period. Single points of failure are not acceptable. As our proposed application involves a combination of portable devices (ambulance based computers) and fixed information servers, which interact with each other using a mix of wireless and static networks, the possibility of individual failures is numerous. Thus, our application needs to be structured in such a way that it is able to gracefully handle specific environmental failures and yet remain functional.

3.5 Locality of Reference

As we move towards a 'managed care' model for health services the ability to access and process multiple heterogeneous sources of information in a transparent manner becomes a necessity. However, medical information is not only highly fragmented but its sheer volume/size makes it difficult and impractical to transport it between network nodes. Moreover, with an ever-increasing acceptance for mobile and wearable computing techniques within the domain, render it even more difficult to process large quantities of medical data on client nodes due to the restrictive nature of these devices (thin clients) and the unreliable, low bandwidth connection environments they generally tend to operate in. The method is needed, which will enable sorting through the various data sets at the source itself, identifying bits that are most relevant to the case-at-hand, then presenting this significantly smaller data set to the client. The ambulatory scenario too is set in such highly mobile and restrictive surroundings. The ability to overcome client limitations and process information closer to its source has great relevance to our application.

3.6 Scalability

Another core issue to be considered while designing an ambulatory application is scalability. A citywide ambulance service typically spans over many tens if not hundreds of organizations ranging from small specialist clinics to full-size hospitals. Each of these organizations is essentially a node in a citywide network. The notion of scalability is two fold. Firstly, our application should be able to cater to the large number of dynamic nodes changing over time (as new hospitals open and some older ones close down). Aside from the pure number of nodes, scalability also concerns the type of individual nodes very different, both in terms of functionality and resources. Our application should therefore be able to take such differences into account and scale to them with minimum effort or changes on the part of its users.

3.7 Customisability

Essentially, this requirement tags along with the scalability factor required by our application. Review of systems deployed in healthcare showed, that current hospital information systems (HIS) tend to be solutions that are tailor-made and targeted to suit the contexts of the particular hospital. This means that the domain of healthcare has traditionally been dominated by single-vendor, proprietary systems [FGG98]. The current lack of standards across organizations implies that every organization has system internals, which are very different from the other (database implementation, security policy etc.). For applications, to be able to interact with these systems they need to be adaptable enough to be able to speak the same 'language' as the systems themselves. As our ambulance-based application will need to interact with a number of hospitals in real-time, hence it needs to be designed in a way so that it is customizable enough to adapt to the changing requirements.

4. Illustrative Scenario

We illustrate the agent-based application model employing a sample scenario. Consider a situation in which a middle-age man suffers from a cardiac arrest while on his way to work. The on-lookers immediately call 000 providing details such as the location of the patient, the nature of the accident, along with some form of identification (ID) and if possible a name (we assume that the patient will have some form of ID such as a driving license or a medical insurance card on him). The dispatch centre then relays this information to the most appropriate ambulatory service, which promptly dispatches an ambulance to the scene. The paramedics use the dashboard-mounted terminal on the ambulance to filter out the group of hospitals that are in close proximity to the patient's location. They also key in the available details such as ID/name, gender and type of treatment required. Once this is done, the paramedics use the travel time to the accident scene to launch two mobile agent assistants. These agents use the data entered by the paramedics to autonomously and asynchronously traverse the various nodes of the network to carry out specific information retrieval tasks on behalf of the paramedics. Finally they collaborate their results and present it to the user in a transparent manner. We call out first mobile agent as the *'hospital assistant'*. This agent is tasked to retrieve the latest resource details form the list of hospitals assigned to it. As mentioned, paramedics' shortlist a group of hospitals based on the patient's location (they may not do this in which case the agent will visit all hospitals in the city irrespective of their distance form the scene of accident). Each hospital is essentially a node in our distributed medical environment. Once launched, the mobile assistant uses the list of hospitals it was assigned and migrates to each of the hospitals one by one. When it arrives at a destination hospital node it interacts locally with the stationary hospital agent to retrieve the latest resource consumptions details for the hospital. As part of its interaction with the stationary agent the assistant acquires information about 1) Facility type offered by the hospital (24-Hr Emergency Centre, Cardiology Ward etc.) 2) Total number of doctors present 3) Number of doctors available to attend patient at present 4) Total number of nurses present 5) Number of nurses available to attend patient at present 6) Total number of beds at the hospital 7) Number of beds available. After receiving the requested information, the assistant then visits the next hospital on the list and repeats its request. It keeps doing this till it reaches the last hospital on its list. Upon completion the agent moves back to the ambulance to relay its results.

While the hospital assistant retrieves the latest resource utilization details from various hospitals, the second mobile agent, which we refer to as the *'data assistant'* simultaneously retrieves the patient's medical history along with the latest route status for the selected hospitals. Based on the patient's ID details, the data assistant travels to a central medical repository (Medical Archives) interacts with the stationary archive assistant and acquires the medical details of the patient along with his past medical history. As part of the of the patients details the assistant acquires the 1) Name of the patient 2) Health insurance number (if available) 3) Driving license number (if available) 4) Address details 5) Known drug allergies. As part of the patients medical history the assistant acquires information about 1) Hospital/Clinic visited by the patient 2) Date of visit 3) Details of visit (medical procedures performed etc.) 4) Name of physician attended. It is important to note that for our scenario we assume that every individual has a unique identifier (ID) that is used to tag all his medical related details. If no ID details were entered by the paramedic (or were not available in the first place) the data assistant automatically skips going to the archive node and migrates directly to the road authority website for the latest situation on roads leading to each of the selected hospitals. After retrieving the relevant patient details and the necessary road updates, the agent moves back to the ambulance to relay this information. However, before both the hospital assistant and the data assistant publish their results, they coordinate their results with each other. The UI then uses this set of results to rank the list of hospitals in order of suitability. In order to rank the hospitals the following are taken into consideration 1) Current resource availability at each hospital 2) Special medical facility offered by each hospital 3) Patient's medical history with each hospital and 4) Current road status information.

By the time paramedics reach the scene of emergency their onboard terminal already reflects the list of nearby hospitals ranked form the most relevant to the current case, to the least relevant. This is information is also supplemented with the patient's medical details and current road situation for each hospital. Once at the scene the medics stabilise the patient, administer medications based on the medical history (such as allergies) and load him in. It is important to note that although the application ranks the hospitals in their order of priority, the final decision with respect to selecting the most appropriate hospital still rests with the paramedics. In essence the application provides for a decision support tool that aids the paramedic by empowering him with real-time, mission critical information at the point-of-care. The paramedics review the ranked hospitals and the ranking details associated with each one and then make their final selection. On the way to the selected hospital, a message is sent to the associated hospital agent, which then updates the relevant hospital databases with the details of the patient 'en route' to the hospital. By the time the ambulance reaches the hospital this information has already enabled the emergency room staff to prepare for the arrival of the patient.

5. Mobile Agent Based Architecture

To support the above scenario we propose an application structured around the mobile agent paradigm. Figure 2 shows the architecture that we have adopted. This architecture involves a mix of stationary (hospitals) and mobile nodes (ambulances) supported by a combination of static and wireless networks. These network of nodes communicate and share information using mobile agents. The various agents autonomously and asynchronously migrate between nodes and interact with other stationary or mobile agents in order to accomplish their assigned tasks. The application gets initiated when the paramedic selects a list of hospital based on the accident location, keys in the patient ID and clicks on the 'Launch Agent' button on the on the user interface component of the *ambulance node*. This deploys the two mobile assistants associated with the application – *'hospital assistant'* and *'data assistant'*. The hospital assistant is tasked to visit a list of hospitals identified by the paramedic. Once launched, it migrates to the various hospital nodes, interacts with the stationary *'hospital agent'* and retrieves the latest resource utilization details. The data assistant on the other hand is tasked to retrieve the patient's medical details and get the latest road status information for the selected hospitals. Once launched, it migrates to the *'medical archives node'* and interacts with the stationary archive agent to retrieve the patient medical history based on his ID. It then travels to the *'road authority web site'* and gets the latest updates on road status information for each of the selected hospital. When their respective tasks are done, the mobile agents then move back to their home node (ambulance which initially launched them). Upon arriving back at the home base, they coordinate with each other to collaborate their results before publishing it to the user interface.

The six major components, which comprise the proposed architecture are detailed in the following section.

5.1 Ambulance Node

The ambulance node is the only mobile node of our architecture. It involves an ambulance based, wireless-enabled, mobile terminal that houses our application's user interface. This mobile terminal could either be a PDA carried by the

paramedic or a dashboard mounted computer. Due to the limited nature (size, computational power, memory) of these devices, the ambulance terminal is essentially a 'thin-client', which is mainly used to provide the paramedics with a graphical interface (GUI) to be able to interact with the application. The main functionalities offered by the GUI are:
To allow paramedics to shortlist the hospitals they are interested in based on the location of emergency (for e.g. hospitals within the same suburb or neighbouring suburbs)

- To key in the available patient details (ID/name/gender)
- To specify if a special medical facility is required (e.g. sports medicine, cardiology etc.)
- To be able to launch the mobile agents – hospital assistant and data assistant in order to carry out the various information retrieval tasks.
- To display the list of selected hospitals ranked in the order of suitability in context to the current patient (as received from the mobile agents)
- To display the patient details including his past medical history (as received from the mobile agents)
- To display the latest route status information for each of the selected hospitals
- To allow the paramedic to verify the ranking details associated with any of the selected hospitals.
- When the paramedic has made his selection of the hospital, the UI should allow him to be able to send a message to the associated hospital agent in order to make a booking for the patient 'en route' to the hospital.

As the ambulance node is mobile in nature, it needs to be supported by a citywide wireless network that spans across all the medical institutions. It then uses this wireless network to communicate with the various static nodes (hospitals, web site etc.) of our proposed architecture. Wireless networks have their own set of special requirements (such as cell hopping etc.) but we do not consider this within our scope. We are only concerned with the various properties exhibited by mobile agents, which make them well suited for being deployed in such environments. In addition to providing a user interface to the application, the only other feature the ambulance node supports is an *agent system*. This system provides the basic set of functionality required by the node in order to launch and receive mobile agents.

5.2 Hospital Node

Each hospital node represents a particular hospital/clinic, which exists with the geographical area where the ambulatory services operates. The hospital node consists of a stationary hospital agent and a proprietary hospital information system (HIS). The HIS is composed of a number of different components that support the various information needs of the particular hospital. For our purpose we will consider only the databases associated with resource utilisation within the hospital. The resources are doctors, nurses and beds. In addition to the hospital agent and the HIS, the hospital node also supports an agent system. This agent system provides the infrastructure required to host and execute the stationary hospital agent as well as to be able to accept incoming mobile agents (mobile hospital assistant). When a mobile agent arrives at the node, it interacts locally with the stationary agent to acquire the required information. The stationary agent then services these requests by accessing the relevant databases of the HIS.

5.3 Medical Archives Node

The review of current medical systems showed that medical records are highly fragmented and distributed across various organizations. Although not many countries currently support the idea of a nationwide medical archive that serves as a central repository for an individual's medical details/past history, we foresee that an integration of this information is likely to occur, in the not so distant future, in order to achieve the desired service levels and cost reduction in the healthcare field.

In our proposed architecture the *medical archive node* component essentially serves this purpose. The node consists of a stationary archive agent and a database repository, which store the medical information for the entire population. Individual patient records are retrieved based on a unique patient identifier (we have used driving license number and the health insurance number for our scenario). Just like the other nodes in the architecture, this node too supports an agent system in order to accept and support the execution of agents. The role of the stationary archive agent is nothing more than a *'shopkeeper'*. In other words when a mobile agent (data assistant) visits the node to gather information for a given patient ID, it already carries with it all the necessary logic need to retrieve and process this information. The stationary archive agent only serves to provide the necessary database connection details and the necessary information for the database tables, which store the relevant information. The mobile agent then uses these details to retrieve the details associated with the current patient. Security is a major issue when it comes to accessing medical information in electronic formats. However, we assume that the security is provided by the underlying infrastructure of the node

5.4 Road Authority Website

Apart from taking into account factors like the current resource availability, special medical facilities offered by individual hospitals and the past medical history of the patient we also factor in the current road status information for the various hospitals when ranking them in the order of their suitability. The road authority node is actually a HTML based website, which provides real-time updates regarding the road status for individual hospitals which exists in the city. The road status is given in terms of total percentage of roads blocked due to present traffic conditions, construction work or other blockages that may exists. The *road authority node* component is made up of a stationary agent with a web front end to it. When a mobile agent (data assistant) visits this node it interacts with the web page agent to retrieve the road updates for each hospital it has on its list of selected hospitals.

5.5 Mobile Hospital Assistant

The *hospital assistant* component is one of the two mobile agents employed by our architecture. This component serves as the 'link' between the mobile ambulance node and the other static nodes representing the various hospitals/speciality clinics. The primary task of this agent is to travel to the various hospital nodes and retrieve the latest availability of resources. For the purpose of our model we consider three types of resources – doctors, nurses and hospital beds. The hospital assistant agent gets instantiated when the paramedic clicks on the 'Launch Agent' button of the user interface. As part of its instantiation the assistant is passed the list of hospitals, which the paramedics have short-listed. On deployment, the agent contacts a central resource registry in order to 'discover' the various hospital nodes it is scheduled to visit. For our implementation, the central resource registry service (similar to a yellow pages directory) is actually a Grasshopper Region [Gra01]. Since the hospital assistant is a mobile agent it is capable of migrating to its destination node in order to execute the assigned tasks. The agents start off by moving to the first hospital node on its list. Upon it successful arrival at its destination, the assistant interacts with the stationary hospital agent present at this node and queries the desired resource status information. The interaction takes place locally within the hospital node using standard Grasshopper communication services. Each interaction is made up of a series of 'request' and 'replies' between the two agents. The information exchanged as part the interaction includes:

- Name of the hospital
- Special medical facilities provided by the hospital (e.g. cardiology, sports medicine etc.)
- Total number of doctors currently on duty at the hospital
- Number of doctors that are currently available to attend a patient
- Total number of nurses currently on duty at the hospital
- Number of nurses that are currently available to attend a patient
- Total number of hospital beds
- Number of beds currently available.

When the mobile assistant has acquired the required information, it moves on to the next hospital on its list. On arrival, it again interacts with the stationary agent present at this node repeating its series of requests. The mobile agent keeps doing this till it has finished with all the hospitals on its list. After having visited all the hospital nodes it was tasked too and acquiring the required information the mobile hospital assistant moves back to its home node (the ambulance client which deployed it). Once home, it interacts with the data assistant agent to collaborate the results. The collaborated set of results is then communicated to the UI, which uses it to rank the hospitals in terms of their suitability to the current patient. This sorted list of hospitals is then displayed to the paramedics. The following factors are taken into consideration when ranking the hospitals:

- Availability of individual resources (doctors / nurses / beds)
- If the facilities offered by a hospital is suitable to the current case
- Number of times the current patient has visited a particular hospital
- Current route status of a particular hospital

It is important to realise that the distributed and heterogeneous environment in which the hospital assistant operates in, is subjected to multiple points of failure. This becomes especially true since our environment is composed of a mix of stationary and mobile nodes interacting together using a combination of static and wireless networks. The possible points of failure identified for the mobile hospital assistant are:

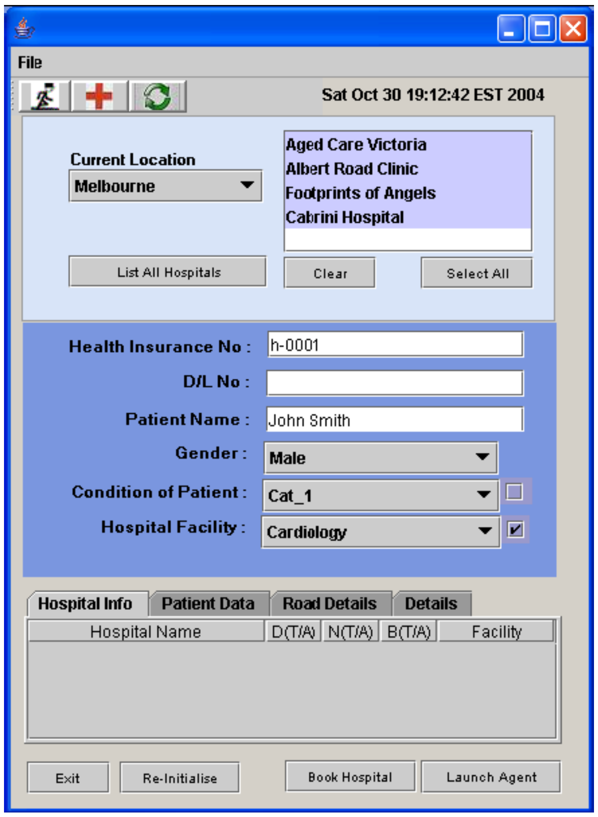
- The assistant gets lost while transiting between two nodes and is unable to arrive at its destination.
- The node on which the assistant currently resides on crashes.
- The assistant is unable to move back to the ambulance (due to possible link failure or the ambulance node may be down)

It therefore becomes necessary that the hospital assistant be designed in a way such that it is fault tolerant. In order to counter the above point of failure we propose the following:

Before every migration the assistant creates a clone of itself. This clone is an exact copy of the assistant and is capable of executing in a similar manner to its parent. Once it has cloned itself, the parent copy then moves to its destination node, upon successful arrival it sends a message asking the clone to remove itself. If the clone does not receive this acknowledgement in a specified timeframe, (this is the estimated time required to travel between two hospital nodes) it is assumed that the migration attempt failed. When this occurs, the clone takes over the tasks of the parent and repeats the entire process of ‘cloning’ and then moving. Adopting this strategy helps safeguard against the assistant getting lost during transition.

In order to safeguard against the failure of a node on which the assistant currently resides on, we propose the use of the Grasshopper Agency’s persistence service. This service is part of the core functionality of the agency and is used to store the data states of all currently hosted agents onto a persistent medium (hard disk). Thus if a node was to crash due to an unforeseen event, its agency has all the necessary information to reconstruct the currently hosted agents upon start-up.

If for some reason the hospital assistant finds that it is unable to move back home (due to possible network failure, home node currently unavailable etc.), it possesses the ability to wait at its current location. When this happens, the assistant goes into ‘hibernation’ for a predefined time period, wakes up and then tries to move back again. It keeps doing this till it is able to successfully get back home or till a specified wait period expires (whichever is earlier). If the assistant is still unable to get back home after the wait period has expired it is assumed that an application error has occurred and the assistant stops its execution and removes itself.



Sat Oct 30 19:12:42 EST 2004

Current Location: Melbourne

Aged Care Victoria
Albert Road Clinic
Footprints of Angels
Cabrini Hospital

Health Insurance No : h-0001
D/L No :
Patient Name : John Smith
Gender : Male
Condition of Patient : Cat_1 ☐
Hospital Facility : Cardiology ☑

Hospital Name	D(T/A)	N(T/A)	B(T/A)	Facility

Figure. 1. DSS user interface

The scenario described above in section 5 involved a man who suffers form a heart attack while on his way to work. A call from the dispatch centre alerts the ambulance of this emergency. Based on the information available, paramedics filter out a group of hospitals that are in close proximity to the location of emergency. They key in the relevant patient details and also indicate any specific medical facility required (cardiology in this case). Figure 1 shows the DSS user interface based on the information entered.

The paramedics then utilize the travel time to the scene of emergency to launch a set of two mobile agents. The first mobile agent (referred to as *hospital assistant*) starts off by contacting the Grasshopper Region to ‘discover’ the four hospital nodes that paramedics have short-listed. When these nodes have been successfully discovered, the agent migrates to the first hospital on its list (*Aged Care Victoria* in this case). Upon arrival, it interacts with the stationary agent present at this node and retrieves the latest resource utilization details for the hospital. Figure 2 shows this interaction process, which takes place via a proxy object using Grasshopper communication service.

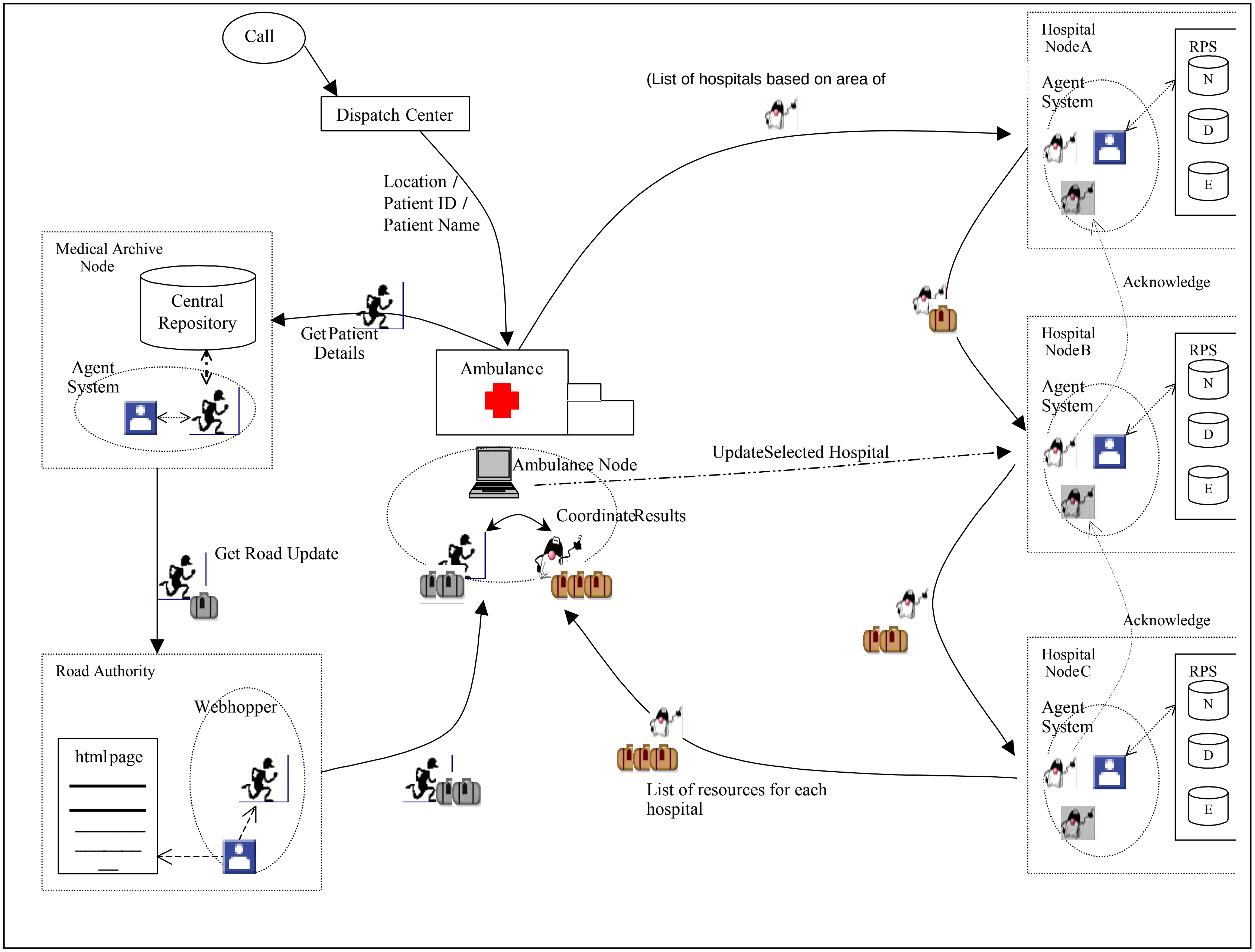


Figure 2. Mobile Agent Based Architecture for emergency DSS

As can be seen from the above figure, a single interaction process is actually made up a series of requests. By being able to migrate to its destination host, the hospital assistant makes this entire process local thus bypassing the network.

Having obtained the requested information, the mobile agent moves to the next hospital on its list (*Albert Road Clinic*), repeating its series of requests. It keeps doing this till it arrives at the last hospital on its list (*Cabrini Hospital*), following which the mobile assistant moves back to the ambulance node.

While the hospital assistant retrieves the resource utilization details, another mobile agent (referred to as *data assistant*) simultaneously gathers route status updates for the selected hospitals along with the relevant medical history of the current patient. It starts off my migrating to the medical archives node in the network and obtains the relevant database connection details by interacting with the stationary archives agent. It then uses these connection details to retrieve the medical history associated with the current patient based on the identification details specified. Figure 4 illustrates the process of obtaining a patient's medical details.

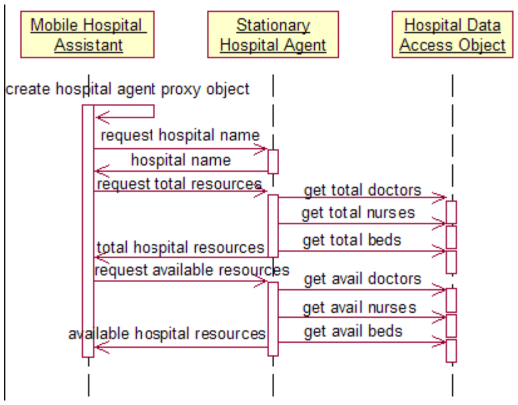


Figure 3 Obtaining Resource Utilization Details

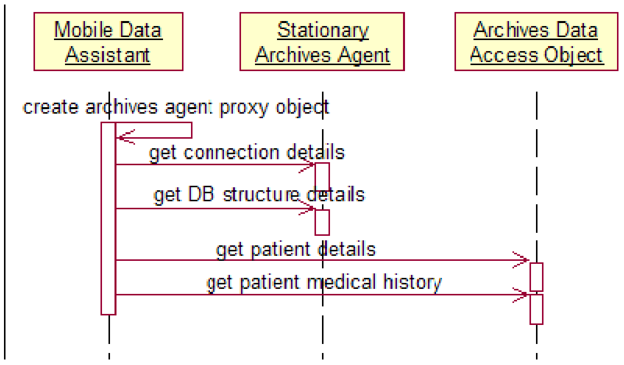

Mobile Data Assistant
Stationary Archives Agent
Archives Data Access Object
create archives agent proxy object
get connection details
get DB structure details
get patient details
get patient medical history

Figure 4 Obtaining Patient's Medical Details

Following this, the data assistant moves to the road authority node to gather current route status updates for each of the four hospitals. Road status is obtained in terms of total percentage of roads blocked due to various factors (such as traffic situation, construction work etc.) Figure 5 shows the sequence diagram for obtaining the route status updates.

When both the mobile agents have finished with their assigned tasks, they move back to the ambulance node. Subsequently, they coordinate with each other to collaborate on their result sets. Based on results obtained, the four short-listed hospitals are ranked in their order of suitability. This ranked list of hospitals is then published to the user interface and gets shown to the paramedics as illustrated in Figure 6.

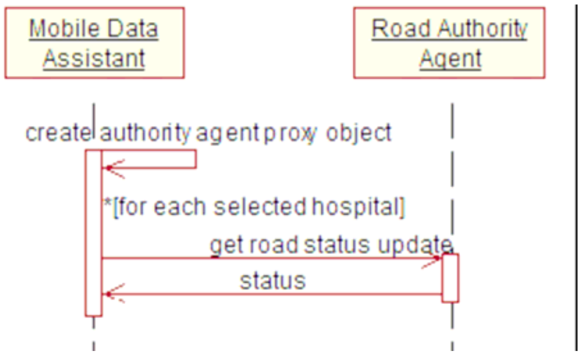
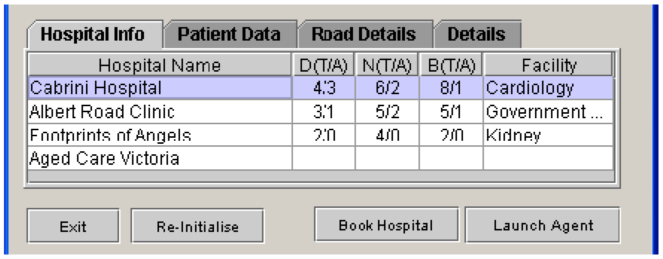


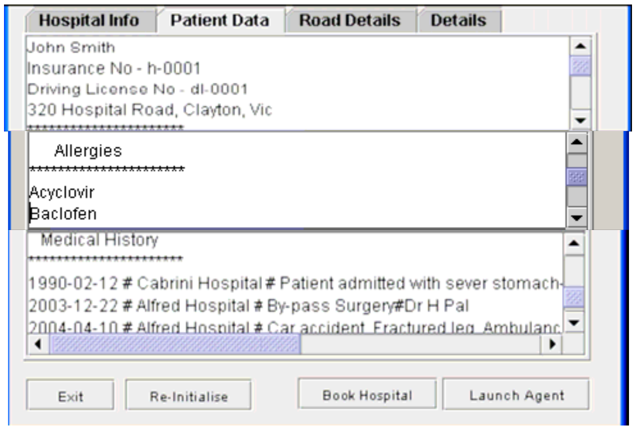
Figure 5 Obtaining Route Status Updates



Hospital Name	D(T/A)	N(T/A)	B(T/A)	Facility
Cabrini Hospital	4/3	6/2	8/1	Cardiology
Albert Road Clinic	3/1	5/2	5/1	Government ...
Footprints of Angels	2/0	4/0	2/0	Kidney
Aged Care Victoria				

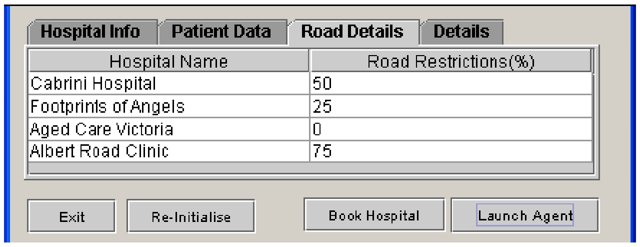
Figure 6 Hospitals Ranked in terms of Suitability

Therefore, by the time paramedics reach the scene of emergency; the ambulance's onboard terminal already reflects the list of nearby hospitals ranked form the most relevant to the least. This information is further supplemented with the patient's medical history (Figure 7) and the current route status update for each selected hospital (Figure 8).



John Smith
Insurance No - h-0001
Driving License No - dl-0001
320 Hospital Road, Clayton, Vic
****
Allergies
****
Acyclovir
Baclofen
Medical History
****
1990-02-12 # Cabrini Hospital # Patient admitted with sever stomach
2003-12-22 # Alfred Hospital # By-pass Surgery#Dr H Pal
2004-04-10 # Alfred Hospital # Car accident, Fractured leg, Ambulanc

Figure 7 Patient's Medical History



Hospital Name	Road Restrictions(%)
Cabrini Hospital	50
Footprints of Angels	25
Aged Care Victoria	0
Albert Road Clinic	75

Figure 8 Route Status Update

Paramedics review this ranked list of hospitals along with the associated ranking details and make their selection. On the way to the selected hospital, a message is sent to the respective hospital node with details of the patient 'en route' to the hospital. Thus, by the time the ambulance reaches the selected hospital, this information has already enabled emergency room staff to prepare for the arrival of John Smith publishing it onto the user interface. It is assumed that mobile agents have access to context-on-demand infrastructure and can request current context, eg, location of other ambulances, general statistical data about the mass casualty event, road traffic status, level of training and experience of ambulance personnel, etc. These context attributes differ in their importance for the on-board DSS. However, the importance varies over time.

7. Verifying Application Requirements and Design Criteria

We had identified some of the key requirements that need to be addressed by our application for it to be successfully deployed in the dynamic and variable context of healthcare. In this section we revisit these and detail how our mobile agent based implementation meets the desired design criteria.

We had earlier identified that the healthcare environment is not only distributed by nature but also exhibits a high level of heterogeneity across individual organizations. Therefore, for any application to be successfully deployed in such environments it is essential that it is able to seamlessly integrate the many heterogeneous nodes in a transparent manner. Our architecture employs mobile agents to visit the various organization nodes and acquire the desired information. Because these agents are only dependent solely on the execution environment (agent system) and not on the underlying hardware/software/transport layers of the node itself, makes them naturally heterogeneous. Various tasks can be assigned to these agents; they can then autonomously and asynchronously visit the various heterogeneous nodes of the network to execute them. Returning home (ambulance) to convey their results once they have finished.

Another key aspect of our approach is the ability to support a disconnected client node. As the ambulance moves around the city, there are areas where it is out of network reach (inside a tunnel for example). Therefore, it becomes necessary for our application to be flexible enough to take such temporary but unplanned changes in the environment into account. The fact that the agents deployed in our application are mobile and are able to take autonomous decisions of behalf of an entity independent of the entity itself makes them particularly well suited for such mobile computing environments. Once these agents have been initialised and dispatched, they are capable of executing autonomously and asynchronously. They carry on with the assigned task without any further interventions required form the users. This effectively implies that they only time the ambulance node needs to be connected to the network is initially to launch these agents and finally to receive them back.

The mobile nature of this application also means that the computing devices needed to support the paramedics will be constrained in terms of their shape, size and weight. We propose the use of a PDA or a dashboard mounted terminal. Due to their limitations these devices are effectively 'thin-client', restricted in terms of their processing power, memory and bandwidth availability. By deploying mobile agents to carry out the various retrieval tasks, we are able to overcome the limitations exhibited by our ambulance client. As our agents are not restricted by physical machine boundaries, they can travel around the network to execute the various tasks closer to the information source. The advantage here lies in the fact that agents are able to execute computationally intensive tasks (such as information filtering and retrieval) at the destination nodes itself and return only the most relevant information back to the client. All that the client needs to do is to display these results.

In addition to the mobile settings, part of the dynamicity and variability of the context and conditions of our application arises because it involves a combination of static and mobile nodes interacting using a mix of static and wireless networks. This results in a possibility of a high failure rate. However, as we have already stated that reliability is an essential requirement in critical application domains such as healthcare. Single points of failure are not acceptable. Structuring the application based on the mobile agent paradigm gives it the flexibility required to be able to gracefully handle specific failures and yet remain functional. As the agent abstraction consists of code along with its state of execution and coupled with the fact that our agents are mobile and autonomous in nature allows them to react dynamically to a given situation (such as node failure). The net result is that employing mobile agents makes it easier to build fault tolerant and robust systems that exhibit a higher degree of survivability (as compared to conventional client-server oriented applications).

By using mobile agents to process and acquire information, it is possible to provide a classic example for locality of reference. Rather than having to send large amounts of data over the network to be integrated and processed on the ambulance client we dispatch a set of mobile agents. These agents migrate to the various nodes, process data locally and return back with only what is required. The concept of moving computation to data rather than the other way around provides for a two-fold advantage. Firstly, it cuts down on the network load and helps overcome latency. Secondly, by processing things remotely the agents are able to overcome the resource limitations of our ambulance node. This ties along with our goal of being able to support thin-clients in a mobile setting.

Basing our proposed architecture on the mobile agent paradigm also allows it to be more easily customisable (compared to traditional technologies) and flexible enough to effectively scale up to the various types of nodes in the citywide network. Our application needs to cater not only for the changing number of organization nodes but also their different types (hence, scalability and customisability to a certain degree go hand in hand). By using agents we are able to encapsulate the ‘protocol’ needed to perform a given task. Hence, when the agents move around the network visiting remote hosts they establish ‘channels’ customised to the ‘know-how’ they themselves carry. This flexibility allows for agents to speak the same language of the organisational node they are visiting, making it possible for a single mobile agent to visit a multitude of heterogenous nodes and yet be ‘tailored’ to every one of them.

Thus the key benefits of mobile agents as important tools for the proposed application are:

- Overcome limitations of the ambulance client
- Provide locality of reference resulting is reduced network load and latency
- Ability to adapt dynamically to their execution context resulting in increased fault tolerance
- Autonomous and asynchronous operation in a heterogenous environment.
- Disconnection support
- Customisability

Some of these benefits can be achieved one way or another by using a combination of different existing technologies. We choose mobile agents because they offer a single, general framework in which large-scale distributed applications can be implemented more efficiently.

8. Evaluation

To demonstrate the possible benefits of utilizing mobile agents for mobile context-aware DSS, our system needs to be evaluated based on its effectiveness and efficiency. While the former deals with overall system accuracy, the later involves its performance. As part of the evaluation, we aim to ascertain the performance impact of mobile agents in terms of the computational time required to archive the desired functionality. In order to do so we monitor various aspects of the mobile agent life cycle and measure its effect on the overall cost of using agents in our proposed architecture. The system evaluation aims to determine:

- The time taken for a client to deploy a set of mobile agents
- The performance delays introduced due to overheads in transporting agents between nodes
- The time taken to interact with agent as well as non-agent based components, utilizing a combination of local and remote Grasshopper communication mechanisms [Gra01]
- The total round trip time for a typical usage scenario.

We do realize that the above set of experiments do not explore the entire range of agent capabilities (eg interoperability etc.). However, results provided bring some understanding to performance measures associated with agents.

Due to the highly controlled nature of the healthcare environment and sensitivity of medical data involved it was not possible to run our experiment in an actual ambulance service setting. Instead a controlled environment was set-up that simulated the characteristics of a distributed ambulance network. The experimental environment consisted of five desktop PCs powered by an Intel Pentium 4 (2.66 GHz) processor, each running on Windows XP Professional operating system, connected together via a non-dedicated 100 MBPS Ethernet LAN that went through multiple hubs. The reason for using a LAN segment, which was part of a larger public network, was to simulate the average network traffic and latency that would be experienced in a real setting. Each of the five PCs represented a particular node in the proposed architecture. Three of them were used to represent a specific hospital server, while the remaining two represented medical archives and road-authority nodes respectively. In order to support stationary as well as migrating agents, all nodes were installed with Java Runtime Environment and Grasshopper Agent Platform. The hospital and archive nodes also support a group of MS Access databases that maintain hospital resource utilization details and patient's medical history. A laptop computer, representing a mobile ambulance node further complemented the stationary PCs. To simulate the environment in which the ambulance operates, the laptop is connected to the network via a 10 MBPS 802.11b wireless LAN. As with other nodes the mobile node too supports the Grasshopper Agent Platform.

The reason for having separate machines to host the various nodes was to force the Grasshopper components to utilize the network in order to communicate with each other rather than being able to connect internally via ports if the were housed on the same machine. It is important to note that due to difference in the internal clocks of separate machines timings used in our experiments would be erroneous. In order to overcome this problem we synchronized all machines with an Internet time-server (time.nist.gov). This method, although not entirely fool proof, significantly reduces the difference between individual clocks thus allowing timings of the test to be usable.

Many of today's distributed applications are based on the traditional client-server model. While this technology managed to successfully cater to an ever-increasing array of application requirements, it however has its limitations. The model is poorly scalable, does not consider the topology of underlying networks and the inherently static nature of its components makes it difficult for applications to be readily customized. Mobile software agents are an emerging technology in the area of large distributed and heterogeneous computing environments and have received a great of attention because they represent a paradigm shift in the way distributed systems are traditionally designed and implemented. The benefits they provide include amongst several others, overcoming the limitations of a client computer, the ability to autonomously represent an entity (included disconnected ones), improved locality of reference, increased flexibility and customisation. Most of these advantages stem from the 'mobile' attribute of the paradigm that allows it to overcome the static nature of present day networks.

In spite of the claimed benefits, mobile agent technology is in a state of paradox primarily because the field has promised so much yet delivered so little in practice. The current lack of existing commercial applications indicates that there is no real need for mobile agents and that there are no applications that absolutely 'require' them. Our review showed, although not all application contexts needs mobile agents yet there are many other that find agents to be the most effective technique in realizing part or all of their tasks. Current research has primarily focused on utilising mobile agents as a single general framework for building distributed systems rather than attempting to find specific application domains where agents have potential deployment. We were not looking to identify and provide that illusive 'killer-application' to 'jump-start' the use of mobile agents but were rather interested in the benefits they claim to provide and the potential they carry in terms of delivering a truly distributed application in a heterogeneous environment.

To make our case for mobile agents, in this thesis we proposed utilizing them as the key enabling technology in realizing a distributed application to support the highly mobile and dynamic environment of ambulatory services. We have presented an architectural model that employs a set of agents to asynchronous and autonomously traverse through a set of heterogeneous medical organizational nodes, carrying out various information retrieval tasks at each node. Subsequently they coordinate their results with each other to rank a pre-selected list of hospitals in the order of suitability for the current patient. By integrating the autonomous, goal driven and pro-active nature of mobile agents into the highly mobile and variable context of an ambulatory environment, our model was not only able to overcome the static orientation of traditional distributed networks but also allowed for effective, well-informed diagnosis and decision support by providing the paramedics with latest hospital resource utilization details and the patient's medical history in real-time.

9. Analysis of the results

We presented an implementation of our proposed agent based architecture, which was based on the functional and non-functional application requirements set out section 4. We also created an illustrative emergency scenario in order to demonstrate the validity and feasibility of our proposed model. From our assessment of the paradigm we were able to identify some of the major technical advantages it has to offer. While we note that none these individual advantages are sufficiently strong enough to justify the use of agents and can be achieved by a combination of other means, we also realize that no single computing technique shares these aggregate advantages. Therefore, the true strengths of mobile agents is that they can provide a single, open, generalized framework that allows a wide range of distributed applications to be developed and implemented in an efficient and effective manner. Our assessment also indicated that applications based on agents present issues and challenges that were unique to the paradigm. The most significant of these, especially in the sensitive context of healthcare, is the need for highly secure agent execution environment, as security and privacy is the single most important issue when dealing with medical information. In addition, our deployment experience showed that presently none of the available agent toolkits provide any kind of support for a debugging tool making it difficult to profile and / or debug agent applications.

We evaluated our system to assess its overall feasibility and accuracy. As part of the evaluation we ascertain the performance impact of mobile agents by analysing the computational costs involved to achieve the desired functionality. This is shown trough an actual measurement of the overheads associated (in terms of Δ time) with the basic mechanisms of an agent oriented technique. From the results obtained we notice that agent migration and remote inter-agent communication involve substantial overheads. These overheads could have implications in terms of acceptability of the system in the real-world ambulatory environment due to the potential delays introduced. However, the more qualitative strengths of agents, such as their ability to handle disconnections, could still make them an ideal choice in ambulatory environments even if overall completion times are worse. From a performance stand point, we can conclude that resorting to mobile agents without utilizing the inherent advantages offered (such as locality of reference) and instead using them as just another communication technique will in fact further increase the overall network load and latency.

Overall, our experience showed that there is a strong case for using agents in wide variety of distributed system. Although, we should not expect to see agents replacing existing techniques, instead they will be used as a combination of techniques to provide the best solution. However, the evolutionary path that will take us from the current state to the widespread use and adoption of agents is only possible if the various technical challenges and other issues of the paradigm have been sufficiently met (lack of infrastructure, security etc.).

10. Future Work

In this research we demonstrated the efficiency and usefulness of mobile agents in distributed systems by applying them to the highly mobile and dynamic area of ambulatory services. Though our implementation and extensive experimental studies have shown the feasibility of this approach, we have however made certain assumptions in our system design so as to ease the actual implementation process. Most of these assumptions are made due to the highly controlled and sensitive nature of the healthcare domain or because of time limitations on our part. There are several extensions, we have identified that could be applied to the proposed model to further equip it for a real-world deployment

- **Location Awareness**

In the current implementation we do not factor in the key element of distance while ranking the selected list of hospitals. Ideally, our model should be integrated with some form of location tracking technology (such as GPS) that will enable it to determine the exact distance of each hospital from the location of emergency. This will not only make the ranking process more intuitive but will also enable the model to cater to a wider variety of emergency scenarios. For example consider a scenario is which the patient is triaged critical (category 1). In this case the nearest hospital should be selected rather than the one, which has a higher level of available resources.

- **Enhanced Security Features**

Security and privacy are of primary concern when dealing with medical data. However, this is an interesting area as providing a higher level of security generally means reducing availability and timely delivery of services. So far, we have utilized the in-built security mechanisms provided by Grasshopper platform. We note that this level of basic security will not be sufficient when dealing with actual patient data. Therefore, moving forward the model would need to incorporate a more comprehensive security policy. Future work could include arming the agents with authentication mechanisms that prevent unauthorized access to server data. Along with authentication, organizational nodes could also define and manage data access policies for different agent types. Further, a simple encryption technique could be put in place to secure rudimentary communication.

- **Incorporating generic web-based nodes into the ambulatory environment**

Instead of utilizing the Webhopper extension to enable an agent to interact with a website (as we did in the case of the road-authority node), a better solution would be to allow agents to ‘talk’ directly to such nodes. Webhopper was used to maintain consistency across all nodes that comprised our architecture as well as to take advantage of the communication and security mechanisms of the underlying agent platform. However, with enhanced security features in place, doing away with Webhopper will not only make the design more generic but also scalable by allowing standard web-based nodes to be included as part of the architecture.

- **Deployment in real-world settings**

To realize the true potential of our proposed application, both in terms of its feasibility as well as its accuracy and efficiency, it should ideally be deployed and tested in an actual ambulatory environment with the size and complexity of real data sets. However, this would involve the collaboration of a number of healthcare organizations (hospitals, clinics etc.) and require access to their information systems. We note that due to highly sensitive and closed nature of this environment, an actual deployment could face many challenges. Moreover, our research prototype has been developed based on a few key assumptions in relation to the operating environment (such as unique patient identifier, agent host on every node), which may not hold true in the real world.

11. Conclusion

In this paper we have demonstrated how the agents technology can be used to provide decision support in a dynamic, context sensitive environment. Healthcare example was selected as an appropriate field due to the fact that it lends itself very well for such implementation from the point of view of business requirements as well as recently increased interest in using technologies by medical practitioners [SBW+05}.

Various arguments and reasons have been put forward by proponents of the agents technology to emphasize the potential this paradigm has to offer. However, despite the claimed potential, opponents believe that most of these benefits can be achieved one way or the other by using a combination of already existing technologies minus the shortcomings agents currently exhibit (for example such as security concerns and lack of infrastructure). In fact agent technology is in a state of paradox [Brad97]. Though the field has enjoyed concomitant research progress and is a 'buzzword' within both the commercial and academic worlds, yet the rate of uptake in terms of commercially fielded systems has been extremely slow. This is despite the fact that there are many potential application areas where mobile agents can be ideally deployed. While numerous agent system have been developed and agents have been applied to a wide variety of applications ranging from purely commercial [KFS02] to military [MCW00] to educational [LeH02], none of these has been able to 'jump-start' the use of agents [MDW99].

Unlike traditional paradigms, agents exhibit the property of being autonomous and interactive. Coupled with mobility, they are capable of performing dynamic and intelligent inference tasks during their execution. Utilizing a framework based on the mobile agent paradigm provided for a higher degree of flexibility by allowing applications to dynamically adapt to the changing demands of their execution environments.

Acknowledgement

This project was funded by Monash University grants in 2004-2005. The research team included Chief Investigators: A/Prof F.Burstein, A/Prof A. Zaslavsky, Dr Jocelyn San Pedro, Dr Leonid Churilov, Dr Jeff Wassertheil (Partner Investigator), Mr Nitin Arrora (Programmer and research student).

References

[ALW02] Allen, J. and Wilson, J. (2002), "Securing a Wireless Network" *In Proceedings of the 30th ACM SIGUCCS Conference on User Services*, 20-23 November 2002, Providence, Rhode Island, USA, pp 213-215.

[AMM03] Arshad U, Mascolo C, Mellor M (2003) *Exploiting Mobile Computing in Health-care,* In: Proceedings of the 23rd International Conference on Distributed Computing Systems (ICDCS'03), May 2003.

[BBD+03] Bilykh I, Bychkov Y, Dahlem D, Jahnke JH, McCallum G, Obry C, Onabajo A, Kuziemsky C (2003) *Can GRID Services Provide Answers to the Challenges of National Health Information Sharing?*, In proceedings of the Centre for Advanced Studies conference on Collaborative Research, Toronto: 39-53.

[Brad97] Bradshaw, J.M. (1997) *Software Agent,* AAAI/MIT Press, Cambridge, MA.

[BRE98] Brézillon, P. (1998) Modelling and Using Context in Applications, International Journal on Human-Compute Studies, 48 (3).

[BRE99] Brézillon P (1999) Context in Artificial Intelligence, Computer and Artificial Intelligence, 18, 4: 321-340.

[CHK00] Chen, G. and Kotz, D. (2000) A Survey of Context-Aware Mobile Computing Research. Technical Report TR2000-381, Dept. of Computer Science,Dartmouth College, November 2000.

[CHK94] Chess D, Harrison C, Kershenbaum A (1994) Mobile Agents: Are They a Good Idea? IBM Research Report, RC 19887, Oct 1994.

[DiS01] Dikaiakos MD, Samaras G (2001) Performance Evaluation of Mobile Agents: Issues and Approaches, Performance Engineering, LNCS, 2047, 2001:148-166.

[FGG98] Forslund DW, George JE, Gavrilov EM (1998) *TeleMed: Wide-Area, Secure, Collaborative Object Computing with Java and CORBA for Healthcare,* In proceedings of the 7 th International Symposium on High Performance Distributed Computing, July 1998: 30-33.

[GGH00] Grimson J, Grimson W, Hasselbring W (2000) *The SI Challenge in Health Care,* Communications of the ACM, 43, 6, June: 48-55.

[GOP02] Glitho RH, Olougouna E, Pierre S (2002) *Mobile Agents and Their Use for Information Retrieval: A Brief Overview and an Elaborate Case Study,* IEEE Network, 16, 1, Jan: 34-41.

[Gra01] Grasshopper Basics and Concepts: Release 2.2, IKV++GmbH., Berlin, Germany, March 2001, Available online at http://www.grasshopper.de

[GSJ+01] Grimson, J.; Stephens, G.; Jung, B.; Grimson, W.; Berry, D. and Pardon, S.; *Sharing Health-care Records over the Internet,* IEEE Internet Computing, Vol. 5, No. 3, May 2001, pp. 49-58.

[KFS02] Kowalczyk, R.; Franczyk, B. and Speck, A.; *InterMarket – Towards Intelligent Mobile Agent e-Marketplaces*, 9th Annual IEEE Conference and Workshop on the Engineering and Computer-Based Systems. 2002.

[Lan98] Lange DB (1998) Mobile Objects and Mobile Agents: The Future of Distributed Computing?, In proceedings of European Conference on Object-Oriented Programming, LNCS, 1445: 1-12.

[LaO99] Lange DB, Oshima M (1999) Seven Good Reasons for Mobile Agent, Communications of the ACM, 42, 3, Mar: 88-89.

[LeH02] Lee CH, Han SG (2002) *Educational Contents and Retrieval Systems using Mobile Agent,* In proceedings of the International Conference on Computers in Education (ICCE'02), IEEE.

[LMP+01] Liu, A.; Martens, R.; Paranjape, R. and Benedicenti, L (2001) *Mobile Multi-Agent System for Medical Image Retrieval,* Canadian Conference on Electrical and Computer Engineering, Vol. 1, May 2001, pp 65-69.

[MCW00] McGrath, S.; Chacon, D. and Whitebread K (2000) *Intelligent Mobile Agent in the Military Domian*, In proceedings of the Autonomous Agents 2000 Workshop on Agents in Industry, Barcelona, Spain, 2000.

[MDW99] Milojicic D; Douglis F,Wheeler R (Eds.) (1999) *Mobility: Processes, Computers and Agents,* ACM press, 1999.

[MoB97] Morton S, Bukhres O (1997) *Utilizing Mobile Computing in the Wishard Memorial Hospital Ambulatory Service*, In proceedings of the ACM symposium on Applied Computing, San Jose, California , 287-294.

[MSE+03] Mabry SL; Schneringer T; Etters T, Edwards N (2003) *Intelligent Agents for Patient Monitoring and Diagnostics*, In proceedings of the ACM symposium on Applied Computing, Melbourne, Florida, 257-262.

[MW96] Merriam-Webster (1996) Merriam-Webster's Dictionary of Law, Merriam Webster, Sep 1996.

[PAD04] Padovitz, A., Loke, S.W., Zaslavsky, A., Burg, B. (2004) "Towards a General Approach for Reasoning about Context, Situations and Uncertainty in Ubiquitous Sensing: Putting Geometrical Intuitions to Work", 2nd International Symposium on Ubiquitous Computing Systems (UCS'04), Tokyo, Japan , 2004

[SBW+05] San Pedro, J.; Burstein, F.; Wassertheil, J.; Arora, N.; Churilov, L. and Zaslavsky, A.; On Development and Initial Evaluation of Prototype Mobile Decision Support for Hospital Triage, To appear in proceedings of the Hawaii International Conference on System Sciences (HICSS'38), Jan 2005.

[TuS97] Tung, B. and Sankaran, S.; *Group Decision and Negotiation Support in Telemedicine: An Application of Intelligent Mobile Agents as Non-Human Teleworkers,* In proceedings of the 30th Hawaii International Conference on System Sciences, 4, Jan: 120-129.

[WMB03] Wendelken, S.M.; McGrath, S.P; Blike, G.T (2003) *A Medical Assessment Algorithm for Automated Remote Triage,* In proceedings of the 25th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, 4, Sept : 3630-3633.