Group Support in Collaborative Networks Organizations for Ambient Assisted Living

Paulo Novais[1], **Ricardo Costa**[2], **Davide Carneiro**[1], **José Machado**[1], **Luís Lima**[2], **José Neves**[1]

[1] DI-CCTC, Universidade do Minho, Braga, Portugal

[2] College of Management and Technology - Polytechnic of Porto, Felgueiras, Portugal

Abstract. Collaborative Work plays an important role in today's organizations and normally in areas where decisions must be made. However, any decision that involves a collective or group of decision makers is, by itself, complex. In this work we present the VirtualECare project, built in terms of an intelligent multi-agent system able to monitor, interact and serve its customers, which are, normally, in need of care services, and assisted with tools based on open standards, like OSGi an R-OSGi.

1. Introduction

In the last years there has been a substantially increase in the number of people needed of intensive care, especially among the elderly, a happening that is related to population ageing [1]. However, this is not exclusive of the elderly, as diseases as obesity, diabetes, and blood pressure have been increasing among young adults [2]. While a new realism, it has to be dealt with by the health sector, and particularly by the public one. Thus, the implication of finding or not new and cost effective ways for health care delivery are of particular importance, especially when one want them not to be removed from their "habitat" [3]. Following this line of thinking, the VirtualECare project [4] is presented in section 2, like similar ones that preceded it [5]. In this paper we are going to center our efforts on the Group Decision modules of the VirtualECare project.

Please use the following format when citing this chapter:

Novais, P., Costa, R., Carneiro, D., Machado, J., Lima, L., Neves, J., 2008, in IFIP International Federation for Information Processing, Volume 286, Towards Sustainable Society on Ubiquitous Networks, eds. Oya, M., Uda, R., Yasunobu, C., (Boston: Springer), pp. 353 – 362.

In the last years we have assisted to a growing interest in combining the advances in information society - computing, telecommunications and presentation – in order to create Group Decision Support Systems (GDSS). Indeed, the new economy, along with increased competition in today's complex business environments, takes the companies to seek complementarities in order to increase competitiveness and reduce risks. Under these scenarios, planning takes a major role in a company life. However, effective planning depends on the generation and analysis of ideas (innovative or not) and, as a result, the idea or brainwave making and management processes are crucial.

Our objective is to apply the GDSS approach to problem solving to a new area. We believe that the use of GDSS in the healthcare arena will allow professionals to achieve better results in the analysis of one's Electronically Clinical Profile records (ECPs). This achievement is vital, regarding the explosion of knowledge and skills, together with the need to use limited resources and get better results.

2. VirtualECare

The VirtualECare project main objective is to present an intelligent multi-agent system able to monitor, interact and provide its customers with health care services of the utmost excellence. This system will be interconnected, not only to other healthcare institutions, but also with leisure centers, training facilities, shops and patient relatives, just to name a few.

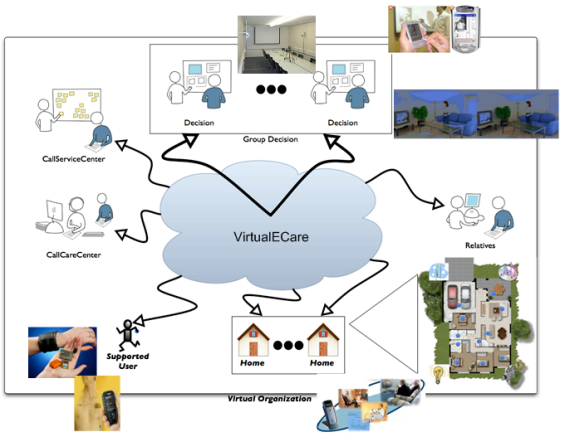

Fig. 1. The VirtualECare Architecture

The VirtualECare Architecture is a distributed one with their different modules interconnected through a network (e.g. LAN, MAN, WAN), each one with a different role (Figure 1). A top-level description of the architecture machinery is given below:

- SupportedUser – elderly people with special health care needs, whose clinical data is sent to the CallCareCenter and redirected to the Group Decision Support System;

- Home – the elderly natural environment, provided with sensors, with the clinical data being sent to the Group Decision Support System through the CallCareCenter, with the remaining one being redirected to the CallServiceCenter;
- Group Decision – it is in charge of all the decisions taken at the VirtualECare platform (our work will be centered on this key module);
- CallServiceCenter – Entity with all the necessary computational and qualified personal assets, capable of receiving and analyze the different data and take the necessary actions according to it;
- CallCareCenter – Entity in charge of computational and qualified personal resources (i.e. healthcare professionals and auxiliary staff), capable of receiving and analyze the clinical data, and take the necessary actions, accordingly;
- Relatives – individuals that may have an effective role on the organized assignment of the tasks of their love ones, being able to give valued information about them and to intervene, in a complementary way, in recognized emergency (e.g., loneliness).

In order to the Group Decision Support System to make its work, it has to collect the judgment of specialized staff (e.g., nurses, paediatrics, cardiologists). There is also the need to have a digital profile of the SupportedUser, allowing for a better understanding of his/her special needs. In this profile one may have quite a few types of relevant information, ranging from the patient Electronic Clinic Process to their own personal preferences (e.g. musical, gastronomic) passing by their own personal experiences, which can be used to better understand and satisfy their needs and expectatives.

This solution will help healthcare providers to integrate, analyze, and manage complex and disparate clinical, research and administrative knowledge. It will provide tools and methodologies for creating an information-on-demand environment that can improve quality-of-living, safety, and quality of patient care.

3. Group Support in Collaborative Networks Organizations

By definition, any Collaborative Network Organization (CNO) may support collaborative work, which presupposes the existence of a group of people that has as mission the completion of a specific task [6]. The number of elements involved in the group may be variable, as well as its perseverance. The group members may be at different places, meet in an asynchronous way and may belong to different organizations. Collaborative work has not only inherent advantages (e.g., greater pool of knowledge, different world perspectives, increased acceptance), but there are also some drawbacks (e.g., social pressure, domination, goal displacement, group thinking) [7].

Group Decision Support Systems (GDSS) intend, as we shall see, to support collaborative work. In this work we will call “meeting” to all the processes necessary to the completion of a specific collaborative task. A meeting is a consequence or an objective of the interaction between two or more persons [8]. Physically, a meeting can be realized in one of the four scenarios: same time / same place, same time / different places, different times / same place and different times / different places. Each one of these scenarios will require from the GDSS a different kind of support.

Until now we discusses collaborative work and present group members as the only persons involved in the process, however, it is very common to see a third element taking part in the course of action, the facilitator. The meeting facilitator is a person welcomed by all the members of the group, neutral and without authority to make decisions, which intervenes in the process in order to support the group in the identification of a problem and in the finding of a solution, in order to increase group efficiency [8].

According to Dubs and Hayne [9], a meeting has three distinct phases, as it is depicted in Figure 2.

Fig. 2. Meeting Phases

In the Pre-Meeting phase the facilitator prepares the meeting, i.e., establishes the meeting goals, proceeds with the group formation (being in no doubt that all the participants have the necessary background), selects the best supporting tools, informs the meeting members about the goals and distributes the meeting materials.

In the In-Meeting phase the participants will be working in order to accomplish the meeting goals, and the facilitator has the task of monitor the elapsing of the meeting (e.g., to observe the relationship between the group members) and to intervene if necessary.

In the Post-Meeting phase, it is important to evaluate the results achieved by the group, as well as by how much each group member is acquit with the achieved results (satisfied/unsatisfied). Still, in this phase, it is very important to identify and store information that can be useful in future meetings (e.g., how to actualize the participant's profile for future selection).

Quite a few modules compose the VirtualECare Group Decision Architecture as it is depicted in Figure 3:

- Setup module – will be operated by a facilitator during the pre-meeting phase that will do several configuration and parameterization activities;
- Multi-criteria module – will be operated by a facilitator during the pre-meeting phase, being in charge of the definition of the evaluation criteria and scales and, eventually, in deleting dominated alternatives;
- Argumentation module - This module is based on the IBIS (Issue Based Information System) argumentation model developed by Rittel and his colleagues in the early 70's. According to this model, an argument is a statement or an opinion, which may support or pointed out one or more thoughts (Figure 4).
- Voting module - This module is responsible for allowing each intervenient of the decision group component to "vote" for his/her preferred option, normally the one most similar to his/her "belief" (Figure 4).

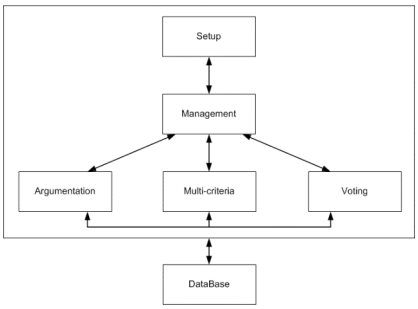

Fig. 3. VirtualECare Group Decision Architecture

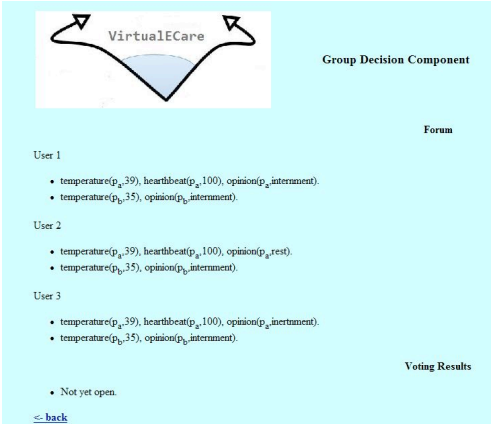

Fig. 4. Forum

4. Implement Issues

The Group Decision is made of different machinery, which must be interconnected, despite their differences. As the decision can be based on information obtained from physical sensors, software agents, external entities and other sources, finding a way of unifying these heterogeneous sources of information is a very important task. To address this challenge, we adopted a service-based architecture relying on OSGi [10] (Figure 5).

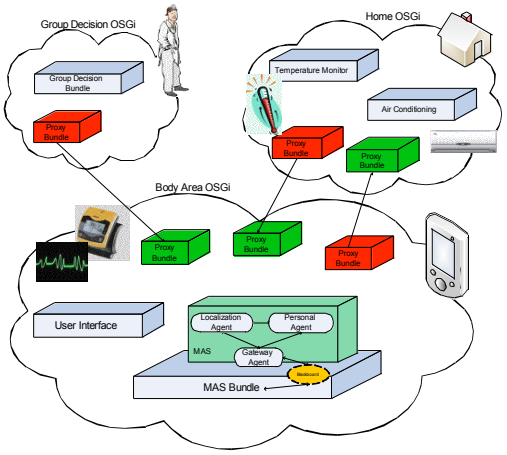

Fig. 5. Architecture from the OSGI point of view

OSGi is an initiative that intends to establish standards in Java Programming [11, 12], more specific, allow the sharing of Java Classes through the adoption of the services paradigm [11, 12]. The use of this technology allows developers to build Java Applications in a modular basis. The resulting modules are called bundles, which may, not only to provide services, but also to use services provided from other bundles. In OSGi, a bundle can be installed, started, stopped or uninstalled in runtime and without the necessity of any "system" reboot.

Through the adoption of the OSGi specification, we intend to adopt standards that will not only fasten the development phase but also greatly improve the dynamics of the architecture. It also makes it easy to add new components to our architecture as long as they are OSGi compatible.

Following this standard, we created the concept of OSGi cell. An OSGi cell is a collection of bundles, which provide and use services needed for accomplishing their tasks. In this specific case, the conclusion of the group decision is taken through the interaction of the bundles inside the group decision cell. Moreover, we hide the different components needed for the group decision (Multi-Agent Systems, physical sensors, among others) behind bundles, so they can interact among themselves.

4.1 - Unifying the sources of information

As assumed before, the sources of information may, and probably will, be very heterogeneous. Therefore, the exchange of information among the several parties is a difficult task. By "hiding" each one of these components behind bundles, we ensure that they will all speak the same language. That brings us, however, the additional work of making each different component bundle compatible. Our objective is to make each bundle able to offer services and functionalities of the component "behind" it, and provide those services to any other component that may need them, to ensure its normal operation.

4.2 - Sensors

A physical sensor for instance does not need, under this setting and approach, to use any service from other workings, since it simply provides the information it reads from the environment. Therefore, bundles, which implement sensors, are just required to present the service on which the sensor is specialized. This is the simplest bundle which, each time the service is requested, reads the value that the sensor makes available on that particular moment (real time), converts it to a standard format (e.g. float, integer, etc.) and returns that value as the result of the service invocation. Indeed, we are able to build a meteorological station bundle in which there are several outside sensors, such as wind, rain, luminosity and humidity sensors. The meteorological station bundle can receive the information relative to all of the sensors and the register services, each one corresponding to one of the sensors. In this way, any other bundle of the OSGi cell can use the information from the weather forecast, simply by requesting the respective service from the meteorological bundle.

4.3 - Human Experts

Another elements that make part of the GDSS are the human experts. These are specialized assets that may apply their knowledge and experience in specific areas on the decision making process. We can think of a physician, as an example. The physician is faced with a set symptoms presented by his/her patient, the respective clinical history, sets the list of problems and provides the respective plan of action (treatment). In this case, the bundle behind the physician is quite complex. Whenever the physician is asked to present a diagnose, he/she must have access to his/her Clinical Electronic Process) [13]. Therefore, the physician bundle must contact the relevant services (such as services provided by the patient's personal agent or by the hospital's clinical registry) and provide the relevant information to the physician. This bundle, besides registering the physician service, also needs to draw on services from other bundles, bundles that may, eventually, be outside the OSGi cell (e.g. the personal agent of the patient is probably running on its hand held device) (the interface between the physician and the bundle may be as simple as a GUI that shows the data acquired from other bundles and where the physician may write its conclusions).

4.4 - Multi-Agent Systems

Multi-Agent Systems are a key component in a GDSS platform and we must have a way of integrating such technology with the rest of the bundles. The objective is to allow an agent to run in a way that its functionalities (e.g. methods) can be provided as services to other bundles. It would not be advisable to convert each agent into an OSGi bundle since that would increase development time and throw away the advantages of MAS. Our choice was to create a bundle behind each agent, the MAS bundle. This bundle can deal with one or more agents and implement the methods declared in

the interface of those agents as services of their own. Moreover, this bundle must be able of start and stop agents, which will correspond to the start and stop of the services being provided by those agents. The bundle is responsible for, after receiving an invocation for an offered service from any other bundle, to forward the invocation to the correspondent agent and delivering the result of the invocation. Note that an agent, when trying to satisfy a method invocation, can possibly need to use services from other bundles that are currently available, so agents can also access services through this MAS bundle.

We are now ready to make a more detailed description of the MAS bundle. It has two methods for controlling the bundle that will be used by the client or administrator to start or stop the bundle. This corresponds to starting a container and the necessary agents and killing them, respectively. As this MAS bundle registers the services of the agents it creates, it declares the public methods of the agents on its interface so that they are visible to the other bundles as regular services. As for the interface between the MAS bundle and the Jade System, a JadeGateway agent is being used. This agent is created when the bundle is started, along with the other agents. Its task is to act as a bridge between Jade and non-Jade code. Whenever a request from a service arrives to the MAS bundle, it knows to which agent that request should be forwarded. A shared object (which we call blackboard and contains some fields such as the name of the recipient agent or the method to be invoked) is created and sent to the gateway agent. The gateway agent then contacts the recipient agent, invoking the requested method and waiting for the answer. After having received the answer, it is written to the blackboard and the command is released. At this point, the MAS bundle resumes the execution and the result of the invocation can be read from the blackboard and returned as the result of the invocation of the service from another bundle. Likewise, if an agent needs to use a service from another bundle, it contacts the MAS bundle, which is responsible for contacting the correct bundle; invoking the service and forwarding the result back to the agent. Finally, the MAS bundle implements a method for shutting down the agent container, which corresponds to uninstall all the services being provided by the agents.

The more specific issue of the interaction between agents inside each platform is also addressed. Agent communication is indeed a very important subject since it implies directly with the performance and behavior of the whole system. FIPA establishes several agent-related standards, being one of them the Agent Communication Language (FIPA-ACL). This standard defines how to syntactically and semantically construct a message. It specifies the parameters a message should have (e.g. sender, content, performative) and how to use them. The communication between the agents of our architecture complies with FIPA-ACL standard. By doing so, we solve some challenges and enlarge the compatibility of our architecture with outside agents that follow the same standard. At this point, any agent that complies with FIPA-ACL can run inside a container that is controlled by a MAS bundle and at the moment it starts, the methods declared on its interface are added as one more service provided by the bundle.

5. Interconnecting

We have already seen how to adapt some of the most important components in our GDSS platform in order to be OSGi compatible and how to enable them to communicate. There must be, however, a way for users of this platform to access the service or services that are, eventually, remotely provided. This is where R-OSGi comes into play.

The main idea behind R-OSGi is for remote services to be accessed by bundles, like they would be if they were local ones, in a completely transparent way. What we do is to add an additional bundle to each cell that provides at least a remote service. This bundle is responsible for checking the service registry of the OSGi cell it is in, and searches for services which should be provided remotely, and announce them in an external port. Remote bundles, which want to subscribe its services, will make a connection to that port and subscribe it. Moreover, each cell should also start a bundle for each service (or bundle of services) it wants to remotely access. This bundle subscribes the remote service as soon as it is needed and registers it in the local OSGi cell, as if it was the bundle providing it. After that, any local bundle can use the service without the need to know if it is accessing a remote or a local service. When adopting R-OSGi, we not only have a way of providing the group decision service. We also expand the group decision capacity, since the main bundle or the other bundles will be able of using external resources (bundles in other OSGi cells), if needed.

It is now clear that the service of the GDSS should be made remotely available, so that the users can have access to it through R-OSGi. There is, therefore, a main bundle on which the decision is made. This bundle registers the main service of the GDSS and the proxy bundle starts providing it remotely. Behind this main bundle of course, there can be a MAS, an expert or any other mechanism for making the decision. When the service is invoked, this decision mechanism, through the main bundle it is in, contacts the relevant bundles (local or remotes), collects the data it needs to make a decision and returns the result of the judgment.

6. Conclusion

In the healthcare arena one aims at a distinguished deliverance of healthcare services to the population in general, and the elderly in particular, without delocalizing or messing up with their routines. Indeed, in this paper it is described a VirtualECare project, with special incidence on its GDSS features, that support asynchronous and distributed meetings, aiming at multi-criteria decision problems, in order to answer to questions being referred to above.

We have also show how we use OSGi to implement a GDSS mechanism, on which we may integrate several different components, enabling them to work together. Moreover, by using R-OSGi, we are able to remotely provide this service and extend the group decision abilities by enabling it to use external services, greatly expanding its potential.

References

1 'Healthcare 2015: Win-win or lose-lose?', in Healthcare 2015: Win-win or lose-lose?' (IBM Global Business Services, 2006.
2 Riva, G.: 'Ambient Intelligence in Health Care', CYBERPSYCHOLOGY & BEHAVIOR, 6, (3), 2003
3 Marreiros G., Novais P., Machado J., Ramos C. e Neves J., An Agent Based Approach to Group Decision Simulation using Argumentation, in Proceedings of the Agent Based Computing: Workshop III - ABC 2006, Wisla, Poland.
4 Costa R., Novais P., Machado J., Alberto C., Neves J., Inter-organization Cooperation for Care of the Elderly, in Integration and Innovation Orient to E-Society, Wang W., Li Y, Duan Z., Yan L., Li H., Yang X., (Eds), Springer-Verlag,Series: IFIP International Federation for Information Processing, ISBN: 978-0-387-75493-2, 2007.
5 Camarinha-Matos, L.M., Castolo, O., and Rosas, J.: 'A multi-agent based platform for virtual communities in elderly care', 2003.
6 Camarinha-Matos, L.: 'New collaborative organizations and their research needs', 2003.
7 Marreiros G., Santos R., Ramos C., Neves J., Novais P., Machado J. and Bulas-Cruz J., Ambient Intelligence in Emotion Based Ubiquitous Decision Making, International Joint Conference on Artificial Intelligence (IJCAI 2007) - 2nd Workshop on Artificial Intelligence Techniques for Ambient Intelligence (AITAmI'07), Hyderabad, India, 2007.
8 Schwarz, R.M.: 'The Skilled Facilitator: Practical Wisdom for Developing Effective Groups' (Jossey Bass, 1994. 1994)
9 Dubs, S., and Hayne, S.C.: 'Distributed facilitation: a concept whose time has come?', 1992.
10 Carneiro, D., Costa, R., and Novais, P.: '(R-)OSGi: The VirtualECare Open Framework', 2008.
11 Initiative, O.S.G.: 'Osgi Service Platform, Release 3' (IOS Press, 2003. 2003)
12 Chen, K., and Gong, L.: 'Programming Open Service Gateways with Java Embedded Server(TM) Technology' (Prentice Hall PTR, 2001. 2001)
13 Machado J., Abelha A., Neves J. e Santos M., Ambient Intelligence in Medicine, in proceedings of the IEEE-Biocas 2006, Biomedical Circuits and Systems Conference, Healthcare Technology, Imperial College, London, UK.