Distributed Guideline-Based Health Care System

David Isern
Universitat Rovira i Virgili (URV)
Department of Computer Science and Mathematics
Tarragona, Catalonia (Spain)
Email: disern@etse.urv.es

Antonio Moreno
Universitat Rovira i Virgili (URV)
Department of Computer Science and Mathematics
Tarragona, Catalonia (Spain)
Email: amoreno@etse.urv.es

Abstract— **This paper proposes to combine *Medical Guidelines* with *Multi-Agent Systems* in order to obtain a system that could be used in a real medical environment for improving the current use/management of the resources. We propose in this paper an agent-based architecture, which allows to represent different entities in a medical centre and the relationships between them. A medical guideline specifies sequences of actions that could be performed in front of a particular pathology. The proposed system allows doctors to follow these guidelines when performing medical visits, and to access the results of clinical tests stored in an electronic medical record. Several service agents are connected to the physical resources providing clinical data of patients, which is added automatically to the medical record. All agents (internal of the system and user's personal agents) have their own knowledge and some roles to accomplish. This paper is focused on a healthcare domain, but this approach could be adapted to other domains. Moreover, another goal of this work is to illustrate that our system is flexible and scalable, and it could be improved with new services/features if required.**

I. Introduction

In a medical centre there are a lot of professionals such as nurses, doctors, administrators, etc., and resources such as x-ray, gym, rehabilitation unit, etc. All of them play a defined role in the organisation of any department where they are included. The coordination between different departments in the same hospital or between different hospitals is very frequently needed. We would like to emphasize three different characteristics of health care systems:

- *Heterogeneous data*: health care organisations exchange a lot of data from different sources (an x-ray image, a blood test, a medical visit, an electronic medical record, etc.) and it is important to integrate them properly.
- *Autonomy*: services, departments, clinical practitioners and patients are autonomous entities with their own knowledge, beliefs and goals.
- *Complex coordination*: There are several kind of interactions to coordinate: human-resource, resource-resource and human-human. This coordination has temporal constraints which must be considered.

Before going on, it could be interesting to analyse the importance of information technologies in the health care domain. The example of Heidelberg University Medical Center in Germany was depicted in [8]. In that hospital, approximately 8000 employees care for approximately 50 000 inpatients and 250 000 outpatients each year. The Heidelberg University Medical Center encompasses over 100 wards, approximately 1700 beds and 70 outpatient units. The yearly gross budget is approximately 500 million Euro. Each year approximately 250 000 physician letters, 20 000 surgical reports, 30 000 pathology results, 100 000 microbiology results, 250 000 radiology results, and 1 000 000 clinical chemistry results are generated. Around 300 000 new medical files are set up each year containing approximately 6 million documents. Files are generally archived for 30 years. Currently, the data volume for digital storage is estimated at 5 terabytes per year, tendency increasing due to new radio-diagnostic methods. Approximately 1000 archived medical records are accessed each day. Over 3000 computers are networked, including approximately 40 larger computer systems. This allows integrated, computer supported information processing in the areas of hospital management, diagnostic ancillary services, on wards and outpatient units. Over 2000 computers serve physicians and nurses at their work places on these wards and outpatient units as health care professional workstations. These systems provide health care professionals with access to most of the mentioned documents as part of an electronic medical patient record, to record data, or to allow the creation of new documents. Consequently, applied medical informatics research could ensure/improve efficient, medically advanced and yet affordable future health care systems.

This paper proposes to combine *medical guidelines* ([23]) and *multi-agent systems* ([24]) in order to obtain a system that could be used in a real environment for improving the current use/management of the resources. Moreover, another goal is to illustrate that our system, in essence, is flexible and scalable, and it could be improved with new services/features if required. The following sections describe the main features of our system and the open issues that can be studied.

The rest of the paper is organised as follows. First of all, section II describes what is a medical guideline with an example, and comments the current available tools to create and store it. Section III describes the application of multi-agent systems in health care domains and its main benefits. After that, section IV describes the agent-based system called HeCaSe2 that combines agents and medical guidelines in order to improve the collaboration and coordination between services and humans. Finally, section V gives some conclusions and some future lines of research.

II. Medical Guidelines

A Medical Guideline (from here on, GL) is a representation which stores a flow of operators and actions to handle a specific pathology. A GL is created by experts and it is published/shared as a document in different forums such as Open Clinical[1]. These published GLs contain all available information (instructions, sentences, recommendations, actions, decisions, etc.) related to a specific pathology using a textual-based representation. At this point, as our work attempts to computerise GLs, we need to have a structured-based representation. Recently, this fact is being tackled by many researchers with successful results and nowadays there are different widely used languages such as Asbru, *PROForma*, GLIF, PRODIGY, etc. ([16], [23]). There are also several tools available for creating and storing GLs such as TALLIS (based on PROForma, [20]), Protégé-2000 ([18]) and GLARE ([21]).

A GL is a suggestion for doctors to solve a problem. To follow a GL is not mandatory and sometimes doctors may change the actions to be performed, depending on the situations, or they could also change the observations to be made if a value is not available. For these reasons, different researchers have proposed to create GLs from the experience, in a field called evidence-based learning ([17]). This approach could be used to create flexible GLs adapted to each doctor or pathology, using data mining techniques.

Studies have shown that computer-based clinical decision support systems (DSS) may improve clinician performance and patient outcomes. Guideline-based clinical DSSs have been proposed for this purpose ([23]).

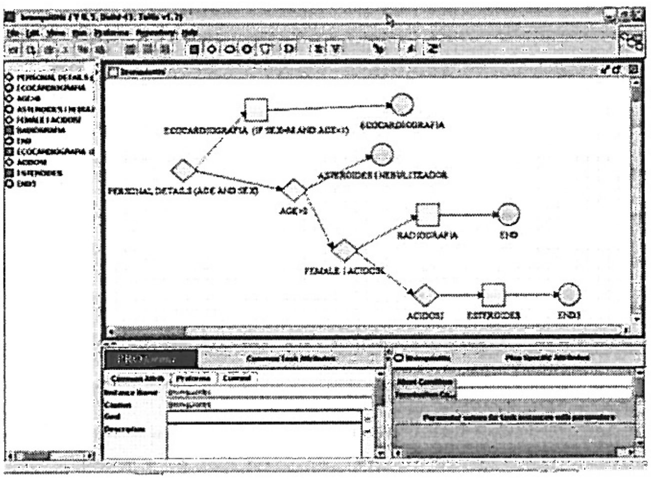

Fig. 1. GL using TALLIS: actions, enquiries and decisions

A. PROForma

As it has been reported in the previous section, several approaches have been proposed to design knowledge representation systems that make possible to support clinical procedures. One of them is PROForma, which was developed by the Advanced Computation Laboratory at the Imperial Cancer Research Fund (now Cancer Research UK) under the EU 4th Framework PROMPT project (1996-1999). Under the name of Arezzo, it is one of the standards to specify clinical guidelines.

PROforma supports four basic classes of tasks:

1) *Plan*: a sequence of several sub-tasks or components. Plan components usually have an order to specify temporal, logical or source constraints.
2) *Decision*: is used for choosing a candidate from a given set, using arguments pro and contra.
3) *Action*: a procedure that has to be executed outside of the computer system, like an injection.
4) *Enquiry*: requests information needed to execute a certain procedure.

In Fig. 1 is depicted a screenshot of TALLIS showing a guideline that covers respiratory pathologies as bronchiolitis ([17]).

Data definitions (in an object oriented point of view, classes and methods) enable PROforma to communicate with external elements (databases, agents, web services). With the support of data definitions (semantics) from the protocol defined in the guideline, it is possible to define the kind of data associated with each data item. Besides, the protocol defines which values it can accept and where the data comes from (following preconditions and postconditions in each case). The data definitions are specified in a separate file that is loaded when starting the protocol [22]. In some cases, a GL can specify a timeout for executing critical medical actions; for instance, a biopsy has to be performed before two days.

III. Agent-Based Health Care System

In the last years, several researchers have applied the agent paradigm in medical informatics. Most of the actual agent-based technologies in medicine could be classified, following [19], as a) *Patient centred information management* (e.g. [2]), b) *cooperative patient management* (e.g. [10]), c) *patient monitoring and diagnosis* (e.g. [12]), and d) *remote care delivery* (e.g. [11]).

The adequacy of agent-based systems to health care problems was analysed in [15]. The main reasons to use agents in this kind of problems are the following:

- *Physical distribution of data*. To distribute data around several machines allows to distribute the knowledge.
- *Proactivity*. An agent can perform a task that may be beneficial for any agent/user. For instance, the user's personal assistant may keep, in the user's health profile, the information that he/she has had heart conditions in the past. When the user travels to a foreign city, the personal assistant could immediately (without an explicit demand from the user) search all those medical centres in which there are heart specialists, and have this information ready in case the user needs it.
- *Social behaviour*. The agents are designed to collaborate and coordinate their actions in order to solve a complex task. In the real world, humans and services play the same role.

[1] Website: www.openclinical.org

- *Autonomy.* From the definition of agent, it runs autonomously and with independence from the rest of the system - if it needs something, it can request it to other agents -. Any agent has its own knowledge, roles and goals to accomplish.
- *Decomposition of a complex problem in sub problems.* In MAS there are techniques of distributed problem solving.
- *Information retrieval.* Information agents and interface agents can allow the retrieval of data from different sources and, if it is necessary, to integrate it into a desirable format.

IV. HeCaSe2: Providing Distributed Guideline-Based Health Care Services

HeCaSe2 is an agent-based system based on its predecessor, HeCaSe. First of all, we shall summarise the main characteristics of HeCaSe and the main goals of this new version. After that, we will analyse in detail the proposed architecture and the services provided by the system.

A. Overview

Health Care Services (HeCaSe) is an agent-based prototype that models the medical institutions of Catalonia with the following characteristics ([9], [13], [14]):

- It provides a decomposition of the problem that allows to model the real entities of the medical domain as agents; it is based on the structure of medical centres in Catalonia [3] (each centre has a set of departments, and each department has a set of doctors).
- It implements an ontology for the medical domain.
- It implements security measures that ensure the confidentiality and privacy of medical data, such as Secure Socket Layer (SSL) for message ciphering, Public Key Infrastucture (PKI) to manage the keys, and Hash functions for authentication and access control of the user.
- It develops agent services as reusable as possible by: (1) using standard languages for agent language communication, and for the content and representation of the knowledge domain, and (2) providing detailed service models to describe the individual functioning and objectives of each agent, including descriptions of actions, protocols used and example messages.
- It provides mechanisms to allow the user the configuration of his user agent, creating the user's profile. This information is used to guide a negotiation or to make proposals according to the user's preferences.
- To provide ubiquitous access to the system from anywhere, any time.

One of the goals of this paper is to make a step forward and provide real coordination and cooperation between humans and medical services using guidelines (GLs). In order to achieve this goal, we developed an architecture that defines several kinds of agents and their roles.

As explained in section II, a guideline is a representation that stores sequences of actions, enquiries and decisions to take in front of a patient with a certain pathology. In a real scenario, when a doctor performs a medical visit, it is frequently necessary to check previous results stored in the medical record; for instance, if a doctor wants to ensure if a patient has appendicitis, he needs to explore the patient and also to know the level of white corpuscles. Any blood test is made by a service (identified in the system as service agent that can perform blood tests) that receives the results from a laboratory. The doctor will arrange an appointment between the patient and that service through their agents, Service Agent and User Agent respectively (see Fig. 2). Finally, in another visit, the doctor will check all the results sent by the SA and may take a decision about the possibility of the patient to have appendicitis.

Although this is a simple example, we would like to emphasise two important aspects:

- On the one hand, the use of GLs can improve the quality of services given by doctors, because it allows to retrieve all data related to the patient in a structured way. The doctor can take decisions depending on the data available at the moment.
- On the other hand, a doctor needs to collect different results and supervise the actions/decisions of the GL. A GL is a guide but it is not mandatory to be followed. Sometimes, doctors change enquiries or actions depending on the available data. These changes are not a problem, and they provide valuable information that could be used in the future to propose changes in the GL, learning from the experience. The doctor's personal agent (DA) will send all the events related to any GL to a Guideline Agent (GA) in order to monitor/save all changes.

B. Description of the MAS

The basic architecture of the MAS is depicted in Fig. 2. The architecture shows the agents defined, the interactions among them, and also the interactions between humans/resources and agents.

At the top of the architecture is placed the user, who interacts with our system through a User Agent (UA). This agent stores static data related to the user such as the national healthcare number, name, address, phone number, and information for allowing a secure access to the system (login, password, keys). It also stores dynamic data such as the agenda of the user. The static data will be used to identify the user in the system (authentication and ciphering). The agents of the system will exchange required data automatically in each step, *e.g.* a doctor needs to know personal details of a patient before the medical visit, in order to retrieve his/her medical record from a database. The dynamic data is very useful to guide negotiations between any UA and other agents, because an UA can avoid coincidences in those negotiations, *e.g.* if the user works from 9:00AM to 14:00PM, his agent would arrange meetings during the afternoon and night. Moreover, we want to provide pro-activity to this agent in order to provide healthcare warnings to the user in pre-defined tasks.

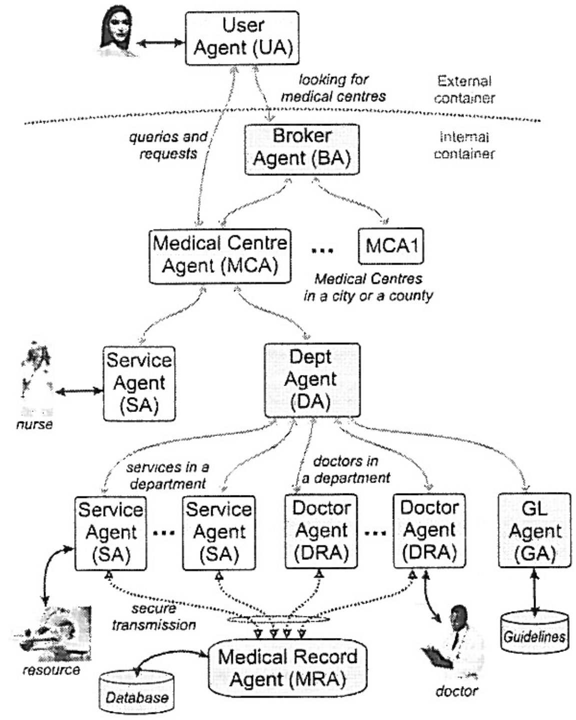

Fig. 2. MAS Architecture

All UAs can talk with a Broker Agent (BA). The BA is the bridge between users and the medical centres, and it is used to discover information about the system. All UAs can ask this agent in order to find medical centres satisfying certain criteria. The BA covers the medical centres located in a city or in an area.

Any user can access the system through the Medical Centre Agent (MCA) that centralises and monitors the outsiders accesses. This is an open architecture and, depending on the situation, there will be more or less agents of each type, and more or less interaction between them. In a general case, a MCA monitors all of its departments, represented by Department Agents (DAs), and a set of general services (represented by Service Agents (SAs)), such as a blood test service. Then, each department is formed by several doctors (represented by Doctor Agents (DRA)) and more specific services (also modelled as Service Agents). Moreover, in a department there is a Guideline Agent (GA) that performs all actions related to guidelines, such as to look for a desired GL, to store and/or to change a GL made by a doctor (this will allow us to learn future changes of GLs), etc. This GA contains only GLs related to the department where it is located (the knowledge is close to the entity that will use it).

At the bottom of the architecture there is the database agent which stores all patient medical records in the medical centre, called Medical Record Agent (MRA). This agent provides a *secure* access to the data using authentication and ciphering through a Public Key Infrastucture (PKI) ([14]).

C. Scheduling services

When a user wants to arrange an appointment with a doctor, or a doctor must arrange a visit of a patient with a service, it is required to schedule a meeting according to different constraints (timetable of services or doctors, and agenda of the user).

In Fig. 3, the services and functionalities provided by the agents in a department are depicted. Following the example of appendicitis, the doctor retrieves the GL of that disease and observes that he needs to know the temperature and the level of white corpuscles. The doctor can take the temperature *in situ* and then he will input the data to the system (updating the medical record) but he needs an external test. The DRA automatically looks for a SA that can provide this data. When the DRA finds the required SA, it negotiates the best date to perform the test, taking into account any constraint of both the user (agenda of the user stored in the User Agent (UA)) and the service (timetable).

Any SA is linked to physical resources (e.g. an x-ray machine, a blood analyser) or humans (e.g. a nurse), and both provide data requested about a patient through its agents when it is required. In the case of humans, we can use a computer connected to the system or a mobile device. In the case of doctors the same paradigms may be used, and the DRA could be stored in a mobile device (with less features) or in a computer (by default).

As we said before, in some cases a GL can specify a timeout for executing critical medical actions. If there is a timeout, it must be considered as a constraint in the scheduling process. Through the negotiation process, the doctor can notice if a test can be made before the timeout or not and, if it is not possible to achieve, he could consider another alternative action to be performed. If this architecture is adopted by several medical centres, when a doctor requires a test, the DRA looks for the service in the same medical centre but also in another medical centres. Fig. 4 shows how a Doctor Agent (DRA) negotiates an appointment with a Service Agent (SA_2) located in another medical centre.

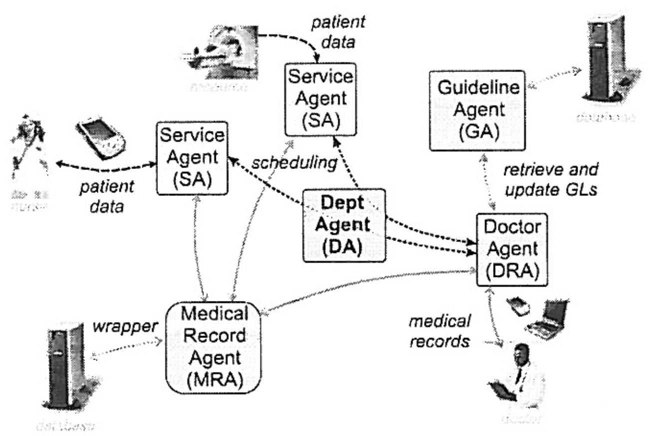

Fig. 3. Services implemented in a department. Interaction of agents with humans and resources.

To discover a required Service Agent is a process that is achieved automatically through different agents. Each SA has a description that serves as a comparison between the required action and the provided one. See Fig. 4. For instance, as it is shown in Fig. 4, if a doctor (DRA) in the medical centre MCA_1 looks for a service SA_2 located in another medical centre MCA_2, the DRA begins the search asking, in the first place, to the MCA_1 (through the department DA_1). As the service is not found, the MCA_1 automatically asks the Broker Agent in order to find a medical centre in the city with the service SA_2. When the broker finds MCA_2, this agent forwards the request to its Service Agent (SA_2) that answers the request and the message comes back to the DRA through the same agents.

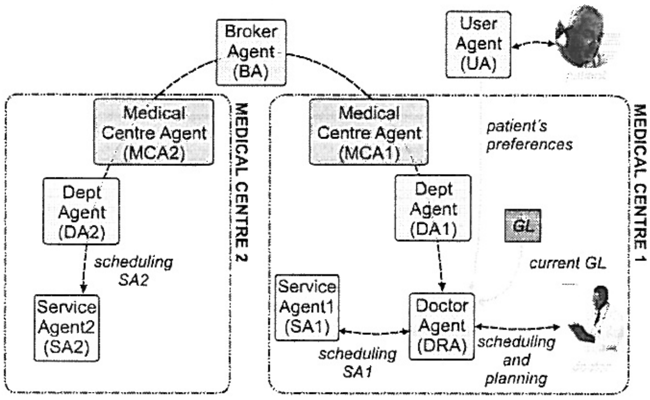

Fig. 4. Scheduling services located in different departments

D. Knowledge sharing: metadata and rules

As we said in previous sections, all agents exchange data as required. All agents need to know exactly the rules of the messages and the meaning of the data received/sent. For these reasons, we need 1) an agent communication language, and 2) an ontology to provide semantic information to the data.

First of all, an agent communication language is a set of rules that define how the information is exchanged among agents. Our system uses the language defined by FIPA, called FIPA Agent Communication Language (FIPA ACL) [7]. Moreover, from the set of available protocols, we have used the *FIPA Request Interaction Protocol* to send a simple action request, the *FIPA Query Interaction Protocol* to ask something and, finally, the *FIPA Contract Net Interaction Protocol* to negotiate appointments among two or more agents [6].

The main goal of an ontology is to share the knowledge about a given domain among several entities (in our case, agents, resources and humans). An ontology is a set of primitive concepts that we can use for representing a whole domain (or a part of it) for which the ontology was designed. Any ontology defines a set of terms (vocabulary), a structure of the statements in the model, and a semantic interpretation of these terms [1].

Attending the scope of validity, following [1], ontologies can be classified as:

i) *General ontologies* - are definitions of general concepts that have a wide scope and are applicable for several domains.

ii) *Domain ontologies* - are knowledge domains for a specific area.

iii) *Task ontologies* - are the most specific type of ontology.

For example, if a SA sends a document containing a blood test, the message contains information about the patient, date when the test was made, information about the service, a description of the test (*e.g.* white corpuscles, red corpuscles), the guideline identifier, and the set of values for the test. In that example, we used all three ontologies, (a) the general ontology defines data structures as date or the type of identifier for the guideline, (b) the domain ontology defines the concepts related to the medical domain, such as the term corpuscle (with its variations), blood test, and (c) the task ontology describes terms and concepts for individual problem-solving methods. In the example, these terms are not explicitly depicted, but as this message is the result of a previous negotiation with preferences/constraints, that kind of negotiation is included in that type of ontology.

Nowadays, this area is in constant evolution and there are different languages available to represent ontologies. Also, different graphical tools to make easy both creating and updating ontologies have been developed ([5]).

We used Protégé-2000[2] as ontology editor and Resource Description Format (RDF) as standard ontology representation language ([4]). Moreover, Protégé-2000 allows the programmer/user to extend the ontology if it is required, and also allows to export the ontology to another language such as DAML+OIL[3]. OWL[4].

V. Conclusion

The use of agents in health care has experimented an important growth in the last years. One of the main benefits of this paradigm is to allow the interoperability of pre-existing systems for improving its general performance.

We have designed an agent-based architecture respecting the health care national organisation, but it could be adapted to other situations. The architecture defines the interaction between agents but also between humans and agents. The interaction human-agent is made through personal agents that could be located in computers or mobile devices. The implementation of the first version of the Multi-Agent System was made using a FIPA-compliant tool called JADE[5]. It also has a plug-in to develop agents in mobile devices called JADE-LEAP.

[2]Protégé-2000 is a free tool made by Stanford Medical Informatics and available at http://protege.stanford.edu

[3]DAML+OIL is the evolution of RDF and allows to define more specific relations among concepts. As RDF, DAML+OIL is a structured language based on XML

[4]Ontology Web Language (OWL) is the latest language defined by the W3C Consortium, following DAML+OIL

[5]Java Agent DEvelopment Framework is a FIPA-compliant set of libraries that facilitates the creation of MAS. It can be downloaded from http://sharon.cselt.it/projects/jade

This paper explains the combination of agents with medical guidelines in order to provide coordination among different types of agents. Using a GL, a doctor can consider all the available data and take an informed decision. If the doctor needs another test, the agents negotiate the best alternative according to the preferences/constraints of the user, the doctor and the services. The services can be located in the same medical centre or in another one (with a previous discovery process).

The main features of this system are pro-activity and autonomy. The architecture defines a set of roles for each entity in our system (resource or human), but we suppose that these entities are autonomous with their beliefs, desires and intentions. Agents act autonomously but also carry out actions in advance when it is required; for instance, before performing a medical visit, the DRA collects all information available about a patient.

The system implements services as reusable as possible by using standard languages for communicating and for the content and the representation of ontologies. It could easily allow the addition of new agents or features.

ACKNOWLEDGMENTS

This work is partially supported by the Universitat Rovira i Virgili project 2003-ACCES-09. The authors would like to thank the Cancer Research Lab. (UK) for allowing them the use of TALLIS. The authors also want to acknowledge the rest of colleagues in the department for their feedback, especially D. Sánchez and A. Valls.

REFERENCES

[1] M. Becker, C. Heine, R. Herrler, and K.-H. Krempels. OntHoS - an ontology for hospital scenarios. In John L. Nealon and Antonio Moreno, editors, *Applications of Software Agent Technology in the Health Care Domain*, Whitestein Series in Software Agent Technologies, pages 87–104. Birkhäuser Verlag, Basel, Switzerland, 2003.

[2] J.-P. Calabretto, D. Couper, B. Mulley, M. Nissen, S. Siow, J. Tuck, and J. Warren. Agent support for patients and community pharmacists. In *35th Annual Hawaii International Conference on System Sciences (HICSS'02)*, Hawaii, USA, Jannuary 7-10 2002. IEEE Press.

[3] Generalitat de Catalunya. Servei Català de la Salut (ICS), 2003. http://www.gencat.net/scs.

[4] W3C Consortium. Resource Description Framework (RDF) - W3C Semantic Web Activity, 2003. http://www.w3.org/RDF/.

[5] O. Corcho, M. Fernández-López, and A. Gómez-Pérez. Methodologies, tools and languages for building ontologies. where is their meeting point? *Data and Knowledge Engineering*, 46(1):41–64, July 2003.

[6] FIPA. FIPA Communicative Act Library Specification. Technical Report Spec. number XC00037J, Foundation for Intelligent Physical Agents (FIPA), October 3, 2001.

[7] FIPA. Foundation for Intelligent Physical Agents (FIPA), 2003. http://www.fipa.org.

[8] R. Haux, E. Ammenwerth, W. Herzog, and P. Knaup. Health care in the information society. a prognosis for the year 2013. *International Journal of Medical Informatics*, 66:3–21, 2002.

[9] D. Isern, D. Sánchez, A. Moreno, and A. Valls. Personalisation of medical services. In I.Rudomin, J.Vázquez-Salceda, and J.L.Díaz de León Santiago, editors, *e-Health: Application of Computing Science in Medicine and Health Care*, Research on Computer Science, pages 18–31. Instituto Politécnico Nacional, Mexico DF, Mexico, 2003.

[10] S. Kim, R. Herrler, C. Heine, and K.-H. Krempels. Agent.Hospital - agent-based open framework for clinical applications. In *Twelfth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WET ICE 2003)*, Linz, Austria, June 9-11 2003. IEEE Computer Society.

[11] Stefan Kim. Ubiquitous healthcare: The onkonet mobile agents architecture. In Andreas Koop Hans-Bernd Bludau, editor, *Second Conference on Mobile Computing in Medicine, Workshop of the Project Group MoCoMed, GMDS-Fachbereich Medizinische Informatik und GI-Fachausschuss 4.7*, volume 15 of *Lecture Notes in Informatics*, pages 105–118, Heidelberg, Germany, April 11th 2002. Springer Verlag.

[12] G. Lanzola, L. Gatti, S. Falasconi, and M. Stefanelli. A framework for building cooperative software agents in medical applications. *Artificial Intelligence in Medicine*, 16(3):223–249, 1999.

[13] A. Moreno, D. Isern, and D. Sánchez. Provision of agent-based heath care services. *AI Communications. Special Issue on Agents in Healthcare*, 16, 2003.

[14] A. Moreno, D. Sánchez, and D. Isern. Security measures in a medical multi-agent system. In I. Aguiló, Ll. Valverde, and Escrig M.T., editors, *Artificial Intelligence Research and Development*, Frontiers in Artificial Intelligence and Applications, pages 244–255. IOS Press, Amsterdam, The Netherlands, 2003.

[15] J. L. Nealon and A. Moreno. Agent-based applications in health care. In John L. Nealon and Antonio Moreno, editors, *Applications of Software Agent Technology in the Health Care Domain*, Whitestein Series in Software Agent Technologies, pages 3–18. Birkhäuser Verlag, Basel, Switzerland, 2003.

[16] M. Peleg, S. Tu, J. Bury, P. Ciccarese, J. Fox, R. A. Greenes, R. Hall, P. D. Johnson, N. Jones, A. Kumar, S. Miksch, S. Quaglini, A. Seyfang, E. H. Shortliffe, and M. Stefanelli. Comparing models of decision and action for guideline-based decision support: a case-study approach (part 1 of 2)(part 2 of 2). Technical Report SMI-2002-0922 and SMI-2002-0923, Stanford Medical Informatics, Stanford University School of Medicine, 2002.

[17] R. Pujol. Generació de protocols i guidelines clínics a partir de l'evidència mèdica. Technical Report Projecte Final de Carrera, Enginyeria Tècnica Informàtica de Gestió, University Rovira i Virgili (URV), September 2003.

[18] R. D. Shankar, Samson W. Tu, and Mark A. Musen. Use of protégé-2000 to encode clinical guidelines. Technical Report SMI-2002-0944, Stanford University School of Medicine, 2002.

[19] V. Shankararaman, V. Ambrosiadou, T. Panchal, and B. Robinson. Agents in health care. In V. Shankararaman, editor, *Workshop on Autonomous Agents in Health Care*, pages 1–11. ACM-AAAI, ACM Press, 2000.

[20] D.R. Sutton and J. Fox. The syntax and semantics of the proforma guideline modeling language. *Journal of the American Medical Informatics Association (JAMIA)*, 10(5):433–443, Sep-Oct 2003.

[21] P. Terenziani, G. Molino, and M. Torchio. A modular approach for representing and executing clinical guidelines. *Artificial Intelligence in Medicine*, 23(3):249–276, November 2001.

[22] A. Vollebregt, A. ten Teije, F. van Harmelen, J. van der Lei, and M. Mosseveld. A study of proforma, a development methodology for clinical procedures. *Artificial Intelligence in Medicine*, 17:195–221, 1999.

[23] D. A. Wang, M. Peleg, S. Tu, E. H. Shortliffe, and R. A. Greenes. Representation of clinical practice guidelines. In *Medinfo*, London, UK, September 2-5 2001. IOS Press.

[24] M. Wooldridge. *An introduction to Multi-Agent Systems*. John Wiley and Sons, 2002. ISBN 047149691X.