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 environment for improving the current use/management of the resources. It has been defined an agent-based architecture, which allows to represent different entities in a medical centre and their relationships. A medical guideline specifies sequences of actions that could be performed in front of a particular pathology. This system allows doctors to perform medical visits by interpreting medical guidelines and to supervise the results collected from a medical record. Several service agents are connected to the physical resources providing data of patients, which is added 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 in other domains. Moreover, another goal of this work is to illustrate that our system, in essence, is flexible and scalable and, it could be improved with new services/features if required.

I. INTRODUCTION

In a real medical centre there are a lot of entities such as nurses, doctors, administrators, etc., and resources such as x-ray, gym, rehabilitation, etc. All of them play a defined role in the organisation of any department where they are included. Also, the coordination between different departments in the same hospital or between different hospitals is also present. This is a brief description of a real scenario but we would like to emphasize three different characteristics of these systems:

- **Heterogeneous data**: this organisation 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 required to integrate them.
- **Autonomy**: services, departments, people 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. Also, this coordination has temporal constraints which must be considered.

Before to continue, 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 to 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, apply medical informatics research could ensure/improve an 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 it is required. In the following sections it will be explained the main features of our system and the open issues that can be devised.

The rest of the paper is organised as follows. First of all, section II describes what is a medical guideline with an example and the current available tools to create and store it. Section III describes the application of multi-agent systems in health care domains and the 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 achieve/handle an specific pathology. A GL is created by experts and it is published/shared as a document in different forums such as Open Clinical1. These published GLs contain all available information (instructions, sentences, recommendations, actions, decisions, etc.) related to an specific pathology such as poliomelities, etc., using a textual-based representation. At this point, as our work attempts to computerise GLs, we need to have an 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 action/s to be performed, depending on the situations, or could also change observations if a value is not available. For these reasons, different researchers proposed create GL from the experience, also 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]).

III. AGENT-BASED HEALTH CARE SYSTEM

In last years, several researchers have been 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 delivering (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.
- **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.
- **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 into a desirable format.

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

A. PROForma

As it has been reported in the previous section, several approaches have been made to design knowledge representation systems that make possible to support clinical procedures. One of them was 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 (all subclasses of the main class task):

1) **Plan:** a sequence of several sub-tasks or components. Plan components usually have an ordering to specify temporal, logical or source constraints.
2) **Decision:** is used for choosing a candidate from a given set, using arguments pro an 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 screen shot 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, it is possible to define the kind of data with regards to the 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.
• **Proactivity.** An agent can perform a task that may be useful 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.

• **Autonomy.** From the definition of agent, it runs autonomously and with independence from the rest of the system - if it needs something, it could request it. Any agent has their own knowledge, roles and goals to accomplish.

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. Besides, we will analyse in detail the architecture proposed and the services provided.

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 Infrastructure (PKI) to manage the keys, Hash functions for authenticating 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 an step forward and provide real coordination and cooperation between humans and medical services using guidelines (GLs). In order to achieve these goals, we developed an architecture that defines several kind of agents and its roles.

As is explained in section II a guideline is a representation that stores sequences of actions, enquiries and decisions with a set of final steps. In a real scenario, when a doctor performs a medical visit, it is required to check previous results stored in the medical record; for instance, if a doctor would like 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 and 3). 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 or not appendicitis.

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

- On one hand, the use of GLs can improve the quality of services given by doctors, because it can retrieve all data related to the patient in a structured way. The doctor can take decisions depending on the data allowed 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 and/or actions depending on the available data. These changes are not a problem but provides a 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, all interaction among all of them, and also the interactions between humans/resources and agents.

At the top of the architecture is placed the user, which interacts with our system through their 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) and the agents of the system will exchange required data automatically in each step, i.e. a doctor need to know personal details of a patient before the medical visit, in order to retrieve their 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, i.e. if the user works since 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 warn something to the user in pre-defined tasks.
BA covers the medical centres located in a city or in an area.

Knowledge is close to the entity that will use it). Only GLs related to the department where it is located (the GL, to store and/or to change a GL made by a doctor (this allows 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 Infrastructure (PKI) ([14]).

C. Scheduling services

When a user wants to arrange an appointment with a doctor, or a doctor must to 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 observe 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 need 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 their 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 system or a mobile device such a Pocket PC. In the case of doctors, it could be used the same paradigm, 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 action to perform. 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 centre. Fig. 4 shows how a Doctor Agent (DRA) negotiates an appointment with a Service Agent (SA) located in another medical centre.

![Fig. 2. MAS Architecture](image)

![Fig. 3. Services implemented in a department. Interaction of agents with humans and resources.](image)
To discover a required Service Agent is a process that is archived 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\textsubscript{1} looks for a service SA\textsubscript{2} located in another medical centre MCA\textsubscript{2}, the DRA begins the searching asking, at first term, to their MCA\textsubscript{1} (through the department DA\textsubscript{1}). As the service is not found, the MCA\textsubscript{1} automatically asks to the Broker Agent in order to find a medical centre in the city with the service SA\textsubscript{2}. When the broker finds MCA\textsubscript{2}, this agent replies the request to its Service Agent (SA\textsubscript{2}) that responds the request and the message comes back to the DRA through the same agents.

**V. Conclusion**

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

We have designed an agent-based architecture attending the health care national organisation, but it could be adapted in 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 both in computers or/and mobile devices. The implementation of the Multi-Agent System was made using a FIPA-compliant tool called JADE\textsuperscript{3}. It also has a plug-in to develop wireless agents called JADE-LEAP.

---

\textsuperscript{2}Protégé-2000 is a free tool made by Stanford Medical Informatics and available at http://protege.stanford.edu

\textsuperscript{3}DAML+OIL is the evolution of RDF and allows define more specific relations among concepts. As RDF, DAML+OIL is a structured language based on XML.

\textsuperscript{4}Ontology Web Language (OWL) is the latest language defined by the W3C Consortium following DAML+OIL

\textsuperscript{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 type of agents. Using a GL, a doctor can consider all available data and take a decision. If the doctor needs another test, the agents negotiate the best alternative according 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 discovering process).

The main features of this system are the pro-activity and the 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 its beliefs, desires and intentions. Agents acts autonomously but also carry out actions in advance when is required, for instance, before to perform 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

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, specially D. Sánchez and A. Valls.

REFERENCES