ELSEVIER

Artificial Intelligence in Medicine 27 (2003) 259–282

Artificial Intelligence in Medicine

www.elsevier.com/locate/artmed

A multi-agent system approach for monitoring the prescription of restricted use antibiotics

L. Godo[a,*], J. Puyol-Gruart[a], J. Sabater[a], V. Torra[a], P. Barrufet[b], X. Fàbregas[b]

[a]*Artificial Intelligence Research Institute (IIIA), Spanish Council for Scientific Research (CSIC), Campus UAB, 08193 Bellaterra, Spain*

[b]*Mataró Hospital, Carretera de Cirera, 08304 Mataro, Spain*

Received 15 February 2002; received in revised form 4 November 2002; accepted 4 November 2002

Abstract

Hospitals have a specified set of antibiotics for *restricted use* (ARU), very expensive, which are only recommended for special pathologies. The pharmacy department daily checks the prescription of this kind of antibiotics since it is often the case that, after a careful analysis, one can get the same therapeutic effects by using *normal* antibiotics which are much cheaper and usually less aggressive. In this paper, we describe a multi-agent system to help in the revision of medical prescriptions containing antibiotics of restricted use. The proposed approach attaches an agent to each patient which is responsible of checking different medical aspects related to his/her prescribed therapy. A pharmacy agent is responsible for analyzing it and suggesting alternative antibiotic treatments. All these agents are integrated in a hospital distributed scenario composed by many different kinds of software and human agents. This patient-centered multi-agent scenario is specified using the design methodology of Electronic Institutions.

Keywords: Multi-agent systems; Electronic Institutions; Decision support systems

1. Introduction

Many effective anti-microbial drugs can be used nowadays to treat infectious diseases. They constitute one of the costliest categories of drug expenditures in hospitals, can cause

* Corresponding author. Tel.: +34-93-5809570; fax: +34-93-5809661.

E-mail addresses: godo@iiia.csic.es (L. Godo), puyol@iiia.csic.es (J. Puyol-Gruart), jsabater@iiia.csic.es (J. Sabater), vtorra@iiia.csic.es (V. Torra), pbarrufet@csm.scs.es (P. Barrufet), xfabregas@csm.scs.es (X. Fàbregas).

0933-3657/03/$ – see front matter
doi:10.1016/S0933-3657(03)00006-X

toxicity, and have implications on the microbiological ecology of the hospital. Among 20 and 40% of hospital patients are treated with antibiotics, which accounts for at least 25% of pharmacy department budget [19]. Their use can be inappropriate in terms of the antibiotic choice, dosage, route or length of treatment. In 1996, the Commission of Infectious Diseases of the Mataró Hospital[1] audited the most used restricted antibiotics (cefuroxime, amoxicillin-clavulanate, ceftriaxone, aztreonam, ciprofloxacin and cefoxitine). The percentage of correct antibiotic including choice of antibiotic, dosage, drug change according to bacteriological sensitivity tests, length of treatment and finally, the modification of drug, dosage and route according to clinical evolution, was 66% [13]. An inappropriate use of antibiotics can result in unnecessary exposure to medication, persistent or progressive infection, emergence of multi-resistant hospital pathogens that can produce super-infections and colonizations, and an increase of costs. Several strategies have been used to decrease the inappropriate use of antibiotics. The development of guidelines for the clinical practice, letting physicians get the feedback regarding their own antibiotic prescribing practices, and computer assisted decision support systems have been used effectively to improve antibiotic prescribing in hospitals [4].

It is of common practice in a hospital to have an antibiotic formulary which is annually updated and mailed to all professionals of the institution. The formulary lists the antibiotics available within the hospital, indicating therapeutic recommendations, dosage, dose intervals and special situations. In the formulary, antibiotics are distributed in two groups: those of free and restricted use.

Antibiotics of restricted use (ARU) are expensive drugs. Usually they are of wide spectrum although some of them are specific for the treatment of certain types of microorganisms. They are intended to be used for the empirical treatment of serious infections produced by microorganisms which are resistant to other kinds of antibiotics. Therefore, as soon as results of sensitivity tests are available, this kind of antibiotics should be replaced, when possible, by alternative active antibiotics with narrower spectrum and cheaper. These are the antibiotics classified as of free use.

In this paper, we describe the development of an agent-based decision support system to improve the treatment of infectious diseases which are initially being treated with antibiotics of restricted use. This application is to automatize the current process of revising and proposing alternative antibiotics which is manually carried out by the pharmacy department at the Mataró Hospital but which is very similar in most of the hospitals.

The paper is organized as follows. In the next section, we describe in detail the current (manual) revision procedure of antibiotic therapies since this will be the base for the development of our agent-based application. In Section 3, we review main related works in the literature on knowledge-based systems for monitoring antibiotic therapies. Most of them are based on the classical expert system approach. In Section 4, we argue the advantages of the agent-based approach in front of the traditional stand-alone (expert system) approach and propose for our application the development of a multi-agent system

[1] The Mataró Hospital, located near Barcelona (Spain), is a 344 bed (plus 14 of ICU) acute-care general hospital. In 2001, 15,048 patients were admitted, and 108,584 emergencies attended. The total expenditure in antibiotics was 300.506 EUR, which represents 21% of the total amount of drug expenditures.

integrated in a hospital environment, where each patient is attached a *guardian angel* agent taking care of the revision of their therapies. To do this, we view the application environment as an Electronic Institution, and use this model in Section 5 to design and specify an infrastructure where the application agents can interact. The basic components of the specified Electronic Institution (agents, roles, dialogic framework, scenes, performative structure) are described. Finally, in Section 6 we briefly describe the architecture of the main types of agents involved in the application. We conclude with some final remarks and discuss future work.

2. Current procedure

In this section, we describe a revision procedure of antibiotic therapies which is currently in use at the Mataró Hospital (see Fig. 1 for a graphical description).

Every patient has assigned a so-called *medical order form* which contains previous and current therapies (name of the drugs, dosage, frequency and type of administration—oral or intra-venous), together with the patient identification data, the provisional diagnosis and other clinical data like weight, height or previously reported allergies to drugs.

Every day (at least once per day), based on the previous prescriptions, the current state of the patient and new available information about tests if any, the physician updates the prescription, which becomes effective during the next 24 h. This updated information is recorded (with explicit reference to date and time) in the same medical order form and signed by the physician. Three hard copies of the medical order form are considered, two are stored as patient records and a third one is sent to the pharmacy department. There, pharmacists revise them and check possible problems.

According to the standard procedure, they first check whether the prescribed medicines belong to the Hospital Pharmaco-therapeutic Guide (the list of medicines that the hospital uses). If so, according to the patient's allergy information, they study the prescribed drugs and report the non-appropriate ones. Incompatibilities between drugs to be simultaneously administrated to the patient are also reported. Finally, they check the correctness of the dosage, frequency, type of administration and length of the treatment. This is done according to different patient's data, e.g. dosage and frequency is adjusted in relation to weight, oral administration is suggested when there are no digestive problems, etc.). In case of any inconsistency, this is communicated to physician, who makes the final decision.

Additionally, the pharmacist may also propose alternative (free use) antibiotics when the therapy prescribed in the medical order form includes a restricted antibiotic. This procedure can take place when patient's pathogen microorganism(s) has (have) been isolated, i.e. there exist positive cultures, and sensitivity microbiological analysis (antibiograms) are available (one for each of the isolated pathogen). This analysis provides a first set of possible alternative antibiotics, those to which the microorganism is sensitive to, and discards those which the microorganism shows to be resistant to. To further sieve and prioritize the set of alternatives, other criteria are taken into account, for instance the administration route, location of the pathogen microorganism, possible allergies, renal failure, etc. Finally, the cost of the different antibiotics is also taken into account to break possible ties in the ranked list of alternatives.

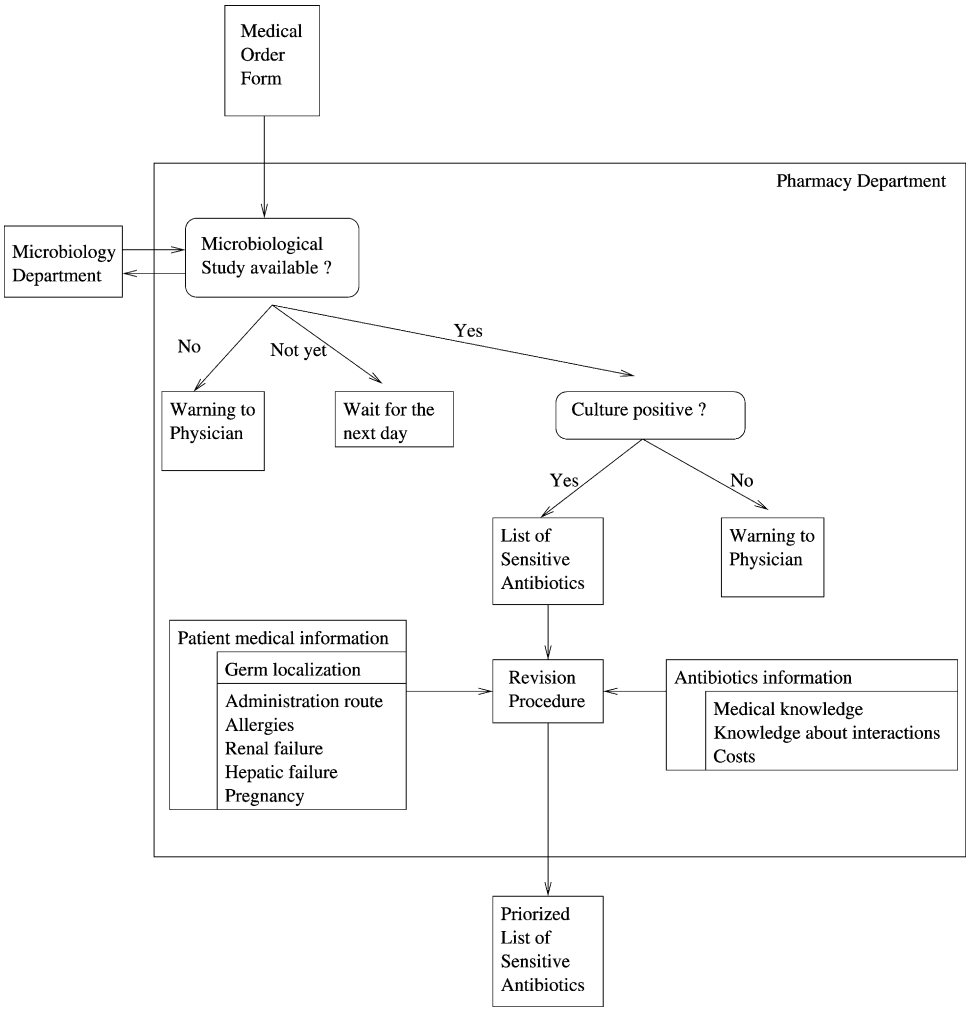

Fig. 1. Manual revision procedure of antibiotic therapies.

Some aspects have to be taken into account in the above procedure to come up with alternative antibiotics to those ARU prescribed in a medical order form:

(i) Only a physician can request an antibiogram. Thus, if the pharmacist detects patients prescribed with some ARU without microbiological analysis, a warning is forwarded to the corresponding physician.
(ii) If an antibiogram has been ordered but the results are not still available, the microbiological study is identified as pending, and its status is daily checked until it is available.
(iii) Only physicians can change a treatment. Thus, once a proposal for an alternative therapy is made by the pharmacy department, the physician is informed and no action is undertaken without his/her approval.

Table 1
Antibiogram for *Escherichia coli*

Antibiotic	Route	Sensitivity	Type
Amoxicillin-clavulanate	Oral	Yes	Free
Amoxicillin-clavulanate	i.v.	Yes	Free
Aztreonam	i.v.	Yes	Restricted
Gentamicin	i.v.	Yes	Free
Ceftriaxone	i.v.	Yes	Restricted
Cefuroxime	Oral	No	Free
Cefuroxime	i.v.	No	Free
Ciprofloxacin	Oral	Yes	Free
Ciprofloxacin	i.v.	Yes	Restricted
Imipenem	i.v.	Yes	Restricted

"i.v." stands for intra-venous.

For the sake of clarity, we give an example of how this process works. Note that inspite of its apparent simplicity the process is knowledge intensive.

Example Let us suppose that the pharmacy department receives a medical order form corresponding to a patient with an infection disease which has been prescribed a therapy with ceftriaxone, an ARU, and for whom a microbiological study (antibiogram) has been requested. A renal failure is reported in the form, but no further problems (allergies, administration route, etc.) are reported. The report by the microbiology department shows a positive culture by *Escherichia coli* in blood, and the microbiological sensitivity test for this microorganism yields the result displayed in Table 1.

So, after the sensitivity test, the list of possible antibiotics is reduced to those that are sensitive, i.e. only cefuroxime (both i.v. and oral) is discarded. Since there is no problem with the administration route, no antibiotic can be discarded by this reason, and furthermore, those which can be administrated by oral via will be preferred. Next, hospital guidelines for *Escherichia coli* infections, together with location criteria (microorganism has been found in blood culture) are applied. All these criteria, which belong to the pharmacological domain knowledge and are specific for each pathogen agent and for each patient's situation, provide a prioritized list of antibiotics, already taking into account the preference for those of free use in front of ARUs. In this example, we assume as output the ordered list in Table 2.

Because of the patient's renal failure, gentamicin is discarded from this list. Therefore, since in principle there are no further issues to consider, and since the most preferred antibiotic, amoxicillin-clavulanate, is of free use no further refinement of the list is needed, and amoxicillin-clavulanate oral would be the suggested alternative therapy for the patient.

Let us imagine another scenario where the patient cannot take pills, so all antibiotics with an oral administration have to be discarded. Then, the remaining drugs are those of Table 3.

So, amoxicillin-clavulanate i.v. would be the recommended therapy.

Finally, assume that, after being treated with amoxicillin-clavulanate i.v., the patient shows signs of allergy. In that case, we should restrict ourselves to choose one among the

Table 2
Prioritized list of antibiotics

Antibiotic	Route	Sensitivity	Type
Amoxicillin-clavulanate	Oral	Yes	Free
Ciprofloxacin	Oral	Yes	Free
Amoxicillin-clavulanate	i.v.	Yes	Free
Gentamicin	i.v.	Yes	Free
Ciprofloxacin	i.v.	Yes	Restricted
Aztreonam	i.v.	Yes	Restricted
Ceftriaxone	i.v.	Yes	Restricted
Imipenem	i.v.	Yes	Restricted

"i.v." stands for intra-venous.

Table 3
Prioritized list of antibiotics with intra-venous administration

Antibiotic	Route	Sensitivity	Type
Amoxicillin-clavulanate	i.v.	Yes	Free
Ciprofloxacin	i.v.	Yes	Restricted
Aztreonam	i.v.	Yes	Restricted
Ceftriaxone	i.v.	Yes	Restricted
Imipenem	i.v.	Yes	Restricted

"i.v." stands for intra-venous.

Table 4
Prioritized list of restricted use antibiotics according to the cost

Antibiotic	Route	Sensitivity	Type
Ceftriaxone	i.v.	Yes	Restricted
Aztreonam	i.v.	Yes	Restricted
Imipenem	i.v.	Yes	Restricted
Ciprofloxacin	i.v.	Yes	Restricted

"i.v." stands for intra-venous.

four remaining drugs, which are all of restricted use. Then the cost criteria could also be taken into account. According to the current hospital data, we have

$$\text{price}(\text{ceftriaxone}) < \text{price}(\text{aztreonam}) < \text{price}(\text{imipenem}) < \text{price}(\text{ciprofloxacin})$$

Therefore, in that case, the final list of prioritized suggested antibiotics will be the one displayed in Table 4.

3. Related work on decision support systems for antibiotic therapy

One of the first decision support systems for antibiotic therapy is Health Evaluation through Logical Processing (HELP) [28], an integrated hospital information system that

combine both communication and advice functions. The clinical data base integrates information from areas such as admitting, radiology, surgery, pathology, nursing, respiratory therapy, and the laboratory. The HELP system has been developed over the past 20 years at the LDS Hospital in Salt Lake City, UT, USA. The system is based on the use of different knowledge frames, which are specialized modules that allow the system react when new patient data is introduced by generating alarms, reminders, diagnostic suggestions and even some therapeutic recommendations [10,11]. Some particular decision support systems for antibiotic therapy that have been developed inside the HELP environment focuses on the improvement of the antibiotic treatment for microbiological confirmed infections [27], the antibiotic surgical prophylaxis [5] and the empirical antibiotic treatment [9].

Clinical and financial effects of these programs are annually evaluated at the LDS Hospital. Observed clinical effects are a significant increase in the improvement of antibiotic surgical prophylaxis, progressive changes in therapy prescriptions by physicians due to the alerts generated by the decision support systems, and a decrease of the rate antibiotic associated adverse events. Main observed financial effects are a steadily decrease of the percentage of total pharmacy drug expenditures represented by antibiotics and a decrease of the defined daily doses per 100 occupied bed-days (DDD) [26].

Based on previous experiences, a computer anti-infections management program was used and evaluated in the intensive care unit (ICU) at the LDS Hospital from July 1994 to June 1995. The patients cared with the aid of the anti-infection management program were compared with patients admitted to the same unit during the 2 years before that period. The use of the program led to significant reductions in orders for drugs to which the patients had reported allergies, excess drug dosages and antibiotic-susceptibility mismatches. There were also relevant reductions in the mean number of days of excessive drug dosage, adverse events and cost [12]. The anti-infections management program at LDS differs in several important ways from other attempts to improve patient care through the use of computer support system. First, the report shows improvement in clinical outcomes and not just the performance of physicians. Second, the computer program was designed by clinicians for use by clinicians at the LDS Hospital. Third, the project was designed to help on physicians' decision. Fourth, the system is understood as a reliable source of information and provides physicians with the means to make sound decisions about the use of anti-infectious drugs in an intensive care setting, where quality matters most [14].

Another expert system for improving anti-microbial therapy was developed in the North Caroline Baptist Hospital in 1991. The expert system simultaneously examines data on patient demographics, cultures results, associated susceptibility test results, cutoff values for susceptibility and anti-microbial therapies downloaded from different databases. The system output consists of one out of four potential problems: no therapy is being given despite the presence of pathogens, the pathogens isolated are resistant to the therapy being given, the therapy cannot be matched with susceptibility data of the isolated pathogens, or the therapy was discontinued too quickly. It was found that a therapy was more likely to be improved when the responsible physician was contacted about the potential problem indicated by the report [22].

Q-ID [39] is a decision support system which uses a series of knowledge bases about infectious diseases to make recommendations for empirical treatment or to check the appropriateness of a current antibiotic therapy. From disease manifestations and risk

factors, a differential diagnosis for the patient is generated. To generate empirical treatment recommendations site-specific data on sensitivity to antibiotics of each organism is used as an estimate of the likelihood of achieving maximum benefit for each disease on the patient. Combining this data with drug and patient specific factors, the system recommends the most adequate antibiotics for a patient.

ICONS [6,35] is an antibiotic therapy advice system for patients in an ICU who have caught an infection as additional complication. This program uses case-based reasoning (CBR) to solve a current problem based on similar previously documented cases. The main task of the system is to present a suitable empirical therapy advice for ICU patients with bacterial infections.

Leibovici and Andreassen address in [1,21] all the important decision points on the first day of managing a patient suspected or known to have a bacterial infection. This system is based on causal probabilistic networks, used to calculate the probability distributions for the relevant output variables, and decision theory, used to balance the therapeutic benefit of antibiotic therapy against the detriment associated with antibiotic drugs.

A semi-automatic program for monitoring anti-microbial therapies [17] has been developed at the Hospital del Mar (Barcelona, Spain). The application integrates information from the pharmacy department and the microbiology laboratory and selects all patients treated with an ARU. Then a clinical pharmacist follows a similar procedure to that displayed in Fig. 1 to revise the therapy and propose some kind of modification or adequation. In the study reported in [17], the most frequent types of interventions proposed were a change of dosage, a change in the administration route and a change of antibiotic. The number of interventions made, in relation with the total of treatments, was fairly constant during the study, approximately 12.5% of ARU treatments warranted an intervention and 92% of them were accepted. They also show that the interventions represented a significant decrease in the total antibiotic expenditures.

Finally, let us comment about the system *Terap-IA* [2,3], a decision support system for the recommendation of antibiotic treatment for pneumonia diseases. Assuming evidence for one, two or three microorganisms possibly causing pneumonia in a patient, the goal of the system is to find the best combination of antibiotics to treat the patient, in the sense that the combination should cover all the microorganisms under consideration. Actually, as described later on (see Section 6.2) in this paper, we adapt in part for our application the general architecture of *Terap-IA* to design the pharmacy agent in charge of revising medical order forms with prescriptions of antibiotics of restricted use.

4. Modelling of the application in terms of a multi-agent system

In the previous section, we have referred to a number of decision support systems. These are a particular kind of what is commonly known as knowledge-based (KB) systems. Knowledge-based systems are very popular in medical domains since they provide easily understandable knowledge representation languages (for instance rule-based languages) which are usually able to deal with incomplete and uncertain knowledge (e.g. fuzzy rule-based systems, Bayesian networks), which is a very common characteristic in medical applications.

Despite of these facts, KB systems are not widely used in the daily clinical practice. One of the reasons is that traditional KB systems are self-contained isolated applications, they are not situated in their natural hospital environment. For instance, a classical expert system is a program which receives inputs from the keyboard and produces outputs to the screen without interacting with other similar computer applications or databases.

The considerable effort in programming traditional KB systems in the medical domain can be more useful when situating that system in a hospital environment. For instance, we can think of a system that obtains data about a patient from the administration department or the clinical files, it orders a particular microbiological analysis to the laboratory, looks up for the results, and finally introduces this result in its knowledge base. This kind of KB systems present a more intelligent behavior [20].

These intuitions lead us to the notion of (software) *intelligent agent*. We will not discuss here on the definition of intelligent agent (see for instance [40]), rather we shall consider software agents in the strong sense, that is, entities with autonomy, proactive and deliberative behavior, and communicative and social ability.

Following the agent metaphor, in this paper we propose to consider the decision support application for improving antibiotic therapies inside a hospital viewed as a multi-agent system, composed by both human and software agents. These software or artificial agents can be human agent assistants, databases, knowledge-based systems and, in general, computer programs with communication abilities within the local hospital network or even in the Internet. In particular, besides considering assistant agents attached to human agents (physician, pharmacist, etc.) or agents doing the main tasks involved in the procedure of revising a therapy (getting clinical information, deciding alternative therapies, etc.) we shall consider to have an agent attached to each patient, call it *guardian angel*, that would be responsible in general of taking care of the several aspects of the patient's stay at the hospital. Here, in this application, the *guardian angel* agent will restrict itself to initiating actions in order to get the ''best'' antibiotic therapy prescribed.

There are a lot of different methodologies oriented to multi-agent system analysis and design [18]. For instance, *AUML* [25] is an extension of the Unified Modeling Language (UML) that makes it suitable for the design of agents and multi-agent systems. Another example is *Gaia* [41], a methodology founded on the view of a multi-agent system as a computational organisation consisting of various interacting roles.

From the organizational point of view, a hospital is indeed an *institution* where agents have concrete roles and abilities with different chains of authority, sometimes hierarchically organized, where there are organizational groups composed again by agents (for instance, nursery, laboratories), and where individual agents and groups communicate among them, usually with precise protocols, etc. This *institutional* view of agent's societies has led to Artificial Intelligence researchers to develop a design methodology for complex multi-agent systems, called *Electronic Institutions* [24]. In this paper, we adopt this methodology to design our multi-agent system for revising therapies which contain antibiotics of restricted use. Furthermore, and very important, we take advantage of ISLANDER [8]—a tool developed at the IIIA—for the specification and verification of Electronic Institutions. ISLANDER defines a textual language and has an editor that currently permits the graphical specification of Electronic Institutions. In the near future, this tool will allow the automatic generation of agent templates and the necessary infrastructure to run the specified Electronic Institution.

In the next section, we describe and specify the multi-agent system in terms of Electronic Institutions, whereas in Section 6, we describe the architecture of some of the involved agents.

Let us remark that, although we shall restrict ourselves in this paper to model that part of the hospital environment directly related to our current application, the approach outlined above we think is general enough for a reliable, sound and incremental introduction of software agents in a hospital information system. The approach allows the *agentification*—using *wrapper* methodology—of tested legacy systems, it does not disturb the normal procedures of the hospital and scales gracefully.

5. Specifying the application as an Electronic Institution

Electronic Institutions are a suitable model to design complex multi-agent systems with heterogeneous agents playing different roles and interacting in different forms [7,23,34]. This model deals with the scalability and other requirements discussed above.

The main components of Electronic Institutions are: the agents and their roles, the dialogic framework, the scenes, the performative structure and the normative rules. In the following sections, we successively show how these notions are used to model and specify the interactions of the different agents involved in our modeling of the process described in Section 2.

5.1. Agents and Roles

Agents are the players in an Electronic Institution and each one adopts some role, that is, a pattern of behavior. The illocutionary actions performed by an agent are constrained by the Electronic Institution according to the role it is playing. In a hospital organization, we can find agents playing the roles of physicians, nurses, patients, and so on. Specifically, in our application we have the following roles:

5.1.1. Guardian angel (ga)

This is the *alter ego* of the patient in the agent world. When a patient is admitted into the hospital, a new *guardian angel* is created and assigned to him/her. This agent has all the patient's medical information (or knows where to obtain it). During the period the patient is in the hospital, the *guardian angel* takes care of the patient by checking every aspect of his/her treatment and proposing improvements. In our application, the *guardian angel*'s goal is revising medical order forms with prescribed ARUs and finding alternative less aggressive or cheaper antibiotics with the same therapeutic properties. It is important to notice that the *guardian angels* do not have the necessary medical knowledge to propose alternative treatments. Instead, in our system, the knowledge expertise is delegated to specific agents, and *guardian angels* must interact with these agents to take advantage of their capabilities.

5.1.2. Physician secretary (phs)

On top of being the access door for a physician to the agent world (acting as a human interface), the *physician secretary* is also the personal assistant of the physician.

The *physician secretary* knows the schedule of the physician and his/her work preferences. Each physician in the hospital has his/her own virtual secretary.

5.1.3. Laboratory manager (la)

This agent takes care of the laboratory data. It is responsible for managing all the analysis request and deliver the results once they are available. In other words, this is the *agentification* of the laboratory data base.

5.1.4. Pharmacy expert (pha)

As the name suggests, this is one of the expert agents present in the system. Using the patient data and the analysis result, this agent proposes possible changes in a medical order with ARUs in order to improve the treatment. This agent implements the current procedure for antibiotic revision described in Section 2. The agent is described in Section 6.2.

5.1.5. Nurse agent (nu)

Similarly to the *physician secretary*, this agent is the access door for the nurses to the agent world. For example, one of its duties is to collect the medical orders from the patient agents once it is requested by the human nurse.

5.2. Dialogical framework

The concepts involved in the communication among agents have to be fixed. Agents interact through illocutions or speech acts [36]. The hospital institution has to establish the acceptable illocutions defining the ontology—the common language to represent the world—and the common language for communication and knowledge representation. The dialogical framework is composed by an ontology, a representational language for the concrete domain, a set of illocutionary particles and a communication language.

In our application, the ontology includes all the medical and pharmacological terms, as for instance the name of antibiotics or germs. The representation language for the domain is first-order logic. The set of illocutionary particles used are: *query*, *inform*, *request*, *offer*, *accept*, *withdraw* and *refuse*.

An example of a communication language expression of the illocution *request* is the following:

$$request(?x_i : ga, ?x_j : phs, authorization(medical\_order\_98))$$

where that expression can be interpreted as a request sent by an agent x_i playing the *ga* (guard angel or personal assistant of the patient) role[2] to another agent x_j playing the *phs* (personal assistant of the physician) role for the authorization of a given medical order.

5.3. Scenes

Interaction between agents occurs within a scene. A scene is basically a group meeting, and it is composed of a set of agents playing different roles and communicating with a

[2] The expression $x{:}y$ stands for the agent x playing the role y.

well-defined communication protocol. An Electronic Institution is composed by several scenes connected among them.

To specify a scene, the first step is to identify which are the roles that will participate in the scene. Agents with these roles can enter and exit the scene—maybe to go to another one. The second step is to define the communication protocol. The communication protocol is fixed. The scene is structured as a graph where the nodes represent the different states of a conversation and the arcs are labeled with expressions of the communication language. It is mandatory an initial state and a final state. Agents can enter or exit to and from different states, this is represented by white and black little triangles labeled with the roles allowed to do it.

In our application we can identify the following scenes:

- *Patients Room*. All the active *guardian angels* can be located in this scene to be queried by other agents that need some medical information related to a specific patient. In order to simultaneously perform the assigned tasks, they use *clones* that can move freely among the other scenes while they are also present in the Patients Room.
- *Physicians Room*. Similarly, Physicians Room is the scene where it is possible to locate all the *physician secretaries*.
- *Laboratory*. This is the scene where an agent has to go if it wants to request an analysis or to gather the result of a previous one. One or more *laboratory managers* are waiting there to attend these demands.
- *Pharmacy*. The pharmacy scene is the headquarters of the *pharmacy experts*. In our application there is only a single type of expert (an expert in ARUs) but in a more general situation agents could find in this scene experts specialized in different areas related to drugs.
- *Dialog scenes*. This is a generic term to denote the following set of scenes: *Consulting Room*, *laboratory results*, *pharmacy locutory* and *patient quest*. These scenes are used to provide a framework for the dialogs between agents. Each time an agent wants to talk to a *physician secretary*, a *laboratory manager*, a *pharmacy expert* or a *guardian angel*, one of these scenes is created to accommodate the agent and a clone of its partner during the discussion. For instance, if a *guardian angel* wants to talk to a *laboratory manager* first it has to go to the 'laboratory' scene. Once there, it has to request a meeting with the *laboratory manager* and if the *laboratory manager* accepts, a new 'laboratory results' scene is created. This new scene is used by the *guardian angel* and a clone of the *laboratory manager* to discuss whatever is necessary.
- *Waiting Room*. This scene is used by the *guardian angels* to wait for a physician authorization or for the result of an analysis.
- *Hall*. It is a sort of crossroads that allows agents to move from one scene to the other.

For the sake of conciseness, we limit ourselves to describe the following two scenes: the Physicians Room scene and the Consulting Room scene.

5.3.1. Physicians Room

Fig. 2 shows the state diagram of this scene. As described above, this scene is a room where *guardian angels* and *physician secretaries* meet. Each *guardian angel* tries to obtain attention from a *physician secretary* in order to talk with it about something of

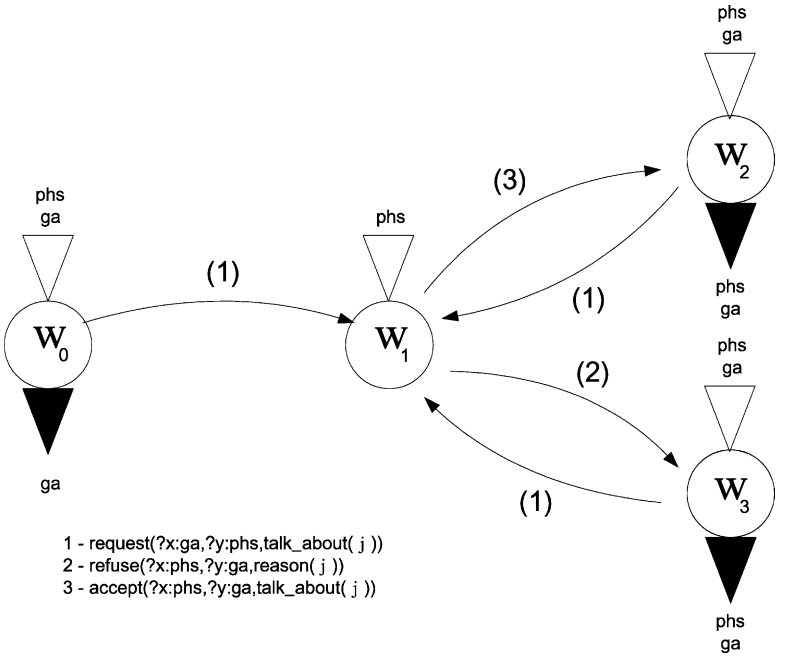

Fig. 2. Physicians Room scene.

its interest. The dialog will take place in the scene described in the section below (Consulting Room).

The communication protocol consists in a request of the agent playing the role of *ga* to talk about something with an agent playing the role of *phs*. The last agent can accept or refuse to talk.

In this scene the agents enter in an initial state W_0. The next state W_1 will be reached by the expression number 1,

$$request(?x : ga, ?y : phs, talk\_about(j))$$

only started by an agent playing the role *ga*.

From W_1, the next state can be W_2 or W_3, depending whether the agent receiving the request accepts or rejects it. In the first case, the two agents maintaining the conversation exit from the scene (and they go to a dialog scene—the Consulting Room). In the last case, the agent playing the role *ga* exits from the scene.

5.3.2. Consulting Room

In this scene takes place the dialog between a *guardian angel* and a *physician secretary.* For each dialog, a new Consulting Room is created. When the dialog finishes, both agents leave the scene that is automatically destroyed. Fig. 3 shows the states diagram for this scene. W_0 is the initial state.

All dialogs between a *ga* and a *phs* starts with a request to ask for an authorization to perform an antibiogram or to change a medical order (1). The *phs* can inform to the *ga* that

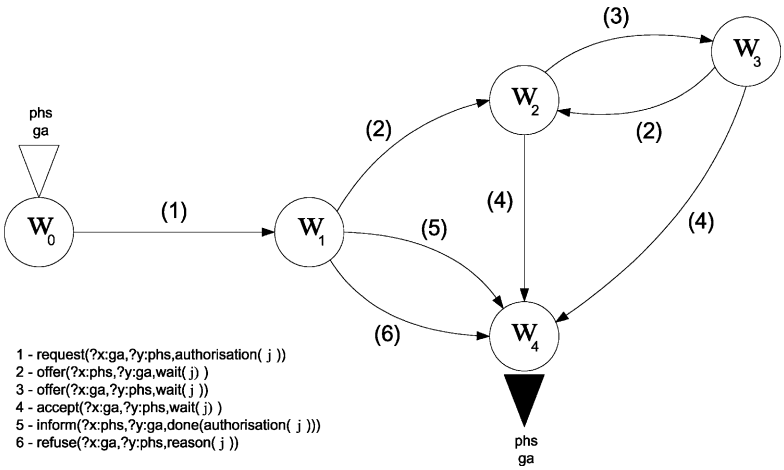

Fig. 3. Consulting Room scene.

this authorization is not possible for some reason (6), that it is already authorized (5) or the time when the *phs* thinks the request can be processed by the physician (2). The time when the physician will process the request is estimated by the *phs* based on the knowledge it has about the physicians' schedule and habits. At this point, the *ga* calculates if the estimate is suitable for the interests of the patient it is representing. To do that, the *ga* takes into account several factors like the scheduled time for the next medicine administration to the patient, the estimated time it will take to perform an antibiogram, the laboratory timetable, and so on. If the schedule proposed by the *phs* does not fit the requirements, the *ga* can ask for a revision of the time proposing a new schedule (3). Then, a short negotiation process takes place between the *ga* and the *phs*. In case both agree that a confirmation by the physician is urgently needed due to time constraints, then the *phs* will contact the physician using a mobile phone, a PDA, or other mechanisms to accelerate his/her answer. The media used to contact with the physician depends on the urgency of the subject.

5.4. Performative structure

Scenes can be connected, composing a network of scenes, which captures the relationship among them. The specification of a performative structure contains a description of how the agents depending on the role they are playing can legally move from one scene to another.

For a complete description of performative structures in Electronic Institutions refer to [7,8,23]. Fig. 4 presents all the graphical elements used in the performative structure of the application, showed in Fig. 5.

The main elements of a performative structure are the scenes. If the scene is defined as a multiple scene, it means it is possible to have several instances of the same scene running in parallel. For instance, each time a *la* agent and a *ga* agent want to talk, a new *laboratory results* scene is created (indicated by the '*' at the end of the arc). Because the *laboratory*

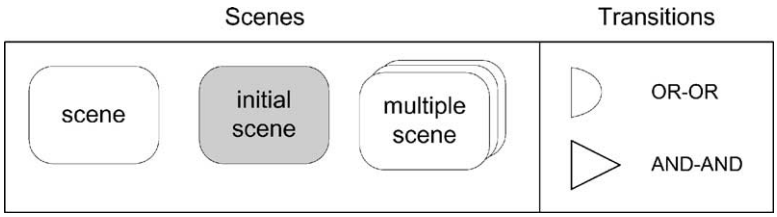

Fig. 4. Graphical elements of a performative structure.

results scene is a multiple scene, it is possible to have more than one instance running in parallel and hosting different pairs of (*la*, *ga*) agents talking at the same time.

Scenes are connected among them by *arcs* and *transitions*. Labels in *arcs* show which roles are allowed to move from one scene to another. As an example, the labels in the arc between the *Consulting Room* and the *Waiting Room* indicates that only agents with role *ga* are allowed to move from the first scene to the transition and from the transition to the second scene. A loop in an *arc* means that the agent playing the role indicated in the label of the loop, instead of moving from one scene to the other will send a clone. Using clones, an agent can be present in several scenes at the same time. All the clones of an agent share the information. For example, the arc between the *Patients Room* and the *patients' quest* has a loop. It means that the agent playing the *ga* role will send a clone to talk with the agent playing the *nu* role in the *patient quest* scene.

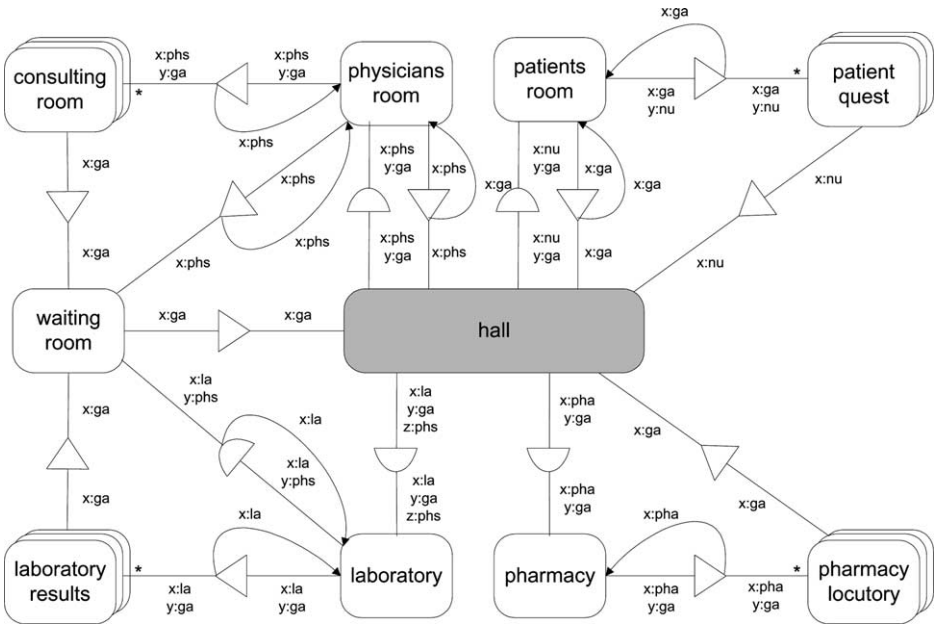

Fig. 5. Performative structure of the application.

Finally, *transitions* must be regarded as a kind of routers that contain local information about the scene instances that they connect. In our application we use two different types of transitions:

- And/And. They establish synchronization and parallelism points since agents are forced to synchronize at their input to subsequently follow the outgoing arcs in parallel. For instance, the transition between the *pharmacy* scene and the *pharmacy locutory* indicates that in order to move from the former to the later, *pha* agents and *ga* agents must go together in pairs.
- Or/Or. They behave in an asynchronous way at the input (agents are not required to wait for others in order to progress through), and as choice points at the output (agents are permitted to select which ongoing are, which path, to follow when leaving). The connection between the *hall* and the *laboratory* is an example of this kind of transition. Agents with role *la*, *ga*, and *phs* are allowed to move from the *hall* to the *laboratory* without any kind of synchronization among them.

5.5. Tracing the interaction flow

In order to show how is the everyday life in the above described Electronic Institution we will follow a *guardian angel* through all the stages involved in the process of revising a medical order. For the sake of clarity we are not going to detail all the possibilities but just those we think are relevant to understand the application.

- *Stage 1. Getting the medical order.* After the daily checking of the patient, the physician connects with his/her virtual secretary and introduces the medical order form. The medical order, that is a XML document, has to be delivered to the guardian angel of that patient. A clone of the physician secretary goes through the 'hall' to the Patients Room scene, requests a meeting with the guardian angel and, using a patient quest scene, delivers the medical order to a clone of the guardian angel created specially for the meeting.
- *Stage 2: Antibiogram authorization*. The guardian angel analyzes the medical order looking for those aspects sensible to be revised. Although in this application we only focus our attention in the ARU, in a general case there could be different aspects to be considered. When the medical order prescribes ARUs, it is necessary to obtain an antibiogram for the patient. However, this cannot be done without the authorization of the physician so the guardian angel sends a clone to the Physicians Room and meets with his/her virtual secretary to ask for an authorization. If the physician is not on-line, the secretary will notify to the guardian angel what time he/she is supposed to be connected again. The guardian angel evaluates if the scheduled time is OK to be ready before the next sample extraction. If the situation requires a quicker answer, the physician secretary can contact with the physician by e-mail or using for instance a mobile phone with WAP protocol. Once there is an agreement, the guardian angel waits in the Waiting Room until the authorization arrives. When the physician secretary receives the authorization sends a clone to the laboratory with the authorization. The guardian angel in the Waiting Room is also informed.
- *Stage 3: Antibiogram results*. To know whether a sample extraction has to be done, human nurses use their *alter ego* in the agent world to query each guardian angel.

If there is such a request for the extraction, samples are extracted and sent to the laboratory. Later, once the results of the antibiogram (again an XML document) are introduced into the laboratory data base, the laboratory agent delivers the results to the guardian angel that was waiting in the Waiting Room.

- *Stage 4: Modifying the medical order form.* Once the guardian angel has both the medical order and the antibiogram, he moves to the pharmacy scene and requests a revision to the pharmacy expert. The pharmacy expert generates a change proposal. Before changes are performed in the medical order, they have to be authorized by the physician. Therefore, the guardian angel goes back to the Physicians Room to ask for a new authorization to apply changes in the medical order. If changes are authorized, the medical order form is modified.

6. Agent architectures

In this section, we first describe the general aspects of the architecture we use for all the agents in the application. This general structure can be then specialized for agents with a specific task. In particular, in the second part of this section, we detail some aspects of the deliberative part of the pharmacy agent—which models the most complex task.

6.1. Agent's general structure

Although each agent has a different task in the system, and therefore requires different kind of reasoning capabilities, all of them share a common base structure. The characteristics of the environment have a great influence in the way agents have to be designed. In our case, these characteristics can be summarized as follows:

- It is a persistent environment. A hospital is open 24 h a day and 365 days a year. Therefore, the agents should be constantly running without interruption.
- It is an asynchronous environment. Messages can arrive at any time. An agent should be always ready to deal with new messages. This, as we will see, implies some kind of parallelism in the agent design.
- The priority of tasks may change along time. The priority of a task increases as long as the deadline to finish it is getting closer. For instance, if it is necessary to make a test for which a physician's authorization is mandatory and the laboratory is going to close very soon, the notification to the physician asking for the authorization becomes a high priority task. If the authorization does not arrive quickly, the patient will have to wait several hours until the laboratory starts to work again. It is important to take into account that how and when this priority should change is task dependent. This point and the previous one implies that an agent cannot be blocked waiting for an answer or spend a lot of time performing a single task. Periodically it has to check the environment (in our case the message queue) to decide which is the most sensible thing to do next. To do that, agents use threads to parallelize task execution.
- Tasks can be canceled. Canceling a task implies not only stopping its execution but taking some ''cleaning'' actions that will be different depending on the current progress of the task.

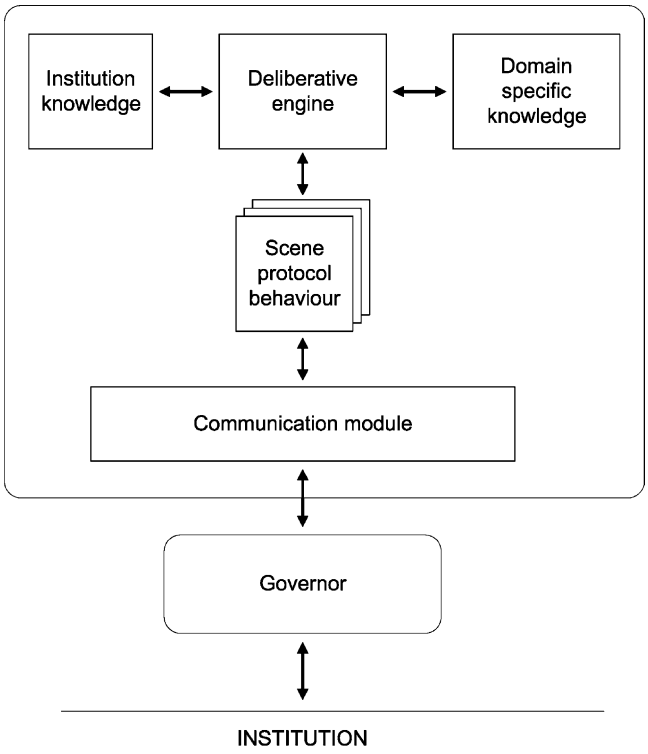

Fig. 6. General structure of an agent.

Fig. 6 shows the common structure of the agents in the application. The agent is attached to the institution through a *governor*. This *governor* has three main functions: the first one is to facilitate the agent connection hiding the low level details of the communication processes. The second is to control the actions of the agent according to the current state of the institution. And finally, inform when required by the agent about the current state of the scenes and the institution in general.

The agent architecture is divided into five modules:

1. *Communication module*. It deals with the high level communication details, managing all the incoming and outgoing messages.
2. *Domain specific knowledge*. It contains the knowledge of the application.
3. *Institutional knowledge*. The specification of the scenes and the performative structure.
4. *A set of scene protocol behaviors*. A scene protocol behavior models the current state of the agent in a scene. There is a scene protocol behavior for each clone of the agent.
5. *Deliberative engine*. Using the domain specific knowledge, the institutional knowledge and the current state of the agent represented in the scene protocol behaviors, it make decisions about the next action—taking into account the restrictions imposed by the Electronic Institution and the goals of the agent—to be performed to achieve the goals of the agent.

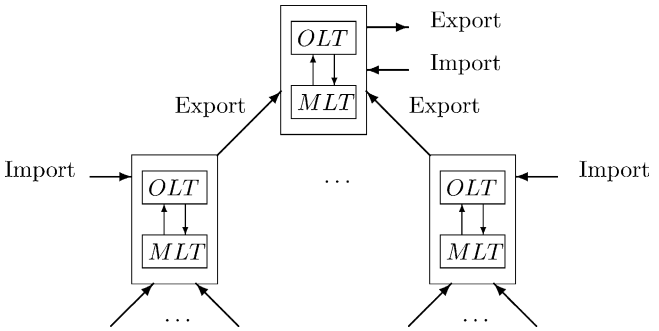

Fig. 7. Structure of a Milord II module hierarchy.

6.2. The pharmacy agent

The deliberative part of the pharmacy agent is programmed using Milord II. Milord II is a modular language for knowledge-based systems. A knowledge base in Milord II consists of a hierarchy of modules interconnected by their export interfaces. Each module contains an Object-Level Theory (OLT) and a Meta-Level Theory (MLT) interacting through a reflective mechanism (see Fig. 7). An OLT is composed by many-valued propositional rules, and a MLT by Horn-like predicate rules.

A module can be understood as a functional abstraction between the set of components it needs as input and the type of results it can produce. From the logical point of view, Milord II makes use of both many-valued logic and epistemic meta-predicates to express the truth status of propositions. For further details in these logical topics the reader is referred to [15,29,37,38]. For a general and complete description of the language see [30–33].

Milord II has been used to develop knowledge-based medical applications [16]. One of these systems is *Terap-IA* [2,3], which is a support system for recommending antibiotic treatment of pneumonia infections. In that application a conceptual architecture for the treatment of infectious diseases was designed and applied, as a particular instance, for the pneumonia treatment.

In the pharmacy agent, we have used the same general *Terap-IA* architecture, so we have modelled the solution to the problem as a series of tasks that are sequentially performed (see the boxes with text in bold in Fig. 8). The procedure designed and supervised by physicians is equivalent to the process described in Section 2.

(A) The system starts by considering groups of antibiotics which are *sieved* taking into account relevant data of the patient. Namely, the task consists in finding the degree of adequacy of every group of antibiotics to that patient. It is performed taking into account data about pregnancy, allergic reactions to antibiotics, renal diseases or genetic alterations. The result of this task is the adequacy of every group to a concrete patient. It is independent of the disease we want to treat.

(B) Then, taking into account the diagnosis of the patient, we dynamically generate (by selection) a set of treatment tasks, one for each microorganism.

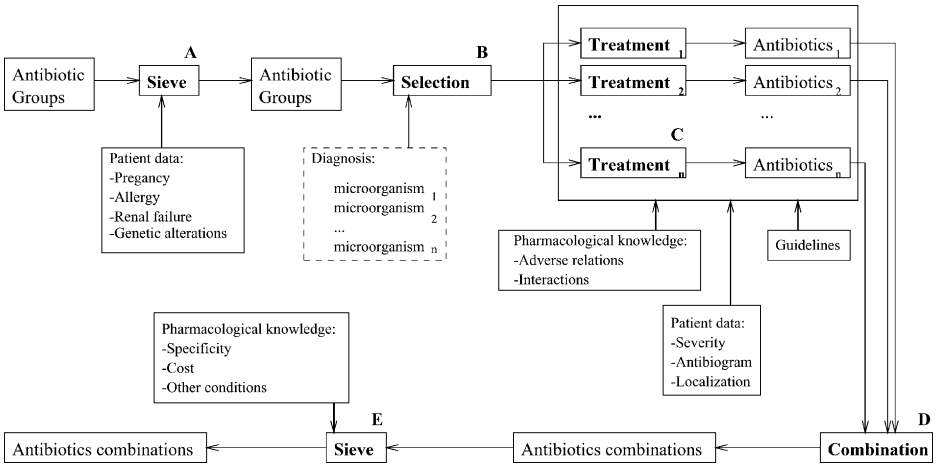

Fig. 8. Architecture of pharmacy agent.

(C) Taking into account the adequacy of every group of antibiotics (task A) for the patient, every treatment task obtains the best set of antibiotics, for the treatment of the microorganism it is specialized on, that belongs to those sieved groups. A treatment task uses data about the patient, pharmacological knowledge about the interactions between drugs, and their contrary effects (previously known adverse reactions to some antibiotics in a concrete patient) and particular guidelines of the hospital.

Important patient's data are the severity of illness—determining the administration route—the localization of the microorganism, and the result of the antibiogram.

(D) The sets of antibiotics adequate for each microorganism are then *combined*. Combinations are produced by means of several criteria. For instancem, do not combine antibiotics of the same group, nor those of the same spectrum (equivalent anti-microorganisms activity) nor with different administration route; we prefer mono-therapy (one antibiotic) to combinations.

(E) Finally, a new *sieve* is applied, now over the ordered set of antibiotics combinations taking into account the specificity and cost, giving as final result an ordered set of combinations that are the most adequate for the treatment of the patient.

The final result of this process is an adequacy-ordered set of treatments (combinations of antibiotics) for a concrete patient.

We would like to remark some aspects about the pharmacy agent with respect to the system *Terapia-IA*. Actually, despite using the same conceptual architecture, there exist between them three main differences that should be noted. The first one is that the domain of *Terap-IA* is only the treatment of pneumonia while the pharmacy agent deals with treatments for any infectious disease. The second difference is about the assumed status of the patient's diagnosis. *Terap-IA* recommends an antibiotic therapy for a set of pathogens which are only known as possible candidates of causing the patient's pneumonia disease. However in our application, the pharmacy agent recommends an antibiotic therapy for

pathogen microorganisms actually infecting the patient (normally only one), acknowledged by positive cultures which are available to the agent. Moreover, the third and last difference is that the pharmacy agent needs for its task to have also available the patient's microbiological sensitivity tests of antibiotics (antibiogram) for the microorganisms it is dealing with. These differences are important to clarify the following point about the feasibility and complexity of the pharmacy agent. *Terap-IA* is a very complex system because of the working hypothesis of incomplete information and uncertainty about the pathogens possibly causing the disease when a first treatment is urgently needed, despite of the lack of data, when a patient with pneumonia is admitted in a hospital. By a simple extrapolation, it seems extremely difficult to develop a system like *Terap-IA* for the empirical treatment of any infectious disease. However, the situation is totally different with the development of the pharmacy agent since, by hypothesis, this agent knows a priori both the microorganism causing the infection and the antibiogram of the patient and thus the task of obtaining a new treatment for replacing a current (provisional) one is radically simpler.

7. Conclusions and future work

In this paper we have described a multi-agent-based approach to develop a decision support system for the task of revising and proposing modifications in prescribed antimicrobial therapies using antibiotics of restricted use.

From a hospital organizational point of view, the development and use of an application as the one described in this paper has clear advantages, both medical and economical. Such a system avoids unnecessary, inefficient and broad spectrum antibiotic treatments, and decrease toxicity problems in case of allergies or renal failures. Moreover, costs also decrease since prescriptions with ARUs, usually very expensive, decrease as well.

Actually, from an AI point of view, the main interest of the proposed application is twofold:

(i) the feasibility of the development of a patient-centered multi-agent application in a hospital environment, i.e. where the whole hospital can be viewed as a multi-agent system, with each patient having attached an agent that interacts with other kinds of agents (pharmacists, physicians, laboratories, etc.) in order to achieve its goals; and
(ii) the use of the model of Electronic Institutions as a sound methodology to specify the protocols and interactions that take place among the different agents involved in the application.

Although the described application focuses on a particular task, these aspects are general enough to think of extending this approach to a whole hospital organization to address the automation of similar knowledge-based tasks performed by human agents in hospital environment. However, there are important points that should be considered in such extensions. One of the most relevant issues is about ontologies. In that line, it is interesting to be aware of current projects aiming at providing standard ontologies for medicine, for instance Unified Medical Language System (UMLS) and Medical Subject Heading (MeSH)—used for indexing and finding information in MEDLINE—of the National Library of Medicine.

Acknowledgements

We acknowledge the Spanish Comisión Interministerial de Ciencia y Tecnología (CICYT) for its continued support through the project SMASH (TIC96-1038-C04). This research has also been partially supported by the European Community project SLIE (IST-1999-10948). The authors would like to thank the anonymous referees for their valuable comments on the contents of this work.

References

[1] Andreassen S, Leibovici L, Schønheyder HC, Kristensen B, Riekehr C, Geill Kjær, et al. A decision theoretic approach to empirical treatment of bacteraemia originating from the urinary tract. In: Horn W, Sharar Y, Lindberg G, Andreassen S, Wyatt J, editors. Joint European Conference on Artificial Intelligence in Medicine and Medical Decision Making, AIMDM'99, lecture notes in computer science, vol. 1620. Berlin: Springer; 1999. p. 197–206.

[2] Barrufet P. *Teràp-IA*: sistema expert d'ajuda al tractament de les Pneumònies, PhD Thesis. Barcelona: Universitat Autònoma de Barcelona; 2000.

[3] Barrufet P, Puyol-Gruart J, Sierra C. Terap-IA: a knowledge-based system for pneumonia treatment. In: Alpaydin E, Fyfe C, editors. Proceedings of the International ICSC Symposium on Engineering of Intelligent Systems, EIS'98, La Laguna, Tenerife, Spain, vol. 1. Canada: Academic Press; 1998. p. 176–82.

[4] Belongia A, Schwartz B. Strategies for promoting judicious use of antibiotics by doctors and patients. Br Med J 1998;317:668–71.

[5] Classen DC, Evans RS, Pestotnik SL, Horn SD, Menlove RL, Burke JP. The timing of prophylactic administration of antibiotics and the risk of surgical wound infection. New Engl J Med 1992;326: 281–6.

[6] Day D, Lubowski TJ, Yamaga CC, Main J, van Vleet J, Ambegaonkar A. Computer-assisted evaluation of antibiotic regimen coverage and cost. Clin Ther 1999;21(8):1418–25.

[7] Esteva M, Rodriguez JA, Sierra C, Garcia P, Arcos JL. On the formal specifications of electronic institutions. In: Dignum F, Sierra C, editors. Agent-mediated electronic commerce, the European AgentLink perspective, lecture notes in computer science, vol. 1991. Berlin: Springer; 2001. p. 126–47.

[8] Esteva M, de la Cruz D, Sierra C. Islander: an electronic institutions editor. In: Castelfranchi C, Johnson WL, editors. Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2002), Bologna, Italy, vol. 3. New York: ACM Press; 2002. p. 1045–52.

[9] Evans RS, Classen DC, Pestotnik SL, Lundsgaare HP, Burke JP. Improving empiric antibiotic selection using computer decision support. Arch Intern Med 1994;154:878–84.

[10] Evans RS, Pestotnik SL, Burke JP, Gardner RM, Larsen RA, Classen DC. Reducing the duration of prophylactic antibiotic use through computer monitoring of surgical patients. Drug Intell Clin Pharmacy 1990;24:351–4.

[11] Evans RS, Pestotnik SL, Classen DC, Burke JP. Development of an automated antibiotic consultant. MD Comput 1993;10:17–22.

[12] Evans RS, Pestotnik SL, Classen DC, Clemmer TP. A computer-assisted management program for antibiotics and other antiinfective agents. New Engl J Med 1998;338:232–8.

[13] Force Ll, Barrufet P, Priu R, Badia JM, Gurrera T, Sauca G. La auditoria como método para evaluar la utilización de antibióticos de uso restringido en el hospital y el impacto de las intervenciones de mejora. In: VIII Congreso Nacional SEIMC. Mallorca, España: Palma de Mallorca; 1998.

[14] Garibaldi RA. Computers and the quality of care—a clinician's perspective. New Engl J Med 1998; 338(4):259–60.

[15] Godo L, van der Hoek W, Ch Meyer JJ, Sierra, C. Many-valued epistemic states. Application to a reflective architecture: Milord II. In: Bouchon-Meunier B, Yager RR, Zadeh LA, editors. Advances in intelligent computing, lecture notes in computer science, vol. 945. Berlin: Springer; 1995. p. 440–52.

[16] Godo L, López de Mántaras R, Puyol-Gruart J, Sierra C. Renoir. Pneumon-IA and Terap-IA: three medical applications based on fuzzy logic, Artif Intel Med 2001;21:153–62.
[17] Grau S, Monterde J, Carmona A, Drobnic L, Salas E, Marn M. Monitoring of antimicrobial therapy by an integrated computer program. Pharmacy World Sci 1999;21(4):152–7.
[18] Carlos A, Iglesias CA, Garijo M, Gonzalez JC. A survey of agent-oriented methodologies. In: Muller JP, Singh M, Rao AS, editors. Intelligent Agents V, lecture notes in artificial intelligence, vol. 1555. Heidelberg: Springer; 1999.
[19] Jozefiak ET, Lewicki JE, Kozinn WP. Computer-assisted antimicrobial surveillance in a community teaching hospital. Am J Health-System Pharmacy 1995;52(14):1536–40.
[20] Lanzola G, Gatti L, Falasconi S, Stefanelli M. A framework for building cooperative software agents in medical applications, Artif Intell Med 1999;16:223–49
[21] Leibovici L, Fishman M, Schønheyder HC, Riekehr C, Kristensen B, Andreassen S. A causal probabilistic network for optimal treatment of bacterial infections, IEEE Trans Knowledge Data Eng 2000;517–28.
[22] Morell R, Wasilauskas B, Winslow R. Personal computer-based expert system for quality assurance of antimicrobial therapy. Am J Hospital Pharmacy 1993;50:2067–73.
[23] Noriega P. Agent-mediated auctions: the fishmarket metaphor, Monografies del IIIA, IIIA–CSIC, vol. 9. 1999.
[24] Noriega P, Sierra C. Towards layered dialogical agents. In: Müller JP, Wooldridge MJ, Jennings NR, editors. Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages (ATAL-96), Intelligent Agents III, lecture notes in computer science, vol. 1193. Berlin: Springer; 1996. p. 173–88.
[25] Odell J, Parunak H, Bauer B. Extending UML for agents. In: Wagner G, Lesperance Y, Yu E, editors. Proceedings of the Agent-Oriented Information Systems Workshop at the 17th National Conference on Artificial Intelligence. New York: ACM Press; 2000. p. 3–17.
[26] Pestotnik SL, Classen DC, Evans RS, Burke JP. Implementing antibiotic practice guidelines through computer-assisted decision support: clinical and financial outcomes. Ann Intern Med 1996;124: 884–90.
[27] Pestotnik SL, Evans RS, Burke JP, Gardner RM. Therapeutic antibiotic monitoring surveillance using a computerized expert system. Am J Med 1990;88:43–8.
[28] Pryor TA. The help medical record system. MD Comput 1988;5:22–33.
[29] Puyol J, Godo L, Sierra C. A specialisation calculus to improve expert system communication. In: Neumann B, editor. Proceedings of the 10th European Conference on Artificial Intelligence, ECAI'92, Vienna. New York: Wiley; 1992. p. 144–8.
[30] Puyol-Gruart J, Godo L, Sierra C, Specialisation calculus and communication. Int J Approximate Reasoning 1998;18(1–2):107–30.
[31] Puyol-Gruart J, Sierra C. Milord II: a language description. Mathware Soft Comput 1997;4(3):299–338.
[32] Puyol-Gruart J. MILORD II: a language for knowledge-based systems. In: Monografies del IIIA, IIIA–CSIC, vol. 1. 1996.
[33] Puyol-Gruart J, Godo L, Sierra C. Handling fuzzy information in Milord II. In: Morel G, Vernadat FB, editors. Proceedings of the Ninth Symposium of the International Federation of Authomatic Control in Manufacturing, INCOM'98, vol. 3. 1998. p. 85–90.
[34] Rodríguez JA. Towards a tes-bed for trading agents in electronic markets, PhD Thesis, Universitat Autonoma de Barcelona; 2001.
[35] Schmidt R, Gierl L, Case-based reasoning for antibiotics therapy advice: an investigation of retrieval algorithms and prototypes. Artif Intell Med 23 (2001) 171–86.
[36] Searle JR. Speech acts. Cambridge: Cambridge University Press; 1969.
[37] Sierra C, Godo L. Modularity, uncertainty and reflection in Milord II. In: Proceedings of 1992 IEEE International Conference on Systems, Man and Cybernetics. 1992. p. 255–60.
[38] Sierra C, Godo L. Specifying simple scheduling tasks in a reflective and modular architecture. In: Treur J, Wetter Th, editors. Formal specification of complex reasoning systems. Chichester: Ellis Horwood; 1993. p. 199–232.

[39] Warner H, Blue RS, Sorenson D, Reimer L. New computer-based tools for antibiotic decision support. In: Masys DR, editor. Proceedings AMIA Annual Fall Symposium. Bethesda: AMIA; 1997. p. 238–42.
[40] Wooldridge MJ, Jennings NR. Intelligent agents: theory and practice. Knowledge Eng Rev 10(2):1995.
[41] Wooldridge M, Jennings NR, Kinny D. A methodology for agent-oriented analysis and design. In: Etzioni O, Müller JP, Bradshaw JM, editors. Proceedings of the Third International Conference on Autonomous Agents (AGENTS'99). New York: ACM Press; 1999.