POLICY-RICH MULTI-AGENT SUPPORT FOR E-HEALTH APPLICATIONS [1]

Lars Braubach[(1)], Winfried Lamesdorf[(1)], Zoran Milosevic[(2)], Alexander Pokahr[(1)]

[(1)] Hamburg University, Department of Informatics, Distributed and Information Systems (VSIS), Vogt-Kölln-STr. 30, D-22527 Hamburg, Germany

[(2)] CRC for Enterprise Distributed Systems Technology (DSTC), c/o The University of Queensland, Level 7, General Purpose Building South, Q 4072 Brisbane, Australia

Abstract: Modern hospital environments represent complex, distributed, and cross-organisational enterprises with a variety of complex and distributed systems applications. They include a multitude of resources at different places, they have to accommodate real-time requirements, and they have to support rather complicated and, in many cases, unforeseeable business processes. Many processes must strictly follow certain sets of rules, and both process and resource usage have to be dynamically optimized to guarantee the best service to all patients at all times. This paper discusses how state-of-the art agent technology, enriched with expressive policy constraints, can be used to support provision of better quality care for patients and more efficient health service delivery to health professionals. We apply results of our research in multi-agent systems and policy modelling to a set of requirements in the e-health domain.

Key words: Health applications, scheduling, distributed systems, (multi-) agent technology, event based policy monitoring.

[1] This work has been funded, in part, by the Co-operative Research Centre for Enterprise Distributed Systems Technology (DSTC) through the Australian Federal Government's CRC Programme (Department of Education, Science, and Training) and by the Deutsche Forschungsgemeinschaft (DFG) through the priority research programme (SPP) 1083 “Intelligent Agents in Real-World Business Applications”.

1. INTRODUCTION

Modern health care environments represent fairly complex examples of distributed enterprises and distributed applications. They provide a variety of rather complex and heterogeneous resources at different places, they have to accommodate requirements of quite complicated and – in many cases – unforeseeable workflow ("business") processes, which again have to strictly follow certain sets of rules and restrictions. Hospital environments today and health jurisdictions in general are faced with a rather high amount of complexity due to their inherent dynamics of the processes and distributed organization structure.

In this paper we consider new features of health care systems and identify some emerging technologies as enablers to deliver better quality care to citizens and more efficient service delivery to health professionals. We investigate a broader set of issues and a broader set of solutions based on distributed agent technology than the resource scheduling aspects as presented in (Paulussen et al. 2003). We consider other health areas where this technology could be applied, particularly in cases where agents could be used to monitor activities and policies of various actors in the system.

Section 2 highlights new features of health systems and possible technologies that could help deliver more effective and efficient services. Sections 3 and 4 outline distributed agent technology and a specific policy language respectively. These two solutions are to be integrated to support the features identified, as presented in section 5. Section 6 summarises key observations from the paper and outlines possible future research directions.

2. NEW FEATURES OF HEALTH CARE DELIVERY

The nature of health care is changing because of a number of societal factors including the increasing pressure on health professionals and more educated population at large. This section summarises some of the new features and requirements related to the service delivery in the health domain. We provide several ideas how some information and communication technology (ICT) systems could be applied to support these new features.

2.1 Continuity of care and a patient-centric focus

There is an increasing pressure on health providers to deliver better and more effective care to consumers while ensuring economic efficiency of service provision. This requires seamless interoperability between business

processes crossing organisational and jurisdictional boundaries and thus better integration and interoperability between the underlying ICT systems of various service providers. Consider, for example, the *care continuum* principle which denotes consumer care that spans public services, community services and acute services of public health providers but also, general practitioners and other non-government organisations. The main premise of this principle is that care of citizens (i.e. population) should start from early preventions, increasingly involving community services and that the acute services of hospitals should be utilised only when necessary. Another similar principle, the *continuity of care*, is oriented towards a single consumer going through an episodic session. This principle aims at ensuring smooth continuity of service delivery to the consumer while being discharged from one and assigned to one or more other health providers.

These two principles require a uniform view of a consumer (from both the individual and population perspectives) across all service providers and possibly all health jurisdictions with an implication for unique and reliable information about customers. This will ensure smoother delivery of care to consumers on their clinical path, and more efficient service delivery to service providers. For example, various *electronic health records (EHR)* initiatives form significant steps towards providing smoother delivery of care to citizens in their lifetime and also at the episodic level – in particular taking into account cross-organisational nature of service delivery. One example is Australian Health*Connec*t initiative where current trials focus is on keeping standardised summary information about health events (e.g. health consultations and diagnostic test results) and recording EHR data such as patient's current medications and principal diagnoses (Health*Connect*).

2.2 Integration of health applications

Many existing systems in the health industry have been developed over many years using various generations of ICT technology, often adopting solutions of various vendors in an ad hoc manner. As a result, a typical health application system consists of many independently developed silos, frequently resulting in redundancy and duplication of information leading to inadequate and delayed treatment to consumers, e.g. unnecessary repetition of previous tests or examinations. This also adds unnecessary costs to the already stretched public funding of health jurisdictions.

In order to address this problem, many public health organisations are going through a transformation of their systems and processes, with an increasing focus on producing *enterprise architectures* for consolidating existing and delivering more interoperable future ICT systems. Examples are recent efforts in the UK, where there is a requirement that enterprise

architectures for health also need to be aligned with UK's e-Government Interoperability Framework (eGif). Similar initiatives, but with a less cross-government alignment are in the US (eGov) and in Australia (NEHTA).

2.3 Sensitivity and privacy issues

While the emerging ICT systems will provide facilities for better recording of patients' health information as in case of electronic health record, there are many new challenges such as making sure that information about patients is exchanged and accessed according to patients' consents. For example, a patient may demand that only certain health professionals are allowed, while others are not, to view their EHR – and this consents need to be enforced. This is of particular importance in the environment of independent cooperating healthcare facilities and raises issues of defining the consent rules and monitoring the access and information exchange between independently governed health providers.

One solution for a privacy-preserving transfer protocol which ensures that access to the health information at the receiving facility continues to be governed by the patient's consent is discussed in (O'Keefe et al.).

2.4 Coordination of health-care delivery

The increasingly collaborative nature of health delivery, involving various health service providers, coupled with best-care practice guidelines and care plans for individual patients, requires a systematic approach to describing steps involved in episodic treatment of consumers. Some of the steps can be applied sequentially while others could apply concurrently and there is often information exchange between these steps. All these aspects require a way of describing collaborative service delivery, in a way similar to business processes in many industries, but also with more flexibility to reflect urgent or unexpected occurrences that need to be dealt with.

Possible ways of supporting such a collaborative environment is through the event-based coordination and choreography technologies as discussed in (Berry and Milosevic, 2005) or by utilising a distributed agent technology to support distributed and autonomous decision making.

2.5 Scheduling of resources

In many countries, the aging population and increasing mobility of people place significant pressure on medical staff and on the resources in hospitals, clinics and other health organisations. This requires sophisticated facilities for scheduling of resources. Although this area has been subject of

previous research, the new health environment requires increasingly dynamic allocation of resources. Decentralized coordination among all involved parties (e.g. a local practitioner and a hospital) assures timely treatment of patients, and better utilization of scarce medical resources.

Solutions based on agent technology, e.g. as described in (Paulussen et al., 2003, and Paulussen et al., 2004) are well suited to address both the dynamics and uncertainty of the domain, as well as the inherently distributed and decentralized decision making in the context of resource scheduling.

2.6 Monitoring and public accountability

The size and complexity of health systems, and involvement of many actors in service delivery, require better monitoring of services. The large number of significant events that may require urgent attention introduces the need for a powerful notification mechanism to ensure safety of health delivery. To this end, event-based monitoring engines could be deployed to support run-time detection of sub-optimal or inadequate service delivery and generation of reminders to the staff for timely delivery of health services.

Further, a more collaborative and patient-centric delivery of health services will rely on standardised electronic health records (EHRs), and will also increasingly require more transparent access to key policies and best practice guidelines. This is also to support new philosophy in health delivery towards evidence-based treatment. While some policies refer to security and privacy as described above, other policies will also apply to the public accountability of health professionals. They will need to express duties and responsibilities of doctors, nurses and other service providers. One approach to describing enterprise policies that could be refined for the health domain is described in (Linington et al., 2004, Milosevic et al., 2004a, 2004b).

3. SUPPORTING E-HEALTH APPLICATIONS WITH AGENT TECHNOLOGIES

The previous section has introduced application characteristics of the health domain that put forth new demands on the technical infrastructure used to build ICT systems supporting health service delivery. *Agent technology* (see, e.g., Jennings and Wooldridge, 1998) provides a modelling paradigm and software infrastructure to support treating relevant subsystems of health service providers as autonomous self-dependent actors. The agent paradigm provides a natural means for communication with domain experts, as there is a direct mapping of organizational structures to technology. Moreover, software agents allow for system integration by wrapping

('agentifying') legacy code respecting the decentralized heterogeneous IT structure, existing in the health care sector (Nealon and Moreno, 2003).

3.1 The general BDI agent model

To support e-health applications, software agents could be designed and implemented to work on behalf of patients to ensure that care goals are satisfied or care plans are satisfactorily executed. They can be also implemented to support health professionals or health regulators to ensure that respective policies and guidelines are met. In general, the agent model is well suited for such applications because it is capable to capture the anthropomorphic aspects of patients or health providers in a quite natural way, through the so called "Belief-Desire-Intention" (BDI) model and theory which is based on mentalistic notions. The inner structure of such agents can be subdivided into beliefs, goals, and plans (Rao and Georgeff, 1995):

Beliefs represent what this agent knows/assumes about its environment. Beliefs are created through actions such as getting current state of environmental parameters including measuring their values, observing events of change or time passing, getting information about agents such as their location behaviour, progress of their processes and so on. *Goals* describe what is expected of agents to perform, what kind of state is to be achieved, or maintained or what policy should hold. *Plans* are concrete courses of action, performed by the agent to achieve its objectives; for example what are acceptable options for scheduling a patient with cardiac problems through various health departments to minimise her waiting time.

In addition to these mentalistic notions, BDI also offers a mechanism that allows deducing the agent actions in a rational manner from the current context. Thereby, goals can be seen as one motivational source for generating actions, as for a goal being pursued the BDI mechanism selects applicable plans under consideration of the current situation represented by the beliefs. These plans will subsequently get executed until the goal is reached or becomes unreachable (Braubach et al., 2004).

3.2 The multi-agent platform Jadex

The BDI model as briefly sketched above is implemented in the newly developed multi agent platform *Jadex*, which enhances the traditional BDI architectures in several aspects (Pokahr et al. 2005). Most notably, Jadex supports explicit goal representation and has built-in deliberation support for possibly conflicting goals.

In Jadex we distinguish between four goal types that address fundamentally different behaviour: perform, achieve, query and maintain. A

perform goal is directly associated with actions that should be performed. An example could be “perform ward round”. An *achieve* goal describes a world state that should be brought about, e.g. “achieve x-ray examination for patient Miller”. *Query* goals are a means for an agent to retrieve information, for example “query charge nurse of the children’s unit”. Finally, *maintain* goals can be used to monitor a certain world state and re-establish this state whenever it gets violated, e.g. a “maintain antibiotics available” could be used to automatically reorder urgently needed medicaments.

Figure 1 depicts the Jadex architecture. At an abstract level, an agent can be seen as a black-box receiving and sending messages. Incoming messages, internal events and goal expressions are input for the agent’s reaction and deliberation mechanism. Deliberation is thereby performed on two levels.

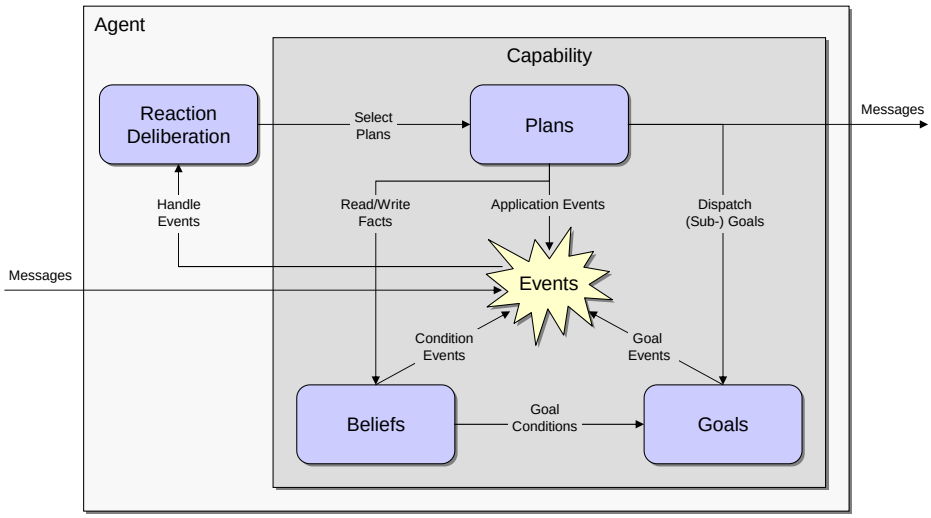

Fig. 1 Jadex architecture

On the goal level the component is responsible for deciding which goals currently need to be pursued. On the plan level, the traditional BDI mechanism as outlined in (Rao and Georgeff, 1995), is applied to select an option to bring about a given goal. To this end, all plan candidates applicable in the current context will be searched from the library of plans that contains the complete procedural knowledge set of an agent. From this set the agent will select the most acceptable by performing meta-level reasoning and will attempt to carry this plan out. If the plan fails the plan selection process starts over again, meaning that the agent can recover from plan failures by searching for alternative ways of achieving a goal.

When the agent processes incoming messages or internal events its behavior is more reactive as only deliberation on plan level is performed. At

runtime plans can access the full range of BDI specific and general purpose actions, such as accessing the belief base, creating new top-level or sub-goals, and sending messages to other agents.

A Jadex agent consists of two kinds of specifications. On the one hand the static structure of an agent type is defined in an agent definition file (ADF), including the specification of its initial beliefs, plans and goals. The ADF is an XML file, which complies with the Jadex language as defined in a BDI meta-model within XML Schema. This structure specification is augmented by a declarative expression language for all elements such as goals and beliefs, which follows the Java syntax. The plan code, on the other hand, is written in Java and can access the BDI specific agent characteristics through an API.

4. EVENT-BASED POLICY MONITORING

The combination of event-based technology capabilities and business pressures towards better governance, more accountability and better compliance, have placed more emphasis on *event-based monitoring* of enterprise systems. In the past, monitoring aimed at providing dashboard-like information about the status of the system and generation of appropriate alarms and notifications to signify undesired states of the system.

However, the current regulatory and legal policies require more proactive and real-time detection of policies or best practice violations of key actors involved in enterprise systems. This is true for both commercial and government environments. In addition, some inadequate medical treatments and health service delivery in a number of countries have identified a need for similar approaches to the public accountability in health domain. The situation is however more difficult in this domain, considering the complex nature of services and resource pressures in many health jurisdictions.

To partly address these issues, we believe that it is valuable to consider how some of the benefits of the agent technology discussed above could be augmented with policy-based constraints associated with agent's behaviour. This in turn would facilitate support for the event-based monitoring of selected activities of patients and health professionals.

In health applications, there is a wide range of policies that apply, with different measures to be taken in case of violations. These range from those that do not have significant risk on patients to those that have high risks and are not allowed to be violated at all. Accordingly, there are various measures that could apply such as preventative enforcement for the policies whose violations would have life-critical consequences to more discretionary enforcement in cases where failures to satisfy policies do not have so severe

consequences. Below, we provide some examples of policies which may apply in the health domain. For example:

"A nurse is obliged to report to the doctor at least one day after the patient's examination".

"A doctor must not reveal the results of an examination before the patient gives permission for that".

"A doctor is allowed to prescribe (only) medicine which costs less than $ 1000".

"*The waiting time for any patient before being assigned to a resource shall not exceed 2 hours".*

In our previous work we have developed a generic policy language that is inspired by a deontic logic formalism. This language was used as part of a broader business contract language (BCL), as presented in (Milosevic et al., 2004a, 2004b). The language provides a declarative way of expressing key policies such as permissions, prohibitions and obligations similar to the natural language expressions. In brief, the key concepts of the language are:

- Behavioural constraints, expressed in terms of *event patterns*; an event pattern describes the relationship between events that reflect one or more actions of actors, or other occurrences such as changes in states of environmental variables; event patterns specify some combination of events of relevance for a policy; a singleton event type is used when listening for occurrences of a single event
- The organisational *role* to which the policy (i.e. the corresponding event pattern) applies
- The *modality*, i.e. obligation, permission or prohibition;

By applying this policy language, the policies as introduced in natural language introduced above could be expressed as follows.

Policy: NurseReporting
Role: Nurse
Modality: Obliged
Condition: AdviseDoctor after (PatientExamination) and before (PatientExamination add 1 day)

In this obligation policy above, the *AdviseDoctor*, *PatientExamination*, and *PatientExamination* are event types and their relationship is expressed as an event pattern. In this example we use *after*, *before*, *and* and *add* relationship operators. For a more detailed description of event operators available in the language see (Milosevic et al., 2004a, 2004b).

Policy: PrivacyRespect
Role: Doctor
Modality: Not Permitted
Condition: RevealExaminationResult before PatientGivesPermission

The example above is for a prohibition policy while a permission policy is:

Policy: Doctor
Role: DoctorGradeA
Modality: Permitted
Condition: PrescribeMedicine (PrescribeMedicine.cost < $1000)

This last example also shows the use of a singleton event pattern and how this event pattern exploits information from the *cost* parameter of the event *PrescribeMedicine* to check the satisfaction of this policy.

Policy: PatientWaitingTime
Role: resourceAgent
Modality: Obliged
Condition: PatientAssignmentToResourceTime before (PatientArrivalTime add 2 hrs)

Note that in the first and last policy, the *add* operator is used to describe a relative time point.

It is worth mentioning that this policy language was developed based on the consideration of a number of formalisms for the expression of deontic modalities and normative concepts and on a more pragmatic analysis of various business situations, in particular in the e-contracting area. Thus, this is an example of a domain specific language for the purpose of expressing enterprise policies. Our experience so far suggests a high expressive power of the language, for the expression of both the behavioural and the structural aspects of enterprise systems. Further description of the language is beyond the scope of this paper; more details could be found, e.g., in (Linington et al, 2004 and Milosevic et al., 2004a, 2004b).

5. POLICY-ENRICHED AGENT SOCIETIES

This section investigates options for incorporating key aspects of the policy language described above. This is to add further capabilities to the software agents developed as part of Jadex system so that it is also possible to support policy-constrained applications. The aim here is to consider how options for applying policy-enriched agents could be applied to a wide variety of problems in e-health applications.

5.1 General implementation considerations

Policies and constraints as presented above are high level, abstract descriptions of behaviour expected from people (e.g. nurses or doctors) or agents (e.g. representing hospital resources). The abstract descriptions do not specify how these policies are handled by an underlying software system. Policy constraints could be just monitored (taking certain actions when they are violated) or directly enforced (i.e. prohibiting all actions that are not permitted). Mapping these policies to concrete agent implementations has to make these design decisions explicit.

Direct enforcement of policies is only possible, when users and agents perform all actions through the system. For example, when a request for an

action is issued to the system, the system can check if proper permissions are available, before actually performing that action. Actions directly performed or required by persons thus cannot be enforced by the system, but could be monitored. A special type of agents, *monitoring agents* can be designed and implemented to observe the behaviour of people and other software agents, and possibly to take appropriate steps when certain events occur, e.g. the detection of existing or possibly arising policy violations. Depending on the criticality of policies, monitoring agents might just log violations for later evaluation, or initiate complex emergency procedures.

5.2 Applying policy agents to the hospital domain

In order to support observation of behavioural constraints and reaction to policy violations in the hospital domain, we introduce another type of agents, namely *policy agents*. They act according to their goals which expressed in terms of pre-defined policies and the plan options associated with each of the policies. Each plan would thus express specific actions through which the corresponding policy is to be satisfied. One style of expression could be by using business process concepts. Note that we assume that policy rules are typically defined as part of the environment in which software agents or people are located but, in this context, we are not concerned with the process of arriving at these policies.

In general, the task of a policy agent would be to detect existing or possibly forthcoming policy violations (such as in the execution of care plans) and to cooperate with other software agents for various purposes. One such purpose would be to notify other agents about possible problems so that these second group of agents, namely *notification agents,* can then send notifications, reminders or warnings to health professionals, patients etc.

So, a policy agent can, for example, recognize if some correlation of events or alarms (e.g. as a result of patient reaction to a medical treatment or as a result of a surgery) may signify a potential adverse effect of serious health consequences. If so, the policy agent could be further involved in monitoring subsequent escalation procedures. The procedures could involve different measures to be taken in cases of progressively higher levels of priorities for the actions to be undertaken by health professionals.

In terms of agents that monitor patients, i.e. *patient monitoring agents*, they can, e.g., monitor the state of patients or elderly people at home and if necessary either 'decide' whether to send reminders to the patients themselves or directly trigger some other agents in hospitals etc. Note that these software agents could also run on various mobile devices.

In summary, agents augmented by the policy constraints, could be used in a collaborative care delivery in order to help doctors and clinical

professionals and also the patients in making sure that agreed processes, best practices, and regulatory policies are satisfied.

5.3 Implementing BCL with BDI agents

We now present a way to implement BCL constraints by incorporating policy expressions as part of agents into Jadex and, thus, deriving concrete agent implementations for policy constraint applications. In order to do so, we propose a design where the system *monitors* all obligations, but *enforces* permissions. In other words, an agent would have to monitor policies with the "obliged" modality but should not allow its principal to execute actions that are "not permitted". A natural way to represent such policies in BDI agents is to map obligations to *maintain goals* and map permissions to *preconditions of plans* (i.e. actions), as described below.

Monitoring Obligations

First, we use 'maintain' type of goals in BDI agents for monitoring obligations. For this, such goals should be created and dropped in accordance to certain events in the domain. In terms of the example above, whenever a nurse agent is informed about a performed examination on a patient, it will create a maintain goal to keep track whether the nurse has advised a doctor about the patient's state within one day after the examination. In a similar way, the agent can be also programmed to remind the nurse, e.g. say one hour, before the day has expired.

Along the same line, the examples as presented in section 4.3, e.g. the NurseReporting and the PatientAssignmentToResourceTime policies which are modelled as obligations, can be mapped to the following corresponding Jadex maintain goal specifications:

```
<!-- "A nurse is obliged to report to the doctor at least one day
after the patient's examination" -->
<maintaingoal name="NurseReporting">
   <maintaincondition>
      AdviseDoctor after (PatientExamination)
      and before (PatientExamination add 1 day)
   </maintaincondition>
   <creationcondition>PatientExamination</creationcondition>
</maintaingoal>

<!-- "The waiting time for any patient before being assigned to a
resource shall not exceed 2 hours" -->
<maintaingoal name="PatientAssignmentToResourceTime">
   <maintaincondition>
      PatientAssignmentToResourceTime before
     (PatientArrivalTime add 2 hrs)
   </maintaincondition>
   <creationcondition>NewTeatmentAdministered</creationcondition>
</maintaingoal>
```

The examples present how policies could be mapped to Jadex agents, if the agent would be able to parse and interpret the BCL statements. As this is a topic of ongoing work, currently the policy statements as shown above would have to be mapped to Java expressions. In principle, for each obligation considered a corresponding maintain goal type has to be defined. Moreover, it is necessary to individualize the general policies and obligations by introducing variables for important aspects. This means, instead of a global policy enforcement component, the agent has one goal template for each policy, which will be instantiated when the context demands it, e.g. a nurse reporting goal is created whenever a new patient examination has been done (cf. the creation condition). This goal is parameterised by the concrete patient examination for which the monitoring has to be done. Similarly, in the second example a new goal is created whenever a new treatment is administered.

Enforcing permissions

The policy agent serves as a mediator between the system and its principal. When the principal requests certain actions, the policy agent can check for permissions of the respective action, as long as these are issued through the system. In BDI agents, these permission checks are naturally represented in plan preconditions, which specify the context required to perform a certain action. When the agent is currently unable to perform the requested action (due to insufficient permissions of the principal) a failure message can be generated, providing information to the user.

```
<!-- "A doctor is must not reveal the results of an examinations
before the patient gives permission for that" -->
<plan name="RevealExaminationResultPlan">
   <body><!-- plan actions go here... --></body>
   <trigger><goal ref="RevealExaminationResult"/></trigger>
   <precondition>Not before PatientGivesPermission</precondition>
</plan>

<!-- "A doctor is allowed to prescribe (only) medicine which costs
less than $ 1000" -->
<plan name="PrescribeMedicinePlan">
   <body><!-- plan actions go here... --></body>
   <trigger><goal ref="PrescribeMedicine"/></trigger>
   <precondition>(PrescribeMedicine.cost &lt; $1000)</precondition>
</plan>
```

Typically, a doctor could use his computer to reveal examination results by sending them by email to another doctor. As part of this process its personal agent will be requested from the user interface to perform this action for him by creating a new goal, e.g. “RevealExaminationResult”. The corresponding plan, “RevealExaminationResultPlan”, requires a precondition that the patient has given permission. When the doctor is not allowed to reveal the results, the plan is not applicable, and as no other

applicable plan can be found, the goal fails. This is reflected in the user interface by showing an adequate error message to the doctor. The usage of BDI plans for detecting and avoiding constraint violations additionally supports context dependent actions. By specifying more than one plan for a given goal preconditions can be used to choose among alternatives in the current context. For example, for the “PrescribeMedicine” goal, a second plan with the precondition cost of greater than or equal to $1000 could be introduced. This plan could automatically request permission for the subscription by the head of department.

6. CONCLUSIONS AND FUTURE WORK

In this paper we presented several current problems in the e-health domain. We highlighted the difficulties and complexities of the domain, arising from the size, distributed nature of health care (both physically and organisationally), and from possible uncertainty associated with patients’ reaction to treatments. We have then outlined one initial solution for dealing with an important class of complexities in this domain, namely the problems associated with ensuring that the actions of actors in the system are in accordance with the predefined set of rules that apply to the actors. These rules could correspond to the treatment procedures with which patients need to comply with or to the best care practices and evidence-based medicine with which health professionals need to comply. Thus, these rules have a lot of in common with the policies that apply in human societal organisations.

The solution proposed is developed to support those applications that require explicit dealing with the policies. This solution is based on the use of agent technology enriched with the expressions of policy constraints developed in our respective research organisations We found that the BDI agent model in general quite nicely reflects the key policy principle of specifying a set of behaviour choices over basic behaviour. This enables us to model e-health application problems in a direct and natural way involving relatively straightforward mapping from our policy language to BDI and Jadex goals and plans in particular.

In future, we plan to complete the mapping of the policy language to the Jadex BDI based agent system. We also plan to test the policy-enriched agent system using several real e-health applications. A particularly complex, but also one of the most important emerging e-health applications is chronic disease management, such as coronary artery disease, diabetes, and asthma. This segment incorporate all key features discussed in the paper and requires the tracking of patients over time and over various locations

(including in their home) to monitor progression of the disease, compliance with the treatment and preventative care.

7. REFERENCES

Berry, A., Milosevic, Z., 2005; Extending executable choreography with business contract constraints, with A. Berry, submitted to the International Journal of Cooperative Information Systems (IJCIS), to appear, in Vol 14, Nos. 2 & 3, Jun & Sep 2005

Braubach, L., Pokahr, A., Lamersdorf, W., Moldt, D., 2004, Goal representation for BDI agent systems, Proc. Second International Workshop on Programming Multiagent Systems: Languages and Tools, R.H. Bordini, M. Dastani, J. Dix, A. El Fallah-Seghrouchni (Eds.), pp. 9-20

e-GIF, http://www.govtalk.gov.uk/egif/specifications.asp

e-Gov, http://www.whitehouse.gov/omb/egov/c-3-6-chi.html

Health-Connect, http://www.healthconnect.gov.au

Jennings, N., Wooldridge, M., (eds.), 1998, Agent Technology: Foundations, Applications and Markets, Springer Verlag

O'Keefe, C., Greenfield, P., Goodchild, A., 2005, A Decentralised Approach to Electronic Consent and Health Information Access Control, Journal of Research and Practice in Information Technology, Vol. 37, Number 2, Australian Computer Soceiety.

Linington, P., Milosevic, Z., Cole, J., Gibson, S., Kulkarni, S., and Neal, S., 2004, *A Unified Behavioural Model and a Contract Language for Extended Enterprise* Data Knowledge and Engineering Journal 51, p. 5-29, Special issue on contract-driven coordination, October 2004

Milosevic, Z., Gibson, S., Linington, P. F., Cole, J., Kulkarni, S., 2004a, On design and implementation of a contract monitoring facility, IEEE Conference on Electronic Commerce, the 1st IEEE Workshop on Electronic Contracting, San Diego, July 2004

Milosevic, Z., Linington, P. F., Gibson, S., Kulkarni, S., Cole, J., 2004b, Inter-organisational collaborations supported by e-contracts, Proc. 4th IFIP conference on E-commerce, E-Business, E-Government, Toulouse, France

Nealon, J. Moreno, A., 2003, *Agent-based health care systems*. In *Applications of Software Agents Technology in the Health Care Domain*, J.Nealon and A.Moreno Eds., Whitestein series in software agent technology, Birkhäuser

NEHTA, http://www.nehta.gov.au/

Paulussen, T. O., Jennings, N. R., Decker, K. S. , Heinzl, A., 2003, Distributed patient scheduling in hospitals, Proc. *International Joint Conference on Artificial Intelligence*, Acapulco, Mexico

Paulussen, T. O., Zöller, A., Heinzl, A., Braubach, L., Pokahr, A., Lamersdorf, W., 2004, Patient scheduling under uncertainty, H. M. Haddad, *Proceedings of the 2004 ACM Symposium on Applied Computing* (SAC2004), A. Omicini, R.L. Wainwright, L.M. Liebrock (Eds.), Nicosia, Cyprus, ACM Press, pp. 309-310

Pokahr, A., Braubach, L., Lamersdorf, W., 2005, *Jadex: A BDI Reasoning Engine*, in R. Bordini, M. Dastani, J. Dix and A. Seghrouchni, (eds.), Multi-Agent Programming, Kluwer (to appear in 2005)

Rao, A., Georgeff, M., 1995, BDI Agents: From Theory to Practice, Lesser, V. (Eds.) Proceedings of the First International Conference on Multi-Agent Systems (ICMAS'95), MIT Press: Cambridge, MA, pp. 312-319