EU PROVENANCE project: an open provenance architecture for distributed applications

Javier Vázquez-Salceda, Sergio Álvarez, Tamás Kifor, László Z. Varga, Simon Miles, Luc Moreau and Steven Willmott

Abstract. The concept of *provenance* is already well understood in the study of fine art where it refers to the trusted, documented history of some work of art. Given that documented history, the object attains an authority that allows scholars to understand and appreciate its importance and context relative to other works of art. This same concept of provenance may also be applied to data and information generated within a computer system; particularly when the information is subject to regulatory control over an extended period of time. Today's distributed architectures (not only Agent technologies, but also Web Services' and GRID architectures) suffer from limitations, such as lack of mechanisms to trace results. Provenance enables users to trace how a particular result has been arrived at by identifying the individual and aggregated services that produced a particular output. In this chapter we present the main results of the EU PROVENANCE project and how these can be valuable in agent-mediated healthcare applications. For the latter we describe the Organ Transplant Management Application (OTMA), one of the demonstrator applications developed.

Keywords. Provenance, software agents, healthcare.

1. Introduction

The importance of understanding the process by which a result was generated is fundamental to many real-life applications in science, engineering, medical domain, supply management, etc. Without such information, users cannot reproduce, analyse or validate processes or experiments. Provenance is therefore important to enable users, scientists and engineers to trace how a particular result came about.

Most distributed solutions can be seen as networks of computational services at distributed locations, which operate by dynamically creating services at opportunistic moments to satisfy the need of some user. These services may belong to different stakeholders operating under various different policies about information sharing. The results provided by such a composition of services must, however, be trusted by the user and

yet, when the services disband, the following question arises: how are we to obtain the verification of the processes that contributed to the final result?

This problem is especially relevant for distributed medical applications. In such applications the data (containing the healthcare history of a single patient), the workflow (of the procedures carried out on that patient) and the logs (recording meaningful events in those procedures) are distributed among several heterogeneous and autonomous information systems. Communication and coordination between organizations and among members of a medical team are critical issues the distributed application should address, in order to ease information sharing and to provide some support to distributed decision making. One approach to model and implement distributed medical applications is the use of agent-based techniques [10]. Modelling application components as agents with some degree of autonomy eases the development phase as it makes it easier to reflect the decentralized nature of the network of healthcare institutions and actors involved in a healthcare process, and also eases the integration of systems owned and developed by different authorities and also humans in the system, by encapsulating them in agents or agent-mediating interfaces.

Even when using agent technologies, the distributed nature of healthcare institutions sometimes makes it really hard to obtain overall views of the treatments of patients, because documentation of the healthcare history and therapy of a patient is split into independent healthcare institutions. However, more and more healthcare applications tend to move towards a user-centric perspective. In order to provide better, user-centered healthcare services, the treatment of a patient requires viewing the processes and data as a whole. Although agent-based cooperation techniques and standardized electronic healthcare record exchange techniques support the semantic interoperation between healthcare providers, we still face the problem of the reunification of the different pieces of the therapy of a single patient executed at different places. Currently there are some countries that have no unification method for patient healthcare records; each region in the country or even each institution inside a region may have its own medical record system, sometimes not even fully electronic, and with no automatic healthcare record exchange mechanisms. Therefore, it is not uncommon for doctors to depend on the patients themselves in order to include data from previous treatments and tests. Furthermore, in medical (and other critical application) domains, there is also a need to provide ways to analyze the performance of distributed healthcare services, and to be able to carry out audits of the system to assess that, for a given patient, the proper decisions were made and the proper procedures were followed.

In this chapter we present a new approach to both capture the distributed medical treatment of a patient in different health institutions in an integrated, patient oriented way, and to register all meaningful events related to a patient's treatment for further analysis, not only for audit purposes but also for medical staff to detect problems in the medical processes (e.g., bottlenecks or lack of timely information) in the processes they are involved into. Our main hypothesis is that trust in results produced by an agent-mediated distributed healthcare system can be increased if the provenance of each of the particular results can be known (e.g., where the patient was treated, who has been involved in each medical treatment, who has taken decisions and which were the basis for such decisions).

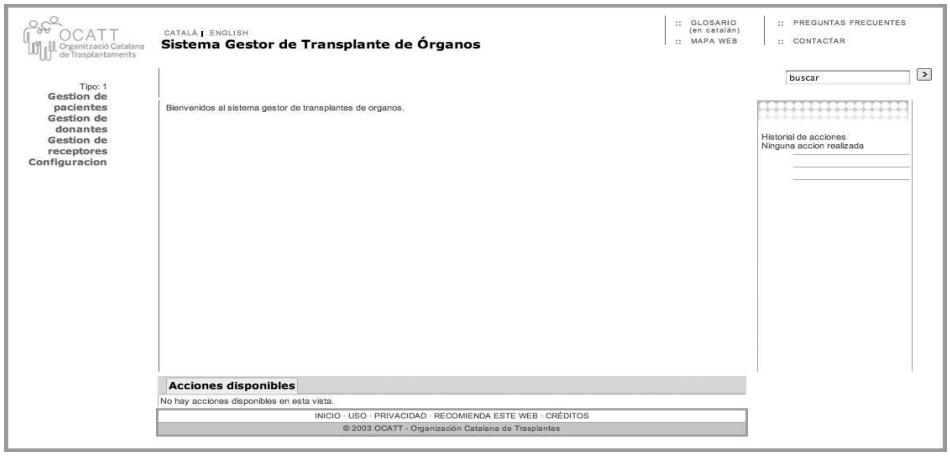

FIGURE 1. The Organ Transplant Management Application (OTMA) user interface

The content is structured as follows: in Section 2 we define the *provenance* concept and describe the the technological developments in the EU PROVENANCE Project; then in Section 3 we briefly present the Organ Transplant Management Application (OTMA), which we will use as example of the use of provenance in agent-mediated healthcare applications; in Section 4 we describe the process undertaken to make the OTMA application *provenance-aware*; in Section 5 we explain how the recorded provenance data can be used to analyse relevant events related to a medical process; in Section 6 we describe the problem on connecting medical process documentation between heathcare institutions; in Section 7 we discuss the privacy issues that may arise by introducing provenance recording in healthcare applications; finally in Section 8 we conclude by summarizing our approach and referring to related work in the literature.

2. Provenance

A key contribution of the IST-funded EU PROVENANCE project, and a technology which underpins the rest of the work described in this chapter, was the development of a *provenance architecture* [6]. Where an application is integrated with an implementation of the architecture, users have facilities to determine the *provenance* of data items produced by that application, i.e., the causes of a data item being as it is. The provenance of an item is extracted from the documentation of processes occurring within the application. In this section, we describe the nature of the provenance architecture, and the structure and use of the process documentation.

2.1. Provenance Architecture

The provenance architecture is comprised of component interfaces, data models, protocols and agent behaviour specifications. Following this approach, each agent independently records documentation regarding the processes it is involved in. The documentation is structured in a form which then allows queriers to trace back through the full, distributed process that preceded a data item's creation or modification.

The provenance architecture has key properties which allow for its wide applicability, scalability and robustness. First, it is technology-independent, allowing it to be deployed in Grid-based applications, Web Service deployments and multi-agent systems in general. Second, no dependencies are required between agents within the system in order to record process documentation: recording is performed independently and autonomously, and no agent is assumed to have access to the state of any other. Third, while conceptually being recorded during execution, documentation of a process can be recorded asynchronously from the process itself. Both the latter two issues are important factors in preserving the performance of large-scale systems. Finally, the application will not be adversely affected if accurate documentation is not available, because few assumptions are made about the documentation. For example, documentation can be complete or partial (for instance, when the computation has not terminated yet); it can be accurate or inaccurate; it can present conflicting or consensual views by the agents involved; it can describe the process at differing levels of detail and abstraction.

Aside from the architecture itself, the project produced an open source reference implementation [1] and a methodology that aids application developers in integrating and exploiting the provenance architecture in their systems [9]. The research was applied not only to healthcare, but also distributed aerospace simulations and bioinformatics experiments, and potential uses were explored in many other sciences [8].

2.2. Process Documentation

The provenance of a data item is represented in a computer system by a set of *p-assertions* made by the actors involved in the process that created it. A p-assertion is a specific piece of information documenting some step of the process made by an actor and pertains to the process. We follow a simple model of process, whereby agents communicate information via *messages*, the sending of one message by one agent and the receiving of that same message by another agent being called an *interaction*. A process consists of a series of exchanges of messages between agents, and processing of the data within those messages by the agents. There are three kinds of p-assertions that capture an explicit description of the flow of data in a process:

- An *interaction p-assertion* is an assertion of the contents of a message by an agent that has sent or received that message.
- A *relationship p-assertion* is an assertion about an interaction, made by an agent that describes how the actor obtained data sent in that interaction by applying some function to input received in other interactions.
- An *actor state p-assertion* is an assertion made by an agent about its internal state in the context of a specific interaction.

Within the architecture, a long-term facility for storing the process documentation described above is defined, called a *provenance store*. A provenance store is used to manage and provide controlled access to the representation of the provenance of a specific data element. As part of the architecture, a recording and two querying interfaces are defined for the provenance store. The *process documentation query* interface allows p-assertions to be retrieved singly or in groups by criteria. The *provenance query* interface returns a trace of all process documentation in the process producing a given data item, i.e., that item's provenance. It allows the results of the query to be scoped to that relevant to the querier, e.g., within a given period of time or at a given level of abstraction.

In the case of agent-mediated healthcare systems, by recording documentation on all the medical processes related to a given patient, one can then re-construct the treatment history of the patient. Therefore, making an agent-mediated healthcare system *provenance-aware* provides a way to have a unified view of a patient's medical record along with its provenance, i.e., to connect each part of the medical record with the processes in the real world producing it and/or the individuals, teams or units responsible for each piece of data within it.

3. OTM/EHCR: applying provenance in agent-mediated healthcare applications

In this chapter we demonstrate the potential usage of provenance in distributed healthcare systems by describing our experience in the domain of Organ Transplant Management. Distributed Organ Transplant Management is an excellent case study of both provenance and the privacy issues of provenance. Treatment of patients through the transplantation of organs or tissue is one of the most complex distributed medical processes currently carried out. This complexity arises not only from the difficulty of the surgery itself but also from the fact that it is a distributed problem involving several locations (donating hospital, potential recipient hospitals, test laboratories and organ transplant authorities), a wide range of associated processes, rules and decision making. Depending on the country where a transplant is being carried out, procedures and the level of electronic automation of information / decision making may vary significantly. However, it is recognized worldwide that ICT solutions which increase the speed and accuracy of decision making could have a very significant positive impact on patient care outcomes. In [12, 13] we presented CARREL, an Agent-Mediated Electronic Institution for the distribution of organs and tissues for transplantation purposes. One of the aims of the CARREL system was to help speeding up the allocation process of solid organs for transplantation to improve graft survival rates. Several prototypes of the CARREL system have been developed using JADE [3]. Although medical practitioners positively evaluated the prototypes, system administrators proved to be very reluctant to manage agent platforms for critical medical applications, and prototypes didn't go through. In [14] a connection between Agent Communication Languages and Web Service Inter-Communication was proposed. This connection allows us to implement agent systems by means of web services which can interact following the same FIPA protocols [5]. With this approach we developed a new

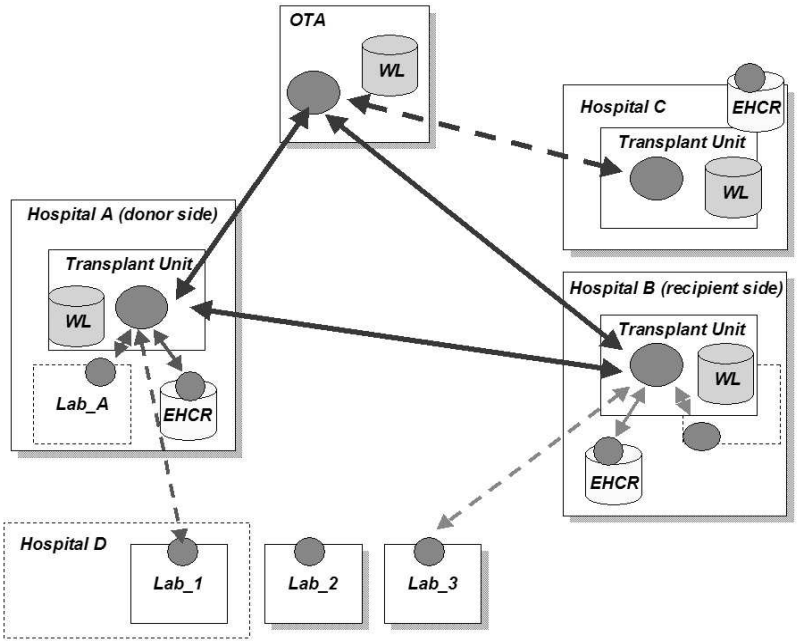

FIGURE 2. Actors in the OTMA system. Actors communicate and coordinate with each other through agents (circles in figure).

prototype, the Organ Transplant Management Application (OTMA) which uses standard web service technology and it is able to interact with the provenance stores in order to keep track of the distributed execution of the allocation process for audit purposes.

Management of the electronic health records distributed in different institutions is provided by the Electronic Healthcare Record System (EHCR). Its internal architecture provides the structures to build a part of or the entire patient's healthcare record drawn from any number of heterogeneous databases systems in order to exchange it with other healthcare information systems. The EHCR architecture has two external interfaces: 1) a Web Service that receives and sends messages (following FIPA protocols [5] and the ENV13606 pre-standard format [4] for the content) for remote medical applications, and 2) a Java API for local medical applications that can be used to access the EHCR store directly.

Figure 2 summarizes the different administrative domains (solid boxes) and units (dashed boxes) that are modeled in the OTMA system. Each of these interact with each other through agents (circles in the figure) that exchange information and requests through messages. In a transplant management scenario, one or more hospital units may be involved: the hospital transplant unit, one or several units that provide laboratory tests and the Electronic Healthcare Record (EHCR) subsystem which manages the healthcare

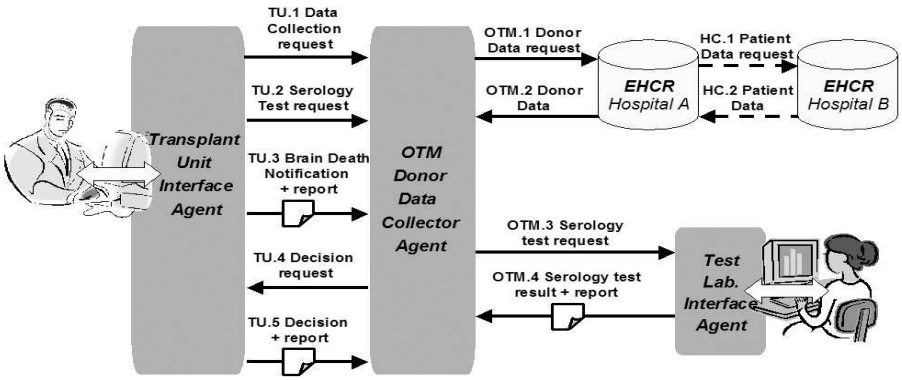

FIGURE 3. Example of interaction in the OTMA system.

records for each institution. The diagram also shows some of the data stores that are involved: apart from the patient records, these include stores for the transplant units and the Organ Transplant Authority (OTA) recipient waiting lists (WL). Hospitals that are the origin of a donation also keep records of the donations performed, while hospitals that are recipients of the donation may include such information in the recipient's patient record. The OTA has also its own records of each donation, stored case by case.

4. Making the OTMA system provenance-aware

Making the OTMA system provenance-aware presented three challenging issues: a) the provenance of most of the data is not the execution of computational services, but decisions and actions carried out by real people in the real world (this is discussed in this section); b) past treatments of a given patient in other institutions may be relevant to the current decisions in the current institution, so information of the processes undertaken in those previous treatments should be connected to the provenance information of a current process (this is discussed in Section 6); c) the agent with provenance information knows much more about the patient than any other agent in the system, so there are privacy risks to be mitigated (this is discussed in Section 7).

In the case of the OTMA system, each organizational unit is represented by an agent-mediated service. Staff members of each unit can connect to the unit services by means of graphical user interfaces (e.g., see the one in Figure 1). The distributed execution of the OTM services is modeled as the interaction between the agents, and recorded as interaction p-assertions and relationship p-assertions. As in the OTM scenario a decision depends on the human making the decision, additional actor state p-assertions are recorded, containing further information on why the particular decision was made and, if available, the identities(s) of the team members involved in the decision.

To illustrate how provenance is handled in the OTMA system, let us see how the provenance of a medical decision is recorded. Figure 3 shows a simplified view over

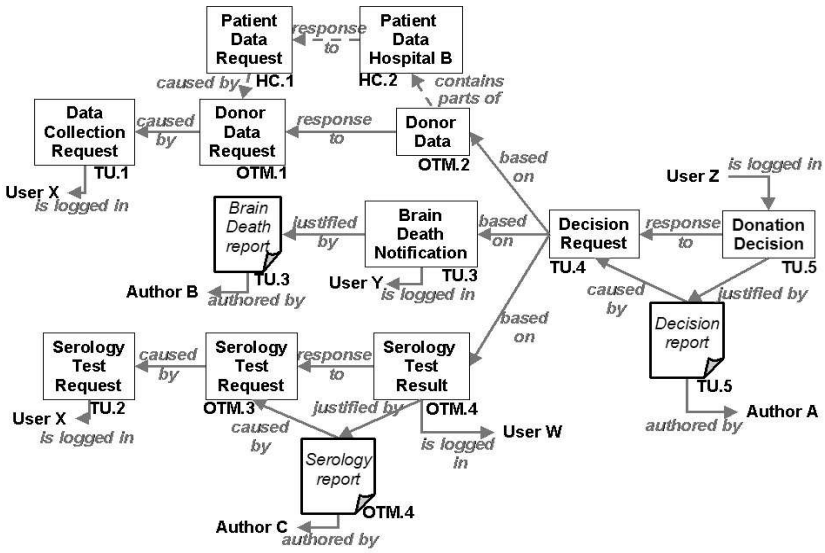

FIGURE 4. Example of provenance trace for the Donation Decision.

a subset of the donation process. In this example a patient (who has previously given consent to donate his organs) enters into a given hospital in critical condition. As the patient's health declines and in foresight of a potential organ donation, one of the doctors requests the full health record for the patient and then orders a serology test[1] through the OTMA system. After the patient enters into a severe comma, a doctor declares a "brain death" condition for this patient and such relevant event is logged in the system (along with the report certifying the brain death). When the system detects that all requested data and analysis results have been obtained, the system sends a request to a doctor to make a decision about the patient being a potential donor. This decision is explained in a report that is submitted as the decision's justification and which is logged in the system.

Figure 3 shows the OTMA agents for this small scenario and their interactions. The Transplant Unit User Interface Agent passes requests (TU.1, TU.2) to the OTM Donor Data Collector Agent, which then gets the electronic record from the EHCR system (OTM.1, OTM.2). Sometimes all or parts of the record are not in the same institution but located in another institution (HC.1, HC.2). The Donor Data Collector Agent also sends the request for a serology test to the laboratory and gets back the result (OTM.4), along with a detailed report of the test. Reports are also passed in the case of the Brain Death notification (TU.3) and the final decision report (TU.5).

Figure 4 graphically represents the subset of the p-assertions produced by the provenance-aware OTMA which are related to the mini-scenario described in Figure 3.

[1] A serology test is usually performed over blood samples to detect viruses (HIV, Hepatitis B/C, syphilis, herpes or Epstein-Barr virus), which, if present in the organ, can pass to the recipient.

The part of the process that happens within the electronic system is represented by interaction p-assertions (regular boxes) for all interactions (TU.x, OTM.x, HC.x), and relationship p-assertions (`response_to`, `caused_by`, `based_on`) capturing dependencies between data. Even though what happens in the system parallels what happens in the real world, as we already said this is not enough to fully determine the provenance of a given decision. To solve this, we connect the electronic process to the real world by adding actor state p-assertions stating who logged the information in the system (`is_logged_in`) and when (not shown in picture), which are the reports that justify a given state in the system (`justified_by`), who are the authors of these reports (`authored_by`) and when the action reported was performed or the decision taken (not shown).

5. Analyzing the distributed medical process through provenance

Storing provenance documentation instead of the, more common, standard log systems, has the advantage that the provenance representation is stored in a way that complex queries can be performed over it, which allows a provenance-aware system to extract valuable information to validate some of the steps taken into a (medical) process, or even to make an audit of the system over a period of time.

In the OTMA system, apart from periodical audits, transplant coordinators also want to ask the following types of provenance questions, related to a given patient (donor or recipient) or to the fate of a given organ:

- Where did the medical information used on each step of the process come from?
- When was a decision taken, and what was the basis of the decision?
- Which medical actors were asked to provide medical data for a decision?
- Which medical actor refused to provide medical data for a decision?
- Which medical actor was the source of some piece of information?
- What kind of medical record was available to actors at each step of the process?
- When was a given medical process carried out, and who was responsible for it?

All these kind of questions can be answered by querying the provenance store. A query will give as a result (a subset of) the provenance representation graph of the process related to the query. If we use as an example the graph in Figure 4, by following the edges from the “Donation Decision” p-assertion we can trace the provenance of the donation decision, how it was based in some data and test requests, how a brain death notification is also involved, who requested the information, where it came from (in some cases it might come from the EHCR of another hospital), and who authored the justifying reports in the main steps of the process.

In those cases (as in Figure 4) where the decision might be based on medical data coming from tests and medical treatments carried out in other institutions, another issue to solve is the following: how to find, retrieve and incorporate the provenance of the data coming from the other institution? If these institutions have also provenance-aware systems and the provenance stores of the different institutions are connected, to solve the aforementioned problem is to solve the issue of matching the different p-assertions related to the same patient. If this match is done, then actors can make p-assertions that

link together the separate sets of p-assertions to create a larger provenance document providing an integrated view of the healthcare history of the patient. The result (not shown on Figure 4) would be that the p-assertions related to Patient Data Hospital B would be linked to the set of p-assertions already part of the provenance of the Donation Decision (by means of the method that we will describe in Section 6).

Collectively the p-assertions can be seen as describing a distributed process, spanning space as well as time. Every relationship described is causal, i.e., between the cause of something happening and the effect of it happening, and is therefore also temporal, i.e., causes always come before effects. Furthermore, extra information can be added to provide further detail. For example, an actor may record, as an actor state p-assertion, the time shown on their local clock. Together, the structured documentation of processes allows a rich set of questions to be asked about what occurred, why, when and by whom and, in the OTMA system, such a process may be a patient's healthcare history

6. Connecting medical process documentation to create a patient's integrated view

As seen in the previous section, in order to be able to create an integrated view of a patient's healthcare history, there must be a series of links between any two p-assertions of the process documentation created by each healthcare institution. The links in the process documentation are interaction p-assertions and relationship p-assertions which connect together the p-assertions of agents in the process. In usual service-oriented applications these links are created by use of a common identifier, called an interaction key, for both parties, sender and receiver, in an interaction. If two agents record p-assertions using the same interaction key, we can determine that their actions are part of the same process, and therefore both are part of the provenance of the process' output. We can record p-assertions with the same interaction key, if the two agents exchange that key, which means they must electronically interact.

In typical business or e-science applications, the agents participating in the process are in contact with each other and exchange documented messages while there is an interaction between them. In this case we say that there is *direct interaction* between the agents. However medical processes, and some other types of processes, are different, because the physicians treating the same patient may not be in direct contact. A typical example would be the following: a patient P is treated by one physician in health institution $H1$; then patient P is released from $H1$; months later the patient goes to another physician in health institution $H2$ with symptoms of another disease. Sometimes there may even be a medical relationship between the two treatments, for example because the second disease is a consequence of the first disease, but neither the patient nor the physicians are aware of this. In this case, the physicians are not in contact with the each other. From the process documentation point of view it is also important to note that the physicians do not know each other's identity, and they may use different identifiers to identify the patient in their process documentations, because the identities cannot be revealed for privacy reasons. This way the p-assertions belonging to the same patient cannot be linked

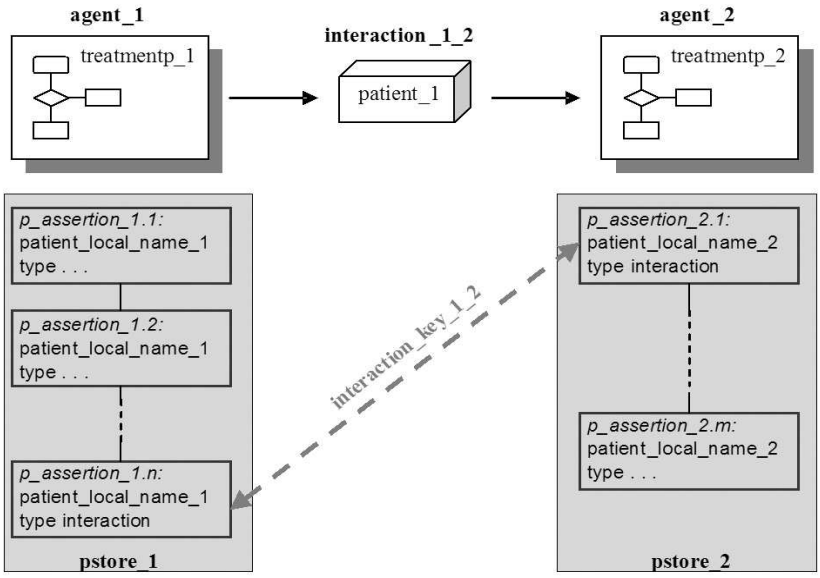

FIGURE 5. Process documentation in strongly connected processes.

together automatically, because the p-assertions cannot be located by the patient identifier. In this case we say that there is *latent interaction* between the physician agents. Note that the patient usually cannot determine the link between the current treatment and the previous one. This is not only because the patient does not remember the previous treatment, but also because the second physician cannot locate the p-assertions made by the first physician, due to lack of known identifiers.

Based on the distinction between direct and indirect electronic interactions we can define two types of processes: *strongly connected processes* and *weakly connected processes*. The processes can be seen as graphs, where the nodes represent the activities executed by the agents alone and the arcs represent the interactions between the agents. The interactions are either latent or direct.

- **strongly connected processes**: A process is strongly connected if the graph representing the process contains only direct interactions. Figure 5 shows the model of a strongly connected process and its process documentation. Agents 1 and 2 represent the physicians who are the actors of treatment processes 1 (`treatmentp_1`) and 2 (`treatmentp_2`). When agent 1 sends the patient to agent 2 in a documented way, the p-assertions about this interaction are recorded by agents 1 and 2. In a medical application we cannot use the globally unique identifier of the patient in the local systems of the agents, because it could be used to determine the identity of the patient. Both agents use a different local identifier for the patient, and when they interact directly and electronically, they agree on an interaction key which is included in their p-assertions about the interaction. This way the process documentations of the

two treatment processes are connected together with interaction p-assertions which contain the same interaction key. Therefore if some agent queries the process documentation using `patient_local_name_1`, then the provenance system is able to link the process documentations created by the two agents using the interaction key, and returns the complete process documentation comprising the provenance of the current healthcare status of the patient.

- **weakly connected processes**: A process is weakly connected if the graph representing the process can be cut into two or more sub-graphs, where the connections between the sub-graphs are only latent interactions. Typically the full healthcare history of a patient is created by a weakly connected process containing strongly connected sub-processes. Collecting the whole process documentation of all treatments of a patient is a bit more complicated in the case of weakly connected healthcare processes, because there is no direct interaction between the agents. The difference from the strongly connected process is that there is no link across the sets of p-assertions of the processes executed by the different agents. We could represent this situation graphically if, in Figure 5 above, we delete both the direct interaction (`interaction_1_2`) and the link between the medical processes documentation (`interaction_key_1_2`). If we want to retrieve the complete provenance of the current healthcare status of the patient, then we would like to retrieve both sets. In addition to this, the agents are unable to connect the two sets of p-assertions, because even if agent 2 finds out somehow that treatment process 2 is some way a consequence of treatment process 1, it does not know the local identifier of the patient used by the other agent and cannot locate the relevant p-assertions made by agent 1. Note that although the patient usually presents to the physicians its global identifier (such as its social security number), this global identifier cannot be used in process documentation for privacy reasons, as discussed later in Section 7.

The basic transplant process of OTMA is strongly connected, because there are always direct interactions between the actors. However when they retrieve the full EHCR of the patient, which may contain data from previous treatments, the transplant process becomes “infected” with the latent interactions of the EHCR creation process.

In order to provide a solution to the problem of process documentation creation resulting from the lack of direct interaction between the agents, we need to find an intermediate way of interaction. This can be done with the help of an institution in a higher hierarchical level, which is in contact with both agents and knows about the patient as well. Medical domains are usually regulated by national and international bodies which assure that there are services which give a global identifier to the patient, such as the national security number. As we said before, the global identifier should not be used in documentation of privacy-aware medical processes, because regulations ordain the separation of medical information and personal identification. The fact that these data cannot be stored together leads to the use of anonymised identifiers to connect medical and personal data. Because of this, agent-mediated healthcare systems usually contain an anonymisation service to convert real patient identifiers to anonymised patient identifiers.

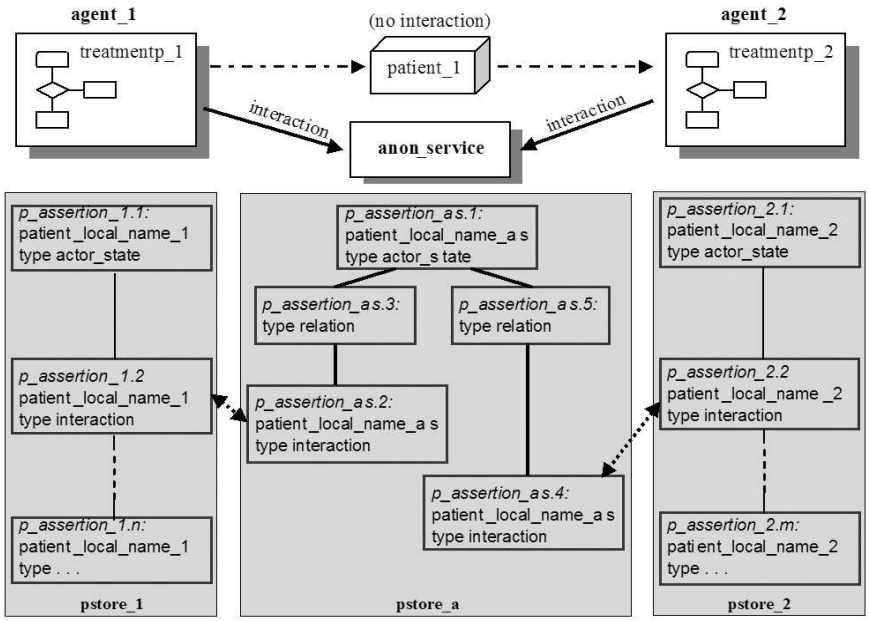

FIGURE 6. Connecting process documentation in weakly connected processes.

Figure 6 shows how the method of creating intermediate interactions and intermediate links in the process documentation works. In the first step of this method, we locate already existing anonymisation service in the application. If there is no such service, then we introduce it into the application. The service is called `anon_service` in the figure.

The second important element of the method is that the anonymisation service makes p-assertions about its own processing. Whenever the anonymisation service is asked to create a new patient identifier, then the anonymisation service puts an actor state p-assertion into the provenance store about the creation of the patient identity. This p-assertion does not contain the global patient identifier, only the anonymised identifier.

The third important element of the method is that each time a new case of a patient is started, the agents notify the anonymisation service. This notification is a direct interaction, therefore it is documented in the process documentation. This is shown in Figure . When agent 1 starts a new case on the patient, it makes an actor state p-assertion about the start of the case and notifies the anonymisation service that the case started. The direct interaction of the notification is recorded in the provenance store with interaction p-assertions on both sides. The anonymisation service knows the identity of the patient, and asserts a relationship p-assertion between the p-assertion related to the creation of the anonymised patient identity and the p-assertion related to the case start notification sent from agent 1 to the anonymisation service. Agent 2 does the same when it starts a

new case on the patient, therefore there will be indirect links between the two agents' processes, and the complete provenance of the patient record can be determined.

Although the anonymisation service is somehow a central interaction node in the system, scalability can be maintained. Concerning the amount of data going through the anonymisation service, there is no real bottleneck, because agents communicate limited amount of data with the anonymisation service. Moreover, the agents contact the anonymisation service only when they start a new case and then later there is no interaction with the anonymisation service during the execution of the case, because agents link further p-assertions to the start case p-assertion created by the anonymisation service. Further, the functionality of the anonymisation service can be distributed in real implemented systems among cooperating services allocated to different hierarchy levels, like countries, regions, insurance companies, etc.

The ability to return the whole process documentation using the method described above allows the agents to improve both the quality of the process documentation and of their own activities.

The quality of the process documentation can be improved if some causal relationship is discovered from the analysis of the real processes, e.g., the current illness of the patient is a consequence of a problem in the previous treatment not discovered before. In this case, the agents can insert additional links to the already existing process documentation created together with the help of the anonymisation service. The additionally inserted links document the real world causal relationships between the p-assertions of the already existing process documentation. Because the links created with the help of the anonymisation service integrate the p-assertions relevant to a single patient into a single graph, any p-assertion in this graph can be located and identified, so the link related to the real world causal relationship can be added.

The agents can improve the quality of their own activities with the help of the integrated process documentation. Now that the process documentation is integrated, agent 2 can retrieve the p-assertions of agent 1 and use this information in the current treatment.

7. Protection of Privacy in Provenance-aware Application

The issue of privacy in healthcare applications is extremely important. As reflected in the famous Hippocratic oath, protection of individuals' health-related data has been a continued concern of the medical body from the very beginning of the medical practice. There exist considerable efforts to put into practice a body of policies which ensure the protection of medical data in a scenario of massive use of computers in the health sector. Regulations define guidelines about the adequate organizational and technical measures that must be taken in medical information systems. The most important of these guidelines concerns the separation of data: as a general rule, the design of data structures, procedures and access control policies must be such that they allow the separation of a) identifiers and data related to a person's identity, b) administrative data, c) medical data, and d) genetic data. Such separation must ensure that no unauthorized person can connect the identity of the patient with his medical or genetic data.

In EHCR systems, and in the OTMA system discussed above, a typical solution for the separation of identity information and medical data is the anonymised identifier. The anonymised identifier is generated from a real patient identifier, and medical data is stored together with this anonymised identifier. If we know the real patient identifier, then we can find the corresponding medical data, but from the medical data we cannot find out the identity of the patient.

An anonymisation method must keep identifiers in secret during remote database management. The database is updated frequently, items are added and removed so we implemented the function on the client side of the database, i.e., in the web application, in order to keep the identifiers unknown for unauthorized people and applications. The function has the following features:

- It generates an unsigned long output for every unsigned long input.
- It uses two parameters to make the algorithm safer and reusable.
- It is deterministic, i.e., the output is always the same for a specific input value.
- It is injective i.e., it generates different output values for different inputs.
- The source code is private. That means that the only person who knows is the developer of the code.
- The final binary is deployed to a properly obfuscated JAR file to make the code breaking harder.

The above methods protect privacy in non provenance-aware healthcare applications, however when we make agent systems provenance-aware, we introduce the provenance store into the system, which needs additional protection, because there is a conflict between provenance and privacy. While for provenance we need as much information as possible about the whole process, for privacy we need to restrict as much as possible the information available, in order to avoid identification of patients and practitioners by unauthorized users.

The introduction of provenance in a distributed healthcare agent system poses two main risks:

- *cross-link risk*: the risk that unauthorised users are able to link some piece of medical data with an identifiable person by cross-linking information from different sources.
- *event trail risk*: the risk that unauthorised users are able to identify a person by connecting the events and actions related to that person (e.g., the hospitals he has visited in different countries).

Comparing the two risks above, the cross-link risk is more considerable than the event trail risk. In order to identify a person by exploiting the event trail risk, information not available in the healthcare information system (e.g., the places where he lived) has to be matched with the information in the healthcare information system. This requires more effort and information to exploit, than the cross-link risk which can be exploited using information available only in the healthcare information system. For these reasons, our current focus is on the cross-link risk.

In the provenance aware OTMA system we applied two techniques to protect privacy, mainly to reduce the cross link risk: a) we do not store sensitive medical data in the provenance store, and b) we use anonymised patient identifiers in provenance stores. Both

of these are supported by the process documentation integration method described in the previous section.

In order to hide medical data from cross-linking, agents do not store sensitive medical data in the provenance store, but only references to such data. This way the provenance store contains only the linkage and the skeleton of the provenance of the medical data, and the healthcare data can be laid on the skeleton by retrieving it from the healthcare information system when needed. The retrieval is done via the EHCR system which is completely under the control of EHCR access rules. With this approach we keep the same degree of privacy of medical data as in the original agent system.

One might think that if we do not store medical information about patients in the provenance store, then no medical information can be inferred about the patient and there is no need to anonymise the patients. However even the fact that the patient was treated can be sensitive information, because the reference to the place where the medical data of the treatment was carried out may contain sensitive information. Such information can be sensitive, because the type of institution can reveal the type of medical intervention, or even the fact that the patient was treated must be treated as part of privacy. Therefore the patient identity has to be anonymised.

The anonymisation procedure should be irreversible: nobody should be able to tell the real identity of the patient by knowing the anonymised identifier. In addition to the anonymisation algorithm mentioned above, the irreversibility is supported by the provenance documentation integration method described in Section 6. The provenance documentation method supports irreversibility of the anonymisation by the way data storage is organized: the anonymisation service does not store the mapping from the real patient identifier to the anonymised patient identifier and computes the anonymised identifier each time it is needed using its own non-trivial algorithm. As a result, the real identifier and the anonymised identifier are not stored together anywhere in the system and the mapping from one identifier to the other cannot be found out without the algorithm of the anonymisation service.

8. Conclusions

In this chapter, we have discussed the important issues of making healthcare agent applications provenance-aware. Provenance-awareness enables users to trace how a particular result has been produced by identifying the individual and aggregated services that produced a particular output. This helps users to get an integrated view of the treatment process executed by distributed autonomous agents, and to be able to carry out audits of the system to assess that, for a given patient, the proper decisions were made and the proper procedures were followed. We discussed the special techniques needed in agent systems to make the autonomous and independent actors provenance aware and produce joint process documentation. We presented provenance awareness through the example of the OTMA agent system in the organ transplant management application domain. We detailed a method of documenting processes by weakly connected autonomous healthcare

agents and showed how this method helps to retain security and privacy of data within the process documentation produced by the agent-mediated healthcare system.

In summary, by transforming OTMA into a provenance-aware application, we augmented OTMA with a capability to produce at execution-time an explicit representation of the process actually taking place. Such representation can be then queried and analysed in order to extract valuable information to validate, e.g., the decisions taken in a given case, or to make an audit of the system over a period of time. Making the EHCR system provenance-aware provided a way to have a unified view of a patient's medical record with its provenance (i.e., to connect each part of the medical record with the processes in the real world that originated it and/or the individuals, teams or units responsible for each piece of data).

There are other approaches in literature which are related to provenance. In those first investigations which started to record the origin and history of a piece of data, the concept was called lineage. In the SDTS standard, lineage was a kind of audit trail that traced each step in sourcing, moving, and processing data, mainly related to a single data item, a logical data record, a subset of a database, or to an entire database [11]. There was also relationship to versioning [2] and data warehouses [15]. The provenance concept was later further explored within the GriPhyN project. The application of provenance in grid systems was extended in two respects: 1) data was not necessarily stored in databases and the operations used to derive data items might have been arbitrary computations; and 2) issues relating to the automated generation and scheduling of the computations required to instantiate data products were also addressed. The PROVENANCE project builds on these concepts to conceive and implement an industrial strength open provenance architecture.

To our knowledge, the application of provenance techniques to agent-mediated distributed healthcare applications is novel. In organ allocation management, there are few ICT solutions giving powerful support to the allocation of human organs which keep records of the distributed execution of processes. The EUROTRANSPLANT system is a centralized system where all information and decisions are made in a central server, and all activity is recorded in standard logging systems. The OTM system of Calisti et al. [7] is a distributed system (developed in collaboration with Swisstransplant) which combines agent technology and constraint satisfaction techniques for decision making support in organ transplant centers. In this case all activity is also recorded in standard logging systems.

Acknowledgment

This work has been funded mainly by the IST-2002-511085 PROVENANCE project. Javier Vázquez-Salceda's work has been also partially funded by the "Ramón y Cajal" program of the Spanish Ministry of Education and Science. All the authors would like to thank the PROVENANCE project partners for their inputs to this work.

More information

More information about the IST-2002-511085 EU PROVENANCE project can be found on the project website:

`http://twiki.gridprovenance.org/`

References

[1] Southampton provenance infrastructure. http://twiki.gridprovenance.org/bin/view/SotonProvenance/WebHome, 2007.

[2] G. Cobena A. Marian, S. Abiteboul and L. Mignet. Change-centric management of versions in an xml warehouse. In *Proc. 27th Int. Conf. of Very Large Data Bases, (VLDB 2001), P. M. G. Apers et al., eds.*, pages 581–590. Morgan Kaufmann, 2001.

[3] F. Bellifemine. *JADE*. CSELT, http://sharon.cselt.it/projects/jade/, 1999.

[4] CEN/TC251 WG I. *Health Informatics-Electronic Healthcare Record Communication- Part 1: Extended architecture and domain model, Final Draft prENV13606-1*, 1999.

[5] The Foundation for Intelligent Phisical Agents, http://www.fipa.org/. *FIPA Specifications*, 2000.

[6] P. Groth, S. Jiang, S. Miles, S. Munroe, V. Tan, S. Tsasakou, and L. Moreau. An architecture for provenance systems. Technical report, Electronics and Computer Science, University of Southampton, October 2006. Available at http://eprints.ecs.soton.ac.uk/12023/.

[7] S. Biellmann M. Calisti, P. Funk and T. Bugnon. A multi-agent system for organ transplant management. in Applications of Software Agent Technology in the Health Care Domain, 2003.

[8] S. Miles, P. Groth, M. Branco, and L. Moreau. The requirements of using provenance in e-science experiments. *Journal of Grid Computing*, 5:1–25, 2007.

[9] S. Munroe, S. Miles, L. Moreau, and J. Vázquez-Salceda. Prime: A software engineering methodology for developing provenance-aware applications. In *Proceedings of the Software Engineering and Middleware Workshop (SEM 2006)*, page 8 pages. ACM Digital, 2006. Published electronically by ACM Digital at http://portal.acm.org/toc.cfm?id=1210525.

[10] J.L. Nealon and ed. A. Moreno. *Applications of Software Agent Technology in the Health Care Domain*. Birkhäuser Verlag, 2003.

[11] S. Khanna P. Buneman and W.-C. Tan. Why and where: A characterization of data provenance. In *Proc. 8th Int. Conf. on Database Theory, (ICDT 2001), LNCS 1973, J. Van den Bussche, V. Vianu, eds.*, pages 316–331. Springer-Verlag, 2001.

[12] J. Vázquez-Salceda, U. Cortés, and J. Padget. Formalizing an electronic institution for the distribution of human tissues. *Artificial Intelligence in Medicine*, 23 (3):233–258, March 2003.

[13] J. Vázquez-Salceda, U. Cortés, J. Padget, A. López-Navidad, and F. Caballero. The organ allocation process: a natural extension of the carrel agent mediated electronic institution. *AI Communications*, 16 (3):153–165, 2003.

[14] S. Willmott, F. O. Fernández Peña, C. Merida Campos, I. Constantinescu, J. Dale, and D. Cabanillas. Adapting agent communication languages for semantic web service inter-communication. In *The 2005 IEEE/WIC/ACM International Con-ference on Web Intelligence (WI'05), Compigne, France, September 2005*, pages 405–408. IEEE Computer Society, 2005.

[15] J. Widom Y. Cui and J.L. Wiener. Tracing the lineage of view data in a warehousing environment. *ACM Transactions on Database Systems*, 25:179–227, 2000.

Javier Vázquez-Salceda
Universitat Politècnica de Catalunya
Campus Nord UPC, Edifici OMEGA
Jordi Girona 1-3
08034, Barcelona
Spain
e-mail: jvazquez@lsi.upc.edu

Sergio Álvarez
Universitat Politècnica de Catalunya
Campus Nord UPC, Edifici OMEGA
Jordi Girona 1-3
08034, Barcelona
Spain
e-mail: salvarez@lsi.upc.edu

Tamás Kifor
Computer and Automation Research Institute
Kende u. 13-17
1111 Budapest
Hungary
e-mail: tamas.kifor@sztaki.hu

László Z. Varga
Computer and Automation Research Institute
Kende u. 13-17
1111 Budapest
Hungary
e-mail: laszlo.varga@sztaki.hu

Simon Miles
Department of Computer Science
King's College London Strand
London WC2R 2LS
United Kingdom
e-mail: simon.miles@kcl.ac.uk

Luc Moreau
School of Electronics and Computer Science
University of Southampton
Southampton SO17 1BJ
United Kingdom
e-mail: L.Moreau@ecs.soton.ac.uk

Steven Willmott
Universitat Politècnica de Catalunya
Campus Nord UPC, Edifici OMEGA
Jordi Girona 1-3
08034, Barcelona
Spain
e-mail: steve@lsi.upc.edu