# The MOSAIC System

## A Clinical Data Exchange System with Multilateral Agreement Support

Magí Lluch-Ariet[1,2,⋆,⋆⋆] and Josep Pegueroles-Valles[2,⋆⋆]

[1] MicroArt
mlluch@microart.cat
[2] Technical University of Catalonia (UPC)
josep.pegueroles@upc.edu

**Abstract.** As more and more data become available, the task of accessing and exploiting the large number of distributed clinical data repositories becomes increasingly complex. Moreover, accessing to a certain data set in a federated data warehouse may have constrains, and multilateral agreements may solve it. Such agreements may be very complex to be solved manually.

Current systems for clinical data sharing do not support multilateral agreements. MOSAIC, intends to give a modular and efficient solution to the clinical data exchange problem with multilateral agreements. The proposed system takes advantage of agent based systems and the current standarised Interaction Protocols together with the current protocols for clinical data transfer.

## 1 Introduction

When a clinician looks for similar cases to compare the data collected from his/her patients, he/she needs to select the clinical centres where to look at, negotiate the access rights to the data, and occasionally interchange some local data with them. To find the correct clinical centre and achieve a good deal is a complex process that can be partially automated. The main aim of the data exchange system presented here is to get the most from the cases available at a certain node, and to get access rights to the data needed from other nodes of the network, when possible.

When bilateral agreements for data interchange do not solve the data access, an automatic process that finds possible multilateral agreements is needed. This process has an associated protocol with the following functionalities:

i) Publish the reference of the data available from a node, ii) Send the request for data access, iii) Send the authorisation and the data access rights, iv) Perform the data transfer, v) Acknowledge the reception of the data, vi) Withdraw the access rights if the agreement is not completely fulfilled, and vii) Start, Commit or Rollback the whole transaction.

⋆ MicroArt / Parc Científic de Barcelona (PCB) C/ Baldiri Reixac, 4 / 8028 Barcelona.
⋆⋆ Departament d'Enginyeria Telemàtica (ENTEL) / Universitat Politècnica de Catalunya (UPC) / Edifici C3 - Campus Nord / C/ Jordi Girona, 1-3 / 08034 Barcelona.

M. Szomszor and P. Kostkova (Eds.): E-Health 2010, LNICST 69, pp. 275–284, 2011.

The work presented in this paper fulfills these requirements and facilitates the achievement of multilateral agreements involving the data sharing among a number of nodes of a network.

The rest of the paper is structured as follows: In section 2, we provide an overview of the state-of-the art in federated clinical datawarehouses and systems for clinical data sharing; in section 3, we describe the architecture of MOSAIC with its main components and the Protocol Architecture of the System; in section 4 the Protocol Design is introduced, explaining how the actors of the protocol interact among them; and in section 5, we conclude the paper with the main challenges that MOSAIC faces and future work for the deployment of the proposed system is analysed.

## 2 The State of the Art in Distributed and Federated Systems in e-Health

Since the availability of electronic data repositories in Clinical Centres, many projects and iniciatives from diferent disciplines have contributed to facilitate the interchange and data sharing among the centres: i) standards related to how the information has to be stored, namely DICOM [3] and the Electronic Health Record [8]; ii) standards related to the interoperability and the transfer of the clinical data, like ISO/HL7 21731 [7] and ISO 13606 [6]; iii) protocols to grant the basic principles of security and privacy, like SSL; and iv) new areas of research that contribute to build distributed systems like The Agent Technology, and the Semantic Web that facilitates the understanding of heterogenous data bases with the use of ontologies. All of them converge to make possible systems and projects like the following.

Cancer Biomedical Informatics Grid (caBIG) [9] is one of the biggest initiatives that provides an infrastructure to build distributed databases, to share data and knowledge, including the Common Security Module (CSM), a comprehensive set of security tools. "The Cancer Genome Atlas" (TCGA) [12] is one of the examples where a distributed database is built using caBIG for its development. However, this infrastructure does not provide specific capabilities for multilateral data exchange.

The Artemis Project [2] has previously dealt in a medical context with the transfer of critical patient data (mainly clinical data). It used a peer-to-peer-like architecture and used web services security protocols for the transfer of patient records. Like in caBIG, multilateral agreements for data exchange are not supported in the Artemis project.

HOPE [4] is a collaborative telemedicine platform for clinical data sharing. It implements a grid infrastructure for a distributed database, providing a common interface independent to the data base of its nodes, to manage data from the patients, accessing to clinical records and images (in DICOM) from PACS. Instead of proposing the exchange of data, this project facilitates the Data Retrieval from an integrated interface.

An example of a federated Datawarehouse and its associated Decision Support System is the HealthAgents project [10,13], that aims to build a worldwide network of clinical centres for the brain tumour diagnosis. This project is focused on collecting data for building classifiers that will be used during the diagnose process, but does not address the exchange of data between the nodes. Agent frameworks for medical purposes have been extensively documented in the literature [1,11], but their use has been traditionally confined to patient control and information [5].

All these systems implement and provide either a good framework for distributed clinical databases where to store the data in a virtual repository or federated databases where the retrieval of the data among the nodes is performed after a direct query to the system. Nevertheless, at the moment, there is not any system that performs an automatic agent-based negotiation for clinical data exchange in a federated datawarehouse.

## 3 Architecture

### 3.1 The Architecture of the System

The MOSAIC system is composed by a set of interconnected nodes each one with its associated Data Mart containing local medical data. The purpose of MOSAIC is the interchange of this data among the MOSAIC nodes. To support this architecture, each node of the network is composed by the following components: i) Web Server (e.g., Tomcat server); ii) DBMS (e.g., MySQL); and iii) FIPA compliant Agent Platform (e.g., JADE).

The nodes of this network may have two different roles: One as "data contributors" (named Contributor Agents) and the other as "data petitioners" (named Petitioner Agents). The communication protocol of the MOSAIC system allows the message and information interchange needed during the negotiation process among the nodes aiming to achieve agreements for the data interchange. The "Yellow Pages Agents" responsible to maintain the information of the network topology are also needed. (Figure 1).

– **The Petitioner Agent.** This Agent is responsible to identify which are the nodes that may contain the requested data, negotiate with them the access rights, try to solve the conditions and constrains (if any) and finally, collect the data, if possible.

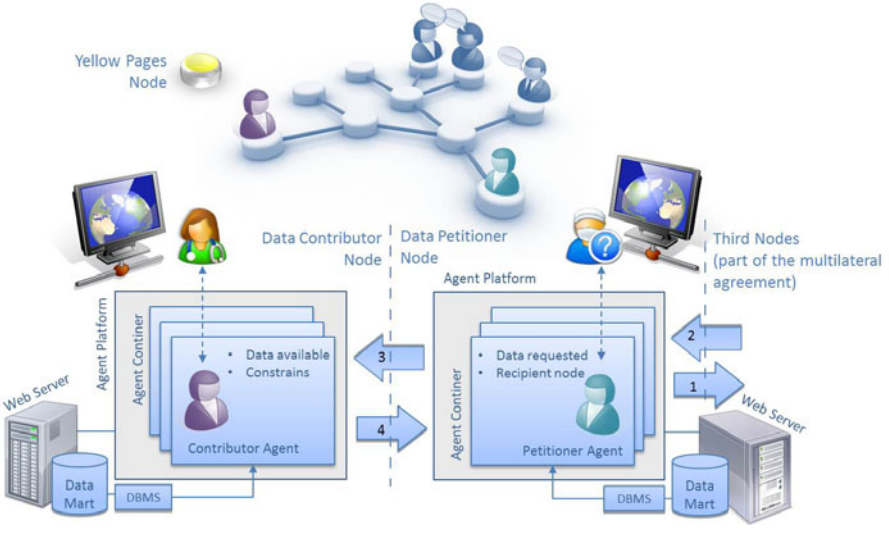


**Fig. 1.** The MOSAIC Architecture showing the data flow between the Agents: Data delivery to third nodes (1); Data collection from third nodes needed to complete the constrains (2); Data transfer to the node where the desired data is hosted (3); and Data transfer to the requesting node (4)

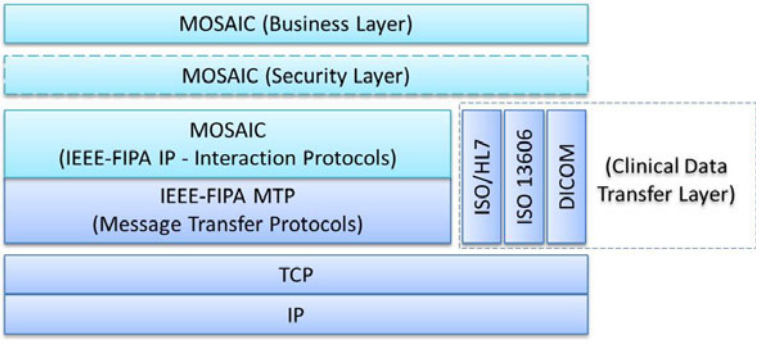


**Fig. 2.** The Stack of the main protocols where MOSAIC sits

- **The Contributor Agent.** This Agent will negotiate (if necessary) the conditions to provide access to the data with the Petitioner Agent, and it will deliver the data when the conditions are fulfilled.
- **The Yellow Pages Agent.** This Agent is to provide the "directory service" to the rest of the Agents in the system. Its main purpose is to inform the Petitioner Agents with the list of active Nodes and their Contributor Agents to whom they may try to request the access to their Data Marts.

### 3.2 The Architecture of the Protocol

MOSAIC aims to follow the current standards for Multi-Agent Systems, and more precisely, those defined by IEEE-FIPA. Beyond the basic features of simply message transfer formalised by the Agent Communication Language (ACL), there is a set of protocols that MOSAIC may take advantage of. Among them, the IEEE-FIPA Interaction Protocols allow to implement part of the dialogs between the Petitioner and Contributor Agents. Additionally, MOSAIC also aims to apply and take advantage of the use of those standards related to the Clinical Data transfer, namely ISO/HL7 21731, ISO 13606 and DICOM protocols.

Figure 2, shows the place where the MOSAIC protocol sits within the current stack protocols of the Agent systems and clinical data transfer.

## 4 The Protocol Design

This section describes the behavior of the Agents responsible to provide data to the network (Contributor Agents: CA) and those requesting data from the network (Petitioner Agents: PA) in order to build multilateral agreements among them and achieve the final goal of the requesting nodes.

### 4.1 Requesting and Collecting Data: The Petitioner Agent

After receiving the request from the user with the details of the data set to be collected from the network, the Multicast Petitioner Agent (PA) consults the Yellow Pages which

are the active nodes of the network with Contributor Agents (CA) running, and sends to all of them a message with the details of the data set that aims to collect and the authorization access request to those that have data that may be part of that data set. Then, the Agent waits for receiving messages from the CAs indicating the existence of the requested data that may be accessed either with or without conditions.

When a set of options are found a set of new Agents (one Agent per option) will be launched in parallel and autonomously will explore each of the possible options for data collection. Each of these Unicast Petitioner Agents (UPA) will ask which are the conditions for accessing the data, if any. For the nodes that do not have any constrain and offer their data openly, the process for collecting the data available will be launched directly. In case there are some conditions to be accepted prior the delivery of the data, the PA asks to the user whether those conditions are accepted or not and sends the user answer to the node which may accept or reject the answer. If the conditions are fulfilled and accepted, the PA will also proceed with the data collection. Finally, it is sensible to expect that a possible condition for accessing to a certain data set of a node is to deliver another data set that the node might be interested to have, and two situations may arise:

- **Bilateral Data interchange.** CA's condition for allowing access to its data set is to receive another data set available at the Data Mart of the PA. This agreement depends only to the bilateral data interchange between the Contributor and the Petitioner Nodes.
- **Multilateral Data interchange.** CA's condition for allowing access to its data set is to receive another data set, not available at the Data Mart of the PA. This forces the PA to look for the requested data set in other nodes of the network. The access to the data set of the CA depends on multilateral agreements among a set of nodes in the network. A "time to live" parameter is proposed to allow the user of the protocol to define a limit at the number of nodes that may be involved in a multilateral agreement, avoiding unmanageable network explorations.

A PA activated by another Agent in order to solve a constrain has to take into account that the final recipient of the requested data set would not be the node from where the Agent is launched and this has to be notified to the possible contributing nodes. Moreover, the data access request does not have to be addressed to all the active Contributor Agents as it has to exclude the recipient node of the requested data.

When the access to the data cannot be solved, the PA will wait a certain time to allow a reconfiguration of the network and its content. After this time, the PA will start a new request. This may happen a number of times until a counter that puts a limit to this interaction expires.

This behavior and the life cycle of the Agent is described with the State Diagram shown in figure 3 and explained below.

- **P1: Start Request.** The Petitioner Agent receives the request from the user with the details of the data set to be collected from the network. The PA consults the Yellow Pages which are the active nodes of the network with Contributor Agents running and sends to all of them a message with the details of the data set that aims to collect and the authorization access request to those that have data that may be part of that data set. After that request, the Agent jumps to state P2.

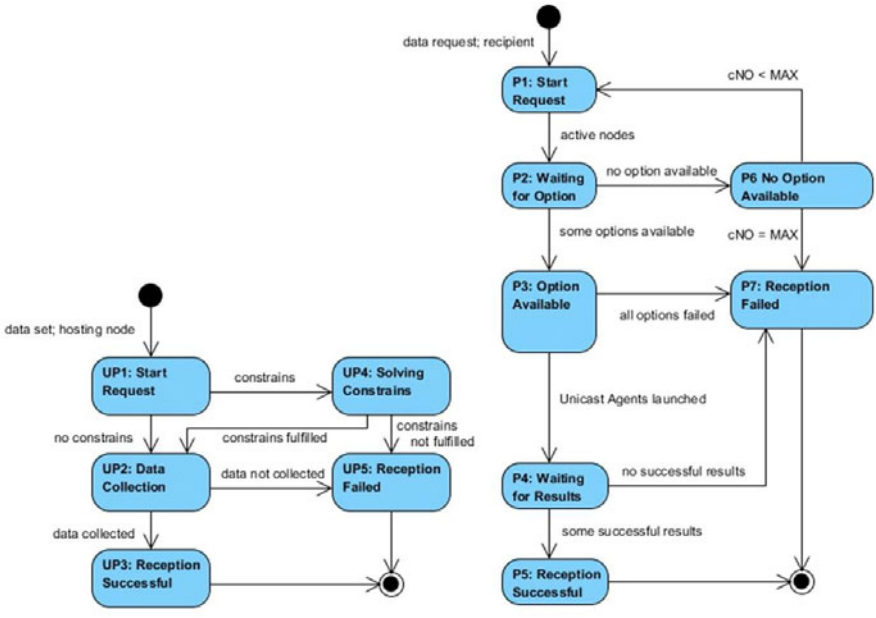


**Fig. 3.** State Diagrams of the Unicast (left) and Multicast (right) Petitioner Agents

– **P2: Waiting for Option.** This is the second state of the PA. Here the Agent waits for receiving messages from Contributing Agents indicating the existence of the requested data that may be accessed either with or without conditions.

  A timer (tWO) indicates the time frame while the Agent will be waiting in this state. After expiring the time indicated the Agent will jump to state P3 (Option Available) if there is at less one answer from some Contributor Agents or to P6 (No Option Available) if there is not any answer from any Contributor Agent.
– **P3: Option Available.** When a set of options are found a set of new Agents (one Agent per option) will be launched in parallel and autonomously will explore each of the possible options for data collection. These new Agents will be named "Unicast PAs" (UPA) and will start with direct request to explore the possible data access to a specific node.

  After launching the set of UPAs the Multicast PA will jump to the state "Waiting for Results" (P4).
– **P4: Waiting for results.** This is the state where the PA waits for the results of the new set of UPAs launched in the previous state.

  A new timer tWR will determine the time limit while the Agent will be waiting for the results of the UPAs. When the tWR expires or all the results from the new unicast Agents are received, the Agent will examine the results for its final action. In case there is not any successful result from any Unicast PA, the main Agent will jump to state "Reception Failed" (P7). Otherwise, the Agent will jump to state Reception Successful (P5).

- **P5: Reception Successful.** This is the final state of the Agent for successful data requests. In this state the PA notifies the user about the success of the request and ends.
- **P6: No Option Available.** In this state the Agent waits a certain time to allow a reconfiguration of the network and its content. This time is defined by timer tNOA. After this time, the Agent will jump back to state P1 "start request". This may happen a number of times until a counter (cNOA) that puts a limit to this interaction does not arrive to zero. cNOA counter is initialised with a value that marks the maximum number of interactions and is decreased with a unit each time the Agent falls in this state.
- **P7: Reception Failed.** This is the final state when there was not any node with the data requested or with conditions that made possible to achieve an agreement for allowing the access to the data. The PA fails in its goal to access to the data and ends.

The Unicast PA (UPA) is automatically activated by a PA in order to manage an active option for accessing to a specific data-set. It receives the request for accessing to the data set, with the reference of the node where it is hosted, plus the reference of the final recipient of the data. It will negotiate with the CA from that node the access rights to that data set and eventually collect the requested data if the constrains are fulfilled.

- **UP1: Start Request.** This is the first state of the UPA where it receives the reference of the data-set to be collected and the reference of the node where it is hosted, the reference of the Contributor Agent with which the negotiations will start and the reference of the final recipient of the requested data (which may be either the requesting node, or a third node if the request is only to solve a constrain for accessing to another data set). If there are no constrains to fulfill, the Agent will move to state "Data Collection" (UP2). Otherwise, the Agent will move to state "Solving Constrains" (UP4).
- **UP2: Data Collection.** In this state, all the conditions, if any, have been already fulfilled and the Unicast PA is authorised to collect the data set. The transfer of the data takes place here and if it concludes successfully the Agent moves to the "Reception Successful" state (SP3). If some problem occurs and the data transfer can not conclude, the Agent moves to the "Reception failed" state (UP5).
- **UP3: Reception successful.** This is the final state of the Unicast PA when the data collection concludes successfully. A message notifying the achievement of the data request to the Multicast PA is sent and the Agent ends.
- **UP4: Solving Constrains.** This is the state where the Agent comes when there are constrains to fulfill. In case there is a condition to provide another data-set, the Agent will check whether it can be obtained from its Data Mart. If so, an authorisation for delivering it to the Contributor Agent will be requested to the user. If the data can not be obtained locally, a new Multicast PA will be launched. A Multicast PA must know whether its activation corresponds to a need to solve a constrain or not. If so, it must avoid sending a request to a possible Multicast Contributing Agent active at the node of the final recipient of the data-set.

  When the data set is available the data delivery will take place launching a UCA. Answers from the users and if needed from the new Multicast PA and / or from the

UCA, are managed in this state. In case all the constrains are fulfilled, the Agent will move to "Data collection" state (UP2). If some constrain can not be solved, the Agent will move to "Reception failed" state (UP5).

– **UP5: Reception failed.** This is the final state of the Unicast PA when for some reason the data collection could not conclude successfully. A message notifying the failure of the data request to the Multicast PA is sent and the Agent ends.

### 4.2 Contributing and Delivering Data: The Contributor Agent

The user responsible of the DataMart in a certain node launches the Multicast Contributor Agent (CA) when he or she wants to put a Dataset available at the network. The user also indicates to the CA which are the constrains to be fulfilled before allowing the access to the data. After its activation, the CA publishes to the Yellow Pages its existence and waits to receive requests from the active PAs. The State Diagrams of the Contributor Agents are shown in figure 4

– **C1: Start Contribution.** This is the first state of the Contributor Agent. It receives as input from the user the reference of the Dataset available and the constrains to be fulfilled before allowing the access to the data. It publishes to the Yellow Pages its existence and jumps to the next state "Waiting for Request" (C2).
– **C2: Waiting for Request.** As its name indicates, the Contributor Agent stays in this state attending possible Data access requests from the network. When a new request arrives the Agent launches a new Agent to process the specific request with the goal of validating the possible data access constrains and delivering the data if the constrains are fulfilled. This new Agent will be named "Unicast Contributor Agent" (UCA).

  The Contributor Agent receives the results from each UCA and notifies the user periodically with those results.

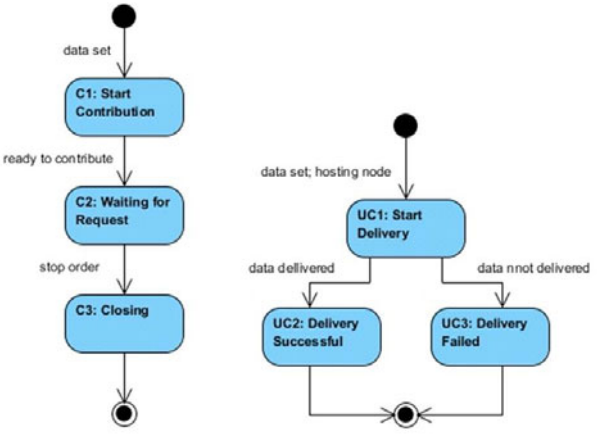


**Fig. 4.** State Diagrams of the Multicast (left) and Unicast (right) Contributor Agents

A stop request from the user implies to jump to the final state "Closing Contributor Agent" (C3). However, before stoping the Agent, it will wait for some time to allow the possible active UCAs to conclude. After expiring that timer, in case there is still some active UCAs, they will be forced to stop before leaving this state.

- **C3: Closing Contributor Agent.** This final state of the Contributor Agent is to notify the Yellow Pages about its end.
- **UC1: Start Delivery.** This is the first state of the UCA. It receives the reference of the data-set to be delivered and the reference of the final recipient node where the data will be transfered, including also the reference of the Contributor Agent to which the negotiations will start. If the delivery concludes correctly, the Agent will move to the final state "Delivery successful" (UC2). Otherwise, if the delivery of the data set can not be concluded for some reason, the Agent will move to the final state "Delivery failed" (UC3).
- **UC2: Delivery successful.** This is the final state of the UCA when the data delivery concluded successfully. A message notifying the achievement of the delivery to the UPA is sent and the Agent ends.
- **UC3: Delivery failed.** This is the final state of the UCA when for some reason the data delivery could not conclude successfully. A message notifying the failure of the data delivery to the UPA is sent and the Agent ends.

## 5 Conclusions

The level of deployment of standards and protocols for clinical data transfer, the availability of hundreds of thousands repositories of clinical data worldwide distributed, and the clinical need of sharing knowledge among the clinicians in order to provide a better and faster diagnose to their patients, inspires the development of the MOSAIC system.

The conditions for accessing to a certain set of data from the network may likely include some data interchange request. This might be solved with bilateral agreements when the data set marked as condition for accessing to the new data is at the node of the PA. However, when this data is not at the local Data Mart, it needs to be found in third nodes and multilateral agreements are required.

It has been shown that Multi Agent Systems are a good framework to build a system for the clinical data sharing in a federated datawarehouse. Finally, the availability of the IEEE-FIPA Interaction Protocols allows to propose a negotiation protocol at application level to cover this needs. Currently we are working in the implementation of the protocol using IEEE-FIPA Interaction Protocols. Future work will address security services for the protocol and optimization of the multilateral agreements.

**Acknowledgments.** We would like to thanks our colleagues from MicroArt, the HealthAgents consortium, and the Telematics Engineering department at the UPC, for the inspiring discussions we had with them regarding the content of this paper. This research has been partially funded by the P2PSEC project (TEC2008-06663-C03-01).

## References


1. Annicchiarico, R., Cortés, U., Urdiales, C.: Agent Technology and e-Health, 1st edn. Whitestein Series in Software Agent Technologies and Autonomic Computing. Birkhuser, Basel (2008)
2. Boniface, M.J., Leonard, T.A., Surridge, M., Taylor, S.J., Finlay, L., McCorry, D.: Accessing patient records in virtual healthcare organisations. In: eChallenges 2005 (2005)
3. DICOM. Digital imaging and communication in medicine (1993)
4. HOPE. Hospital platform for e-health, `http://sourceforge.net/projects/telemed/` (last accessed February 11, 2009)
5. Huang, J., Jennings, N.R., Fox, J.: An agent-based approach to health care management. Int. Journal of Applied Artificial Intelligence 9(4), 401–420 (1995)
6. ISO 13606. Electronic health record communication – part 1: Reference model (2008)
7. ISO/HL7 27931. HL7 version 3 - Reference information model (2006)
8. ISO/TR 20514. Electronic health record – definition, scope and context (2005)
9. Komatsoulis, G.A., Warzela, D.B., Hartela, F.W., Shanbhaga, K., Chilukuric, R., Fragosoa, G., de Coronadoa, S., Reevesa, D.M., Hadfielda, J.B., Ludetb, C., Covitza, P.A.: cacore version 3: Implementation of a model driven, service-oriented architecture for semantic interoperability. Journal of Biomedical Informatics 41, 106–123 (2008) (article in press), doi:10.1016/j.jbi.2007.03.009
10. Lluch-Ariet, M., Estanyol, F., Mier, M., Delgado, C., González-Vélez, H., Dalmas, T., Robles, M., Sáez, C., Vicente, J., Huffel, S.V., Luts, J., Arús, C., Silveira, A.P.C., Julià-Sapé, M., Peet, A., Gibb, A., Sun, Y., Celda, B., Bisbal, M.C.M., Valsecchi, G., Dupplaw, D., Hu, B., Lewis, P.: On the Implementation of HealthAgents: Agent-Based Brain Tumour Diagnosis, 1st edn. Whitestein Series in Software Agent Technologies and Autonomic Computing. Birkhauser, Basel (2008)
11. Merelli, E., Armano, G., Cannata, N., Corradini, F., D'Inverno, M., Doms, A., Lord, P., Martin, A., Milanesi, L., Moller, S., Schroeder, M., Luck, M.: Agents in bioinformatics, computational and systems biology. Brief Bioinform. 8(1), 45–59 (2007)
12. National Cancer Institute and National Human Genome Research Institute. The cancer genome atlas (2005-2010), `http://cancergenome.nih.gov/` (last accessed May 12, 2010)
13. The HealthAgents Consortium. The healthagents project (2006-2008), `http://www.healthagents.cat/` (last accessed May 12, 2010)