An Agent-based Federated Information System for Telecare Environments

Víctor Guevara-Masís, Hamideh Afsarmanesh, Louis O. Hertzberger
Faculty of Science,
University of Amsterdam
1098 SJ, Amsterdam, The Netherlands
Email: {vguevara, hamideh, bob}@science.uva.nl

***Abstract*— In modern networked telecare environments, information sources are typically distributed among several geographical sites and their availability varies over the time. Moreover, care related information is typically located on heterogeneous database management systems, or yet stored in complex legacy systems. The introduction of multi-agent systems in telecare has proved to be highly effective for information exchange. Although they can provide a rich mechanism for information brokerage, they still do not tackle the issue of retrieving data from the heterogeneous sources. In this paper, we describe an approach that combines aspects of federated databases and with those of agent systems in order to access and retrieve information homogenously. This paper explores the architecture of an agent-based information management system, developed under the context of TeleCARE project, with a focal point on 'intelligent' strategies for data assembly during federated query processing.**

I. Introduction

Most of the literature on remote care is focused on "traditional" forms of cooperation among care related enterprises. After centuries of delivering care in hospitals and care centers, the paradigm is however shifting towards treating patients at the point of need. The relationship between the patients and care provider is evolving from encounter-oriented care, to a broader, continuous and ongoing distance care services.

Although technology definitely cannot provide the solution to all care problems of the telecaring, it plays a fundamental role in the application of the new remote and integrated care paradigm [1], [2]. Progress in computer networks and ubiquitous computing, for instance, suggest new opportunities for more care services, which can apply computer remotely controlled hardware, devices, and other customizable and adaptable software systems. Examples of such advanced care services include comprehensive status monitoring, proactive alarm systems, and context-aware reminding systems that lead to significant benefits in the care industry.

Developing the advanced care services raises new challenges in terms of modeling, infrastructures, support tools, and even the very understanding of telcaring networks and its operating rules. The design and development of an affordable infrastructure for telecare applications becomes a crucial requirement if effective deployment of telecare applications take place in a collaborative network environment.

From information management point of view, the architecture of the telecare network is similar to a distributed database environment, where the data is spread over several nodes and there are local schemas that covers the entire shared data sets. Due to significant autonomy characteristics of the telecare environment, however, it requires a different information sharing framework such a federated information system [3]. Unlike other networks, in a telecare network

- every node is autonomously configured to establish which part of its data is shareable with others,
- the critical data that is handled about individuals, and
- there is data model heterogeneity among nodes, in the sense that nodes consist of different data sources (e.g. a large variety of hardware and software resources supporting the care environment).

This paper aims at the design of an agent-based infrastructure for the information management that can support properly remote applications in a federated environment, all within the TeleCARE project [4] and Virtual Laboratory for e-Science project. The IST 5FP TeleCARE project designs and development of a configurable framework solution that focuses on tele-supervision and tele-assistance for elderly support. The project is funded in part by the IST program of the European Commission and as complementary to other initiatives for the integration of elderly in the society, while reducing their isolation. The TeleCARE project researches on an extendable infrastructure so different elderly care developments may be easily plugged, making possible the interoperability in this sector. The Dutch VL-e project (Virtual Laboratory for e-Science) is a multi-disciplinary virtual laboratory environment for remote experiment control and Grid-based distributed analysis in experimental sciences [15]. The aim of this project is to bridge the gap between the technology push of the high performance networking and the application pull of a wide range of experimental applications.

II. Multi-agent systems

The research of Multi-Agent Systems (MAS) is part of the distributed artificial intelligence. It aims to define and establish mechanisms that allow autonomous pieces of software components to structure and share their capabilities and knowledge, in order to reach their individual and common goals. A MAS can be defined as "*a loosely-coupled network of problem solvers that work together to solve problems that are beyond their individual capabilities*" [5].

Although there is no general agreement on the definition of the term *agent*, since there is an ongoing debate on it, many researchers generally consent that for a software program to be called an agent, must exists in a dynamic and partially unpredictable environment and embody the notion of *autonomy*, among other characteristics [6]. A bare definition of an agent is to consider it as a software component, which can *sense* the environment in which exists, and proactive (or reactive) acts upon it to achieve some goals. This aspect of acting (or reacting) upon sensing the environment is considered as the agent being able to *decide for itself* to satisfy its objectives.

A. Mobile agents

The agent paradigm supports **mobility** at different levels. *User mobility* refers to specific techniques to allow relocation of user within a network [7]. *Device mobility* relates to the action of placing a device into different platforms and, and since agents enclose the functionality of a device, they are appropriate for this task. *Mobility of code* is either characterized by *weak mobility* or *strong mobility*. Weak mobility indicate that an agent is deployed on another machine with an initial state. A Java applet or downloading a new version of software can be both considered as examples of weak mobility. Strong mobility, on the other hand, refers to the complete migration (including the execution state) of an agent into another environment.

Agents whose predominant characteristic is strong mobility are generally classified as *mobile agents*. Mobile agents are then pieces of computer software that are able to start a computation on a site, suspend the execution of the computation, arbitrarily at some point, autonomously migrate from one computer to another, in a heterogeneous network, and resume and continue its execution on the destination computer [8]. Mobile agents are stand-alone, executable software, which are being sent to different computers (servers) to carry out specific computational tasks (See Fig. 1). Software mobility brings robustness, performance, scalability or expressiveness to distributed applications in a computer network.

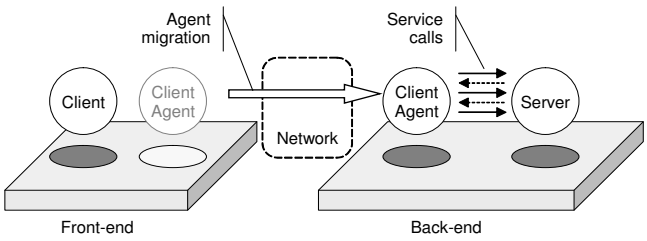

Fig. 1. Mobile agent migration

III. Federated information system

The concept of federated information system is intrinsically related to the concept of *federated database system*, which is defined as "*a collection of cooperating database systems that are autonomous and possibly heterogeneous*" [3]. The concept however has been employed for several different situations and even extended to a broader context and to include additional information systems. Basically, a federated information system refers to a set of heterogeneous, distributed and autonomous components. Besides, each participant have the ability to associate or disassociate itself from the federation or, even, to participate in other federations.

A federated information system comprises independent information sources, which are, up to a different extent, physically *distributed*; *heterogeneous*, regarding technical aspects, such as hardware, software, structure, environment, data model, communications and semantic issues; and *autonomous*, i.e. operationally independent. A basic federated information system can be seen as a client server structure with a three-tier architecture, as shown in Fig. 2.

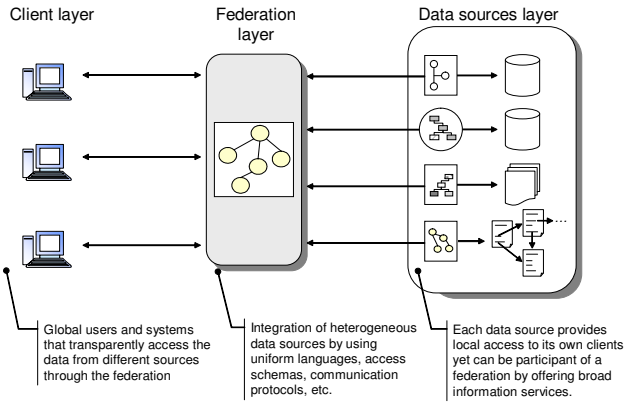

Fig. 2. Federated information system as three-tier architecture

The client layer consists of users and global applications, which take advantage of the federation layer. The federation layer is composed of software components that access transparently the federated data sources, and provides an uniform view of the information to all users. This uniform view involves an uniform access language, an uniform access schema, an uniform metadata set, an uniform communication protocol, etc. The data source layer is integrated into the infrastructure by *mediators*, or by wrappers, to resolve technical difficulties. The federated architecture can be extended with other layers, for instance, for security concerns, presentation, etc.

A. Federated query paradigm

The federated query paradigm performs the query and process data from the federated information sources. A federated query processing is designed to preserve the local autonomy and heterogeneity of the database systems [9]. Since the query processing cannot force a system to accept execution plans, the processing strategy requires some appropriate algorithms to perform the query, such as:

1) Query planning: Activity of establishing a suitable plan of executing remote queries for a given query against the federated schema.

2) Plan execution: Tasks related to executing the query plan, including: (i) optimization activities to decide which

query operations are performed, (ii) carry out the execution of subqueries, (iv) the gathering of partial results, and (v) post processing and data joins, if allowed.

3) Result integration: Activity where the gathered data is homogenized by removing redundancy, identifying identical results, and resolving inconsistent data values.

IV. Agent-based federated information system

Multi-Agent Systems, in particular, those with mobile agents, provide an innovative and rich paradigm for information brokerage in a network environment. In fact, an information system based on agent paradigm may be an alternative to solve difficulties found in traditional information systems [10]. Besides, other advances in agent paradigms (negotiation, reasoning, etc. for example) can further support the growth of electronic commerce and, in particular, together with service oriented architectures, they are likely to have a considerable effect on electronic commerce and delivery of care.

The information management system based on agents, generally speaking, acts as a flexible data mediator from federated data sources [12]. First of all, the agent-based components are designed to support applications that may require a variety of data models, access physically distributed and heterogeneous data, and possess a large amount of users and agents who retrieve data with pre-defined visibility rights. The agents act as mediators in charge of performing the information management activities, with regard of the database repository, and are also responsible over the federated query processing. They are conceived to effectively support the cooperation among sites within a telecare network.

Although the use of mobile agents to retrieve information is relatively novel, various approaches have already considered them for accessing remote databases. For instance, a DBMS-applet that creates and dispatches a mobile agent (or agents if necessary) to the remote database server is proposed in [11]. At the database server site, the mobile agent initiates the JDBC driver, connects to the database, and performs the query specified by the requester client. When the mobile agent completes its query task, it dispatches itself directly back to the client machine, where the DBMS-applet is located.

Our suggestion differs from the above approach in the sense that stationary and mobile agents are in use. Stationary agents provide a local foundation for information retrieval, result processing, data visibility levels, and access rights verification. Mobile agents are in charge of transporting the subqueries and their logic to the remote sites; they do not carry any code overhead related to database connectivity, making them more suitable for migration. Another difference is the fact that our mobile agents do not need to return back to the point of origin, because they only require to transmit back the results.

A. Federated Information MAnagement (FIMA)

The *Federated Information MAnagement* (FIMA) is designed to benefit from the agent characteristics, distributed programming, and federated information systems in order to overcome some of the deficiencies of traditional data sources, within the telecare domain.

The goal of FIMA is to support all data handling and data persistency within a telecare node. Using an approach for object persistency, FIMA it provides a transparent object-oriented interface for flexible processing of federated queries, data handling, schema management, and preservation information privacy through access rights management. Internally, FIMA has mechanisms to directly store and query objects through a standard object-oriented language. FIMA can be further extended to support other less structure object representations, such as XML documents. The provision of transparent storage and manipulation functionalities enables the "sharing of data" required by other internal components of the MAS platform as well as the telecare services.

Fig. 3 shows a high level overview of FIMA and its basic relationship with other components and elements within a telecare node. These components, depending on their role and functionality, are implemented as stationary or mobile agents. With the agent paradigm, every agent (including mobile ones) has its own thread control, is event-driven, and communicates through a proper Agent Communication Language (ACL). A closer look into FIMA identifies these underlying mechanisms.

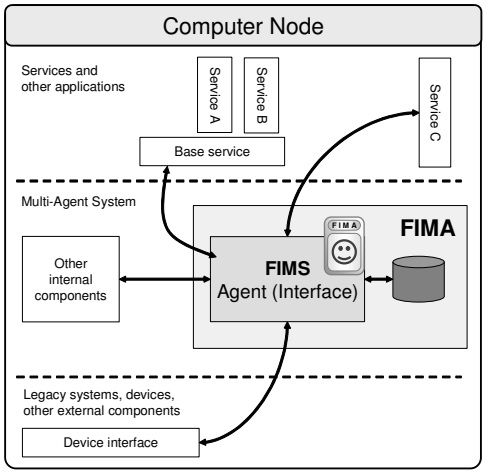

Fig. 3. Relationship between FIMA and other node elements

1) Computer Node: Computer system where FIMA resides. It provides the environment for running all FIMA subcomponents and protection against malicious software.

2) FIMS Agent: Representative of the FIMA in the node. It is a stationary agent that encapsulates and protects all database operations from direct access. The interface provides transparency for the repository by hiding database details, such as the connection settings.

3) Clients: Clients are autonomous agents and users. They represent services, telecare applications, or other components of the node. Clients do not interact with the database directly, rather, they contact FIMS Agent instead.

4) Mobile Agent: Mobile component that is pushed from one node to another for execution of queries.

5) Passport credentials: Certified document containing agent's abilities and qualifications. They are bound to each agent and attest the globally unique identifier of the agent within the whole telecare network. It is usually conceived as an electronic passport.

The federated approach requires a suitable and dynamic mechanism to cope with the ever evolving federated schema. Within TeleCARE, the Dynamic Ontology-based data Structure Generator (DOSG) allows telecare application developers to transform an abstract data model into a concrete database schema. This schema is then used by FIMA. In fact, DOSG enables the federated modeling of information while FIMA enables the management of the data (i.e. storage, retrieval, querying, etc., of data instances). The description of DOSG has been previously reported in [13].

Now that we have introduced the all the basic elements and relations of FIMA, we can proceed to illustrate how the agent system collaborate to perform the query processing.

V. Agent-based query processing

The Federated Query Processing (FQP), supported by FIMA, provides secure access to information distributed over heterogeneous and autonomous nodes within the telecare network. The main goal of the FQP, in FIMA, is to enable authorized clients to query the information, hiding the concrete details about connections, agent creation, migration to nodes, and manipulation of the data.

FQP is a quite complex task, briefly detailed as follows. First, the requester sends a query (which is in high-level format) to the FIMS Agent. FIMS Agent generates another stationary agent designed to handle locally this particular request. The query is then translated into the concrete internal structures of the stored data and a set of sub-queries is then created. These sub-queries are one by one assigned to mobile agents with a proper itinerary. After this step, these mobile agents are dispatched to the remote nodes to accomplish their mission, to perform the local query at remote sites, and to transmit the results back to the original node. Finally, the gathered results are merged at the orignial node, and sent back to the requester, as shown in Fig. 4.

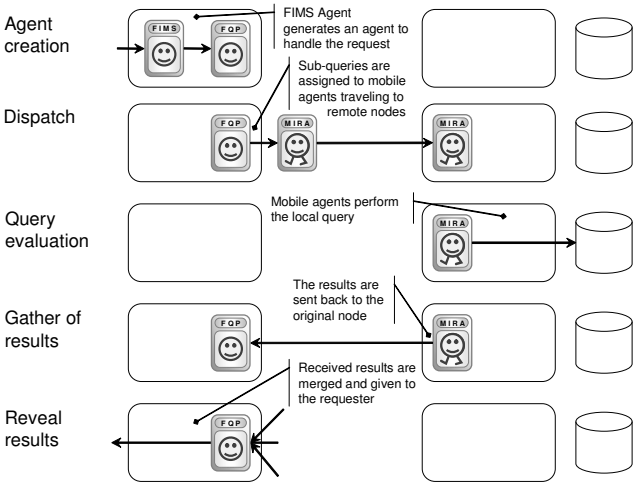

Fig. 4. Federated Query Processing

A. FQP components

FIMA makes usage of both mobile and stationary agents in order to perform the data retrieval. The mobile agents are used as the mechanism to perform the sub-query and for result transmission between the client node (where FIMA receives the request) and the other remote node(s) where information may be located. Stationary agents provide proper processing of information, control of the status of the request, and verification of credentials of the requester. The internal components and their tasks during the federated query processing of FIMA are described as follows.

1) FIMS Agent: It runs as a stationary agent at each site. Whenever FIMS Agent receives a request for a federated query, it generates another stationary agent based on the requester's passport. This new agent is defined as Federated Query Processor Agent (FQP Agent) and runs in a different execution thread. From this point on, all query and access operations are bound to this FQP Agent.

As part of the strategy to enforce the visibility levels and access rights on the information, FIMS Agent checks the "credentials" to validate the access rights of the requester before creating the FQP Agent. The FQP Agent is created with the credentials of the requester and, those credentials are used un turn to create authorized remote agents. In general, this strategy is used to validate the access rights to the information for requesters, no matter if the requester is local or remote.

2) FQP Agent: The FQP Agent performs several query tasks, grouped as follows:

- **Query translation.** A query received in high level format is then translated into internal structures.
- **MIRA creation.** Depending on the type of federated query, and on the targeted itinerary, appropriate Mobile Information Retrieval Agents (MIRAs) are created. If the query has multiple data sources, then multiple MIRA agents are created.
- **Query decomposition.** The original query is divided into a number of sub-queries. These sub-queries are assigned to the corresponding MIRAs according to the number of target nodes.
- **MIRA transmission.** Each MIRA is sent to a remote node, carrying the corresponding sub-query.
- **Query evaluation.** The MIRA performs and executes the query at the remote node. It retrieves the requested information.
- **Result transmission.** The MIRA transmits back to the FQP Agent the results from the sub-query.
- **Information merge.** Once all sub-results arrive, the FQP Agent merges and sends them as a final result to the requester.
- **Resource release.** When the query finishes, FQP Agent releases all resources associated to the query, disposing all the MIRA agents involved during the evaluation and

execution. FQP Agent can be disposed at any stage of the query execution, closing the query processing.

3) MIRA: Each mobile agent that performs the retrieval of information from external nodes ia a MIRA. Employing a MIRA guarantees the possibility of combining some intelligent decision making for information processing during the data retrieval operations. The management of MIRA agents is solely performed by the FQP Agent and it is completely transparent to the requester. From the requester point of view, this transparency noticeably reduces the system complexity.

B. Federated query language specification

There is a need for a language specification for FIMA. Our suggestion is not to reinvent the wheel or design a complete computational federated query language from scratch. Our approach rather provides proper extensions for federated access to a well-known language specification. When designing these necessary extensions, however, not all the defined goals can be entirely fulfilled, due to the conflicts and trade-offs among the goals, languages, resource limitations, and language availability.

The language proposed for FIMA is based on the Object Query Language (OQL). OQL is the de facto standard to query objects and relies on the object oriented model of the Object Data Management Group (ODMG) [14]. In a nutshell, OQL represents and object oriented extension to SQL that uses the known select-from-where clause. OQL is designed specifically around the operational needs to support an object-oriented architecture. The most significant differences between OQL and traditional SQL are highlighted below:

- OQL has the ability to support object referencing within tables. It is possible then to have objects nested within objects.
- Not all SQL keywords are supported within OQL; "irrelevant" keywords were removed from the syntax.
- OQL has the ability to perform mathematical computations inside OQL statements.

After analyzing the above differences, the objectives and general requirements of TeleCARE, and pre-established priorities associated with FIMA design, a number of extensions were established. The language specification is presented as a XML document in Fig. 5.

```
<?xml version="1.0" encoding="UTF-8" ?>
<fqp type="parallel" time-out="2" life-span="10">
   <query>select p from Person p</query>
   <targetList>
      <target>URL node 1</target>
      <target>URL node 2</target>
      . . .
      <target>URL node n</target>
   </targetList>
</fqp>
```

Fig. 5. Language specification for FQP

C. Federated query types

FIMA's design allows the requester to select a *query type* that sets a predetermined behavior on how to execute the query. Each query type gathers the information that reside outside of the node under different ways. The advantage of using different types of access methods is that the user or requester can control the general application performance and overhead. The three federated query types queries can then be optimized for specific purposes.

1) Parallel query type: The parallel query type retrieves information, independent of the number of nodes to visit, in a transparent and parallelized way. For tele-monitoring, which consists of gathering remote readings, diagnostic data, and other environmental conditions of the involved nodes, the parallel query type offers a mechanism to transfer data to care experts (such as doctors, care consultants, etc.) as soon as possible, regardless the location. The rapid gathering of data leads to early diagnoses, assistantship, and prevention of problems. If care givers in an ambulance, for example, require the list of elderly located in a certain street, they can submit a command in parallel to the network.

2) Serial query type: The serial query type processes data from the nodes, one at a time, in a (*node-by-node)* fashion. This approach reduces the amount of resources to use, specially, those of communication and processing load. The main disadvantage however is the low speed due to the consecutive migration of code.

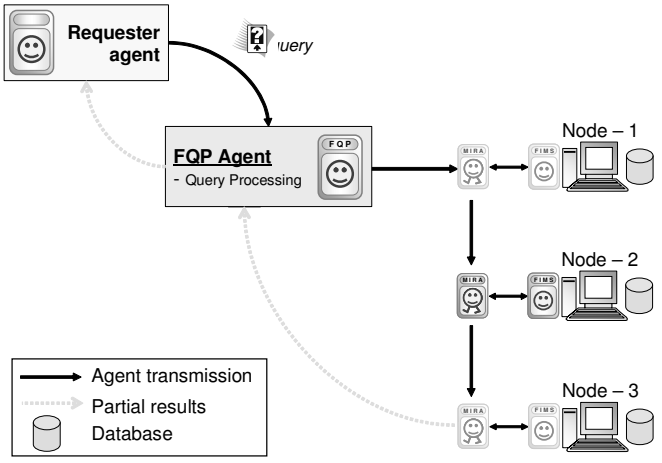

Fig. 6. Serial query type

As shown in Fig. 6, the FQP Agent creates only one MIRA that executes the query request at all required nodes, node-by-node. After the mobile agent completes its execution at each node, it sends the results back to the FQP Agent, which in turns formats and forwards them to the requester. It finishes the query when there are no more nodes to visit in his itinerary.

The serial query type can be use, for example, when elderly may move regularly outside their own homes to residential centers, or viceversa. When this occurs, it is difficult to

locate the right location to gather care data. It must also be distinguished, whether the elderly lives in a residential center, which includes nursing homes and senior citizen residencies, or in a private residence, which comprises the cases of elderly living in a private home. The elderly may also live alone, with others (as a couple or with family members), or in a mixture of these environments. Under this scenario, it is not certain to which node a monitoring query can be sent or where as the information (regarding status monitoring) can be found. Using a serial query type, however, the user can establish an itinerary of residence where to perform the query. If the request is not an emergency, the query is minimized only to the minimum possible nodes, avoiding network overhead.

3) Sequential query type: Sequential query type is an evident evolution of the serial query type. It further adds advantages of scalability, decrease of the processing time, reduces the usage of resources to the minimum necessary. With a community whose requirements escalates and becomes more complex, these factors become essential.

Using a sequential query, the "requester" (being an intelligent decision-maker, i.e. a human or expert system) interacts directly with the query execution, in order to decide whether to stop the processing when the query is satisfied. A sequential query is used when the requester is interactively looking for a piece of information. For instance, when an operator at the care center requires a phone number of an elderly's relative. The operator launches a sequential query over different nodes where he thinks data may be found. When the operator finds the phone number, the query can be immediately stopped avoiding the search on the remaining nodes.

VI. Conclusion

A federated information management approach offers suitable mechanisms to cope with the required flexibility, heterogeneity, autonomy, and privacy requirements for information handled within a collaborative network for telecare. Its combination with mobile agent-based platform in FIMA has proved to be an effective approach to develop a flexible infrastructure supporting a large variety of services within TeleCARE project.

The implemented FIMA provides access to remote data from several nodes and does not require any centralization of data or control. Fig. 7 shows an example how FIMA gathers of information. FIMA supports large numbers of users and agents accessing and retrieving data, while providing visibility rights to physically distributed and heterogeneous data.

Furthermore, FIMA provides three different types of access methods that requesters can use to control the performance of queries. These query are namely a parallel query, where the performance in speed is the key consideration; a serial query, which targets the optimization of resource usage; or a sequential query for decisive user interactivity.

Acknowledgments. Part of this work was carried out in the context of the Virtual Laboratory for e-Science project. This project is partially supported by a BSIK grant from the Dutch Ministry of Education, Culture and Science (OC&W) and is part of the ICT innovation program of the Ministry of Economic Affairs (EZ).

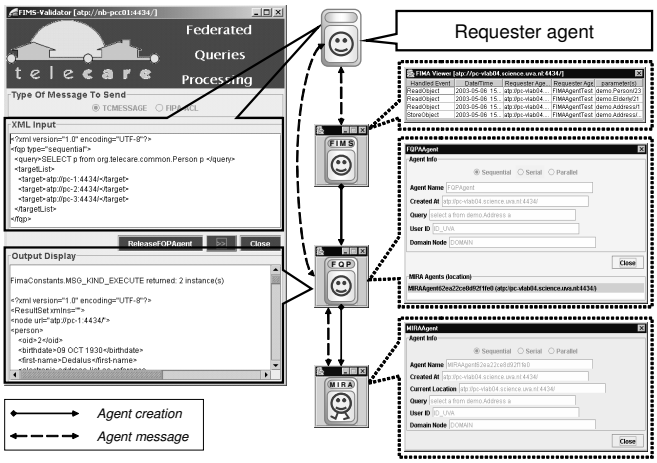

Fig. 7. Retrieving information with agents

References

[1] M. J. Field, *Telemedicine: A Guide to Assessing Telecommunications in Health Care*. National Academies Press, October 1996, iSBN: 0309055318.

[2] NHS Executive, "Information for health: an information strategy for the modern NHS 1998-2005," Leeds: NHS Executive, 1998.

[3] A. P. Sheth and J. A. Larson, "Federated database systems for managing distributed, heterogeneous, and autonomous databases," *ACM Computing Surveys*, vol. 22, no. 3, pp. 183–236, 1990, iSSN: 0360-0300.

[4] TeleCARE, "A multi-agent tele-supervision system for elderly care," Internet Reference, 2006, `http://www.uninova.pt/~telecare`.

[5] N. R. Jennings, "Coordination techniques for distributed articial intelligence," in *Foundations of Distributed Articial Intelligence*, O'Hare and N. R. Jennings, Eds. Wiley, 1996, pp. 187–210.

[6] M. Wooldridge, "Intelligent agents: The key concepts," in *Proceedings of the 9th ECCAI-ACAI/EASSS 2001, AEMAS 2001, HoloMAS 2001 on Multi-Agent-Systems and Applications II-Selected Revised Papers*, ser. Lecture Notes in Artificial Intelligence, vol. 2322. London, UK: Springer-Verlag, 2002, pp. 3–43, iSBN:3-540-43377-5.

[7] B. Landfeldt, J. Chan, B. Thai, and A. Seneviratne, *User mobility in IP networks: current issues and recent developments*. Boca Raton, FL, USA: CRC Press, Inc., 2003, vol. Wireless internet handbook: technologies, standards, and application, ch. 9, pp. 197–225, iSBN: 0-8493-1502-6.

[8] D. Lange and M. Oshima, *Programming and deploying Java mobile agents with aglets*. Addison-Wesley, 1998.

[9] E.-P. Lim and J. Srivastava, "Query optimization and processing in federated database systems," in *CIKM '93: Proceedings of the second international conference on Information and knowledge management*. New York, NY, USA: ACM Press, 1993, pp. 720–722.

[10] J. R. Chen, S. R. Wolfe, and S. D. Wragg, "A distributed multi-agent system for collaborative information management and sharing," in *CIKM '00: Proceedings of the ninth international conference on Information and knowledge management*. ACM Press, 2000, pp. 382–388.

[11] S. Papastavrou, G. Samaras, and E. Pitoura, "Mobile agents for WWW distributed database access," in *ICDE*, 1999, pp. 228–237.

[12] H. Afsarmanesh, V. Guevara-Masis, and L. O. Hertzberger, "Management of federated information in tele-assistance environments," *The Journal on Information Technology in Healthcare*, vol. 2, no. 2, pp. 87–108, 2004, publisher: Health Technology Press.

[13] V. Guevara-Masís, H. Afsarmanesh, and L. O. Hertzberger, "Ontology-based automatic data structure generation for collaborative networks." in *Virtual Enterprises and Collaborative Networks*, L. M. Camarinha-Matos, Ed. Toulouse, France: Kluwer, 2004, pp. 163–174.

[14] R. Cattell, D. K. Barry, M. Berler, and e. al., *The object data standard: ODMG 3.0*. San Francisco, USA: Academic Press, 2000.

[15] VL-e Consortium. VL-e – Virtual Laboratory for e-Science. Internet Reference, 2006. `http://www.vl-e.nl/`.