18

ONTOLOGY-BASED AUTOMATIC DATA STRUCTURE GENERATION FOR COLLABORATIVE NETWORKS

Victor Guevara-Masis, Hamideh Afsarmanesh, L.O. Hertzberger
University of Amsterdam, Faculty of Science
Kruislaan 403, 1098 SJ, Amsterdam, THE NETHERLANDS
{vguevara, hamideh, bob}@science.uva.nl

Different forms of web-based collaborations are continuously configured and growing. Such emerging collaborative networks (CNs) are exemplified by Virtual Organizations, Virtual Communities, and Virtual Laboratories. Design and development of these environments constitute long, incremental, and complex processes. One challenge in this process is the proper definition of required syntax for entities and concepts, to be shared over the networked environment. A current tendency to describe the information/knowledge models for CNs is to use ontology systems. This paper presents an innovative tool called DOSG that assists the developers of CN environments, by taking the ontological definitions of concepts and entities, and automatically generating data structures such as: the relational database schema, Java code, and XML Schema. The DOSG prototype is developed within the TeleCARE project. The main goal of the TeleCARE project is the development of a CN infrastructure, providing tele-assistance for elderly care. DOSG represents one component of the federated information management component of TeleCARE infrastructure.

1. INTRODUCTION

Today, computer systems are present in almost every part of the society, used in science, industry, government, education, as well as at homes for games and fun. Moreover, based on the easily available communication networks and their free and user friendly tools, different forms of *web-based collaboration* are continuously configured and growing. Namely, web-based mechanisms are used as the base, to deal with the increasing need to interoperate within the emerging distributed and collaborative environments. At present, great expectations are associated to the possibilities that *collaborative networked organizations and societies* of future may create, and what they might enable people to achieve. Building proper support systems on top of the web for emerging *Collaborative Networks* (CN) remains however as a main challenge, being tackled by some research and development initiatives.

Among the main emerging example areas of CNs, we can name Virtual Organizations, Virtual Communities, and Virtual Laboratories, as described below:

i) Virtual Organizations (VOs) [6, 9] provide web-based collaboration and sharing of resources and skills among autonomous (profit, non-profit) organizations, towards achieving specific common goals. To outside, a single "virtual external organization" represents the collaboration of all member organization in a VO. In research and development projects, examples of VOs are now being applied to some manufacturing, public/government-based systems, and more recently to service provision such as tourism, and for tele-assistance / tele-monitoring (e.g. for elderly care).

ii) Virtual Communities (VCs) [7, 8] provide human web-based networks, replacing the face to face communication with the "virtual communication", being more and more applied everyday to sharing knowledge, time, or experience among the associated members within a virtual association or community. VCs range from topical associations, support clubs, and keeping in touch for individuals, to the professional VCs providing consulting on specific areas of expertise.

iii) Virtual Laboratories (VLs) [13, 18] facilitate web-based provision of required scientific laboratory resources (HW/SW and information) to collaborating scientist users. VL provides a "virtual experimentation environment", through which it can support sharing of laboratory facilities and services, provided by different member nodes in the network, e.g. sharing of Grid-based VL resources for the purpose of remote scientific experimentations and problem solving in different domains, and even performing inter-disciplinary projects.

Design and development of the VO, VC, and VL are challenging open areas, now being tackled by the research and technological development projects [2, 6, 12]. Although they share a number of common characteristics, clearly each of these virtual environments has a large range of requirements and specificities that are different and some even contradictory with the others. One main shared difficulty however, is that the development of web-based collaborative environments and systems is not a "one-step" event, as described in the following paragraphs. Building all the elements of such an environment is rather an incremental and complex process, involving many people, from a wide variety of disciplines, working for a long period of time.

At the first step, usually a group of people from various disciplines and expertise together build an architecture and a set of components and mechanisms that altogether constitutes a generic *reference model* for the aimed CN. The required expertise primarily depends on the type of collaborative networks. Typically several other expertise in addition to many areas of computer science are required; that may range from management, sociology, and law, to experts in application domains being a scientific domain, engineering, health care, entertainment, tourism, etc. The main areas of computer science required to take part at this stage typically includes modeling and system design programming, human-machine interface, information management, communication, artificial intelligence, operation research and security among others. At the second step, usually a group of developers, typically computer scientists, need to work on the *development* of the base infrastructure, constituting the *horizontal layer* development of a CN. Possibly, this group also develops a few

generic information/knowledge models, an ontology, and some generic base services and tools.

From this point on, other developers from multi-disciplines related to the CN's application domain, in collaboration with computer scientists get involved to gradually and incrementally build a wide range of tools and value-added services on top of the base CN infrastructure. Each of these tools and services in turn, enhances the general functionalities of the whole system. These tools/services constitute the *vertical layer* of the development. A simplified generic reference architecture for CNs is depicted in Figure 1.

Hence, considering the requirements for building collaborative networked environments and systems, their base horizontal infrastructure must, on one hand support the *flexibility, openness* and *extensibility* needed for the incremental addition of forthcoming user required information and knowledge models and other vertical tools/services. On the other hand, the base horizontal infrastructure must provide **facilities for assisting** the future developers/users of the vertical layer for collaborative networks.

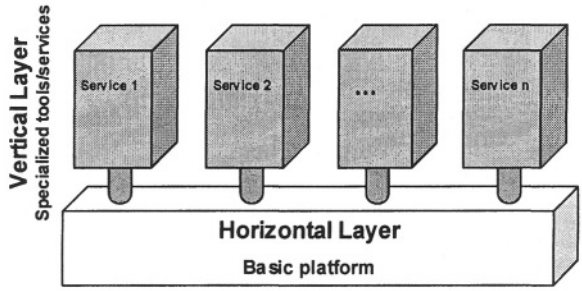

Figure 1 - A simplified generic reference architecture for CNs

In previous publications, several challenges such as flexibility, openness, extendibility, involved in developing base infrastructure for VOs, VCs and VLs [2, 5] were discussed. This paper however primarily zooms in on the challenge of providing assistance for developers of the CNs, and managers/coordinators of such web-based distributed environments and systems. We particularly focus on the area of assisting developers and managers with their "definition of information and knowledge models" that is either: (1) required for modeling the information/knowledge, about the environment, users, or the CN's base funtionality needed by developers, in order to support the co-development and integration among different components of the base horizontal infrastructure, or (2) required for the development of the systems/tools for the vertical layer of the CN. And thus, a main challenge here is to assist their users with the difficulties of: *1*- the definition of database schema for the data that needs to be stored in the database repository, and *2*- the definition of common data structures to support the sharing/interoperability and integration among different tools and supporting software systems in the CN.

In this paper, an assisting component called *Dynamic Ontology-based data Structure Generator* (DOSG) is described that tackles the above tasks and difficulties, and makes them automatic. This assisting component is designed and developed within the IST TeleCARE project [17], focused on the design and development of a VO infrastructure for tele-assistance and tele-monitoring for

elderly care. Therefore, the examples presented in this paper come from the elderly care application domain. In specific, we focus on the distributed services development within the vertical layer of the TeleCARE reference model.

To develop DOSG, after an initial phase of analyzing the TeleCARE's VO infrastructure, we have identified the following four requirements for the information modeling and for the definition of data structures, that need to be assisted and can greatly benefit from DOSG:

1. General data related to the configuration of the distributed virtual environment must be modeled and stored in the database (e.g. information about the nodes involved, individual users, theirs access rights, etc.). This information is required to be provided and maintained by the virtual environment administrator.
2. For each vertical service, the service developers need to model some information to be stored in the database of the center that manage this service, from which other distributed nodes (that use this service) can access the information.
3. Similar to data modeling and storage requirements for vertical services, the developers of the base infrastructure need to model and store their data in the database repository.
4. For each value-added service (service defined on top of other more elementary services), there is some data from other service that needs to be accessed, and vice versa some data that this service generates may need to be retrieved by other services. Therefore, within the distributed collaborative environment, the interoperation and data sharing among vertical services is necessary, to support building of value added services on top of each other.

The DOSG approach and system presented in this paper tackles these problems and assists developers through reducing the difficulties introduced earlier. DOSG is a main component of the federated information management system of the TeleCARE [1].

This paper first provides a brief overview of the TeleCARE architecture with some emphasis on its federated information management architecture, also providing a brief introduction to its DOSG component. Then, it provides a specification of the Protégé ontology management system, and how TeleCARE ontologies are defined using an object oriented approach. It also describes the automatic process of database models generation from the ontological description. This process is described in terms of the data structures needed by vertical services, and the mechanisms and internal elements involved during their translation to RDBMS schema, XML Schema, and Java code. Finally, the paper concludes summarizing its achieved results.

2. TeleCARE - A TELE-ASSISTANCE PLATFORM

The IST 5FP TeleCARE project designs a configurable and collaborative framework solution for tele-supervision and tele-assistance, to support the elderly. TeleCARE integrates a number of technologies and paradigms into one solution, in order to provide an open architecture supporting seamless future expansion. It is based on the integration of: (i) *multi-agent systems* (MAS), including both stationary and mobile

intelligent agents, (ii) *federated database systems,* (iii) *secure communications,* and (iv) some *intelligent services* that are likely to be offered by the emerging ubiquitous computing, intelligent home appliances or their software based applications.

The infrastructure developed for TeleCARE contains a *horizontal* platform, which provides the MAS and the federated information management functionality, and a variety of *vertical services,* which the TeleCARE consortium has further developed on top of this platform.

2.1 The TeleCARE reference architecture

The reference architecture of TeleCARE constituting its cooperation/federation layer for the network nodes is detailed in [8]. The main components of this architecture are however briefly addressed in this section, and depicted in Figure 2 below. The designed architecture of TeleCARE is composed of a three-level architecture, at the bottom the *External Enabler Level,* in the middle the *Core MAS Platform Level,* and on top the *Vertical Services Level.*

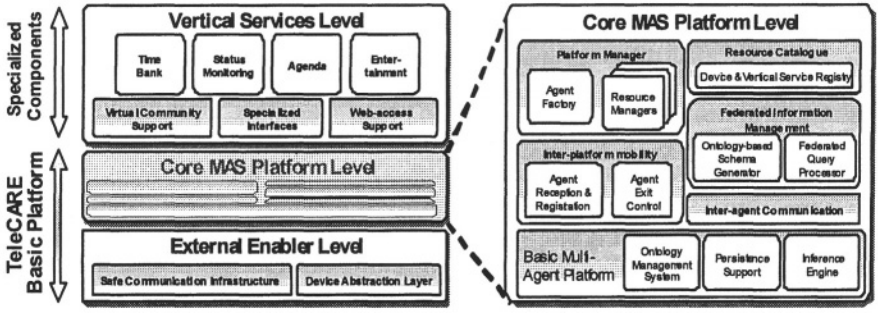

Figure 2 - TeleCARE reference architecture

2.1.1 External Enabler Level

This level supports the remote communication with other nodes and provides interfacing mechanisms to the external devices. This level compromises two segments:

- **Safe communications infrastructure,** providing safe communications and supporting both secure and reliable agent message passing.
- **Device abstraction layer,** interfacing the sensors, monitoring devices, and other hardware (home appliances, environment controllers, etc.) to the TeleCARE environment.

2.1.2 Core MAS Platform Level

The Core platform level is the main part of the reference architecture and it supports the fundamental functionality for agents and their interactions, including the creation, launching, and reception of agents, user authentication, access rights verification, and execution of stationary and mobile agents. This level includes the following main modules:

- **Basic Multi-Agent Platform,** provides the essential multi-agent support, and it is based on Aglets framework [3,11] with the following extensions:
  i. *Ontology management system* – To support the creation of ontologies
  ii. *Inference engine* – To allow intelligent agent interpretation
  iii. *Persistence support* – For basic agent recovery mechanisms
- **Inter-platform mobility,** extension to the basic MAS platform to support generalized mobility of agents, including security mechanisms. This module includes the *Agent Reception & Registration* component and the *Agent Exit Control* component for administration of stationary and mobile agents.
- **Inter-agent communication,** extension to support credentials and coordination of agent communication, independently of the agent location.
- **Federated information management [1],** supporting the information management of TeleCARE nodes and providing the infrastructure for flexible processing of federated queries. It also preserves the information privacy through access rights management to local data. This module includes the *Ontology-based data Structure Generator* and the *Federated Query Processor* components, and is build on top of SAP DB [14], a relational database system.
- **Resource catalogue management,** providing a catalogue of resources, that registers the descriptions of all device and vertical services available at the site, as well as their access rights.
- **Agent factory,** to support the creation, specification, and launching new agents.
- **Platform manager,** configures and specifies the operating conditions of the platform at each site, including user administration and node management.

2.1.3 Vertical Service Level

The applications / vertical services level focuses on specialized tools to support the users of TeleCARE. Such users constitute: the elderly people (who may require specialized user interfaces), the care providers, and relatives of the elderly people (assumed to be able to interact with normal computer interfaces). This level compromises two layers.

- **Base services,** a set of base services on top of the horizontal infrastructure that provides specific support to other value-added services.
  i. *Virtual Community Support* – To support the creation and management of virtual community for the elderly perople.
  ii. *Specialized interfaces for elderly* – Suitable computer interfaces for elderly use.
  iii. *Web-access support* – Web-based mechanisms to access the TeleCARE environment.
- **Vertical Services,** a number of specialized vertical services are implemented as TeleCARE applications, including the time bank, agenda reminder, living status monitoring, and the entertainment service.

3. ONTOLOGY MANAGEMENT SYSTEM

The ontological definitions in TeleCARE constitute the syntactic definition and modeling of a collection of concepts, entities and their inter-relationships. Therefore,

in TeleCARE, developers of modules for the core platform, as well as the developers of vertical services define their required data structures using the Ontology system chosen for this platform. The ontological definitions can then be consulted (and understood) by both humans and software agents alike.

The chosen ontology system for TeleCARE is the Protégé-2000. This system is developed at the Stanford Medical Informatics Lab [16] to support the software developers and domain experts for development of their knowledge-based systems. Protégé-2000 assists its users with the creation and storage of their knowledge base by definition of structured classes. Although Protégé was designed originally for the medical domain, it later grew as a general-purpose set of tools for building knowledge bases systems in any domain, providing convenient mechanisms for concept modeling. Protégé has a component-based architecture that enables the development of new functionality in form of *plug-ins.* One important and strong point of Protégé is that it is a freeware and open source software. The above reasons, namely: the ability to work with different domains, facilities for specific concept definition, having a component-based architecture, and being an open source software, are in particular important to the TeleCARE project. These features match the design principles of the TeleCARE system and preserve the objective of providing a highly configurable framework for collaborative environments. Therefore, Protégé 2002 is chosen as the ontology management system for TeleCARE environment.

4. DOSG DESIGN AND IMPLEMENTATION DETAILS

The *Dynamic Ontology-based data Structure Generator* (DOSG) is a component of the federated information management module of the TeleCARE architecture. It assists the process of common database schema generation to support the shareable information in a collaborative networked environment such as TeleCARE. The main task of DOSG is to transform and translate a Protégé ontology-based definition into the underlying schema model of the federated information management layer of the TeleCARE platform. Particularly, it provides a collection of output schemas with their appropriate class definition and inter-class relationships that are used to initially specify relational database structures. However, DOSG is also an innovative mechanism to leverage object knowledge modeling and providing proper Java object persistence. It assists the developers, from both the basic horizontal platform level and the vertical services level, with their seamless manipulation of their information in different formats.

As shown in Figure 3, based on the ontological definitions provided by users, the DOSG tool automatically generates five different output, namely: (1) the relational database schema, (2) the source code of the Java classes, (3) the XML Schema, (4) the data object mapping that governs the conversion between Java classes and the database system, and (5) the XML mapping that binds the Java classes with XML documents. Below, we first address the generation of data structures in these five formats and then provide a summary description of their usage.

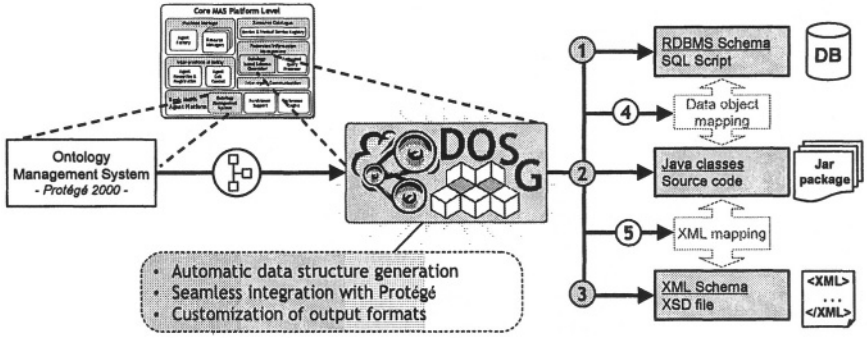

Figure 3 - DOSG schema generation

DOSG is designed as a plug-in to Protégé. It extends Protégé's ontology editor with an interface that allows users to parameterize the automatic data structure generation. DOSG benefits from the integrated Protégé environment by gathering online input related of conceptual schema, while allowing customization of some parameters for this generation process through the DOSG-GUI. Some examples of these parameters are found below. The implementation of DOSG is in Java and, it also uses free / open source software, in specific Castor is used to produce the two mapping definitions [10], while Xerces is applied for the development of XML Schema [4].

4.1 Translation to relational database schema (RDBMS Schema)

A major challenge in building DOSG is its translation between the object-oriented ontology definitions and the relational database schema. The object-oriented ontology supports definition of applications' entities and concepts as objects described by their structure and behavior. Relational database systems on the other hand support the definition of record-based structures. RDBMSs support the storage of data in tables and its manipulation via data manipulation language; internal to the database via the stored procedures, and external to the database via SQL commands. DOSG generates the necessary SQL statements for data storage following the ANSI standard (ANSI X3.135-1992), and targeted to the SAP DB.

Clearly enough, the conversion between the objects and their relational structures is not flawless or perfect. The underlying paradigms are different and the two sets of structures cannot be converted seamlessly [15]. For instance, the difficulties become apparent when selecting an approach to access the data, where in the object paradigm it is possible to traverse objects through their relationships, while in the relational paradigm this linkage can only be performed via the join of the two tables.

There are some fundamental differences that DOSG cannot overcome or translate, however DOSG has also solves several of the problems, allowing a smooth translation between the objects and their relational counterparts, as addressed below. A set of technical decisions had to be made considering the tradeoffs of the two paradigms, and below the most relevant considerations are pointed out.

- **Object identifiers (OID).** To allow data persistence for objects, it is required to assign a unique object identifier (OID) to each object. In relational databases, the unique identifier is referred to as primary key (PK) that allows the

possibility to locate unique records in the repository. Thus, for the translation, DOSG adds a PK when it is defining the relational tables; however the name assigned to the PK attribute can be customized through the DOSG-GUI.

- **Handling the slots.** In general, all the slots or attributes for each class are translated into table columns by DOSG. In relational databases however, it is required to specify a "type" for every column in the table, while Protégé allows certain ambiguity (e.g. using the type `Any`). In such cases, DOSG has no other choice but not to create a column for such a slot.
- **Handling the classes and hierarchies.** The strategy followed by DOSG to solve the class definition is to create one table per class and then, to define its specific attributes. To translate a class hierarchy, as shown in Figure 4, DOSG establishes a relationship among the tables of the subclass and the super-class, which is maintained through the usage of the primary key (PK) and the foreign keys (FK).

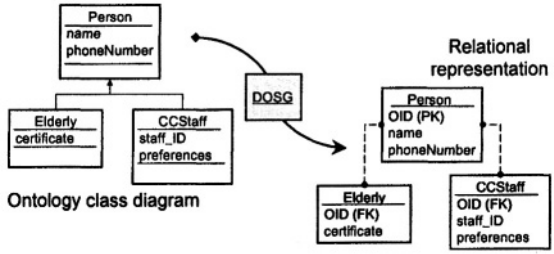

Figure 4 - Mapping the class hierarchies

- **Handling the one-to-one relationships.** In relational databases, relationships are maintained through foreign keys. DOSG implements one-to-one relationships by including the OID of one table into the other, as depicted in Figure 5.

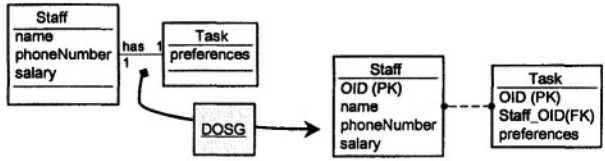

Figure 5. Mapping one-to-one class relationships

- **Handling the one-to-many and many-to-many relationships.** In order to implement a many-to-many relationship, DOSG uses a relationship table, as shown in Figure 6. DOSG employs the same strategy for handling one-to-many relationships, since it gives further flexibility to add more columns in the table.

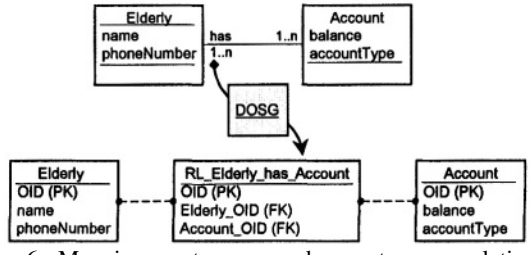

Figure 6 - Mapping one-to-many and many-to-many relationships

- **Value constraints.** Some value constraints from Protégé system are successfully translated to the relational schema by DOSG, e.g.: maximum/minimum values for numeric types, default values, enumeration of strings (`Symbol` type in Protégé).
- **Storing semantics within the database repository.** Strategically, one of the features of SAP DB (TeleCARE database repository) is the support for storing the textual descriptions about the semantics of the data structures (e.g. tables, columns, keys, etc.). Therefore, if the ontology definition includes some textual description of their semantics on classes and attributes, then DOSG also adds these semantics in the database repository.

4.2 Translation to Java source code

Since both the Protégé system and Java language support object-oriented paradigm, this translation and generation of code is more straightforward than the translation to the relational schema. Nevertheless, there are several issues that require attention, and some cases cannot be handled by DOSG, due to the technical incompatibilities. These incompatibilities include features that are available in the ontology system such as multiple class inheritance, but not supported by Java environment directly. Some of these features are the value constraints; the type `Symbol` (which is converted to string type but no constraints are enforced); the type `Any;` and multiple class inheritance. On the other hand, when generating the Java source code through the parameterization of DOSG-GUI, it is possible to take advantage of some features offered by DOSG such as: (1) Java package definition, (2) on-line Java compilation, and (3) Java jar encapsulation.

4.3 Translation to XML Schema

Many organizations and enterprises use the XML documents as a format to exchange information and the XML Schemas as a common data structure definition language. XML documents are widely used to facilitate the information exchange due to the fact that they are both human-readable and machine-readable. The extensibility of XML allows creation of generic models that integrate data from different sources. The XML Schema generated by DOSG describes the logical model of the interchanged information defined by the Protégé ontology.

The XML Schema can support the representation from the object oriented paradigms better than the relational system. However, similar to Java code generation, there are still several issues that cannot be at this stage equivalently translated by DOSG from the object definitions to the XML schema, such as the value constraints, the type `Any`, or multiple class inheritance.

4.4 Generation of data object mappings and XML mappings (class views)

DOSG is able to elaborate the necessary mappings to transform data between the Java classes and the database schema, as well as between Java classes and the XML schema, as represented in Figure 3. The mappings follow the Castor specification [10] which provides bi-directional declarations between Java classes with both the relational data structures and the XML. In fact these mappings describe how the

properties of a Java object can be translated into their counterparts in the RDBMS and in XML Document.

- **Data object mapping.** The data object mapping is a description that renders object instances of Java model to relational database model, and vice versa. This is usually referred to as object-to-relational mapping (O/R mapping). Such mapping definitions are necessary to dissociate the changes in the structure of a Java object model from the changes in the database. The federated information management uses these mapping files, to grant proper access rights and visibility level on the database information.
- **XML mapping.** Similar to data object mappings, the XML mapping is a way to bind Java classes to the XML documents. It allows transformation of the data contained in a Java object model into/from an XML document.

4.5 Advantages and validation/verification of DOSG

Advantages of using DOSG are verified and validated by the software developers of TeleCARE, who were involved in the development of its horizontal and vertical layers. Here we briefly give some examples where these developers benefited from DOSG, by identifying each of the automatically generated formats of data structures. The vertical service developers as well as the developers of the horizontal layer's functionality need to create some **RDBMS Schema** to manage their information. At the same time, **Java class structures and source code** are needed to be defined in order to manipulate the same information in the code that they develop. Additionally, since TeleCARE environment supports the heterogeneity and independence of different sites, their interoperability must be supported. **XML Schema** for this information can enable simple and standardized exchange of XML documents with any other service or platform. Furthermore, for the development of most programs, it is necessary to convert information between these different data structures, preferably at their object level through **data object mappings** and **XML mappings.**

In TeleCARE, the three vertical services of *Agenda Reminder, Living Status Monitoring* and *Time Bank* were in specific used as the validation/verification ground for the DOSG component. The results of this process proved the design and prototypical development of DOSG to be successful.

5. CONCLUSIONS

For the emerging collaborative environments and networked systems an important element is their information/knowledge management. Developers of both the horizontal platform as well as the vertical services/tools for CNs need to model their information/knowledge and provide proper definitions and syntax in terms of: i) database *schema* for the data that needs to be stored in the environment repository, and ii) "equivalent" *data structures* for sharing/interoperability among different services/tools and supporting software systems.

This paper describes an approach to automatically generate both the database schema and the necessary data structures, based on the ontological definitions provided by the developers, within the context of the TeleCARE project. Once the modeling of concepts are defined using the Protégé ontology management system,

the DOSG tool provides to its end users (the domain experts and developers) the possibility to translate these definitions into: i) *relational database schema,* ii) *XML Schema,* and iii) *Java code.*

The automatic and dynamic generation of data structures is a key feature in TeleCARE allows the application experts and software developers to solely concentrate their efforts on their development and modeling their information through a user-friendly ontology system interface. DOSG frees these developers and application experts from the burden of knowing the technical details for developing the database structures, Java code, or XML Schema. However further research is required to overcome the restrictions for specific mismatch in these conversions, either imposed by the involved paradigms (e.g. between object-oriented and relational), or due to software environment limitations (e.g. lack of support for multiple inheritance in Java).

Acknowledgments. This work was funded in part by the IST program of the European Commission. The authors thank the contribution of the TeleCARE consortium.

6. REFERENCES

1. Afsarmanesh H, Guevara-Masis V. Federated management of information for TeleCARE. 1st International Workshop on Tele-Care and Collaborative Virtual Communities in Elderly Care (TELECARE-2004), Porto, Portugal, 2004.
2. Afsarmanesh H, Guevara-Masis V, Hertzberger LO. Virtual Community Support in TeleCARE. 4th IFIP working conference on Virtual Enterprises PROVE'03, Lugano, Switzerland, 2003; pp. 211-220.
3. Aglets.org. The Aglets Portal, http://aglets.sourceforge.net/, 2002.
4. Apache Software Foundation. Xerces2 Java Parser, http://xml.apache.org/xerces2-j/index.html, 2004.
5. Camarinha-Matos LM. New collaborative organizations and their research needs, Processes and Foundations for Virtual Organizations: Kluwer Academic Publishers, 2003; pp. 3-14.
6. Camarinha-Matos LM, Afsarmanesh H. The virtual enterprise concept, in Infrastructures for Virtual Enterprises. In: Luis,Afsarmanesh H (eds): Kluwer Academic Publishers, 1999; pp. 3-14.
7. Camarinha-Matos LM, Afsarmanesh H. Virtual communities and elderly support. In: Kluev VV, D'Attellis CE,Mastorakis NE (eds), MIV'01 - Advances in Automation, Multimedia and Video Systems, and Modern Compuuter Science: WSES, 2001; pp. 279-284.
8. Camarinha-Matos LM, Afsarmanesh H. Design of a Virtual Community for Elderly Support. Infrastructures for Virtual Enterprises (PRO-VE'02), Sesimbra, Portugal, 2002.
9. Camarinha-Matos LM, Afsarmanesh H, Erbe H-H. Advances in Networked Enterprises - Virtual Organisations, Balanced Automation, and Systems Integration, IFIP INTERNATIONAL FEDERATION FOR INFORMATION PROCESSING: Kluwer Academic Publishers, 2000.
10. ExoLab Group. Castor, http://www.castor.org/, 2004.
11. IBM Tokyo Research Laboratory. Aglets Workbench, http://www.trl.ibm.com/aglets/: IBM Japan, 2002.
12. Kaletas E. Virtual Laboratories and Virtual Organizations Supporting Biosciences. 3rd IFIP Working Conference on Infrastructures For Virtual Enterprises (PRO-VE'02), 2002; pp. 469-480.
13. Messina P. The Emergence of Virtual Laboratories for Science and Engineering - iGrid2002 Presentation, http://www.igrid2002.org/ppt/Paul_Messina.ppt, 2004.
14. SAP AG. SAP DB, http://www.sapdb.org/, 2003.
15. Silberschatz A, Korth H, Sudarshan S. Database system concepts, 4th edition: Mc Graw Hill, 2002.
16. Stanford Medical Informatics. The Protege Project, http://protege.stanford.edu/: Stanford University School of Medicine, 2003.
17. TeleCARE. A Multi-Agent Tele-Supervision System for Elderly Care, http://www.uninova.pt/~telecare, 2003.
18. Vary JP. Report of the Expert Meeting on Virtual Laboratories: United Nations Educational, Scientific and Cultural Organization (UNESCO), 2000.