VIRTUOSE: a Virtual Community Open Source Engine

M. Benini, Univ. of Insubria (benini@dsi.unimi.it)
F. De Cindio, Univ. of Milano (fiorella.decindio@unimi.it)
L. Sonnante, The Milano Community Network Foundation (leonardo.sonnante@rcm.inet.it)

1. Introduction

AIReC (the Association for Informatics and Community Networks) groups the Community Networks (CNs) located in the Lombardy region; it has been established under the auspices of the Lombard Regional Government and the Department of Computer Science of the University of Milano. Among its many activities we recall it as the organizer of the First European Conference on Community Networking, ECN'97. The CNs members of AIReC use different software platforms for their activities: the oldest ones chose in the very beginning FirstClass, a proprietary software developed within the Bulletin Board System paradigm; some CNs use Lotus Notes because it was the intranet/extranet platform adopted by the hosting Municipality; finally, a couple of them adopted free software environments.

However, no one is fully satisfied with its own choice and it appears to us that we could not rely upon existing software solutions specific for community: for instance, the CSuite software [Mcm98], [CS] does not match our needs because of its character-based user interface. Therefore, after some preliminary studies developed through Master theses [Ale99], [ADRS99] [For01], in 2000 AIReC obtained a funding by the Lombard Regional Government to undertake a project for developing a CN platform with the following features:

(i) to cope with the needs of the several CNs members of the association. This is not limited to provide all the technical functions for managing a community network, but extends to the migration of the running CNs from the current to the new operational environment. As a matter of fact, this feature might be critical from the user' perspective, since virtual community members are often very conservative. Therefore, customisation is a crucial issue both for end-users (i.e., citizens [Pya98]) and for administrators who need to reproduce the specific look-and-feel of his/her community;

(ii) to enhance the facility of information exchange and discussion sharing among different CNs. Although each local virtual community (VC for short; we now prefer to denote the experiences of community networking with this term, since they are virtual communities where the shared interest is given by belonging to a specific local community) is fully autonomous in making the choices about contents and services to be provided and discussions to be supported, a mutual benefit may derive from co-operation. Thus the platform should make it easy the exchange of information, the sharing of discussions and also – very important point – the sharing of software solutions among the CNs members of the Association;

(iii) the new platform should be of interest for the medium-big realities (municipalities and provinces) which, nowadays, already have their own web site, and may want to enrich it incrementally with elements of discussion with and among citizens and of contents provided by users; the new platform should encounter the needs of the medium-small realities which don't have a web site, as a recent survey shows that these ones are still the majority;

(iv) the new platform should also face the needs of those who have understood the big potential of communities for attracting attention and audience in the commercial sites [HA97], [Kim00]. As well discussed in [Mic00], this idea is particularly interesting in the specific Italian industrial context, which already knows and experimented the economic powerfulness of the network of social relationships within a local community in the so called industrial districts, and more and more asks to renew this tradition to face with the demands of the global market.

Requirements (iii) and (iv) suggest to develop the new platform on the top either of the open-source technology or of the most popular proprietary technology. Requirement (i) does not give any hint in the choice because of the variety of current solutions. Requirements (ii), in particular the wish of sharing software solutions, strongly pushes in favour of open-source technology, for reasons that we would consider evident. Moreover the original view behind community networks and community networking went and goes in the same direction: let's in fact recall that free-nets are the "parents" of community networks and that free-software is the "parent" of the open-source technology: in both cases 'free" stands for free-of-charge but also for "freedom". Furthermore, in the public organisations there exist an increasing interest for the open-source technology because more and more people believe that proprietary technology is too expensive and may leave in private hands the control of public data and procedures. Therefore, we decided to adopt open-source technology. However, to retain an open dialog with the still popular application environments - which are often the only ones used in small municipalities and business companies - we are also pursuing a parallel project for developing a virtual community server built up on the top of the standard applications available in a Microsoft-based environment, trying to keep consistency between the two worlds.

The goal of the project is therefore to develop an open-source server for managing a virtual community (VIRTUOSE - VIRTual CommUnity Open Source Engine - in the following), namely a community network.

The distinguished feature we want to achieve is to homogeneously couple both the publishing of *information*, and facilities for *discussion*, which enriches information with community-generated knowledge (expression which we prefer to the most common "content generated by users"). Moreover, the platform must be open to integration with interactive *services* provided by a specific CN. In fact, let's suppose a citizen asks the *list of movies* in Milano this evening. By accessing the local forum "Cinema" s/he enriches this information with community-knowledge created through discussions, for instance: *"beautiful plot".* At this point, s/he would *book a place* in the closest hall giving the movie. After assisting the movie, s/he comes back and comments in the local forum the

movie or the state of the cinema hall (e.g., from the point of view of the facilities for disabled people).

The same pattern can be applied in many situations, from the choice of a restaurant to the need of a laboratory for clinic analysis. In this last case, comments after service use provides a powerful way of assessing the Quality of Services. Fig.1 schematizes the interplay between these three elements.

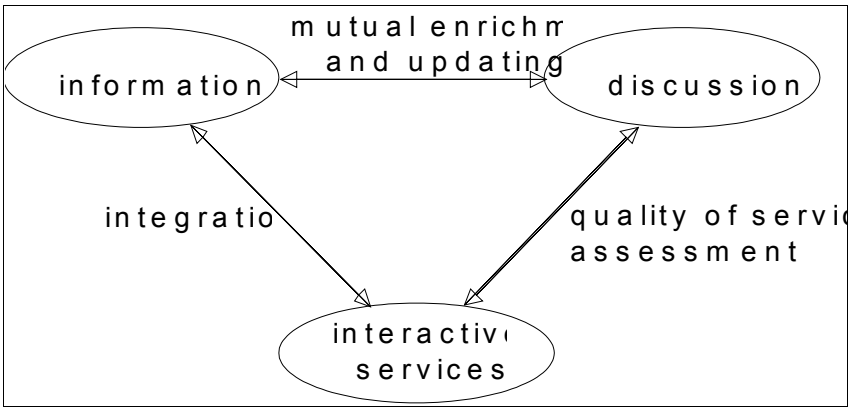

Fig. 1: Interplay among information, discussion and services

Most of the applications developed for the web are basically *publishing* oriented. As a consequence, the so called *application servers*, often concentrate on this need, as in the case of Midgard [MDG]. On the other hand, portals require an integrated technology for publishing information and for enforcing security in the access of interactive services: integration is often reached by ad hoc software development which for instance enriches platforms developed for traditional press media, such as Vignette [Vig], with solutions for secure transactions, for the digital signature, for a single registration. Portals often include discussion forums, but they are seen as auxiliary services, completely disjoint from information and services. Discussion forums are handled by using one of the large variety of dedicated software, either proprietary (e.g., the already mentioned FirstClass and Lotus Notes) or open-source (let's for instance mention Phorum [Pho]; for a quite rich list, see [CW] and [SF]).

Consequently, to develop and implements our view, VIRTUOSE keeps the *message* as the fundamental entity which constitutes both the unit of information to be published and the unit of discussion among people. Messages are grouped into *conferences*. *Users* access conferences through *views*. These four basic entities are presented in detail in the next section. Section 3 discusses the system functionalities and section 4 its architecture.

2. Design choices behind VIRTUOSE

2.1 Guidelines

The guidelines underlying the development of VIRTUOSE can be summarised as follows.

Generalisation

The design principles outlined in the previous section and the many kinds of contents that can be found in a virtual community lead to select some key elements to characterise a virtual community. It is the implementation of these key elements that can be considered the kernel of a technological platform to support virtual community management.

Great importance is especially given to the generalisation of the forum concept, that we named conference and it can be defined as an information containers regarding a specific topic. So examples of conferences are: a distance course of lectures, a poll to collect users opinion about a specific topic; the lists of cultural appointments that happen in a specific place, etc.

Use of specific tools for different needs

Technology selected to implement this project is the same used to manage many dynamic Web site based on databases: Web pages are PHP dynamic pages that fetch data to be shown from a DBMS. This architecture allows specialised servers for different functions. A Web server for displaying interface and client connection management; a DBMS for data management and organisation; a mail server to send and receive e-mails. Thus our choice permits to use the better server for every task, making possible to build our software on a solid base and enabling it to focus on virtual community features.

Use of Open Source software

Using Open Source servers guarantees that everyone wishing to use our software can easily get them without extra costs. Besides this fact, the choice is often technically strong since some open source servers are the standard de-facto in the field, e.g. the Apache web-server or sendmail, the e-mail management server.

Further extensions

Virtual community management may need many features, even very different between apparently similar communities. Our software focuses on some of them, but is designed to be open to integration of other features developed by third parts in a future.

2.2 Key entities

In the design phase, the following key entities have been outlined as characterising a virtual community: Conferences; Views; Messages; Users; Profiles.

Conferences can be defined as information containers regarding a specific topic. This information is included in messages, that is, a set of elements belonging to the same structure. The structure therefore defines the type of the message. Every conference has a set of message types accepted. This wide conception of conference and message concepts, as implemented inside VIRTUOSE, allows community members to create and publish structured contents, but also claims for a tool for filtering. This is the function carried out by views.

Views establish the way in which users and conferences interact. A conference can only be

accessed through a views. Specifically, a view contains:

- a set of message types; it's a subset of message types accepted by the conference, the ones which can be accessed through the view;
- a visualisation layout;
- a set of permissions, i.e. actions that one is allowed to perform on the messages of that conference when accessed through the view.

Messages are the containers for the information belonging to the system and are stored in a structured manner. Each message has a structure that defines a set of fields that constitute the type of the message. The values of the fields are displayed by means of display functions; Permissions of view dictate actions that can be played on messages.

Users are the authors of messages and they are identified and classified by means of their role (s). Roles are modeled by the more general concept of profile, defines the relationship between users and views. So profiles can identify a role or an homogeneous class of users.

2.3 System functionalities

The basic system functionalities have been implemented using the entities defined in the previous section. This are the features that we think distinguish a virtual community environment.

Registration and users management

All users are registered to achieve member recognition in the community. So each user has a userid and a password to log in the system. S/he has also a form collecting some information that can help in identifying her/him, (i.e. a photo, age, job, hobbies) and that s/he can modifies when s/he wants. Main users categories, or profiles, in the technical sense, can be:

- visitors users: can read and write only to some areas and have not to log in;
- registered users: they are the common users and they can use all system features;
- moderator users: they manage one or more conferences;
- administrator users: they have administrative functions for the whole of the system or for a part of it.

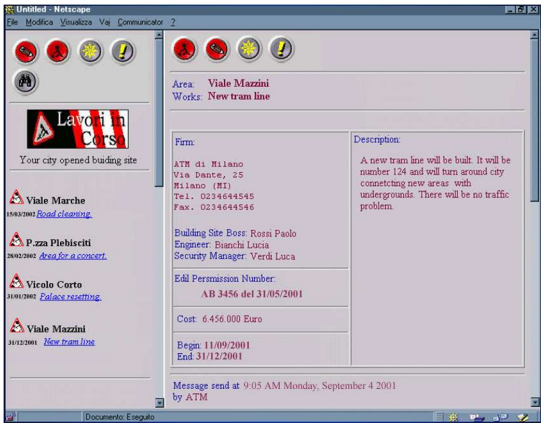

Fig. 2: Publishing view of the conference

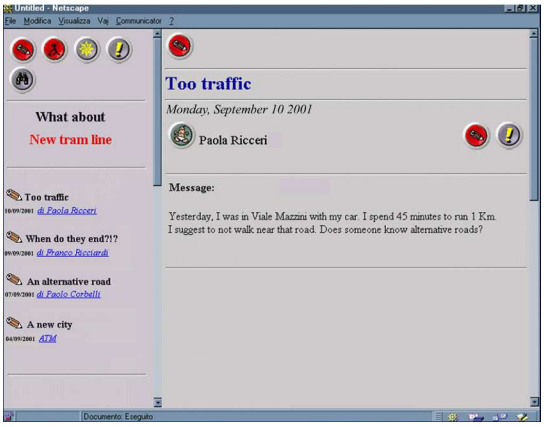

Fig. 3: Discussion view of the conference

Conferencing

According to the previous section' definitions, *conferencing* means all the features concerning content creation by users that can be achieved by sending messages to conferences.

Thanks to views every user can play actions only on messages of the types listed in the view, and permissions set what actions s/he can play.

This kind of conference management offers the following advantages:

- it is possible to set up private conferences, that is conferences which only a subset of users can access to, or conferences where few users can send messages to and the others can read published messages;
- it is possible to set up "publishing views", that is views that show only messages with a structure suited for information publishing;
- setting up a "discussion view" for the same conference, it is possible to make available comments sent by users about the above mentioned messages; according to the above definition, comments are simply messages with a structure where there is a text field;
- users can access different types of messages according to their role (profile);
- users can play different actions on the different types of messages according to their role (profile); in fact, we believe that a role is defined by the actions that a user can take, and by the message types s/he is allowed to manipulate.

Here is an example of how this kind of conference management is an effective way to represent information and discussion around a specific topic, preserving, at the same moment, information and discussion separate but in correlation.

The conference "Roadworks ahead" (Fig. 2) lists the road yards opened by the local public transport company. Information about every road yard is included in a structured message (right side of Fig. 2) and is summarised in the left side list, that shows essential information about (place, summary and date of end of work). This list can be considered the above mentioned publishing view. For the same conference, administrator can set up a discussion view (Fig. 3) that shows users comments about road yards. For this view, essential elements are subject, author and sent date. The two views set up for the same conference "Roadworks ahead", enable the reader to focus either on information matter (list of road yards) or on discussion one, with a tight correlation between both. This is made possible because road yard messages and comment messages belong to the same conference and therefore to the same topic.

To make this conference management versatile enough, it is necessary that the administrator can easily create new types of messages and new views. So a virtual community administrator has an enhanced interface which allows the undertaking of the tasks of creating and maintaining users, profiles, views and conferences.

Profiles and user *Desktops* management

Profiles are used to link a user to a set of views, to identify a role or an homogeneous class of users. In a possible management schema, every user has a personal profile, a generic profile, a moderator profile and one or more interest profiles. Moderator profile links a user with all the moderation views of the conference that s/he moderates: it is empty for the users with no conference to moderate.

Every user also has a *desktop*, so that when a user log in to the virtual community pages enters a costumized home page with her/his own elements, such as some interesting conferences for that user. The Desktop is simply a list of profile-view couples, that is, a list of views linked to the profile which gives access to those views.

Searching features

Searching features enable to search text either in the header of the message or in the body, where the header is the set of fields that are common to all messages and the body is the set of fields that change according to the message type. Search takes advantage of the message types structure because users can select the message type which restrict the search to, among the types available in the current view so that only those messages are examined. This possibility lets user address her/his search only to a category of messages, getting results nearer to her/his interests.

Hosting, private mail and calendaring

Our virtual community environment has also *hosting* (this word means that every user can create and manage her/his Web site on our virtual community server), private mail and calendaring features.

With regard to these features, it is important to point out the following two aspects:

- these features (that are very common on Web site managing virtual communities) are completely integrated in the system, with two main advantages; to make available more services in the same environment increases the users feeling of being a member of the community; the latter advantage is for administrators because it is often very difficult to integrate different tools that are currently available for implementing hosting, private mail or calendaring features;
- the above described conferencing system is very versatile and it has been able to give to the environment a good orthogonality. Mailbox, calendar and Web site are in fact implemented by special conferences with suited message types, views and permissions. This simply configuration let these conferences to carry out their own functionalities.

3. Architecture of the System

So far, the basic concepts underlying VIRTUOSE have been introduced. The architecture of VIRTUOSE closely implement them, providing a hierarchy of abstract machines which starts from a database management system and culminates in a graphical user interface for a Web browser.

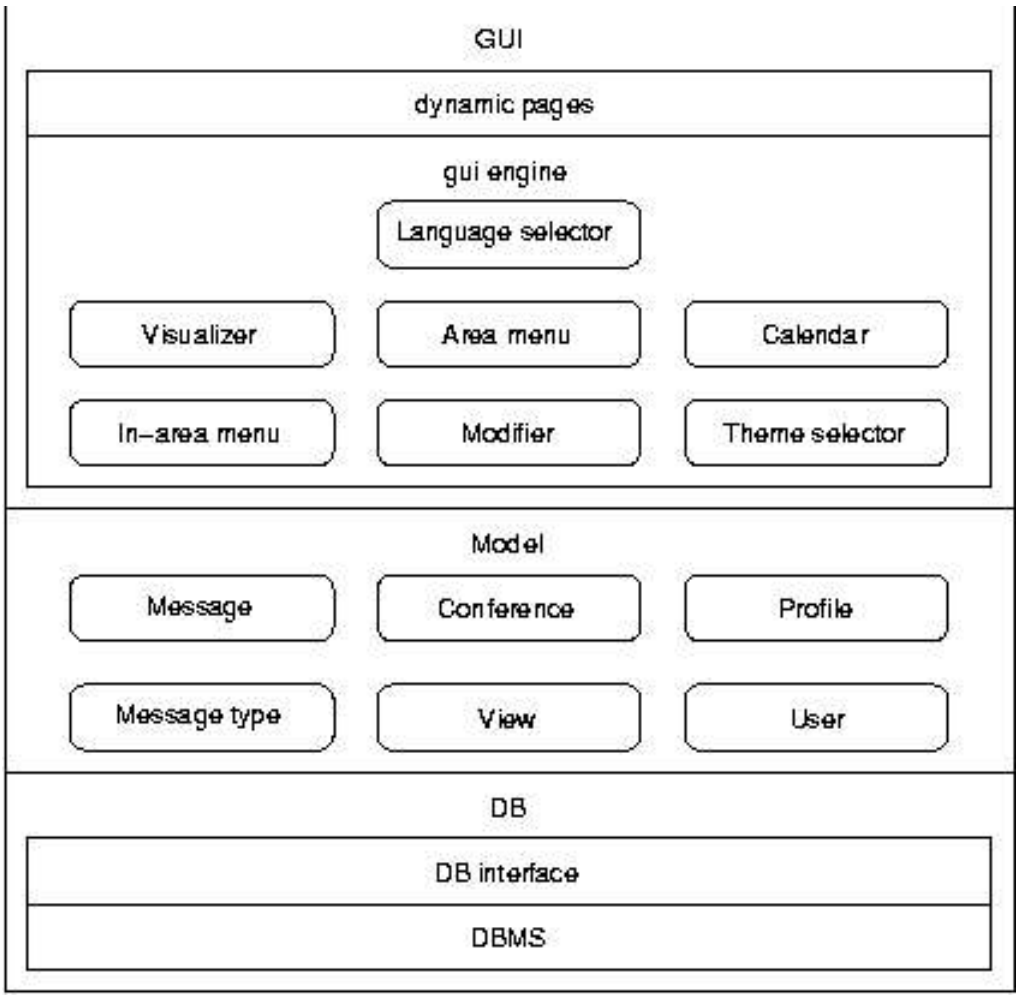

Fig. 4: The internal architecture of VIRTUOSE

As depicted in Fig. 4, there are three major levels in VIRTUOSE architecture: the database (DB), the model and the graphical user interface (GUI). Any level defines a proper set of notions which models the entities it operates on. So, a GUI entity may be reflected to the model level as a set of objects, which in turn, becomes a set of interrelated tuples in the DB. Functionally, any level which co-operates in this reflection of concepts adds features to the information it gathers from the lower levels.

More precisely, the DB level is the interface between the computer and VIRTUOSE. Its role is to represent the data that constitutes the virtual community and to give access to these pieces of information. The DB level has been divided into two sub-levels: the DBMS is implemented in SQL and its purpose is to store the information enforcing data integrity; the DB interface is implemented in PHP and its role is to be the gateway between the database and the rest of the system, abstracting from the specific representation of data and permitting a sort of dynamic behavior of database tables.

In fact, since messages have a structure that can be defined by VIRTUOSE administrator, they cannot be directly represented in the relational paradigm. So, the DB interface represents messages as constituted by two sections: the header, which lies in the corresponding database table, and the

body, that gets stored in a database table which depends on the type of the message. Since message types are not known a priori, the DB interface takes care of adding and removing the database tables that store message bodies, as part of the message type maintenance. In this sense, VIRTUOSE database is not relational, because it does not fit into the strict paradigm of the entity-relationship model, but it uses a sort of dynamic entity-relationship model where the addition and deletion of entities and relations is controlled by an event based rewriting system.

The model level is written in PHP and its purpose is to represent the view of the virtual community for the citizen who browses its content. In this respect, an object oriented approach has been adopted: every entity that plays a role in the semiotics of VIRTUOSE universe has been represented as an object. There is a one-to-one correspondence between objects and the entities and relations in the DB, so that all pieces of information are stored in the database, but the model provides mechanisms to enforce permissions, that is, who is allowed to do what.

In this sense, the model is a partial image of the current state of VIRTUOSE as seen by the citizen who is looking at the system. It contains the information that is being accessed, the permissions associated with the objects being manipulated and the location in the system. In other words, the model is the piece of VIRTUOSE which permits to navigate the virtual community and to control the access and the modifications to data.

The purpose of the GUI is to transform the view of the virtual community provided by the model into a suitable HTML presentation. The GUI level has been divided into two sub-levels, the GUI engine and the set of dynamic pages.

You may think to the GUI engine as the composer of the graphical entities that populate the browser window when a citizen visits the virtual community. The GUI engine provides the composing tools, a set of widgets, such as the calendar, the menus, etc., and a set of visualizers and modifiers to manipulate atomic pieces of information. The dynamic pages retrieve the information from the model and, using the GUI engine build what has to be sent to the citizen's browser.

The GUI engine filters the dynamic pages through a series of personalizations items, which determinate the final graphical layout in the form of an HTML page that is sent to the browser. The personalization takes place at three levels: first, the system administrator may define what is the structure of pages, that is, what widgets, what fields, what text and images will be included; second, the system manager defines the available themes and languages; third the citizen may choose a language and a theme, and this combination determines how the widgets are drawn, the color of the graphical elements, the language of the textual elements, the position of the graphical entities, and so on.

For instance, when a citizen view a message in a conference, a dynamic page is generated. That page is composed by some menus that show the possible actions (reply to the message, change conference, etc.) and a set of fields which constitutes the message. The dynamic page uses the GUI engine to draw the menu widgets and the visualizers to draw the fields according to their

MIME types. The content of the fields is gathered trough the model, which, in turn, checks that the user can read the message, and retrieves from the database the values of the fields which have to be shown according to the view the user has chosen. Every action (e.g. clicking on a menu item) is sent by the browser to the Web server which transmit it to the GUI engine that reacts according to what has been declared inside the code that generated the dynamic page.

From an abstract point of view, the path from the top level of the GUI down to the DB dismount information to obtain data, while the same path in the opposite way, reconstructs information from the data. The sophisticated architecture behind VIRTUOSE is necessary since it fills the gap between our view of a virtual community as a place where information and debate are the principal activities, and the computer view of a virtual community as a structured collection of data.

The layered structure of VIRTUOSE allows to separate tasks according to the sophistication of the entities they operate onto. For example, the notion of user in the DB level is just a row in a database table, while, at the model level, a user may be a citizen, a system manager or a guest, and the distinction can be reduced to what they are allowed to view and to do; at the GUI level, a user becomes a citizen which browses information and partecipates in debates.

From a technical point of view, there are many interesting points which can be discussed in the way we implemented the idea of virtual community in VIRTUOSE, but we prefer not to deepen the presentation since it will need the knowledge of the internals of the system, topic which is far beyond the scope of this article.

4. Aknowledgements

VIRTUOSE has been developed in the framework of the project “Development of services and technologies to turn local communities to account in the Information Society" funded by the Lombardy Regional Government and carried on by AIReC (the Association for Informatics and Community Networks).

5. References

[ADRS99] Aletti M., De Cindio F., Rossi G., Sonnante L., *A Web-based platform for third generation community networks*, presented at the ECSCW'99 Workshop on “Broadening Our Undestanding”, Copenhagen, September, 1999.

[Ale99] Aletti M., *Una piattaforma Web-based per reti civiche di terza generazione*, Master thesis, Department of Computer Science of the University of Milano, July 1999. (in italian)

[CS] CSuite Web site: http://csuite.ns.ca

[CW] *Conferencing on the Web* Web site: http://thinkofit.com/webconf/index.htm

[For01] Fornaciari D., *Studio di fattibilità di una piattaforma web-based per reti civiche in ambiente open source: realizzazione di un prototipo con Midgard*, Master thesis, Department of Computer Science of the University of Milano, February 2001. (in italian)

[HA97] Hagel J., Amstrong G. A., *Net Gain Expanding the Market through Virtual Communities*,

Harvard Business School Press, 1997.

[Kim00] Kim A. J., *Community Building on the Web*, Peach Pit Press, 2000.

[Mcm98] McMahon S., *Open Source Tools for Community Networks*, Cisler S. (ed.), “Technology Issue”, Community Networking, quarterly of Assoc. For Community Networking, December, 1998.

[MDG] Midgard Project Web site: http://www.midgard-project.org

[Mic00] Micelli S., *Imprese, reti e comunita’ virtuali*, Etas, 2000. (in italian)

[Pho] Phorum Web site: http://phorum.org

[Pya98] Pyatok M., Keynote Speach Abstract, proc. of the The Fifth Biennal Participatory Design Conference, Seattle (WA,USA) 1998.

[SF] SourceForge Web site: http://sourceforge.net/

[Vig] Vignette Corp. Web site: http://www.vignette.com