Virtuose, a VIRTual CommUnity Open Source Engine for integrating civic networks and digital cities

Marco Benini[1], Fiorella De Cindio[2], Leonardo Sonnante[3]

[1]Dipartimento di Informatica e Comunicazione, Università degli Studi dell'Insubria, Italy
[2]Dipartimento di Informatica e Comunicazione, Università degli Studi di Milano, Italy
[3]Fondazione RCM - Rete Civica di Milano, Italy
marco.benini@uninsubria.it, fiorella.decindio@unimi.it, leonardo.sonnante@rcm.inet.it

Abstract. This paper outlines the development of digital cities in Italy from 1994 to the present and then shows how this became the basis for the design principles of *Virtuose*. *Virtuose* may be termed *communityware*. It was conceived specifically for managing (local) virtual communities and drew inspiration from social as well as technological design concepts, using an open-environment philosophy. *Virtuose's* basic requirements and its essential design and implementation choices are presented, and a pragmatic example of its application is given. The paper also provides a specific description of how the messaging structure supports both publication and dialog.

1 Introduction

Digital cities emerged in Europe in 1994-95: The Amsterdam Digital City (DDS, *De Digitale Stad*) started in Spring 1994, the Milan Community Network (RCM, *Rete Civica di Milano*) in Autumn of the same year. The Bologna Iperbole experience became operational at the beginning of 1995. Several other digital cities then followed these early initiatives.

In Italy the phenomenon was rather explosive mainly because of its interplay with the political situation, which transported the country from the First to the Second Republic and led to the adoption of a new election laws for mayors and city councils. The Internet was seen as a way for promoting citizen participation in public affairs [Sch00] – now called *e-participation* – and for reinventing citizenship and democracy [Dec00] – now called *e-democracy*.

Inspired by these principles, several municipalities (in Italy and elsewhere in Europe) followed Bologna's example, which became well known after winning the Bangemann award. In other cases, the Milan experience, promoted by the university, provided the inspiration for civic networks that were started as grassroots initiatives by civil society. The distinction between institutional and grassroots initiatives in most cases led to a technological difference: Richer experiences directly promoted by the municipality adopted the Web as their network infrastructure, while grassroots initiatives with lower budgets often adopted cheaper BBS technology. However, despite these differences, all these early experiences were called *civic networks*: As [Mia02] describes, "the primary aim of early civic networks was promoting the publication of official documents and promoting citizen participation in the life of the

municipality". It is important to point out that publishing official documents was seen as a means of assuring transparency: Allowing citizens to access official documents is a prerequisite for their active participation.

In the same year, 1995, another significant digital city was started in Italy by the municipality of Turin. It was named differently – the Public Telematic Service – to stress the fact that, unlike other digital-city initiatives, its main emphasis was on providing online services to the local community: Citizens, as well as professionals and enterprises, could benefit from reducing interaction time with local government (for getting a certificate or a map, for paying taxes, etc.). An argument could be made that, in Italy, the Turin initiative started what we now call *e-government*. It is important to note that the promoters of the Turin Public Telematic Service insisted from the outset, as subsequent research confirmed, that e-democracy and e-government are not mutual exclusive, but rather complement and enrich each other. As a result, one phenomenon we now observe is that the use of the Net for publishing information and delivering online services overrode the role of the Net as an environment for participation. Over the course of the years from 1996-7 to the present, digital cities increasingly turned into the official websites of local government, providing information and transactional services. In most cases, the pioneering spirit of civic networks was lost. In Italy, this change is consistent with a corresponding mutation in the political climate wherein the participation issues of the second half of the nineties nearly disappeared. People were increasingly seen as *users* of ICT applications or as *consumers* of online services, although it ought to be borne in mind that citizens own *a sovereignty right* that should allow them to contribute to shaping the information society [DS03].

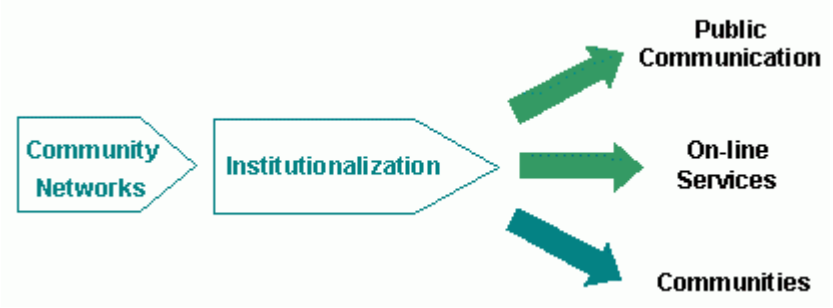

Fig. 1. The specialization of Digital Cities (source: Censis)

Censis, an outstanding Italian national research institute that has been tracking the evolution of digital cities in Italy since 1997 and publishing annual reports, depicts the increasing institutionalization of digital cities in its last two reports (see also [Mia02]). Censis shows that this process has now led to the specialization of digital cities, as shown in Figure 1 (from [Dom02], which anticipated [Cen03]). Different kinds of digital cities can actually be observed:

– digital cities that focus primarily on delivering public information, where the communication pattern is broadcasting from the public sector (which tends in-

creasingly to be not a single entity but a consortium of local authorities in the same area) to the citizens (and in general to the local community);
- digital cities (often called *city portals*) that focus mainly on delivering e-government services, where the communication pattern is two-way interaction according to a predetermined information flow;
- digital cities (often called *civic networks*) that aim to provide the local community with a communication environment for free dialog, where the communication pattern is the peer-to-peer conversation typical of virtual communities.

Censis selects paradigmatic examples for each of these types of community networks[1]; what is relevant for our purposes is that the most recent report [Cen03] strongly emphasizes the need to integrate the three approaches: a good digital city should provide well-integrated information and online services, both for residents and for those with occasional business or tourist interests in the city, and a good digital city should also provide a dialog-oriented communication environment (see Figure 2). Because the present scenario is e-government, i.e., the integration of information and on-line services, it is interesting to consider the reason Censis gives for expanding a city website to include community facilities: "To create online communities among the users of the local-government website means offering a chance to interact with local government in a different way, to recover and develop a sense of belonging and trust in the local and institutional context.".

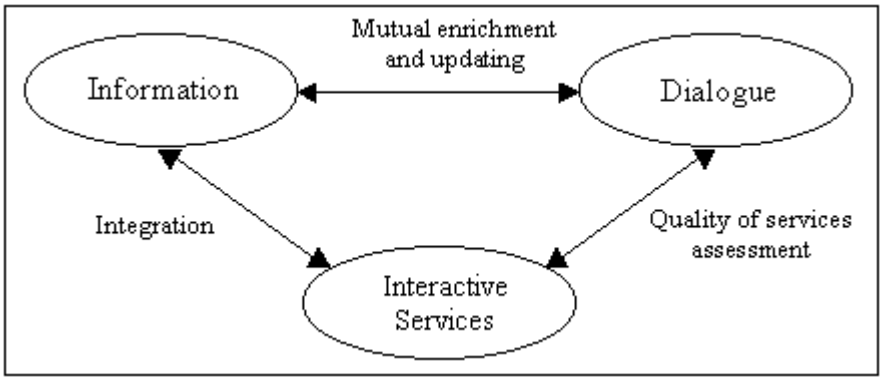

Fig. 2. The interplay between dialog, publication, and services

Because of the cumulative experience of having designed RCM in 1994 and managed it since then, we came to the same conclusion. Our perspective has always been that of a civic network trying to keep its original emphasis on citizens' and civil society's direct involvement. Indeed, in the framework described above, a civic network cannot merely offer an environment for dialog, but needs ways to extract information and knowledge from people's small talk. This is what spurred us to work on the technology that could be used for digital cities.

[1] In the Censis slides, the Milan Community Network (RCM) is presented as the paradigmatic case of the third kind.

2 Software for Community Networks

When we decided to start RCM in 1994, we had several design alternatives to consider, including the choice of software platform. The first natural option we entertained was *FreePort* software [FP], which was developed at the Case Western University to run the Cleveland Free Net and then became a sort of standard available very cheaply anyone wishing to start a community network. Unfortunately, *FreePort* adopted an outdated text-based interface, whereas the standard was already the window, icon-based graphical user interface.

The other option considered was to develop RCM on the Web, then in its early stages of development. However, we were forced to discard this option for two reasons. On the one hand, a large majority of people and policy makers in Italy and Europe considered e-mail rather than full Internet access the universal network service to be provided to everybody, see, e.g., [ABLM95]. On the other hand, the Web had been conceived for enhancing the publishing capabilities of the Internet – which was initially confined to the protocol known as FTP – and poorly supported the kind of peer-to-peer communication that, in contrast, typically takes place through email, mailing lists, forums, and chats. We feared that the shift from a communication platform such as that offered by BBSs to a publishing platform, as the Web was at the beginning (and long remained), might radically change the community network from being a local virtual community where everybody is considered an information provider to a rather standard medium for information broadcasting where the website owner is responsible for providing information to its customers.

Hence, our need was for software that was very easy to learn and to use, while providing standard BBS features. The choice thus fell to *FirstClass* (originally produced by SoftArc Inc., now by OpenText), which we knew as the platform used by the Open University. Unfortunately, *FirstClass* is proprietary software, and, over the years, we encountered the problems typical of using proprietary software, namely: cost, a closed data format, and a fixed, non-modifiable set of features.

As noted, RCM was a reference model for other, mainly grassroots, civic networks, mostly in the Lombardy region. Thanks to the support of the regional government, in 1996 Lombardy's civic networks joined to form the Association for Informatics and Civic Networking (AIReC Lombardia)[2]. The various members of AIReC made different choices for running the civic network. Some community networks adopted *Lotus Notes* because it was the intranet/extranet platform used by the host municipality. Others chose proprietary software quite similar to *FirstClass* called *Worldgroup*. A couple of them opted for Web-based, free software environments. But none of them was truly satisfied with its choice.

Therefore, after considering the *CSuite* software [Mcm98,CS] – an open-source platform adopted by several community networks in Canada – which did not meet our needs because of its character-based user interface, we started some preliminary analysis developed as part of a few master's theses. In 2000, AIReC obtained funding from the Region of Lombardy to undertake the development of a civic-network software platform whose distinguishing feature was to be the homogeneous coupling of both *information* publishing and *dialog* facilities. These facilities enrich informa-

[2] AIReC [AIR] groups the community networks located in Lombardy. It was established under the auspices of the Region of Lombardy and the Department of Computer Science of the University of Milano.

tion with community-generated knowledge (an expression we deem more expressive than the more common "content generated by users"). Moreover, the platform was to be open to integration with interactive services.

The rationale behind this basic design choice can be illustrated through the following example. Let's suppose Fiorella, a citizen, wants to know the list of movies showing in Milan this evening. She can look at the city website, which usually provides this information. However, by accessing the "Cinema" forum on the community network, she enriches this information with community knowledge that was created through discussion. For instance, about a couple of movies, her favorite opinion-maker in the community says: "Beautiful plot." At this point, she makes her choice and books a place in the nearest theater showing the selected movie. After viewing the movie, she comes back and comments on the movie or the state of the theater in the local forum, e.g., in terms of the fairness of the automatic booking procedure.

The same pattern can be applied to many situations, from the choice of a restaurant to the need for a clinical-analysis laboratory. In the latter case, comments following use of the service provide a powerful way for assessing Quality of Service[3]. Figure 2 illustrates the interplay between these elements, i.e., the positive cross-fertilization between the discussion facilities of a community network and the information and interactive-service facilities of the municipal website. This cross-fertilization can help overcome the increasing institutionalization of digital cities mentioned above, while enhancing the impact of the discussions carried on within the community.

The present paper, which significantly extends [BSD02a], discusses the result of the project, i.e., the *Virtuose* software, a *VIRTual CommUnity Open Source Engine*. In the framework of the technologies for digital cities envisaged in [Ish00], *Virtuose* can be seen as a "technology for public participation that supports both content creation and social interaction." Pursuing our driving idea, mentioned in [Ser00], that "*Les maison font la ville, mais les citoyens font la cité*," (J.J. Rousseau, *Du Contract Social*) *Virtuose* focuses on the basics of integrating information and communication, so that digital citizens [Sch02] can be information providers (rather than merely passive consumers of information provided by public- and private-sector organizations) and can actively discuss any issue related to city life.

Advanced graphics facilities such as the presentation of information based on interactive maps [BSGR00] are, of course, useful and effective for enhancing user interfaces. In the above examples, for instance, they might help indicate where, in town, movie theaters, restaurants or clinical-analysis laboratories are located. We actually experimented with such user interfaces in one of the pre-studies for *Virtuose* [ADRS99], although the version we present here concentrates on the kernel engine.

3 Our solution: *Virtuose*

This section is devoted to illustrating and discussing *Virtuose's* basic system requirements, its fundamental design choices, and how these choices have been trans-

[3] The need to assess the quality of public services and their acceptance through citizen feedback is explicitly referred to in Republic of Italy Law No. 150/2000, "Discipline of govemment information and communication activities."

lated into software architecture. Finally, this section analyzes an example of practical application.

3.1 Requirements

As discussed above, the requirements of *Virtuose* are both an abstraction and a synthesis of the needs that emerged from experience managing community networks in the Italian context. Such experience, where AIReC and its members (RCM, RecSando, RCL, etc.) played the leading role, came both from the citizen side and the administration side. Comments, suggestions, requests for improvements, and criticism were collected and evaluated. Thus, albeit indirectly, every category of community users, i.e., normal citizens, moderators, administrators, directors, etc., contributed to developing the specifications for *Virtuose* through experience accumulated over the years.

Consequently, *Virtuose* was designed on the basis of a social rather than technological perspective. This centered around two main activities: publications and dialog.

Both activities were modeled from the outset on a small and general set of simple concepts. The key idea, borrowed from most community-oriented applications [CS, MDG, Vig, Mcm98] is that the information unit is a *message*. Formally, a message is divided into a *header* and a *body*. Both are composed of *fields*, i.e. named and categorized pieces of data. The header contains information that identifies the originator, the destination, the type of content, and the subject, i.e., a textual description of the content. The body consists of the message content.

Every field has a name and a type. The whole message has a type which completely defines the types and names of the fields of its body. In a word, the message type identifies the *structure* of the message.

Unlike from most other applications, *Virtuose* allows message types to be dynamically defined. The community administrator, who maintains the community service, may define new message types at any time, without interrupting service.

Messages are grouped into *conferences*, which are structures defined by a name, and a list of messages, which may be threaded. Conferences can group messages by any criterion an administrator wishes to adopt. They might hold a set of messages on the same topic (e.g., a discussion list on football), messages from an institutional organization (e.g., the book records of a library), or more subtle groupings, like messages to a single community member (e.g., a mailbox). It is very important to note that limiting the conference to accept only messages of a fixed set of types enables the use of the conference concept to model most of the important tools employed in building community software. As a matter of fact, the personal website of a community member is nothing other than a named set of HTML pages, i.e., a set of messages of type "HTML page." Moreover, a calendar can be modeled as a conference where messages are appointments and notes. All these examples have been implemented in *Virtuose*, which supports discussion lists, as well as publication conferences, mailboxes, calendars, and user websites by means of the unified framework just described.

A fundamental aspect of social relations is the *role* of a person. Typically, the concept of role has been confused with the technical concept of *permission*, i.e., the ability to do something. This is only the half of a good technical rendering. Actually,

permissions determine what a community member can or cannot do on the information units, i.e., permissions define a community member's ability to manipulate messages, or, in a social view, to create publications or to take part in a discussion. But, equally fundamental in defining a role is how information is viewed. Depending on the role s/he plays, a community member may or may not view certain types of messages, and the way they are presented graphically may change. For example, in a discussion, the role of moderators is important. They filter messages by avoiding publishing those that may be offensive or irrelevant. When a community member plays the role of moderator, s/he should be able to view all the messages posted to a conference, and s/he may approve some of them and delete others. When the same community member wants to post a message, s/he assumes a different role, one that allows posting but excludes approving. Moreover, in the second role, only approved messages may be viewed.

Therefore, a strong requirement of *Virtuose's* specifications is to model roles in the way just explained. Every community member has an identity, and, thus, can assume a role that defines her or his degree of control when viewing a conference and determines the way messages in the conference are displayed.

A social role is more than just permissions and visual presentation. It may be interpreted as a way to group people, or, conversely, to group conferences. For example, community members interested in music may wear the role "music" to access thematic conferences, which are uninteresting for other members at that moment.

The main requirement of *Virtuose* is to implement conferences and roles as described above, that is, it must allow dynamic manipulation of message types. Every conference has to have a set of admissible message types, every user must have an identity, and every user may assume a role, both in the sense of group membership and in the sense of manipulation power. These requirements are referred to as *flexibility* in the management of the virtual community. The flexibility requirement calls for a non-traditional programming technique: having a small set of basic concepts, the message, its type, the conference, the user, and his or her roles, that can combine dynamically in any way possible. This requires an approach to development that is closer to an artificial intelligence product than to a traditional Web application.

Moreover, *Virtuose* had to be concept software, that is, a system developed specifically to test our ideas of what a virtual community should be and to check that our concepts, as stated above, can be effectively used for a faithful rendering of community life. For these reasons, *Virtuose* had to be compact and easy to modify and to manage, even at the expense of performance or range of features.

Nevertheless, many other features were included in the specifications for *Virtuose*, such as multilingual support, graphical themes, strict adherence to the open source paradigm, etc. Although these features are important, and their analysis might be of interest, we believe a discussion of them is beyond the scope of this article, because the core of *Virtuose* lies in the flexibility requirement, which represents the real novelty of this piece of software and, we hope, a significant contribution to the designers and developers of virtual communities.

3.2 Design and Implementation

Virtuose is a Web application written using PHP [PHP, LT02, RG00] as the server-side development language, PostgreSQL [PSQL, Mom00, Sti01] as the supporting

database management system, Apache [AP] as the Web-server engine, Javascript [Fla01] as the client-side scripting language, and HTML [HTM, CM00] as the presentation-markup language.

From an architectural point of view, *Virtuose* is fairly standard, adopting a variant of the classical three-layer structure [RG00] that divides an application's logic, presentation, and data. The architecture of *Virtuose* is depicted in Figure 3.

The main difference between *Virtuose* and similar applications [CS, Vig, Mcm98] lays in the flexibility required by design and the compactness of coding techniques. Both requirements were discussed in the previous section, but not from the point of view of technical implications.

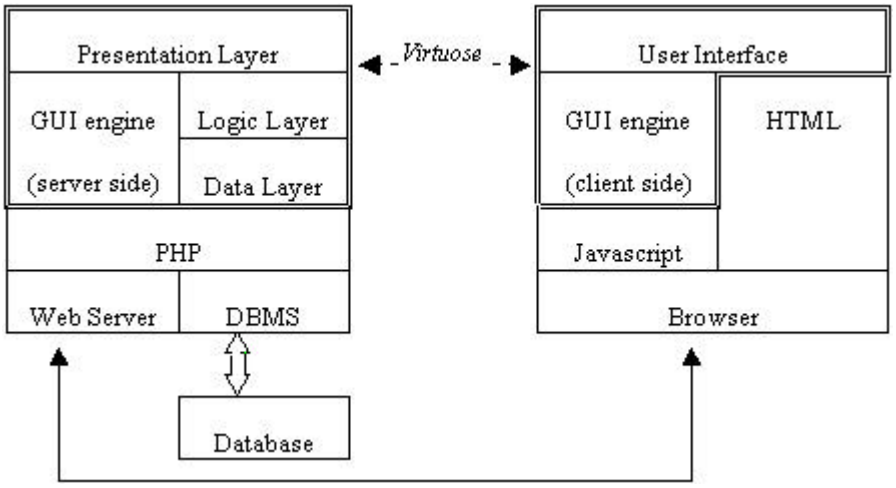

Fig. 3. The architecture of *Virtuose*

In fact, the GUI engine plays the role of an application server [Lea00], although it is really an application framework [RJB98], developed from scratch for the purposes of *Virtuose*.

The flexible nature of message types, which can be defined by the community administrator at any time, requires a dynamic approach in defining the database. Indeed, because messages are subject to searching, they should be represented as the rows of one or more interrelated tables in the database, the columns being message fields. However, fields depend on message types, which are stored in the database as well. Thus the database schema must change dynamically whenever a message type is added or deleted. Hence the data layer of *Virtuose* provides both an interface to the DBMS and the functions assigned to tracking the evolution of the database schema over time.

The flexibility requirement has its strong point in the concept of role, interpreted both as ability to manipulate and grouping. In order to implement this concept, two new entities were introduced in the design: *views* and *profiles*. The former renders the first meaning of the notion of role, while the latter is used for grouping. Their interaction defines what we call flexibility.

A profile is nothing other than a named set of users. A single user may appear in many profiles, that is, a single user may act as a member of many different groups.

A view is, as the name suggests, a way to look at and to interact with a conference. It is defined by a set of message types, that are allowed to appear in the conference, and, for each message type, a set of permissions and a set of graphical templates[4]. The only way to inspect a conference is through a view. The list of messages in the conference is thus filtered by the view, so to display only messages whose type appears in the view. Moreover, when a given individual message is to be displayed, the view permits display of its content only if the read permission for that type is true. Likewise, creating, deleting or modifying a message is allowed only if the view contains the appropriate permission for the message type involved in the operation.

When a message is to be displayed for a user, either as an element of the list of messages or as a single, complete message, the GUI engine selects the right template to use according to the message type. Moreover, the whole message list has its own template in the view structure. Therefore, the community administrator, by creating her or his own templates, may control the way information is presented with no constraints.

In Figure 4, a piece of the database schema is shown: It represents the main entities involved in rendering the concept of role, along with their relations.

A user may be associated with a potentially large set of profiles through the group relation. A profile provides access to a number of views by means of the access relation. A view displays the content of the associated conference by means of the show relation. Thus, a user may look at a conference by choosing a profile that provides access to a view for that conference. Consequently, the user may look at the same conference through many different views, depending on the profile he chooses. A virtual-community administrator may set access to conferences by defining views and by distributing their access to different profiles so as to partition groups of users based on their interests, their responsibilities, and so forth.

A view refers to a conference by means of the show relation. The message types it manipulates are the ones related to permissions by means of the visible relations. The logic layer assures that each visible relation is a subset of the admissible relations. A message has a type and is a member of a conference and is divided into two tables, the header and the body. The body table is dynamic, i.e., there is a distinct table for every message type, whose columns are exactly the records of the field table related to the type.

The logic layer defines the procedures to manipulate the concepts defined in the previous section, mapping them to the database schema, and assuring integrity of data semantics. The presentation layer provides the primitives to capture events from the client side of the application, and to generate presentation of the virtual community by means of HTML templates.

Actually, the picture of the insides of *Virtuose* presented so far is partial. Many entities in the database schema have been omitted for the sake of simplicity and a series

[4] Although, in principle, graphical templates in a view are grouped according to the message type they apply to, there are default templates for all messages, independent from their type, but specific to the view, and there are templates, associated with conferences, that are used as defaults for views that do not provide more specific templates. This template hierarchy is quite involved to describe, although it has been designed to be natural to work with. In the following, we adopt the simplified assumption that every message type has its own templates.

of important functions the logic layer provides have not been included in our description. The most complex piece of code in *Virtuose* is the GUI engine, which introduces many abstractions the need for which is apparent when the details of implementation are considered. As noted, although these considerations may be interesting, we believe that the important part of *Virtuose's* implementation lies in its dynamic-database message management and in the structure of views. For this reason we are limiting our discussion of the internal workings of *Virtuose* to this and must refer readers interested in other topics to the technical documentation [VIR].

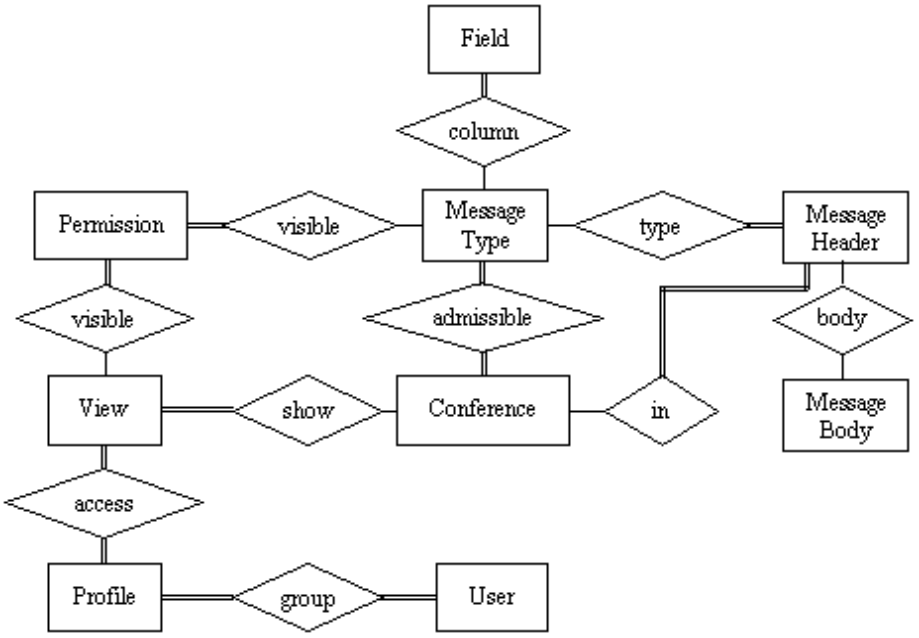

Fig. 4. The relevant part of the database schema

3.3 A pragmatic example

The goal of this section is to offer an example of how the described model of conference management is an effective way to represent publication and dialog around a specific topic, while preserving the separation between the two activities without losing their underlying relationship.

One topic commonly found on community networks consists of the cultural, social, and sporting events and the like taking place in the community network's geographic area. Let's see how the community manager can arrange a conference around this topic using the *Virtuose* platform.

S/he can create a suitable type of message named *event* whose fields are *title, date and place, description,* and *image* with the obvious meanings. This will be the "pub-

lication part" of the conference. S/he also wants to allow people to send comments to the published events as text or as related web links. Therefore, s/he creates a conference named *events* that accepts messages of type *event*, but also *comment* and *weblinks*. The latter is a message type with a structure suitable for inserting a list of URLs with their descriptions, while the former is a message type with a body that consists of a text field. This is the "dialog part" of the service. For each message type, the community manager may also write an HTML template for formatting field content for publication. For example, Figure 5 shows an event message, as it appears in the *Virtuose* interface.

Fig. 5. A message of type *event*.

As already explained, a conference can be accessed only through a view, so the next step is to create some views to allow users to insert events and comments. If the community manager decides to allow all community members to insert their events, s/he may want to choose some of them as moderators, i.e. those users who make events readable by others. No matter what s/he decides, s/he needs a view that grants every user who can access the conference the right to write events in it. If the same view also gives the right to write comments, we might have a configuration like the one shown in Figure 6. On the left side of the page is the list of views, grouped by the profiles the user belongs to. Clicking on a view brings up the list of messages filtered by that view on the right side of the page.

In the list of messages, we can emphasize the differences among message types thanks to different background colors or other visualization clues. We are also able to display different fields of the message: date and place for event messages, author for comments and weblinks. When a community member wants to write a message, s/he has to choose the type of the message s/he is going to create, thus providing implicit information about the semantics of what s/he is going to write. This information can be very useful for search features: If someone is interested in related web sites, he can focus his search on weblinks messages, ignoring all comments and events.

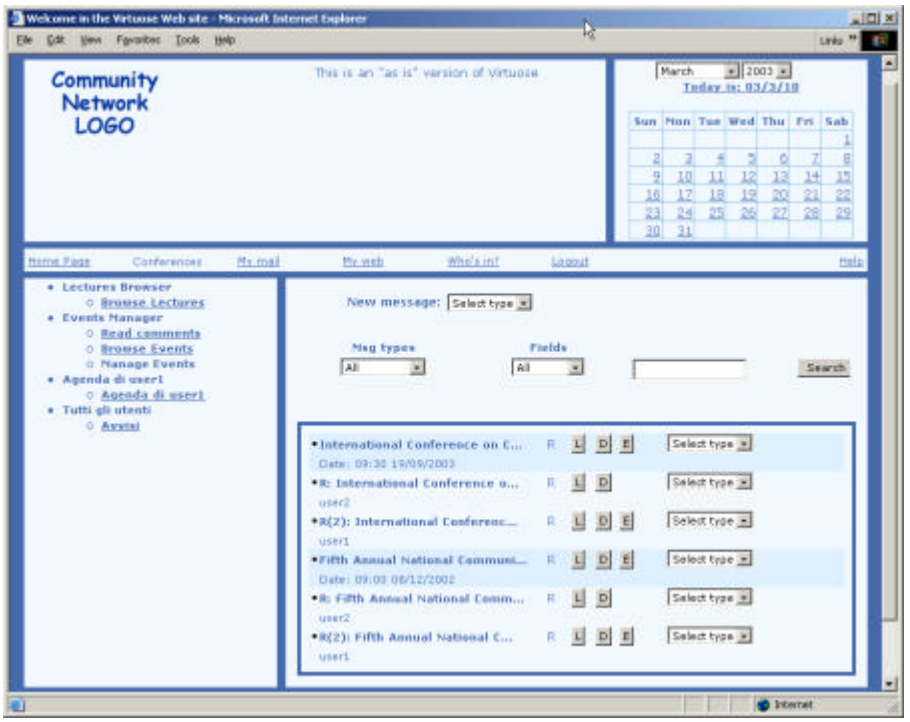

Fig. 6. The list of messages of the Events conference

It will now be readily apparent to the reader that in order to allow distinct access to events and to comments, the community manager need only create two views, with their appropriate permissions, one for events and one for the comment and weblink message types, see, e.g., Figure 7. These two views enable the reader to focus either on matters of information (the list of events) or on discussion, with a tight correlation between them made possible by belonging to the same conference and, therefore, to the same topic.

Of course, given the flexible nature of *Virtuose*, any other policy for defining views may be implemented in similar fashion.

4 Conclusion and Discussion

As we said in the Introduction, the idea and the development of *Virtuose* was basically driven by the need of a software for managing online communities homogeneously coupling the publishing of *information*, and facilities for *dialog*, which enriches information with community-generated knowledge. More in general, this can be seen as the kernel of a software platform for virtual communities. The need for *communityware* was confirmed after we started to develop *Virtuose*.

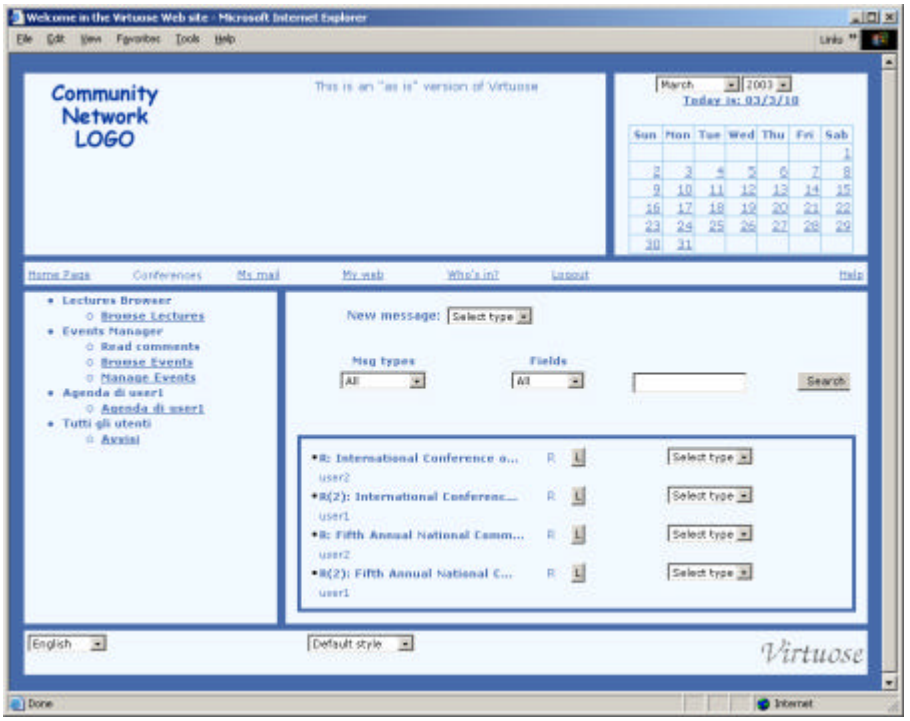

Fig. 7. The read comments view

In March 2001, Etienne Wenger, published on the Web a detailed report surveying community-oriented technologies for supporting communities of practice [Wen01]. He considered and classified a large number of applications from different fields (knowledge management, online conversations, community-oriented e-learning spaces, website communities, and others). None of them, save one, falls into the intersection that represents software providing to a satisfactory extent the features needed to manage a community of practice. According to Wenger's survey, the only software that approximates the requirements is *Communispace* [Com], which is, however, more a service than a software product, as it is available only on an ASP basis at relatively expensive rates. Although the emphasis of Wenger's survey is slightly different from ours, since he focuses on "a technological platform to support

communities of practice across a large organization," the result of his survey confirms that "the ideal system for a general platform for communities of practice still does not really exist."

In Spring 2002, the American Association For Community Networking raised a call for software to manage community networks, promising an award of $2,000 to the winner. Most of the required features were included in *Virtuose*, although the general architecture was quite different. Indeed, the call required collecting in a common framework the standard open-source applications that support the required features (e.g., *PHORUM* for managing forums or *Mailman* for mailing lists). Nevertheless, we decided to submit the first version of *Virtuose* and its associated documentation [BDS02b]. It was surprising to find that *Virtuose* was the only submission, and gratifying to receive, from the promoters of the award, after they had tested the software, a $500 premium to encourage its further development.

Finally, it is worth mentioning that, while we were developing *Virtuose*, a somewhat similar technology appeared, i.e., Internet weblogs, which have now become quite popular. The similarity lies in the fact that weblogs allow generic users to easily publish their own content and to enrich it through readers' comments in threads of discussion. In this respect, weblogs share *Virtuose's* purpose of integrating publishing and discussion. We are experimentally testing a couple of weblog environments as satellites of RCM. This early experience points up the main differences between weblogs and *Virtuose*, namely:

1. Each weblog contains one specific pattern of publishing and discussing ("news" and "forum" sections), while *Virtuose* gives its administrators the flexibility to design their own patterns of integration between information and discussion;
2. Weblogs are much more individual tools, while *Virtuose* is conceived to support communities.

From a more technical point of view, *Virtuose* has been the occasion to develop a solution for managing dynamic databases by significantly extending standard database technology. Although this feature is one of the reasons why the present prototypical implementation suffers from inefficiency, it also proves that *Virtuose* provides much more flexibility than the other similar applications. This is to say that *Virtuose*, in its present state of development, as it will be for the near future as well, is an open framework where ideas and techniques involving communities can be tested. For this reason it can be freely downloaded (www.virtuose.it).

We are developing an XML-based implementation that keeps this flexibility while improving performance. This technological solution was discarded in the current version because of the lack of tools available at that moment in the design phase. Because some apparently mature tools, e.g., the Tamino XML database management system [Tam], have since been brought to market, we are now able to reconsider the choice. This opens up the opportunity to relate *Virtuose's* open architecture more deeply to the concepts of Semantic Web [BLHL01].

5 Acknowledgments

Virtuose was mainly 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 Region of Lombardy and carried out by AIReC (the Association for Informatics and Community Networks) and its members. Subsequent improvements were partially supported by the Italian MIUR (FIRB "Web-Minds" project).

References

[ABLM95] Anderson R.H., Bikson T.K., Law S.A., Mitchell B.M, *Universal Access to E-mail: Feasibility and Social Implications*, Rand, Santa Monica, CA, 1995.

[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 Understanding", Copenhagen, September, 1999.

[AIR] AIReC Web site: http://www.airec.it.

[AP] Apache Web site: http://www.apache.org.

[BDS02a] Benini M., De Cindio F., Sonnante L., *VIRTUOSE: a VIRTual CommUnity Open Source Engine*, Proc. DIAC-02 Symposium "Shaping the Network Society: Patterns for Participation, Action and Change", Carveth R., Kretchmer S., Schuler D. (eds.), CPSR, pp. 40-43, Seattle, WA, May 2002.

[BDS02b] De Cindio F., Benini M., Sonnante L., *VIRTUOSE White Paper*, A.I.Re.C. Report, February 2002.

[BLHL01] Berners-Lee T., Hendler J., Lassila O., *The Semantic Web*, Scientific American, May 2001.

[BSGR00] Bolatto G., Sozza A., Gauna I., Rusconi M., *The Geographic Information Systems (GIS) of Turin Municipality*, in [II00]

[Cen03] Censis ed., *7th Report on Digital Cities in Italy*, February 2003. A summary is available at http://www.censis.it/censis/ricerc.html. (italian)

[CM00] Connolly D., Masinter L., *The 'text/html' Media Type*, RFC2854, June 2000.

[Com] Communispace Web site: http://www.communispace.com.

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

[Dec00] De Cindio F., *Community Networks for Reinventing Citizenship and Democracy*, in M. Gurstein (ed.), *Community Informatics: Enabling Communities with Information and Communications Technologies*, Idea Publ. Group, Hershey (USA), 2000.

[Dom02] Dominici G., *Città digitali: quale limite alla partecipazione?*, presented at the Fifth National Meeting of Civic Networks, COMPA Exhibition, Bologna, September 2002. (in italian)

[DS03] P. Day and D. Schuler (eds.), *Shaping the Network Society*, The MIT Press (forthcoming).

[Fla01] Flanagan D., *Javascript: The Definitive Guide*, 4th edition, O'Reilly & Associates, 2001.

[FP] Case Western Reserve University, Free Port Version 2.3: product overview, freeport-info@po.cwru.edu.

[HTM] World Wide Web Consortium, *HTML 4.01 Specification*, 1999, available at http://www.w3.org/TR/1999/REC-html401-19991224.

[II00] Ishida T., Isbister K. (eds.), *Digital Cities: Experiences , Technologies and Future Perspectives*, Lect.Notes in Comp.Sc. 1765, Springer-Verlag, 2000.

[Ish00] Ishida T., *Understanding Digital Cities*, in [II00].

[Lea00] Leander R., *Building Application Servers*, Cambridge University Press, June 2000.

[LT02] Lerdorf R., Tatroe K., *Programming PHP*, O'Reilly & Associates, 2002.

[Mcm98] McMahon S., *Open Source Tools for Community Networks*, Cisler S. (ed.), "Technology Issue", Community Networking, Quarterly of Association For Community Networking, December, 1998.

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

[Mia02] Miani M., *The Institutionalization of Civic Networks: the Case of Italian Digital Cities*, presented at the Second Euricom Colloquium *Electronic Networks & Democracy*, University of Nijmegen, October 2002.
[Mom00] Momjian B., *PostreSQL, Introduction and Concepts*, Addison-Wesley, 2000.
[PHP] PHP Web site: http//www.php.net.
[PSQL] PostgreSQL Web site: http://www.postgresql.org.
[RG00] Ratschiller T., Gerken T., *Web Application Development with PHP 4.0*, New Riders Publishing, 2000.
[RJB98] Rumbaugh J., Jacobson I., Booch G., *The Unified Modelling Language Reference Manual*, Addison-Wesley, 1998.
[Sch00] Schuler D., *New Communities and New Community Networks*, in M. Gurstein (ed.), *Community Informatics: Enabling Communities with Information and Communications Technologies*, Idea Publ. Group, Hershey (USA), 2000.
[Sch02] Schuler D., Digital Cities and Digital Citizens, in M. Tanabe, P. van der Besselar and T. Ishida (eds.), Digital Cities II: Computational and Sociological Approaches, Lect.Notes in Comp.Sc. 2362, Sprinfger-Verlag, 2002.
[Ser00] Serra A., *Next Generation Community Networking: Futures for Digital Cities*, in [II00].
[Sti01] Stinson B., *PostgreSQL Essential Reference*, New Riders Publishing, 2001.
[Tam] Tamino Web Site: http://www.softwareag.com/tamino/.
[Vig] Vignette Corporation Web site: http://www.vignette.com.
[VIR] *Virtuose* Web Site: http://www.virtuose.it.
[Wen01] Wenger E., *Supporting Communities of Practice: a Survey of Community-Oriented Technologies*, ver. 1.3, March 2001. Available at http://www.ewenger.com/ewbooks.html.