Conference Information System - Analysis Design and Implementation

Master thesis

Katinka Kromwijk
<katinka.kromwijk@gmail.com>

Prof. Dr. Moira C. Norrie
Stefania Leone, Beat Signer

Global Information Systems Group
Institute of Information Systems
Department of Computer Science

15th March 2009

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

globis

Abstract

Conferences are an important tool for researchers to keep up to date with new developments in their research field. A major goal of attending a conference is to gather new information. We identified that information collected at conferences is either scientific or complementary information. Scientific information includes any information that is directly related to the conference and the conference topic including latest research results. On the other hand, complementary information is obtained in the context of a conference, without directly related to the scientific research tracks. For example, the extension of a participant's social network can be seen as a form of complementary information. The increasing availability of personal devices, as well as the popularity of the Internet have changed the way we gather and distribute information. While a few years ago information was mostly available on paper, nowadays there is an increasing amount of digital information that needs to be searched and stored.

The goal of this thesis was to identify the needs of conference participants to manage information before, during and after attending a conference. We present research on currently existing systems and conducted an online survey to identify specific needs of conference participants. Based on these investigations we designed and implemented the conference information system that is presented in this thesis. The presented solution enables the management of any conference-related resources - be it scientific or complementary information.

Contents

1 Introduction

The definition given by the Cambridge dictionary of a conference is *'an event, sometimes lasting a few days, at which there are a group of talks on a particular subject, or a meeting in which especially business matters are discussed formally'*. For scientists, it is an opportunity to meet fellow scientists interested in a specific topic, as well as an opportunity to present their work. Furthermore, they allow scientists to stay up-to-date on recent developments in their area or research. The importance of conferences varies from one research field to another. We are going to concentrate on academic computer science conferences. Indeed, for computer scientists, conferences are the most important way for getting their work published. Generally, the conference committee starts by publishing a call for papers for a conference on a specific topic. All researchers that feel appealed may submit their work for the conference. If a paper gets accepted, the work will be presented during the conference. Generally, at least one of the authors will need to attend the conference and present their work.

Conferences allow conference participants to gather a lot of new knowledge in a specific field in a short period of time. The two main types of information that can be gathered are scientific and complementary information. Scientific information is all information that is specifically related to the conference topic, such as research results presented during sessions. As opposed to scientific information, complementary information is independent of the conference topic. Complementary information includes the extension of ones social network, for example. Not only do conferences provide a broad spectrum of information types, this information also comes from a range of different sources. The two main information sources are the conference organizers and the fellow participants.

We have seen that conference participants gather scientific and complementary information. This classification is not sufficient to effectively describe the information gathering process during a conference. Another important classification is the timing at which the information is collected. It is possible to recognize three phases during which participants gather information. The first phase is before a conference where the participants have access to general information such as the location of the conference. Once the conference started, there is a second phase lasting throughout the whole conference during which most information, be it scientific or complementary, is gathered. During a conference, participants attend talks, participate in social events, etc. Scientific information is mostly gathered during the talks where fellow researchers present their research results, whereas the main source for complementary information are the social interactions between conference participants. Finally, the last phase is after the conference where participants might have access to information such as the pictures of the conference. Even after the conference is over, new scientific information might be gathered when reading the proceedings, for example. During the after conference phase, complementary information such as the pictures might be collected.

In this thesis, we have defined ‘conference information systems’ as systems that manage information that is related to a conference. This term refers to any system ranging from a sheet of paper with the schedule to a complex system that provides conference-related information on personal devices. Traditionally, conference organizers would hand out a big binder containing all necessary information such as the schedule, the proceedings, etc. Currently, some still do this, but there are more and more complementary digital systems such as websites. We can see that there is an evolution in the information channels. Indeed, people carry more and more personal devices with them and the way they gather and store information is changing. Although paper is still present, more and more digital information is available. This ranges from an e-mail to a forum or online pictures. The challenge then becomes to effectively make use of these different channels. With information coming from such a variety of sources, it might be hard to keep it all organized, and participants might quickly get overwhelmed. It can seen that because of the various sources and channels, conference information systems are often very distributed in terms of channels, means and media.

1.1 Goal of the thesis

This thesis attempts to address the issues described in previous paragraphs. In order to do this, it is split up into two phases. First, an analysis of the current existing applications and user needs for conference information management has been made and second, the design and implementation of a conference information management application based on the previous research which will be tested during a conference in July 2009.

Analysis We first started by a literature overview in this field. The literature provides a good overview of the current systems. Based on this, we identified the needs of conference participants in order to develop our own system. As a first step, we conducted an interview with a professional who has been involved in the organization of a conference where a conference information management system was tested. After this, we had a group discussion and as a final step, we conducted an online survey which was filled in by about a hundred and ten computer science professionals.

Design and implementation After having identified the user needs, we implemented our own solution: a conference networking site that should accompany conference participants through all the conferences they attend. The idea behind such a system is that a participant can use it during the three phases of a conference experience i.e. before, during and after the conference. Furthermore, the use of the system is not limited to one conference, but accompanies conference participants during all conferences they attend as it is possible to subscribe to all conferences that have been entered on the networking site.

1.2 Thesis structure

The thesis starts with a chapter giving an overview about the state-of-the-art of conference information systems in literature. This chapter is followed by a few possible scenarios involving different types of conference information systems. This goal of this chapter is to illustrate what could be possible with the current technology, but also place new ideas in a context. At this point, the analysis of the current situation has been covered and we start to identify the needs of potentials users of our conference information system in chapter 4. We then define the requirements of our system in chapter 5. Chapter 6 describes the architecture and the implementation of our application and the thesis ends with future work and some conclusions.

2 Related work

2.1 Overview

Currently, when attending a conference, participants are given a bag containing a lot of information they might need during the conference. The bag contains information about the conference itself such as the proceedings book and CD, the program booklet or the location of an information point. Furthermore, a name badge with the participants names and affiliation, instructions on how to access the WIFI and information and tickets for social events such as the reception and the conference dinner might be included. Generally, conference attendees are also equipped with their personal device such as mobile phones or laptops. Starting from this general scenario, this chapter will try to give an overview on what systems have been designed in order to support the access and distribution of this information. In the following chapter, we will refer to these systems as conference information systems. A conference information system is any system that manages all or some of the information needed during a conference. This information could be in any form. As an illustration, a big binder containing information about a conference is already a conference information system and so is a complex system based on mobile devices.

The main goal of this chapter is to give an overview of the systems that have been developed either in research or in industry, in order to try to enhance the access and distribution of information during a conference. Theses systems are as diverse as the problems they address, therefore, different groups of enhancements to current conference information systems will be treated instead of looking at each solution individually. These enhancements are grouped into the types of information they provide. First, all the systems providing non-personalized information will be looked at, followed by systems providing personalized information. Non-personalized information is information that is the same for all or most of the participants. This can be a static map, a schedule or information about a speaker. In contrast to non-personalized information, personalized information is information that has been personalized to a specific user by taking in some user parameters. These parameters could be, for example, the current time or the current location of the participants. A more detailed classification can be seen on Figure 2.1. The Figure summarizes the classification by showing examples of information that can be found in each category.

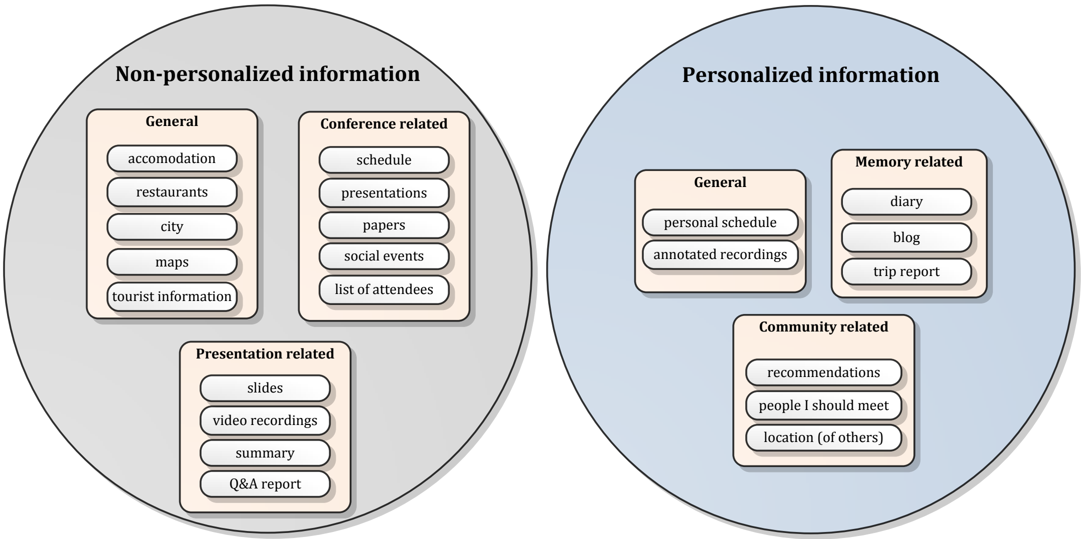


Figure 2.1: Information classification diagram

The systems that will be presented later in this chapter are not only different in the type of information they provide, they have also been developed for different purposes. In addition, some systems are purely conceptual while others have been mainly developed for research purposes. Finally, there are also a few commercial systems available showing that there is a real need for these technologies.

One of the main drivers for the research in this topic is the problem of information overload. While this problem is not limited to conferences only, conference participants are particularly affected by this as the information gathering process happens only during a short period of time. For example, when attending a conference, participants generally have a lot of activities planned on a single day and meet many new people. Chances are that all this information is very quickly forgotten. While many solutions we will discuss in this chapter address this issue implicitly, a section in this chapter is dedicated to systems whose main goal is to help people remember everything about their conference experience.

Previously, we have seen that conference information systems functionalities can be classified by the type of information they provide being it general or personal. However, an important aspect of these systems is that they mostly will try to replace the traditional paper-based support. Therefore, it is also relevant to look at the channels these systems use for the information distribution. This classification can be seen in table 2.1. In this table there are two types of devices mentioned: personal and non-personal. Personal devices are devices that participants

Personal devices only	[14, 20, 18, 1, 2]	[19, 25, 12, 3]	
Personal devices in combination with non-personal devices	[27]	[32, 9, 11, 13]	[10]
Non-personal devices only	[28, 16, 15]	[22, 21]	[7, 17]
	Personalized information	**Both personalized and non-personalized information**	**Non-personalized information**

Table 2.1: Classification personal / non-personal devices

carry with them and that are intended for their personal use only. These devices are mainly mobile devices as well as name badges. A more detailed description of each of the output channels can be found in the next section. Non-personal devices are the devices that are accessible by a group of people. This can be a kiosk, a public display as well as the web. As can be seen in the upper right field in table 2.1, the papers do not present systems providing non-personalized information on personal devices. This does not imply that such systems do not exist, on the contrary, the traditional paper documentation would be an example of such a system.

2.2 Channels

The conference information systems we will discuss in this chapter have a big variety in output channels. This small section will give a small overview on each of them.

Information kiosk An information kiosk, also called an information booth, is generally a stationary computer where information can be looked up. Such a device can be used in many contexts, it could be a computer, at the airport, showing the available hotels or a computer in a shopping mall showing the next event and the maps of the mall. In the context of conferences, this type of device could be used to allow an easy access to schedule information, maps or even the Internet. Typically, these devices provide access to general information as they are shared by many users.

Public displays An alternative to information kiosks are public displays. These, generally big, screens are put up somewhere so they can be seen by a maximum number of people. The screens can have an implicit or explicit external user input. For example, it could be a touch screen, and it could also include a camera so that it reacts to users upon recognition of their face. A sub-category of public displays are proactive displays. Proactive displays are displays that are augmented with sensors. An example of two people communicating with such a proactive display can be seen on figure 2.2. Similarly to the information kiosks, these devices are often used for displaying general information. However, in the case of proactive displays, they mostly provide personalized information.

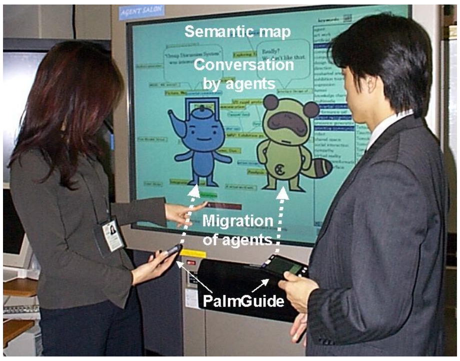


Figure 2.2: AgentSalon: a proactive display enhancing human communication

Mobile devices Whenever mentioning the term mobile devices, we consider them to be any device users can carry with them ranging from a cell phone to a laptop. We also assume that these devices are personal, meaning that they are used only by one person for the length of the conference, at least. Generally, these devices have a small screen and use a considerable amount of energy meaning that power supply must be available.

Name badges Name badges are the badges that are generally distributed to all conference participants and that serve as an identification. These badges are normally made of paper, but later in this chapter we will see that some systems are based on electronic badges. These electronic badges can enhance the traditional badges in many ways. For example, such a badge could just contain some extra leds, but it could also incorporate a touch screen and / or keyboard. The traditional name badge are used to display personalized information, and so do most of the new electronic badges.

Augmented paper Many solutions we will present in this chapter act as a replacement for the traditional paper-based information. It is up to the conference organizers whether they want to provide the paper information as well or not at all. However, people find paper convenient for many reasons: the display quality and size is better than that of mobile devices, it can be viewed by many people at the same time, etc. Of course, it also has disadvantages as it is hard to find supplemental information for a paper-based document. Therefore, a totally different approach to providing information is the use of augmented paper. Augmented paper is actually normal paper on which a pattern of small dots is printed. With the help of a special pen that is equipped with a camera, it is possible to track the pen's location at all time. As an example, we can have a look at the EdFest [8] application. This application was developed for a festival called Fringe. Fringe has many shows that are all described in a program booklet. The idea of this project is to augment the booklet with extra functionality. Figure 2.3 shows what the booklet looks like. When users want extra information about an event, they can "click" on a paper button and get extra information through an audio interface. It is also possible to write comments about shows or to do ticket reservations. Although this system was not developed for conferences, it is relevant as the paper information needed during a conference is very similar to the one during such a festival. The shows of the festival, could be associated to the presentations, workshops, demonstrations and other sessions of a conference. The EdFest system also provided a way to provide location with the help of a GPS as well as information about the city itself.

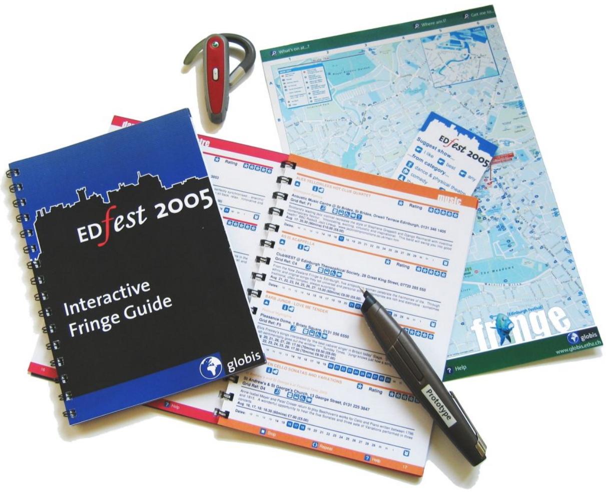


Figure 2.3: The EdFest interface

2.3 Non personalized information

In this section, we are going to look at systems that provide access to non-personalized information. The general group of non-personalized information is split into three sub-groups: general, conference-related and presentation-related. The next few sections each cover one of these groups.

2.3.1 General information

When attending a conference, the basic information conference participants need is general information such as how to get there, the accommodation, the city in general, etc. This information is relevant before and during the conference and is therefore often available on the conference website before the conference starts, but might also be distributed on paper at the beginning of the conference. The next few systems show alternatives to this traditional way of handling general information. The solutions are in two big groups: solutions based on information kiosks and solutions based on mobile devices.

Information kiosks The first group of solutions is based on information kiosks. In the context of a conference, these kiosks can be used to display information about the host city, the restaurants that are close by as well as some maps. One of these systems is the CHI'89 InfoBooth [28]. The CHI'89 InfoBooth is an information kiosk which provides access to information about presentations, fellow attendees and the city where the conference is held. The CHI'89 InfoBooth was a project on its own, but many other solutions provide such information kiosks not as the main information system, but as part of the general project infrastructure [32, 9, 13]. As an example, we can look at the ™Intellibadge system [13], that provided information kiosks allowing to search for a restaurant based on several criteria such as the type of food, distance, etc. Although these information kiosks were part of the ™Intellibadge system, the main channel of this system is an electronic badge. For conference participants, it is very convenient to have this kind of information made available and filtered especially for the conference, but the same information can probably also be found on the Internet. Therefore, some kiosks do not provide the information as such, but allow the users to access the Internet and thereby give them access to all the information they need, providing they take the time to look for it.

Mobile devices As technology improves, it becomes possible to create mobile devices that are cheap enough to loan one to each participant. These mobile devices can integrate either a web access or the information as such like on the CHI'89 InfoBooth [28] we saw previously. As an example, the ICMAS'98 mobile assistant project [25] provides information such as shops and sightseeing spots on a mobile device. This system was developed 10 years ago and a picture of the device that was used can be found on figure 2.11.

The previously presented solutions are already quite old. We believe that the lack of new development in this area is due to the increasing popularity of personal devices and wireless Internet access. Indeed, most conferences now make this kind of information available on their web page so that users can access it from anywhere through the personal devices, whether this device is a big computer or a small phone.

2.3.2 Conference-related information

Conference-related information includes all information about schedules, presentations, as well as about the papers that will be presented and their authors. This information is relevant before and during the conference, but might also be available after the conference for reference. Currently, most conference organizers put this information on their websites. This website is already available before the conference and is mostly maintained after the conference [13, 15] thereby making it accessible to everybody during the conference, providing they are in the possession of an Internet access. The difference with the general information described in the preceding section is that, while general information is mainly static, conference-related information might be dynamic. For example, a last-minute schedule change will not be reflected on paper, but on the electronic devices mentionned earlier it will. A second challenge with such changes is that they might occur at any time and it is not possible to expect participants to go and check regularly whether their schedule has been changed. As a result, some systems still provide access through information kiosks [28] and static access on mobile devices [19], but other systems using big displays [10], chat channels [21] or alerts on mobile devices [25, 12] try to provide more dynamic information where changes can be easily noticed.

2.3.3 Presentation-related information

The last type of non-personalized information is information related directly to the presentations themselves. This includes slides, videos, question and answer session reports, summaries, etc. This information is normally only available after the presentation has been given and is relevant during and after the conference. During a presentation, most people take notes that are in direct relation with what is being displayed by the presenter as well as his annotations. A first class of solutions tries to help the participants by providing them with live access to this data. There are many of these systems, but as they have been mainly developed for class rooms, we will have a look at just one: CORAO. [11]. COROA consists of an interactive whiteboard displaying the slides and on which notes can be taken by the speaker. Figure 2.4 shows this whiteboard. At the same time, the participants have a mobile device where the live version of the slides are displayed. The participants can take personal notes on their own versions of the slides. This project has not yet been tested. A similar, but more complete solution is the conference assistant [12]. The conference assistant is a wearable computer distributed to all users. This wearable computer displays information about the current presentation (whether the presentation is recorded, the name of the speaker, etc.) as well as the current slide. Additionally, users can annotate the slides and take control over the interactive whiteboard when they are asking a question. These devices can also be used to show extra information during demonstrations. When a question is being asked, the speaker has access to the identity information of those participants that are asking the question.

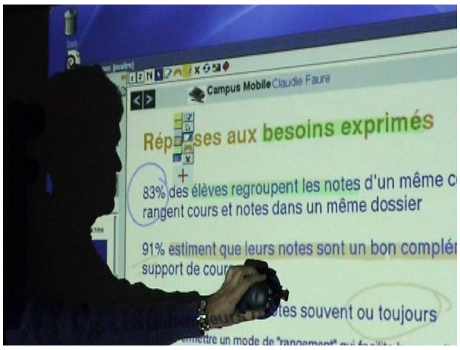

Campus Mobile Claudie Faure
Rép ses aux besoins exprimés
83% des élèves regroupent les notes d'un même c
rangent cours et notes dans un même dossier
91% estiment que leurs notes sont un bon complé
support de cour
tes souvent ou toujours

Figure 2.4: The interactive whiteboard used for the CORAO project

At the end of a presentation, there is generally a questions and answers session. When asking a question, the participant first presents himself by saying his/her name and affiliation. When the question is very interesting or relevant to others, a fellow attendee might want to take note of the questioner's name in order to contact them later. However, it might be hard to catch the name of the person asking a question. A system that tries to deal with this issue is the AutoSpeakerID [22] that displays the name of the person currently asking a question on a screen next to the presenter. This way, all participants can easily see the name and affiliation of this person and contact them later if they want to. Most people found the AutoSpeakerID very helpful. However, some people took it as a game and created false identities. Some people found this funny, while others found it very distracting.

Another problem of question and answer sessions is that, sometimes, one might want the answer to a question during the presentation and it is not pleasant to interrupt the presenter. In [21], a team proposed to introduce chat channels during the conference so that, amongst other functionalities, participants can ask question during the presentation. The speaker can also decide to give some extra references and pointers during the talk. The biggest issue with this solution is what the authors called "continuous partial inattention" as people are always looking at their laptops. Some speakers also find it disturbing that people are staring at their screens all the time and therefore, they do not know whether someone is actually listening or not.

After the presentation, it might be desirable to put a video or audio file of the presentation online. If there are many videos, it is very useful to provide a way to search the videos in order to avoid having to view the entire video. In [7], a way to simplify the whole process of putting these videos online and then being able to search them is presented. The authors propose to search the audio and browse the video. This means that speech recognition is applied to the audio of the presentation in order to provide text-based search and pictures of key frames of the video file are returned as the query result, allowing to easily browse for the desired video.

2.4 Personalized information

So far, we have seen systems that provide the same information for all participants. In this section, we will have a look at systems that provide personalized information.

2.4.1 Personalized information

The first group of personalized information is information that has been personalized on the parameters of a single user. One of the simplest forms of personalization is obtained by allowing each user to create their own schedule. Most conference management systems that support personalization allow their users to do this. However, we are going to look at systems that, not only allow users to create their own schedule, but that also recommend presentations based on community information as well as personal preferences / context. The community-based recommendation systems will be covered in the next section, at this point we will just discuss the recommendation systems based on the user's own attributes. A first approach consists in letting participants choose or submit a list of personal interests or keywords they find relevant. Based on this, the system can recommend some presentations that might be particularly interesting for them. A system providing this type of recommendations is the conference assistant [12]. In this system, all participants are equipped with their own mobile device that, amongst other functionalities, is able to display the conference schedule. On the schedule, potentially interesting presentations are highlighted. A similar approach is taken by UbiCoAssist [16]. The difference between these two systems is that with UbiCoAssist, the list of keywords is automatically generated by a web mining algorithm that looks up the current research topics of each participant. Another technique is to recommend presentations based on the participants rating of previous presentations he attended as well as on the current time. This approach is used by the conference assistant system [32].

Another field where personalization can be relevant is the recording of presentations. In the section about non-personalized information, we have seen a paper that proposed to automatically index videos of presentation by applying speech recognition to the audio file. As an alternative, each participant could create their own audio record of the presentation and annotate it, in order to index the audio file in a personal way, so that searching becomes even more intuitive. The audio notebook is such a system [31]. The idea behind this project is to keep the users habits in terms of note taking on paper, but to augment their notebooks' functionality by combining the written notes with the audio record of the presentation. These written notes will then serve as an index for the recorded file. The notebook prototype can be seen in figure 2.5. A more recent commercial product proposing similar functionality is the Livescribe pen [4] is described in the section on commercial products.

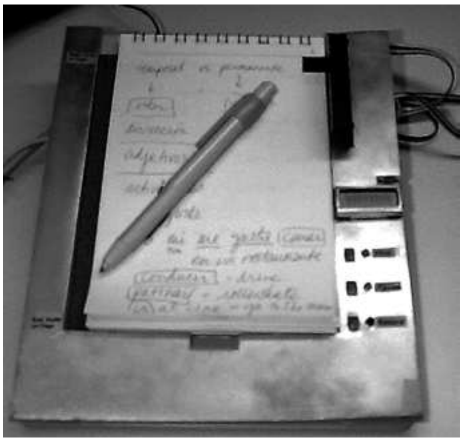


Figure 2.5: The audio notebook

Another approach to personalizing information is by exploiting a users's location. Location is a very broad research field, therefore this paragraph gives an overview only on how it has been handled for conference information systems. Location can be very important when going to an unknown environment as it is usually the case during a conference. People need maps to find their way or try to locate other people to talk to. Many information kiosks provide general maps, but these have been considered in the section about general information. Here, we are looking at personal devices that show the current position of a user. Most of the time, participants want to know their location [24], but sometimes they might also be looking for another person. Such location tracking can be provided using different diffusion channels such as mobile devices, kiosks or web applications [13].

2.4.2 Collaborative filtering

This very broad section covers conference systems that use community information to provide a range of services starting from recommendations of presentations based on people with similar interests to a list of people that are interesting to meet. Collaborative filtering is a general term designating the process of filtering information coming from different data sources such as social connections, ratings, etc. We will first look at applications that help participants to meet with other people that could be of interest to them. The most straight-forward way to achieve this is by giving participants a paper list with the names of people that should be interesting and relevant for them. The Relescope [14] is such a project. The information needed for such a report is gained from proceedings of a special interest group and the algorithm used to find people that could be of interest is called the common neighbours algorithm. This algorithm assumes that two people are more likely to work together the more colleagues they have in common. The users that tested this report were generally enthusiastic, but found it lacked an easy way to find and meet the people on their lists. An interesting result was that new and senior attendees seemed to be interested in meeting the people that are the furthest away from them in the social network graph. For example, a young PhD student might be very interested in meeting an experienced professor from whom he can learn a lot.

Instead of creating artificial encounters as is done when giving participants a list of people they should meet, some projects try to get people to just connect and converse with other participants in their vicinity. Several projects attempted to do this by augmenting name-badges. As name-badges are most of the time distributed at the beginning of a conference, everybody is already accustomed to carrying one. Therefore, name badges are a good way of reaching a maximum number of participants. One of the first projects in this field is the MeMe tags project [9]. MeMe tags are name tags containing an LCD screen that can display a phrase. The idea behind this project is that all participants have their own small phrase stored in their badge that will be sent to everybody they encounter. The people who receive such a message can then choose to keep it or to reject it. By seeing their phrases on other peoples' tags, participants found it easier to go and talk to them. Similarly, the iBand [18] is a wristband that is worn by the participants. Every time two people shake hands, as can be seen in figure 2.6, their contact information is exchanged. By seeing other people wear the same device, many users found it served as an ice-breaker.

The previously described projects aimed at getting people to meet. However, they do not really help people to get to know their common interests. This is the goal of many projects that put up big displays that react to the surrounding participants and display information about them so that they can find a common discussion topic. In this category we can find the AgentSalon [32] and ticket2talk [22], these systems display the interests the participants have in common and thereby make the conversation easier and more efficient. Another project is UbiCoAssist [16] where several users can see their common social network.

As seen with iBand [18], there are also systems replacing the traditional business cards. Although this certainly has many advantages, many people still attach value to the gesture of handing out business cards. Therefore, instead of replacing the traditional business card, projects such as [19] are trying to enhance traditional business cards by putting a color code on

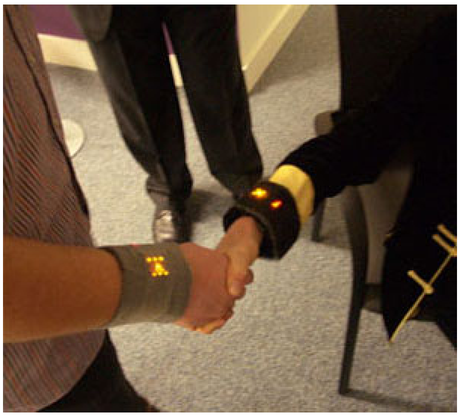


Figure 2.6: Handshake with the iBand

them so that they can easily be captured onto a phone, for example. This way, the gesture of handing out business cards is preserved while, at the same time, it allows to add extra functionality.

Finally, community information can be used to make recommendations to participants with similar interests. One of the biggest issues for conference participants, especially in the case of big conferences, is to be able to choose the right sessions to attend. If data is available about choices of people with similar interests as the user, this user can make a more informed choice on the sessions to attend or not. The community-based conference navigator [15] exactly tries to do this. When users register for a conference, they have the choice between joining or creating a virtual community of interest. Once they are in such a community, they can see what are the most attended presentations and read annotations of others that are in the same community. The conference assistant [12] also provides recommendations by analyzing the schedules of the colleagues of participants thereby imitating our natural way of solving the problem of choosing a presentation: asking the people we know what presentations they think are worth attending.

2.4.3 Memory-related information

Would it not be nice to have a computer memorizing information for us? As seen in the overview, people are more and more overwhelmed with information. Therefore, it would be very interesting and helpful to have a system that helps anyone remembering what they did and when [29]. An idea would be to provide a personal diary which shows the activities of a person. This is the idea that the conference assistant system implemented [32]. During the whole day, it captures the events and encounters of the participants in order to create a diary that is accessible through a web application. However, the participants might also be interested in creating a blog entry about their conference experience. The action-oriented weblog [27] proposes to do this for the participants. As they go through the day, the system captures their activities and creates draft entries for the participants blogs. The latter can then edit the entries later in order to publish them. These two examples are limited to the scope of the conference on-site activities. These solutions could be very helpful in the context of conferences, but could also have their meaning in a more everyday scenario. One of the projects researching on this is the forget-me-not [20] project that introduces a very close relationship between people and their mobile devices, as their mobile device is "aware" of all their activities. In this project, the mobile device is considered as a member of the family of the user.

2.5 Commercial solutions

In the last paragraphs, a few conference systems have been described, which were developed for research purposes mostly. The next three systems are commercial solutions that are currently on the market. Two products, the nTag [2] and the Charmbadge [1] are augmented name badges, SpotMe [3] is a system based on mobile devices, and Livescribe [4] is a pen that records written notes as well as audio.

2.5.1 nTag

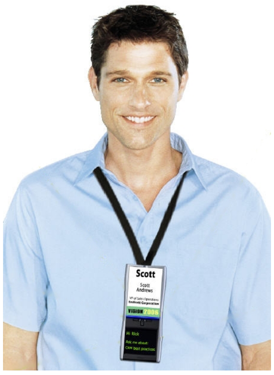


Figure 2.7: nTag name badge

nTag [2] is a system in which all participants receive their own name badge with an LCD touch screen and that integrates several services that could be useful during a conference. The underlying technology supporting nTag is semi-passive RFID, allowing wireless connections. This wireless communication allows participants to exchange electronic business cards and annotate them through the badge's touch screen. Other services provided are messaging, a personal agenda, the reception of event information and greetings that show participants their common interests to help a conversation get started. When participants go home after the conference, they can find all the collected information on a personal web page. This web page is a so-called trip report containing information about the sessions the participants attended, the contacts that were made, etc. An employer might use this as a way to measure the usefulness of the investment and the participants might simply want to recall what they did during the conference or follow up with some contacts. Figure 2.7 shows a man wearing the nTag.

2.5.2 Charmbadge

The Charmbadge [1] is a system based on name badges that mainly supports the exchange of contact information. The contact information is automatically exchanged when people are in each others vicinity. Before the start of the conference, the participants need to provide contact information and answer four questions. The Charmbadge includes four led lamps that lighten up when two people talk to each other to show the number of questions the two participants answered the same. The big difference with respect to the nTag is that the nTag supports a lot of additional functionality while the Charmbadge is specialized in exchanging contact information.

Figure 2.8: Charmbadge

2.5.3 SpotMe

Spotme [3] is a system based on mobile devices. Similarly to the nTag, it provides a lot of functionality, such as helping participants identify people that could be of interest to them, messaging, live audience response, plan meetings, maps, etc. This device supports most of the functionality we have seen in this chapter. Although it is very versatile, its main goal is to support the networking experience of people assisting a conference or any business reunion.

Figure 2.9: Spotme device

2.5.4 LiveScribe

The last commercial product we are going to cover is the Livescribe smartpen [4]. This pen is based on the Anoto technology that consists of printing a pattern of small dots on paper in order that a pen equipped with an infrared camera always knows its exact position. Livescribe extended this functionality by providing a pen that not only allows to record written notes, but also audio as their pens have two integrated microphones. Once the notes are taken and the audio is recorded, the notes can be transferred to the computer and shared online on either the livescribe platform or facebook.

Figure 2.10: Livescribe's Pulse smartpen

2.6 User response

From all the systems we have seen, it is interesting to see how the users reacted to the different systems. In order to be able to define an efficient system, it is important to look at user studies that have been conducted with various of these systems. Therefore, in the next two sections, we'll look first at the positive feedback and response to try to point out what users found most useful and then at the negative feedback in order to try to avoid the same problems. This section does not cover all remarks made by the testing persons, but tries to show the general points that should be kept in mind.

2.6.1 What to keep

When using a conference information system, people often showed the need and desire to have an effective way of planning meetings with other people [32, 14, 22]. Although they are quite happy having systems that support them making new contacts by pointing out relevant participants, they are looking for a way to make these new contacts work [22, 9]. Applications such as the MeMe tags and the iBand are seen as good ice breakers [9, 18] and sometimes even fashionable [18]. An important point to keep in mind is that, when creating applications that help people meet others, new and senior participants are often interested in meeting participants that are the furthest away from them in the social network graph [14].

2.6.2 What to watch out for

Discussion boards, forums In many systems, forums and similar discussion boards are provided, however, most of the time, they do not work as good as the organizers would like them to. One of the causes could be that participants are not aware of their existence [32] or are afraid to be the first one to post to the board [25].

Displays When putting displays into a room, it has to be noted that it can be very distracting to the point of disturbing the normal flow of events. For example, the AutoSpeakerID [22] that showed the names of people asking a question, led to some people not being able to concentrate anymore. Another issue is that not everybody feels at ease when their personal information is shown in public. Some of the users of ticket2talk [22] also pointed out this problem and did not like the way these systems acted as a kind of loud announcement of their presence.

Chat When using a chat channel, it seems that participants are less concentrated on the presentation itself. During experiment [21], the authors referred to this as continuous partial inattention. Additionally, not every participant has the same technological knowledge and some users might be overwhelmed. Finally, some speakers did not like the fact that most of the listeners were continuously staring at the screens.

2.7 Technological progress

This section is a short discussion about the evolution in the field of conference information systems as it was observed from the literature study. The oldest system we saw is the CHI'89 InfoBooth [28] that was realized almost 20 years ago. When looking at comparable systems nowadays, we can see that the basic functionality of such information kiosks has not changed a lot. Similarly, when looking at systems based on mobile devices, systems developed a decade ago [25, 12], are quite similar in functionality compared to current commercial solutions. The big advancements can be seen in community-related functionality. The research is clearly influenced by social networking becoming more and more popular. Although one of the earlier projects, the MeMe tags project [9] already aimed at observing group phenomenons, this was not yet exploited as an extra source of information. Currently, it can be seen that this information is mainly used for recommendations [29, 20]. As can be seen from the literature, although the basic ideas have not evolved significantly, the technologies have, allowing the creation of new, more reliable and faster solutions. This can be noticed very clearly when looking at the mobile device that was used in a system developed much earlier in figure 2.11 in comparison to the spotme device in figure 2.9.

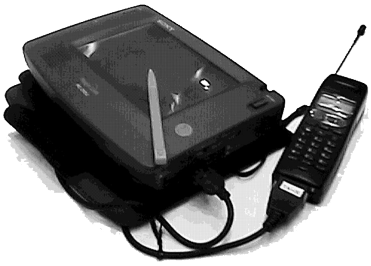


Figure 2.11: PDA for conference assistant that was used in 1998 [25]

3

Scenarios

3.1 Introduction

In chapter 2, we have seen many projects proposing different kinds of conference information systems. In this chapter, we will try to give a few example scenarios demonstrating how these different systems could be used. These scenarios are on purpose a bit extreme and should not be taken as such. They are merely an illustration of what could be done with them. The three figures in appendix A show the different activities that have been identified for before, during and after the conference. They also show which systems already have been realised and which not. Each scenario is split up into three phases: before, during and after the conference.

3.2 Scenario 1: Extreme networking

Conferences are a great opportunity for networking. We have seen in chapter 2 that many systems try to enhance the networking experience of conference participants. This first scenario takes this concept to the extreme and presents a conference information system that is all about networking.

3.2.1 Before the conference

Alice starts by registering normally to the conference. In addition, she fills out a personal profile containing the data she wants to provide to other participants. Furthermore, she creates a list of keywords describing her personal interests and enters her social network information. The system then connects to her social networks and downloads the lists of her friends.

Now that she is registered to the conference, Alice can start to plan her conference journey. The first two things she has to do is to arrange her travelling and accommodation. Of course, she could travel on her own, however, she decides to try to find a companion for her journey. For this she uses the *travel arranger*, a service provided by the conference organization. After having filled in a form with her travel information (departure city, transport means, etc.), the system returns two lists: one containing Alice's acquaintances and one containing other conference participants that could be willing to travel with her. If a suitable partner is found, Alice can contact him. On the other side, if there is no good match, Alice can leave her preferences behind so that someone else can contact her.

Alice is a student and has a very limited budget, therefore she chooses to look for a room-mate. She first uses the *accommodation arranger*, a system similar to the travel arranger seen earlier. As this system did not return any satisfying results, Alice decides to book a double room and try to find a room-mate later. Alice does not want to book any hotel; she wants one where she will have the opportunity to meet other conference participants. Luckily, the conference organization provides an *accommodation heat map*. The accommodation heat map shows a map of the city with a varying colouring depending on the hotels popularity. An example of such a map can be seen on Figure 3.1. After having looked at this map, Alice can easily determine a hotel that is best for her by looking at the popularity / price ratio.

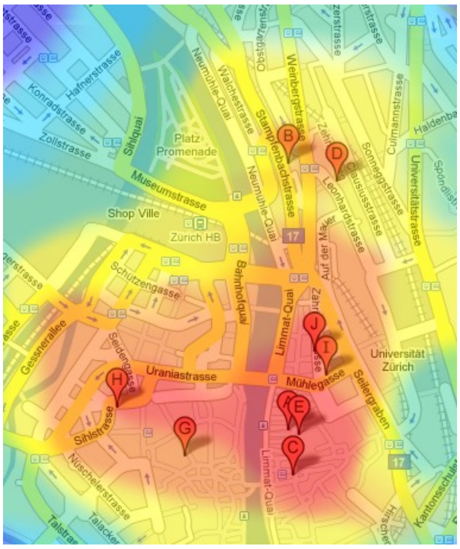


Figure 3.1: Accomodation heat map

The conference Alice is going to is quite far away, thus she decides to get there a bit early in order to visit the city. Although the conference will not have started yet, it is a great opportunity to get to know new people in a relaxed atmosphere. Alice decides to place an ad on the conference web page, asking other participants to join her during these few days.

Finally, it is time for Alice to prepare her schedule. On the schedule she already gets some recommended presentations. These recommendations are based on the sessions that are most popular amongst Alice's acquaintances as well as the ones that match best the keywords Alice entered during registration. At this point, Alice is ready to go to the conference.

3.2.2 During the conference

Upon her arrival at the conference, Alice receives a *personalized name badge* as well as a mobile device. This mobile device is the participants personal conference companion. It contains all general information as well as many services for the participants to network. Alice decides to look for a colleague of hers and sends him a message through the messaging service on her mobile device. After a few minutes, she receives a meeting request from this same colleague. Upon acceptance, the meeting is automatically inserted in her personal agenda. Alice looks on her schedule and notices she still has an hour before her first session and decides to go for a coffee. While drinking her coffee, Alice meets another participant called Marc and they exchange *virtual business cards* thanks to their mobile devices. In order to remember that Marc is working on a similar topic as her, Alice annotates the business card.

While drinking her second cup of coffee, Alice receives a message from the conference organization telling her that her first session has changed room due to the high number of participants that are planning to attend it. She can directly see the location of the new room on her mobile device and finds it without any problems. During the session, Alice can directly see the identity of the person speaking or asking a question on her mobile device. This service makes it easy for Alice to contact one of these people afterwards as she can save their contact information and annotate it.

Again, Alice has some free time and she decides to have a look at the *presentation heat map* showing her what presentations are currently most popular. Two examples of such heat maps can be found on Figures 3.2 and 3.3. The latter is part of the Intellibadge [13] project.

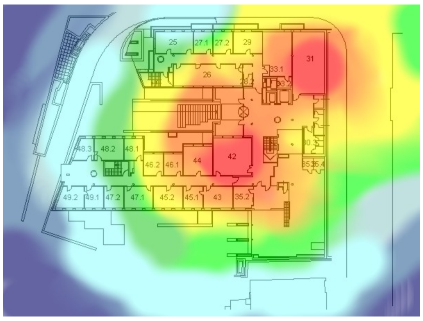


Figure 3.2: Presentation heat map

After a long conference day, Alice feels like having dinner with people that share her interest in object oriented databases. In order to organize such a group meeting, she creates an ad announcing her plans as well as a meeting point. Interested people can directly subscribe and

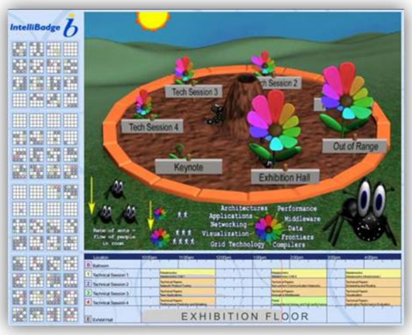


Figure 3.3: Intellibadge: "how does your conference grow?"

Alice sees the live evolution of the participants of her dinner. This way, she can choose an appropriate restaurant.

Back in the hotel, Alice does not remember the name of the participant she met this morning during her coffee. Thanks to the diary on her mobile device she finds Marc and arranges a meeting with him the next day. Furthermore, she feels like meeting someone new and looks on her list of interesting people to meet. With the device she creates a request for meeting.

3.2.3 After the conference

Once the conference is over, it is important to keep in touch with the new contacts. Alice goes to the website of the conference and finds a list of contacts she collected during the conference. From this interface, Alice can choose to add some or all of these contacts to her social network(s). She can also print or receive this list by e-mail.

Alice can now add an application to her social network that is called her conference world map. Alice's *conference world map* shows the conference she has been to and the ones she will attend next. This way, others can see what conferences might be interesting for them too and Alice can see which of her acquaintances will attend the same conferences as her.

After a few days, Alice receives a reminder to contact some of her new contacts. This reminder contains the annotations she made during the conference. In order to have a more efficient communication, Alice can start up by looking up the profiles of her contacts where their current projects and publications might be described. Another way to remember information about her contacts is to look at the pictures that have been taken during the conference. On some of them, she already has been tagged and thanks to the tags, she can easily find out who is on the picture with her.

Finally, Alice also prints out her trip report containing the presentations she attended as well as the contacts she made. This trip report might be useful for her employer in order to have a measurement of the "usefulness" of the conference.

3.3 Scenario 2: It's all about rating

This second scenario treats with rating. Rating is the general term describing the action of assessing or evaluating someone or something. In a conference, it is possible to attribute a grade to, for example, sessions, speakers or papers. This information could be used for handing out a price, for example.

3.3.1 Before the conference

Before the conference, Alice registers using the registration page and creates her person schedule. As Alice already has been to the same conference a year ago, she gets some recommendations based on her ratings from last year. These recommendations can be either based on speakers she liked or on topics she found particularly interesting.

3.3.2 During the conference

At the end of each presentation, Alice gets a small survey on her mobile device, badge or computer. She can also leave some comments for the speaker in an anonymous or personalized way. Furthermore, Alice can post questions (anonymously or not) which will then be posted to the forum where everybody can read the responses. All this information is directly fed back to the speaker. After having been to a few presentations, Alice gets recommendations for other sessions, based on the ratings she gave for the presentation she attended.

Of course, at the end of the conference, it is possible to easily give out a best talk award based on the ratings of all participants and not on a jury.

3.3.3 After the conference

After the conference, Alice wants to look back at the best conference talk. On the conference website, she can find the video of all talks. She can even search for a specific sequence as the videos can be searched. Additionally, Alice can find back the ratings she gave to each presentation and thereby it becomes easy for her to remember the conference and contact the interesting people.

3.4 Scenario 3: Paper is the real thing

This last scenario tries to show a scenario based on a paper interface. Although technology advances, many people still find that writing on paper is very convenient. This paper interface should be coupled to the digital world in order to provide new services. This scenario is based on the research from the Globis group where paper is augmented by using the Anotot technology. The Anoto technology consists in printing an invisible pattern of dots on a document. By using a special pen equipped with a camera, the pen is able to define its exact position, as well as the current document. This technology allows the creation of interactive documents [30].

3.4.1 Before the conference

Before the conference, Alice signs up as usual. There is no special sign-up procedure required in this scenario.

3.4.2 During the conference

During a presentation, Alice can take notes that are automatically digitalized and associated to the current presentation. At the end of the presentation, Alice fills in a survey and this is automatically sent for evaluation to the speaker. Both the notes and the survey are on paper.

When she meets new people, Alice exchanges augmented paper business cards on which it is possible to take notes. The business card, as well as the notes, will be automatically saved together and associated with the time / place the business card exchange took place.

In order to add some dynamism to the paper interface, big displays can be found everywhere. These displays display the current schedule as well as other useful information. When Alice goes to the displays, she can print out extra (interactive) information and use the screens to navigate through her paper documentation. This information can be about the conference, restaurants, maps of the city or even tourist information. As an example, Alice could have a printed copy of a paper. When clicking on a reference, the referenced paper would be displayed on the screen. This paper could then be printed in order to become a dynamic document as well.

3.4.3 After the conference

After the conference, Alice can easily find her notes that are already organized and digitalized. She can also decide to share her notes on a specific presentation with her friends. She can do this online and define who she allows access to her notes. Alice also can see the business cards she received and create relations with the social networks and her outlook.

4
User study

4.1 Overview

This chapter collects the opinions about conference information systems via different methods. First, we summarize an interview with Dr. Daniel Felix who has had the opportunity to organize a conference where the SpotMe [3] conference systems was used. The second part describes the outcome of a group discussion of the Global Information Systems Group (GlobIS) and will serve as a main basis for setting the requirements for the rest of the project. Finally, once the main requirements have been defined, we analyze the results of a survey that has been conducted amongst a set of people who attend computer science conferences.

4.2 Interview

On September 29th 2008, we conducted an interview with Dr. Daniel Felix who has used a commercial conference system during the Interact 2003 conference [5] where he was a member of the organization committee. The system used is called SpotMe [3]. A detailed description of this system has been given in section 2.5 concerning commercial systems. A transcript of the interview question can be found in the appendix B.

As a PhD student, Dr. Daniel Felix used to visit many conferences. For him, the most important services a conference would need is a way to find people, send messages and arrange meetings. The system used during the interact 2003 conference, SpotMe, is a system based on PDAs, where every participant gets their personal PDA. During Interact 2003, this meant that about 300 people were equipped with PDAs. As these PDAs were distributed to all participants, it was not possible to ask for a deposit in order to be sure to get the devices back. As could have been expected, not all devices were returned, which caused a conflict with the supplier. SpotMe was mainly chosen as it provided the ability to send messages to other participants and the organizers as well as receiving messages from the organization committee and other participants. Due to the restricted user interface, it was not very pleasant to send messages via this device, terminals with common keyboards were provided as well, to allow participants to easily send messages. The most used, but at the same time, the most debated feature of this application was its "radar" functionality. The radar allows participants to see all the other participants in its vicinity. While it was claimed by the supplier that the devices only work in the conference area, people have been able to use it in sensible areas such as the hotel or toilet. This, of course, raised some privacy concerns amongst the participants, especially as it was not clear whether one would still be detected with the device turned off. At the time, in 2003, the product was very new and there were still many problems. For example, some areas were not covered well enough with the wireless network causing people to sometimes, not receive all messages. Although the system fulfilled the need of a messaging system, it was still too early in its development process and not reliable enough. This system was introduced as an experiment and therefore did not replace any existing infrastructure. If the reliability would be increased, it could replace the pin boards that allow people to exchange messages during a conference. This device certainly added fun to the conference and made people remember it.

4.3 Group workshop

After two months of research on the topic of conference information systems, we held a small group workshop for the GlobIS group. We started out presenting a survey of the systems described in chapter 2 so that all participants have the same information. After the presentation, there was a discussion in order to determine the goal of our future application. Next follow some of the ideas presented by the participants (some of which are not new):

Meeting people based on their technical know-how Most of the projects enabling networking between conference participants presented previously tried to match people based on their common interests. An alternative approach could be to match people according to a supply and demand principle. Some senior participants such as professors might be looking for PhD students or other qualified people to participate in their projects. A conference might be a very good occasion to match such people as they can get to know each other in a less formal manner than through ads or e-mails.

World map of conferences This application could be useful for several cases. First of all, someone might not be able to travel far in order to assist to a conference and therefore try to find a conference nearby. In contrast, the application might also be used for people wanting to travel somewhere they have not been to yet. Although this is a less serious use of such an application, it might find its audience.

Shared pen to control slides This idea comes from the work of the group with the Anoto technology. With this technology, introduced in chapter 3 it would be possible to imagine a scenario in which everybody has a version of the current slides printed out with the Anoto pattern. During the questions and answers session, a pen equipped with a camera could be passed around in order to control the display of the current slides. Such a project could be based on the PaperPoint application [30].

Submit questions to the moderator During a questions and answers session, there is not always enough time for everybody to ask their questions. Sometimes several similar questions might be asked, leaving no time for questions that would have put another light on the discussion. A possible solution for this problem would be to allow participants of a talk to submit their questions before and during the talk to a moderator who then chooses the most interesting (or frequent) ones in order to give an equal chance to everybody. After the talk, the remaining questions could be posted on a forum or wiki and the presenter could provide an answer.

General tagging applications This application would enable participants of a conference to tag all information related to a conference, from slides to photos. These tags could then be shared or used for retrieving information.

Conference networking site The idea is a social networking site in common for several conferences. The organizers of the conference can put all the information related to a conference on this website, such as slides, videos and schedules. The participants can then register for a conference and have a personalized profile containing all the information that is related to them.

4.3.1 Conclusion

As a result of the group discussion, as well as the results of the survey, we have retained the idea of the creation of a conference networking site that allows to manage all types of information related to a conference. A more detailed description can be found in chapter 5.

4.4 Online survey

This section presents the results of our online survey. Indeed, after the group discussion, we decided to perform an online survey in order to capture the opinions of a wider audience. The survey was intended for professionals that have attended to at least one conference on a topic related to computer science.

4.4.1 The survey

The survey contained twelve questions and took, on average, between five and ten minutes to complete. It was online for approximately ten days and a hundred and eighty four people answered the survey at least partially. However, we decided to take into account only the answers from people that filled out the whole survey, which gives us a hundred and thirteen complete questionnaires to analyze. Based on the IP addresses of the respondents, we analyzed the location of the participants location at the time they filled out the questionnaire as shown in figure 4.1. While we treat each question in detail in the following sections, the full results, including the respondents remarks, can be found in appendix **??**.

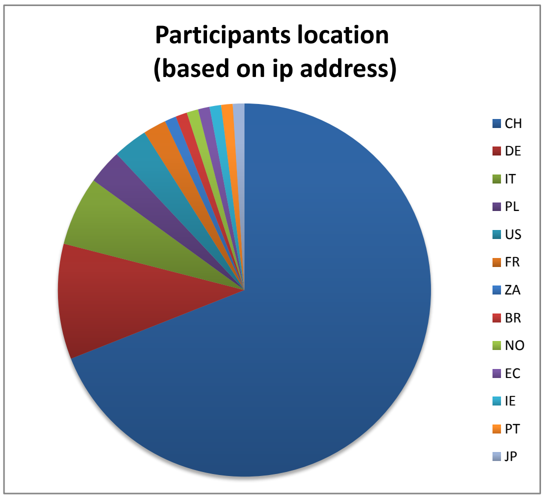


Figure 4.1: IP address distribution

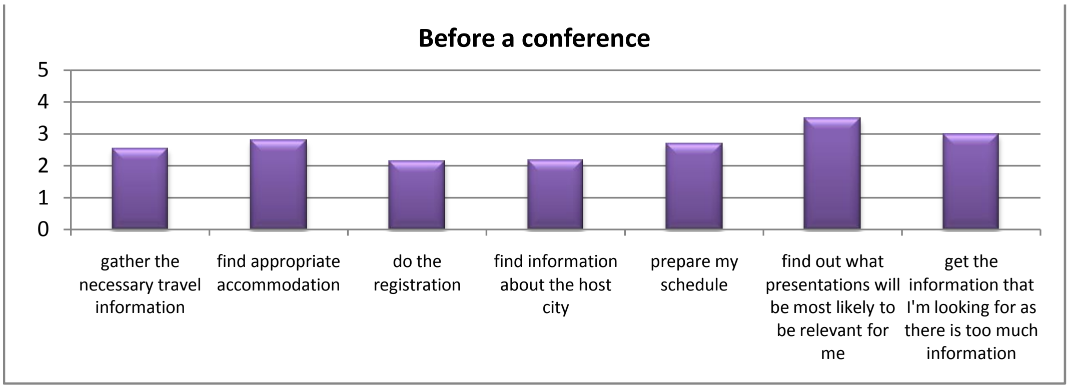
Questionnaire structure

The first questions were "matrix" questions, meaning that there are several tasks described that should each be rated according to their difficulty. Five answers were possible: ++, +, Neutral, - and - -, where ++ stands for very important as it is difficult to achieve the task described in the question. For each of these questions, there are two graphs available, for creating the first one, we assigned grades from 5(++) to 1 (- -). With these grades, an average grade from all participants is computed. Grade three means the positive and negative opinions are balanced. If the grade is higher than three, it indicates that more people are interested in having this task simplified for them and if the grade is negative, less people are interested in this task. The second graph shows the detailed results as well as the distribution of the answers. Each of the "matrix" questions also had an additional remarks field where respondents could fill in their own comments.

The next two questions had several propositions that could be ticked with a checkbox, allowing a user to choose several answers at the same time. Finally, at the end there were a few `yes or no` questions with no possibility for remarks.

Before the conference

This first question tried to find out what issues conference participants are facing before a conference. We asked them to rate seven possible problems they could have before a conference. Figure 4.2 shows the results of this first question.



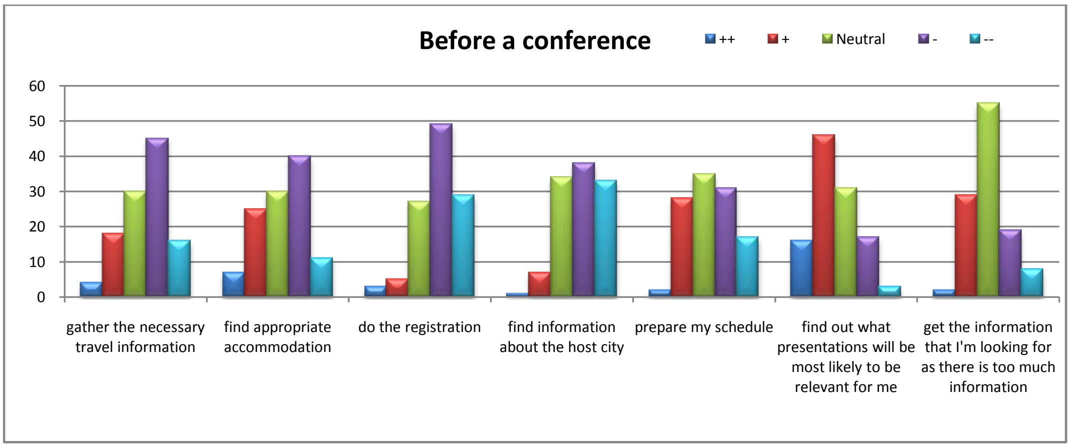


Figure 4.2: Survey results: before the conference

In this first question there is only one task that obtains a grade above 3: "find out what presentations will be most likely to be relevant for me". Another interesting point is that information overload does not seem to be a problem, at least before the conference, as a big majority responded neutral to the last task. There were several remarks from the respondents. Two found that the hotels suggested by the conference organizers are mostly too expensive, so they do not use this service.

During the conference

In order to find out about the needs of conference participants during the conference, we divided the questions into three groups of issues: outside the conference venue, networking issues and conference-related issues.

Outside the conference venue This question tried to discover whether people have difficulty to find touristic information or restaurants and bars outside the conference venue. While touristic information does not seem to be an issue, finding a good bar or restaurant is more difficult as shown in figure 4.3. Furthermore, as one of the participants suggests, the issue is not always to find a good restaurant, but it is also important to find a place where there are people one knows.

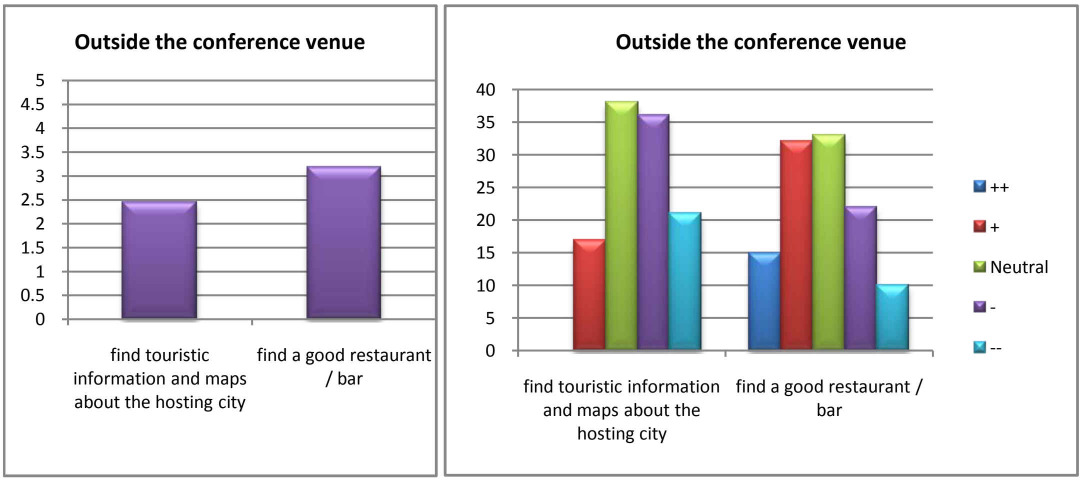


Figure 4.3: Survey results: outside the conference venue

Networking This question tries to get a deeper understanding of the respondents needs for the support of their networking activities. Networking can go from helping people to meet, to remembering names. With the exception of managing business cards, all other activities have a grade of above three. This shows that the current applications do not provide enough support for conference participants. As said by some respondents, managing contacts can be done through a personal agenda, however, remembering peoples and faces or getting to know the right people is something that is usually not supported.

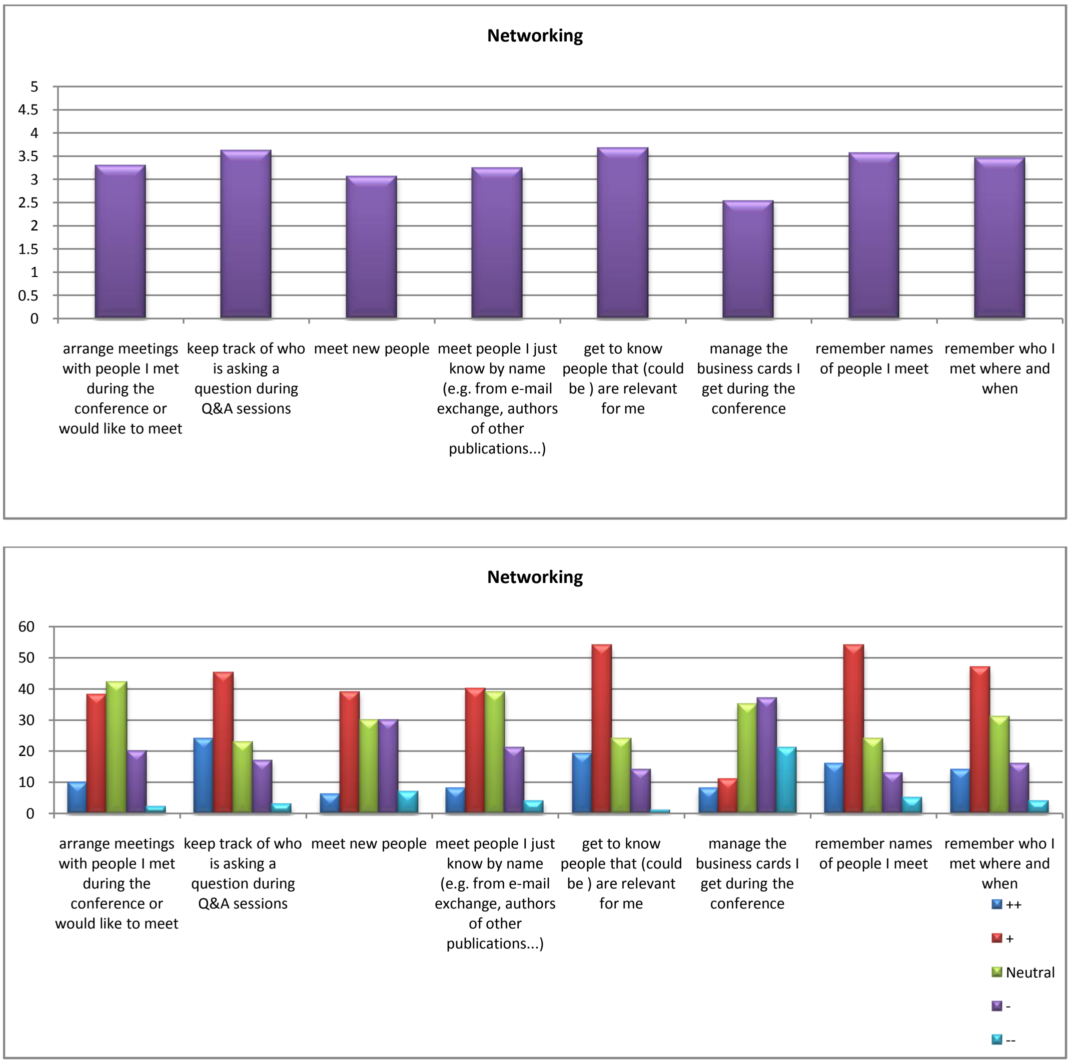


Figure 4.4: Survey results: networking

Conference-related Confirming the trend we have seen in the last few questions, people show interest in having a personalized service that helps them find relevant information. Currently, the tools provided by the conference organizers are often forums or wikis. From the results in figure 4.5, we can see that a lot of people answered “neutral”, which might show their lack of interest in these technologies. A recurring remark was that it often is hard to get access to the slides of the talks.

After the conference

This question tries to find out what services conference participants could be interested in after the conference. The results can be found in figure 4.6. Currently, most conference organizers provide a forum/wiki after the conference, as well as pictures on the conference

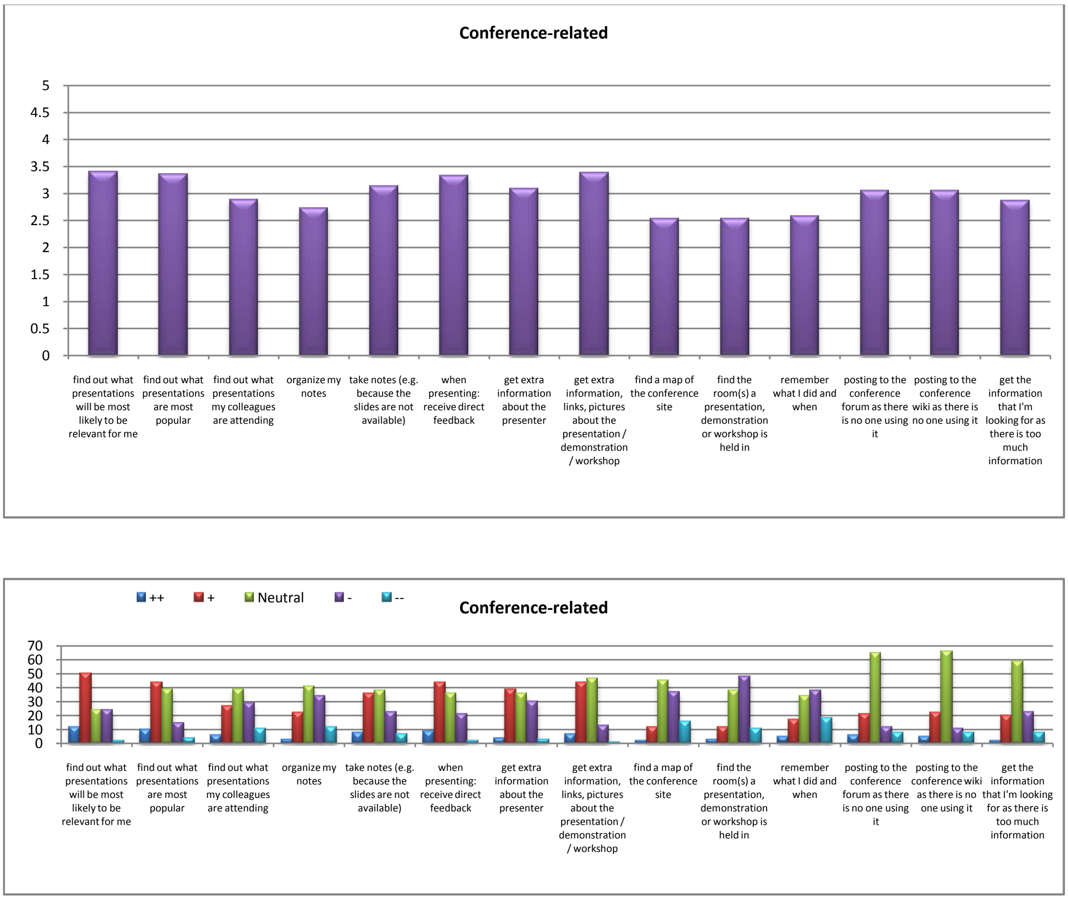


Figure 4.5: Survey results: conference-related activities

website. We can see that many people do not have an opinion about these services. As the previous question also pointed out, people generally like to have the slides and videos of talks they attended. Finally, a conference report seems to be the most requested feature. A conference report is a document that is generated based on information collected during a conference. It can contain information such as the schedule, the fellow attendees names or personal notes.

Channels

The next two questions try to find out what information channels are currently used and what people would prefer as a channel for a new system. Figure 4.7 shows what the respondents would like to have as a channel and figure 4.8 shows the systems they have already used for a conference. A clear result is that the respondents were very favourable on using a software or web application on their own devices. A very interesting result is that while wikis, forums and information kiosks have been used by a significant number of people, they are not very popular when considering the design of a new application. On the other hand, badges with

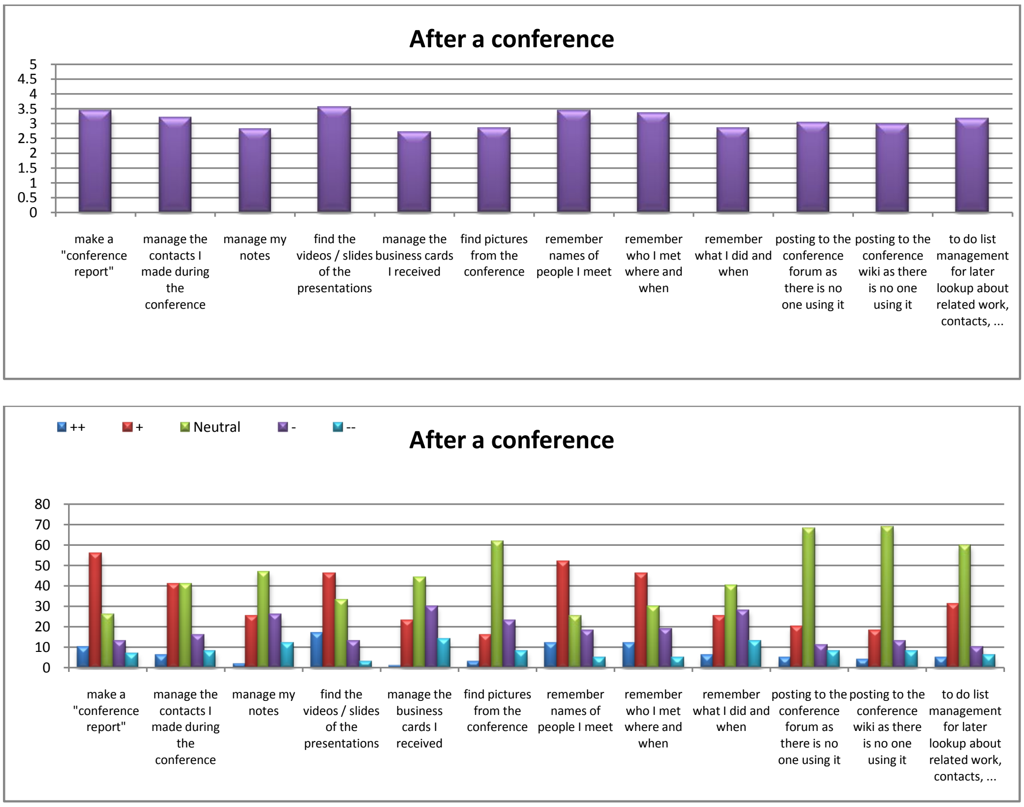


Figure 4.6: Survey results: after the conference

augmented functionality have not been used by many people, but are quite popular when considering all channels for a new system. Finally, one of the respondents proposed the idea of downloading contact lists to Outlook. This is also a very interesting idea as it would not involve users changing the way they work.

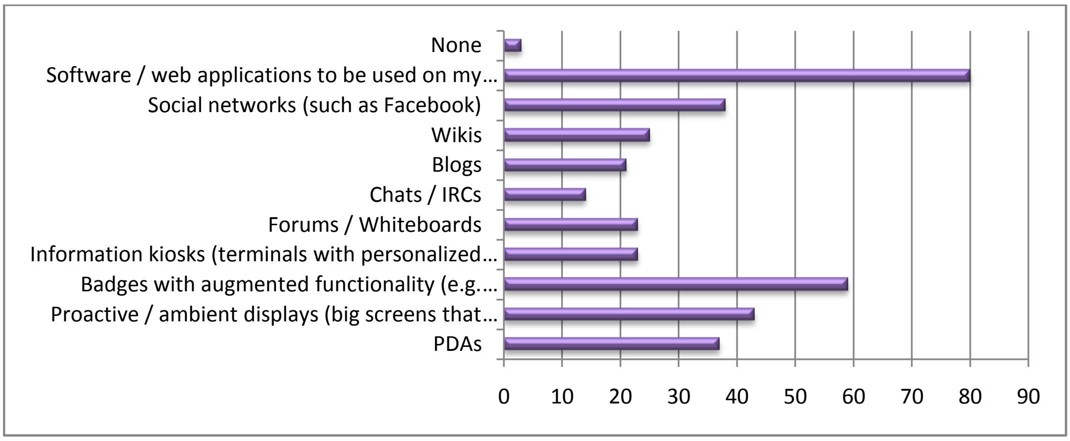


Figure 4.7: Survey results: desired channels

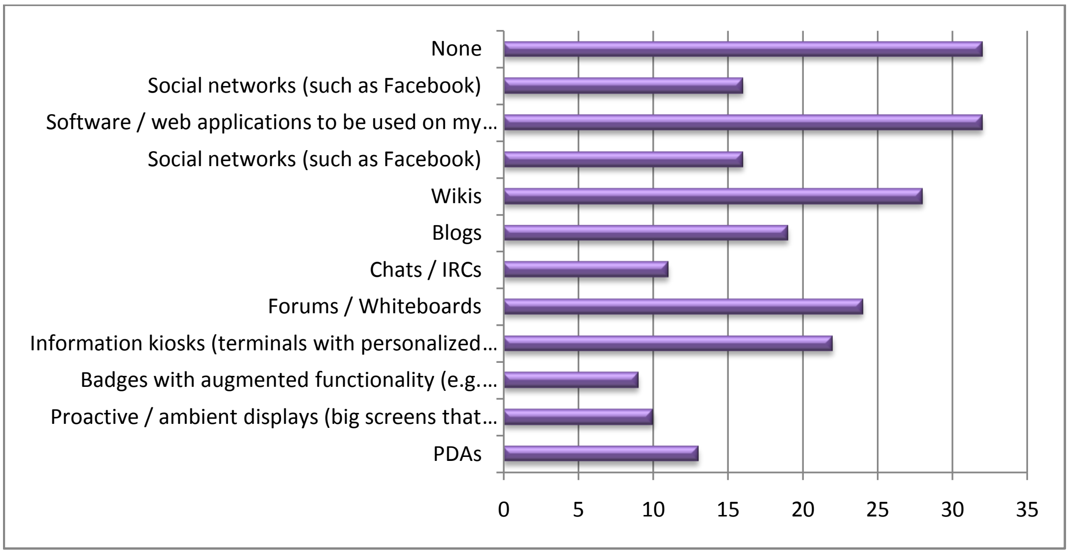


Figure 4.8: Survey results: used channels

Final questions

For the last four questions, we first introduced briefly the system we are working on (a conference networking site that helps you keep track of your personalized information throughout several conferences) and then asked several yes/no questions. While the respondents do not want just another social network for managing their contacts, they seem to be interested in a system that allows them to manage their conference-related information throughout several conferences.

Conclusion

The survey confirmed our initial hypothesis of the need of a personalized conference information system. One of the features with the highest score is the conference report. We have not found a system that automatically generates such a report yet and therefore we are going to focus on providing a system that generates such a report.

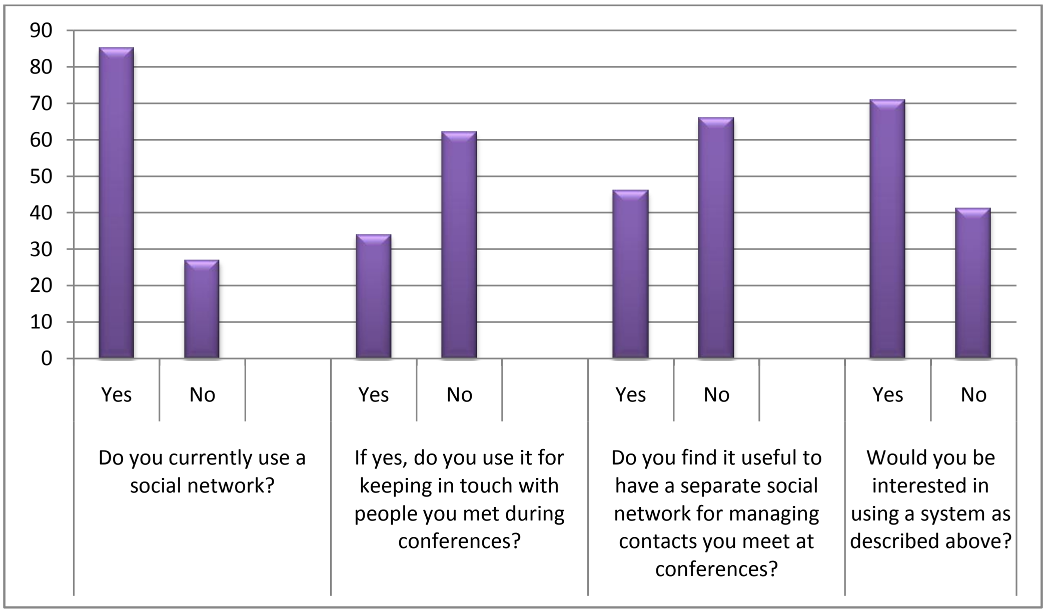


Figure 4.9: Survey results: last questions

5 Design

5.1 Background information

Our application is modeled using the OM data model [26, 6]. This first section introduces the main characteristics of the OM data model.

5.1.1 OM Data Model

The OM data model can be seen as an evolution of the Entity/Relationship model that provides support for object-oriented data management. Object-oriented models have the advantage to prevent the object relational impedance mismatch. An object relational impedance mismatch is encountered when a program is written in an object oriented language and the data is stored in a relational manner. The OM model not only models what data is stored, but also how it is organized, thereby providing a model for the whole application. It is a generic model as it is independent of a particular system. One of the main features of the model is that it provides a distinction between object classification and typing [26, 6]. The model is made up of two levels: the front level providing information about classification and the back level about typing. The typing information describes the representation of stored objects while classification describes the roles of objects, thus describing the application. Let us consider the following example application: we want to model a small contact management application handling people as well as organizations. The model for this application can be seen in figure 5.1.

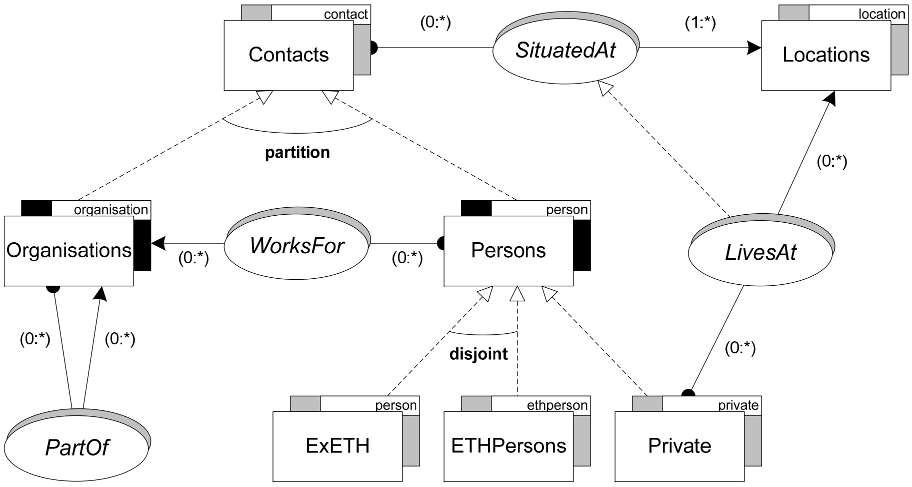


Figure 5.1: Example OM Model

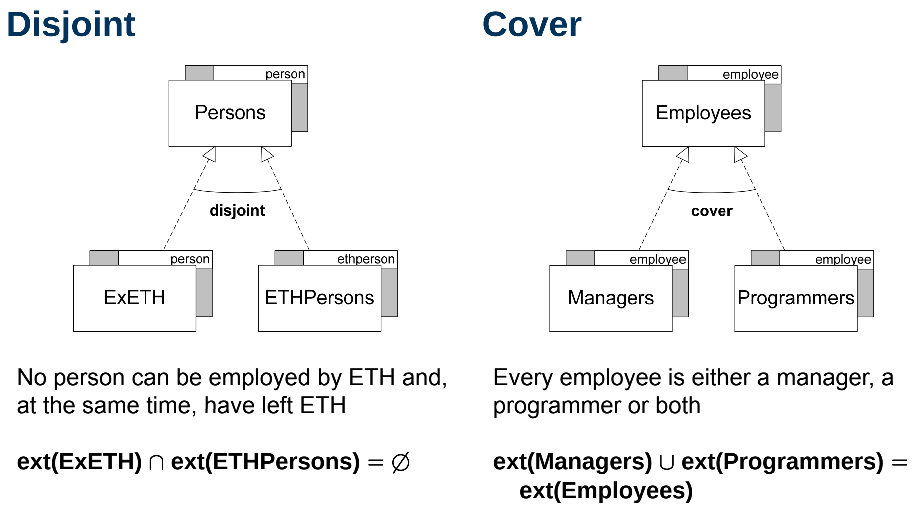
Typing

The back layer of the model handles the typing information. For example, a person object and an exEth object both have the same type `person`. This type contains the information about the format and value of attributes, the operations as well as inheritance. OM supports multiple inheritance, meaning that a type can be a subtype of several types. Another feature is multiple instantiation. Multiple instantiation allows objects to have several types that can be hierarchically unrelated. Furthermore, objects can gain or lose these types dynamically at runtime. For example, in the scenario introduced before, an object of type `ethperson` defines some attributes that are particular to people working at ETH. This could, for example, be an identifier that identifies the ETH employees. If this person were to leave the ETH, he would lose the `ethperson` type, thereby losing the attributes of an `ethperson`, such as his employee id.

Classification

The front layer of the model handles information concerning classification. It consists in grouping objects into semantic groups as well as the relations between such groups. It is also said that the classification defines the roles of an object, therefore it is sometimes referred to as the role model. For example, `Privates` and `ETHPersons` are both collections of persons, but they can be classified into the privates and ETHPersons groups. Such a group of semantically similar objects is called a collection. There are two types of collections: unary and binary. A unary collection represents a group of entities such as the `ETHPersons` collection whereas a binary collections describes an association between two collections (unary or binary). Collections can be of several types depending on the properties we want to provide them with: they can be a set, bag, sequence or ranking. Sets allow no duplicates and define no order, bags allow duplicates, but no order. Sequences may contain duplicates and are ordered. Finally, rankings are ordered, but may not contain any duplicates. Although the model does not provide support for n-ary associations, it is possible to describe complex relationships by having an association as the source and/or target of an association. Similarly to multiple instantiation, the OM model supports multiple classification, allowing objects to have several roles simultaneously.

Subcollections In order to define conceptual dependencies between collections, it is possible to define a hierarchy on collections. For example, in figure 5.1, it can be seen that `ETHPersons` are a subcollection of persons. Person objects member of the collection `ETHPersons` are both of type Person and `ETHPerson`, where `ethperson` is not necessarily a subtype of Person. This shows the clear separation between typing and classification. The OM model provides four types of constraints on such dependencies: cover, disjoint, intersection and partition. The cover constraint that can be seen in figure 5.2 states that each object of a supercollection is also part of at least one subcollection. Disjoint as in figure 5.2, makes sure that an object in a supercollection can be present in at most one subcollection. The intersection constraint as in figure 5.3 is used in the case where a subcollection has several supercollections; in this case, all objects of the subcollections are in both the supercollections. Finally, the partition constraint shown in figure 5.3 makes sure that each object of the supercollection is in exactly one of the subcollections.



No person can be employed by ETH and, at the same time, have left ETH

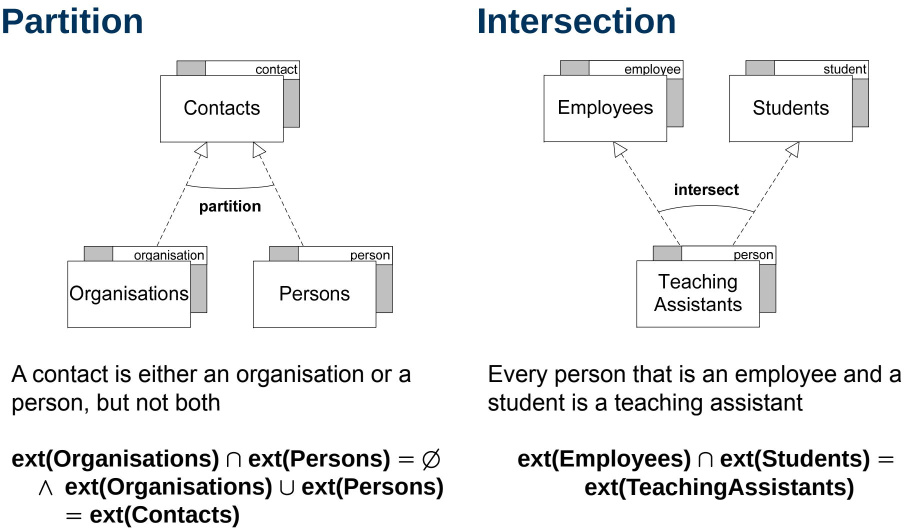
ext(ExETH) ∩ **ext(ETHPersons)** = ∅

Every employee is either a manager, a programmer or both

ext(Managers) ∪ **ext(Programmers)** = **ext(Employees)**

Figure 5.2: Disjoint and cover constraints

Associations As mentioned before, associations are binary collections expressing dependencies between collections. It behaves like a normal collection and can be of type `set`, `bag`, `sequence` or `ranking`. Similarly to the entity-relationship model, it is possible to define cardinalities on the relations, but the notation is different. For example, the (0;*) cardinality on the situatedAt association implies that an organisation can be situated at any number of locations whereas the (1;*) cardinality expresses that each location should be associated with at least one organisation. All associations in OM have a direction from a source collection



A contact is either an organisation or a person, but not both

$\mathbf{ext(Organisations)} \cap \mathbf{ext(Persons)} = \emptyset \wedge \mathbf{ext(Organisations)} \cup \mathbf{ext(Persons)} = \mathbf{ext(Contacts)}$

Every person that is an employee and a student is a teaching assistant

$\mathbf{ext(Employees)} \cap \mathbf{ext(Students)} = \mathbf{ext(TeachingAssistants)}$

Figure 5.3: Partition and intersection constraints

to a target collection. The source collection is called the domain of an association, and the target the range.

5.2 The application

In the last chapters, the current state in conference information systems was captured as well as the interests of different parties. From this point on, we will design and implement an application that is based on the knowledge gained through the readings, discussion and survey. Our application is a social networking platform for managing conference-related contacts and information. The platform should provide the functionality of a typical social networking site, but should also implement conference-related activities.

This conference social networking platform will allow conference organizers to create their own conferences in our system. The application can therefore be seen as a meta conference system. Indeed, with a single user account, a person can manage all conferences they are attending as well as those they are organizing. In order to make this possible, it should be possible for users to change roles over time. During one conference, a user might be a conference organizer, while in another conference, this same user might be a participant and have access to a restricted set of actions.

We also differentiate information that is relevant inter or intra conferences. For example, a conference schedule is only relevant in the context of the conference it belongs to while users contacts will be relevant across all conferences.

The data model can be seen in figure 5.9. As it is complex, we will look at it step by step.

5.2.1 Social networking elements

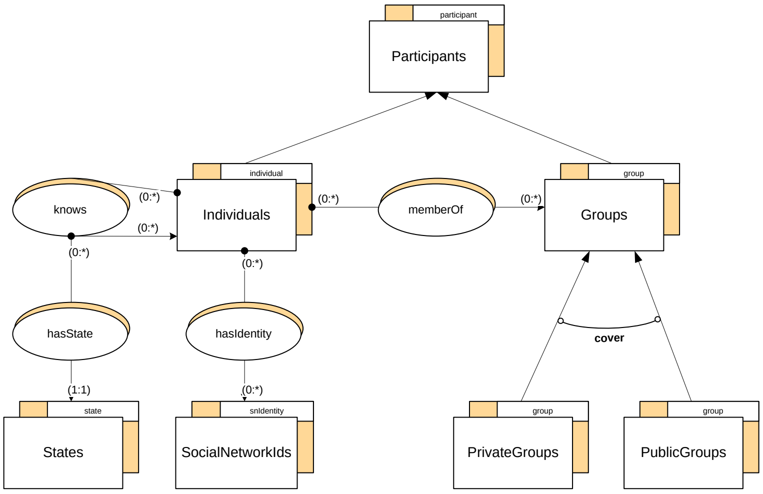


Figure 5.4: Social networking sub-model

First of all, our application is a social network and should therefore provide the necessary functionality. Figure 5.4 shows the elements of the model necessary for that functionality. A `participant` is a user registered to the system which can either be an individual or a group. In the current implementation, we decided to create a new social network without any contact to other social networks. In the future, it must be possible to link our users to their identities in other social networks in order to avoid redundant user accounts. To enable this, each individual can be linked to social networks with the `hasIdentity` association. The entity `SocialNetworkIds` would contain the necessary information to provide this linking. Each individual can be a member of one or several `groups`. The model describes two collections of groups: `private groups` and `public groups`. The distinction between these two types of groups will define the manner they can be joined. Private groups can only be joined by individuals who have been invited to do so, whereas public groups can be joined by any individual. Finally, individuals should also be able to interconnect similarly to interconnections in LinkedIn[1] or friends in Facebook[2]. This can be done through the `knows` association. The knows association also has a `state`. This models the necessity to request for a new relationship, which another user can then later accept or decline. There are three states defined: `pending`, `request` and `approved`. For example, when person1 asks person2 to be his acquaintance, the `knows` association from person1 to person2 will contain the pending state and the association from person2 to person1 the request state. Once person2 accepts the request, both association will contain the approved state.

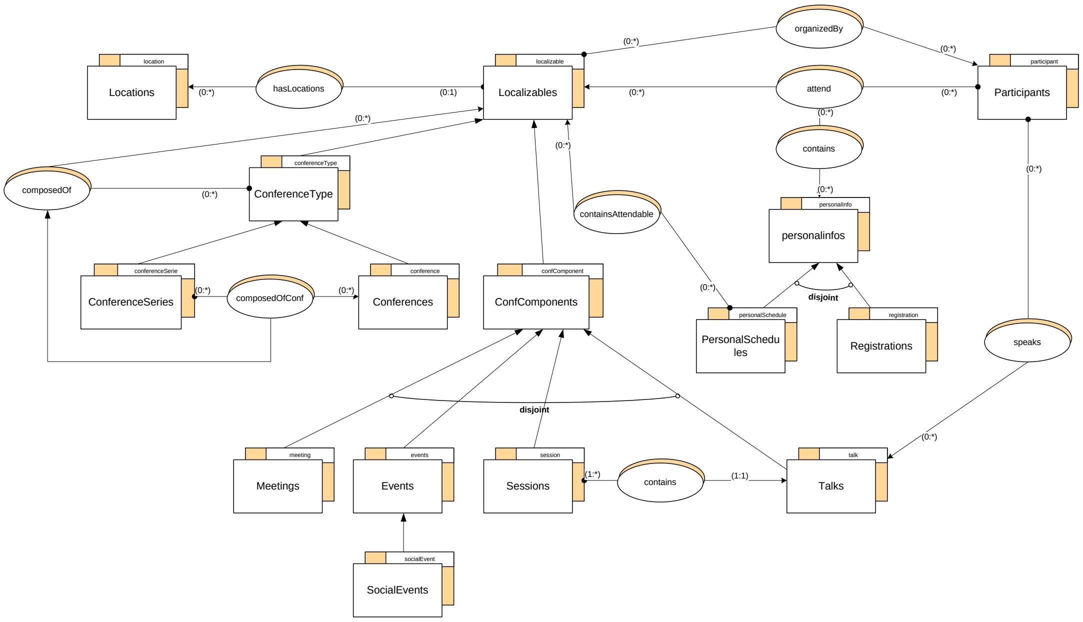


Figure 5.5: Conference-related sub-model

5.2.2 Conference elements

Figure 5.5 shows the submodel defining the conference-related entities of our application. On the right side of the model, we can see the participant type that was described in the previous paragraph. Furthermore, we now have an entity called localizable. A localizable object is any object that can have a location. Therefore, it is this object that has the hasLocations association that allows to define a location for a localizable object. There are two types of localizable collections: ConferenceTypes and ConfComponents. The ConferenceTypes collection contains all conferences and conference series. A ConferenceType object can contain several components through the composedOf association. In order to express that ConferenceSeries can only be composed of conferences, the composedOfConf is expressed as a sub-association of composedOf. Furthermore, the conference schedule can be defined through the composedOf association. Indeed, ConfComponents are basically the elements of a conference schedule, them being part of the formal schedule or not. The latter components can be classified into meetings, events, social events, sessions or talks.

This sub-model also expresses the relationship between a conference and its participants. A participant can attend a conference, conferenceSeries or any conference component. Moreover, personal information can be attached to the attend association. This allows to associate a schedule or a registration, for example, to a conference a participant is attending. A participant can also be the organizer or speaker for one or several

[1] http://www.linkedIn.com

[2] http://www.facebook.com

items of a conference. These are expressed by the `speak` and `organizedBy` associations.

5.2.3 Academic conference elements

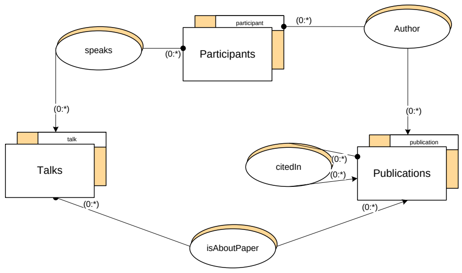


Figure 5.6: Academic conference-related sub-model

As our system is designed for academic conferences, we also modeled a few concepts that are particular to academic conferences. They can be seen in figure 5.6. In this sub-model, we can see that a participant can be the author of a certain number of publications and that these publications can be the subject of talks.

5.2.4 Annotation elements

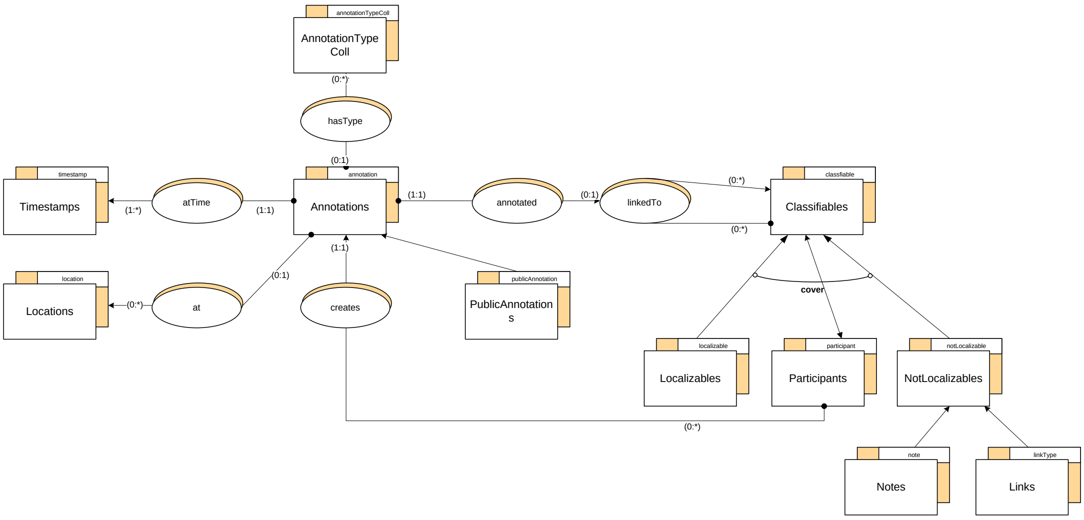


Figure 5.7: Annotation-related sub-model

One of the main functionalities of our system is the creation of a personalized conference report. A conference report is a document that can be printed out before / during or after the conference and that contains all information about a participants conference experience. For example, it can contain their schedule, but also who they met and their personal notes. Figure 5.7 shows the sub-model of the application describing the annotation functionality. Indeed, in order to generate the report mentioned above, our system generates annotations. For this, we introduce the `Classifiable` collection. This collection contains all `localizables`, `participants` and `notLocalizables`. An annotation consists in linking two `classifiables` through the `linkedTo` association. Then, an annotation is attached to this association which contains a reference to the `timestamp` and `location` of the annotation. In order to be able to define different kinds of annotations, an `annotationType` can be attached to the annotation. This type could be used to render a specific type of annotation in the conference report. Finally, while a conference report normally only contains the annotations attached to the participant it is made for (through the `create` relationship), it might be useful under some circumstances, to have annotations that are available to everyone that attended some `classifiable` object. For example, the organizer of a talk might want to attach notes or links to the slides. This information is useful for everybody and should be available on the conference reports of all participants of the concerned talk. In order to enable this, it is possible to put the annotation in the `publicAnnotations` collection.

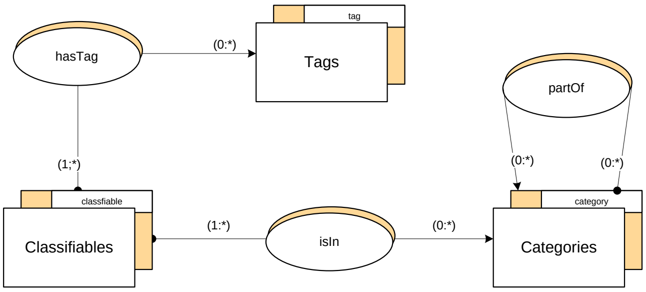


Figure 5.8: Tagging and categorisation sub-model

5.2.5 Full model

Figure 5.9 shows the whole model of the application. It combines the elements described in the last few paragraphs with two extra features: tagging and categorization, that can be seen on 5.8. Indeed, every `classifiable` object can have a `tag` associated to it as well as a `category`. A `category` can be part of another `category`. This allows the creation of a `category` hierarchy.

The application is now fully specified through the model. We have made it as general and flexible as possible, but we will impose some restrictions on the model through the application design that will be described in the next chapter.

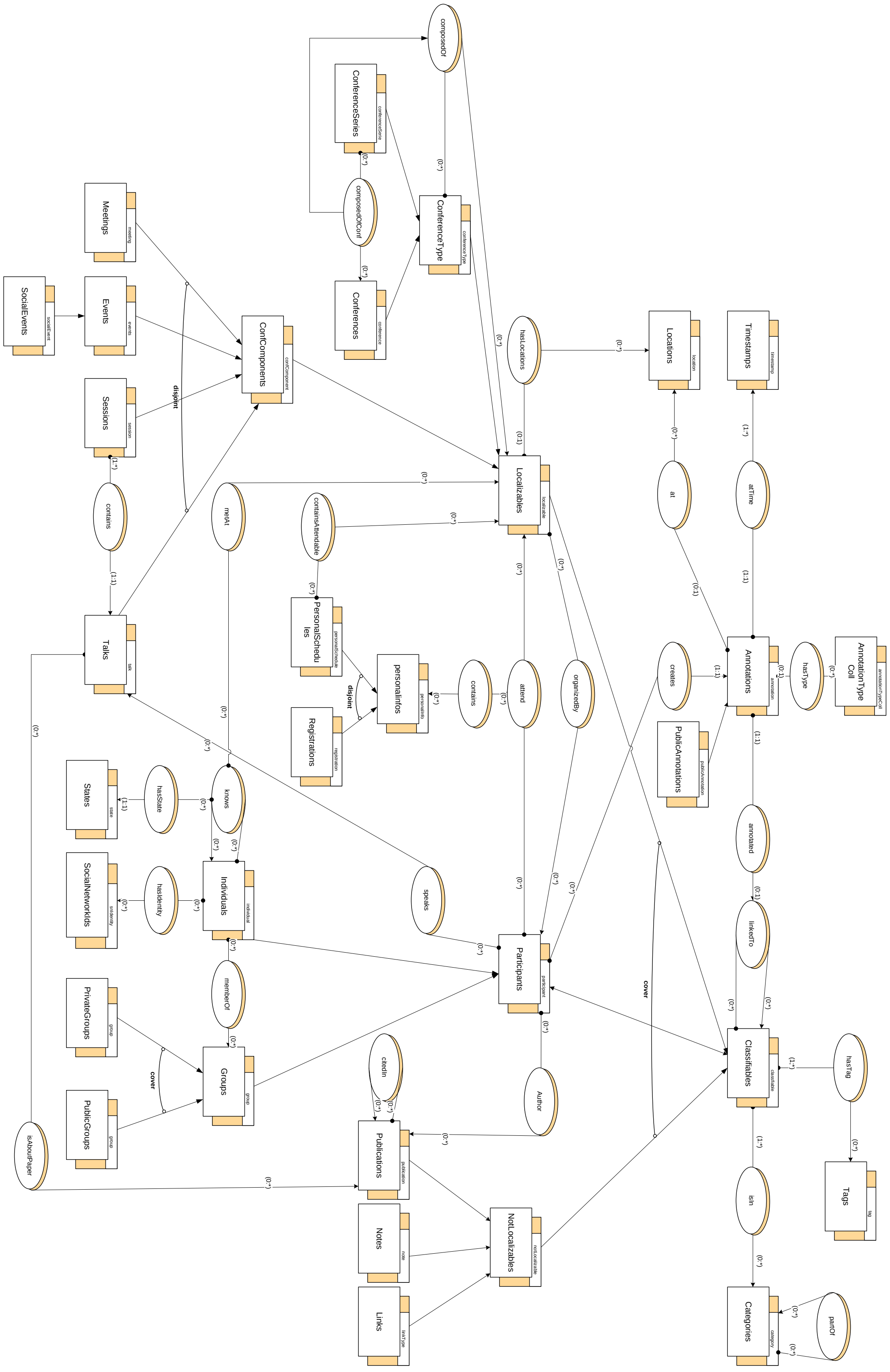


Figure 5.9: Application model

6

Implementation

6.1 Background information

This chapter describes the concrete implementation of our application based on the requirements defined in chapter 5. This first section entitled 'background information' provides an introduction to the two technologies we used, namely Avon and OpenLaszlo.

6.1.1 Avon

Avon [23] is the name of the database implementing the OM data model. It has several storage implementations, one of them using the object-oriented database db4o[1]. Avon is based on a layered architecture that includes a storage layer, a model layer and an interface layer. The storage layer handles the persistence of the database. Although it is implemented on top of db4o, Avon defines its own concepts for storage and provides two components: the storage management interface used by the model layer to store, retrieve, update and delete objects as well as the query engine. The query engine receives the queries given in by the model layer, optimizes and processes them and then returns the results. The middle layer, exposes the OM concepts with the Java object model. It also handles transactions as well as OML. OML is the object model language defined on top of the OM model that provides an algebraic query language, a data definition and manipulation language. The top layer, the interface layer, provides a programming interface to the database for client applications.

[1] http://www.db4o.com/

6.1.2 OpenLaszlo

Our application is provides a web based user interface. Due to the increasing use of rich Internet applications (RIAs), users expect the web applications they use to be reactive. Therefore, in order to provide a good user experience, we decided to build a rich Internet application. A RIA is a web application that has characteristics of desktop applications. Several frameworks, such as *JavaFX*[2], *Adobe Flex*[3], *Silverlight*[4] or *OpenLaszlo*[5] provide support for developing RIAs. Each framework has its advantages and drawbacks. For example, at the beginning of the development, *JavaFX* was still in an alpha stage and was therefore not a real option for us. Finally, we decided to use the open source framework *OpenLaszlo* [33].

OpenLaszlo is separated into two components: the *OpenLaszlo* server and the Laszlo LZX scripting language. Applications in *OpenLaszlo* are designed using a declarative language that is based on XML. The development is done in a text-editor such as notepad or Spket IDE [6] which provides syntax highlighting for LZX files, amongst others. The development cycle of an *OpenLaszlo* application can be split up into two parts: the development and the deployment. During the development phase, an *OpenLaszlo* server is needed that compiles the LZX files into Flash or DHTML, which are the two output formats currently supported. The advantage of this system is that LZX is a format that is independent of the output format. Currently, Flash and DHTML are supported, but we could imagine in the future there could be others. In our case, the *OpenLaszlo* server is a servlet running on a tomcat server. Once the development is finished, the application needs to be deployed. There are two deployment modes provided by *OpenLaszlo*: SOLO and server. SOLO deployment means that the application will run as a Flash or DHTML application independently from any *OpenLaszlo* server. In contrast, for some features, it is required to have an *OpenLaszlo* server running and the application needs to be deployed with a server. Once the application is deployed, it communicates with a server application through HTTP requests. The data sent from the server to the client application needs to be in XML format as *OpenLaszlo* supports a partial implementation of XPath [7] for handling data.

6.2 Conference information system architecture

The conference information system architecture, shown in figure 6.1, is made up of two components: the client and server. The client side component is implemented using *OpenLaszlo* and deployed as a SOLO Flash application. The server side is composed of two layers. The first layer contains the application logic and the second handles the data access. The different layers and components will be discussed in more detail in the sections that follow.

[2] http://javafx.com/

[3] http://www.adobe.com/products/flex/

[4] http://silverlight.net/

[5] http://www.openlaszlo.org/

[6] http://spket.com/

[7] http://www.w3.org/TR/xpath

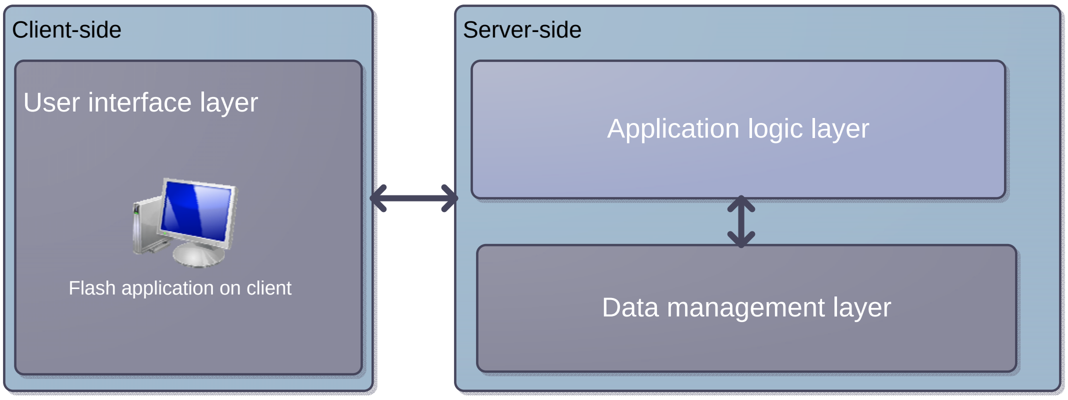


Figure 6.1: Application architecture

6.3 Server-side component

As seen in figure 6.1, the server side of the conference information system is made up of two layers. Both of these layer are implemented in Java and run on the same Tomcat server[8]. We will start to look at the data management layer followed by the application logic layer.

6.3.1 Data management layer

The lowest level of the application architecture is the data management layer. This layer uses the Avon database and provides several data access and manipulation methods for the application layer, as shown by the class diagram in figure 6.2. The `Database` class is used for setting up, opening and closing the database. In order to provide an abstraction from the database to the application layer, each type represented in the database has its own Java class inheriting from the `DatabaseClass` class. The `DatabaseClass` class defines an `objectId` attribute of type string. The `StringAccessDatabase` class will use these ids to retrieve OMObjects from the database. A restriction imposed by Avon is that there can only be one transaction running at the same time. Therefore, the `StringAccessDatabase` class has a second role, it ensures that only one database method can be executed simultaneously, thereby assuring only one transaction runs at any time. This is achieved by making the `StringAccessDatabase` class a singleton with synchronized methods. After the class has transformed the ids into OMObjects, it will call methods from the `ObjectAccessDatabase` classes which will then access the database.

[8] http://tomcat.apache.org

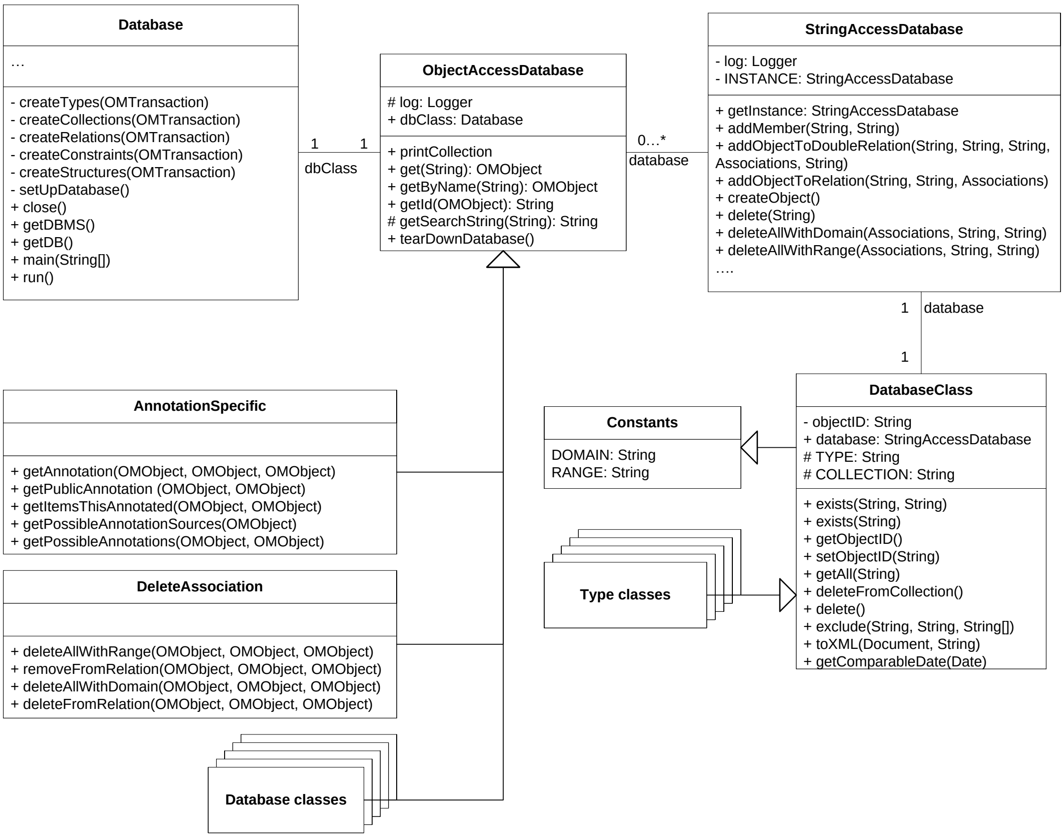

Database
…
- createTypes(OMTransaction)
- createCollections(OMTransaction)
- createRelations(OMTransaction)
- createConstraints(OMTransaction)
- createStructures(OMTransaction)
- setUpDatabase()
+ close()
+ getDBMS()
+ getDB()
+ main(String[])
+ run()
1
1
dbClass
ObjectAccessDatabase
# log: Logger
+ dbClass: Database
+ printCollection
+ get(String): OMObject
+ getByName(String): OMObject
+ getId(OMObject): String
# getSearchString(String): String
+ tearDownDatabase()
0…*
database
StringAccessDatabase
- log: Logger
- INSTANCE: StringAccessDatabase
+ getInstance: StringAccessDatabase
+ addMember(String, String)
+ addObjectToDoubleRelation(String, String, String, Associations, String)
+ addObjectToRelation(String, String, Associations)
+ createObject()
+ delete(String)
+ deleteAllWithDomain(Associations, String, String)
+ deleteAllWithRange(Associations, String, String)
….
1
database
1
DatabaseClass
- objectID: String
+ database: StringAccessDatabase
# TYPE: String
# COLLECTION: String
+ exists(String, String)
+ exists(String)
+ getObjectID()
+ setObjectID(String)
+ getAll(String)
+ deleteFromCollection()
+ delete()
+ exclude(String, String, String[])
+ toXML(Document, String)
+ getComparableDate(Date)
Constants
DOMAIN: String
RANGE: String
Type classes
AnnotationSpecific
+ getAnnotation(OMObject, OMObject, OMObject)
+ getPublicAnnotation (OMObject, OMObject)
+ getItemsThisAnnotated(OMObject, OMObject)
+ getPossibleAnnotationSources(OMObject)
+ getPossibleAnnotations(OMObject, OMObject)
DeleteAssociation
+ deleteAllWithRange(OMObject, OMObject, OMObject)
+ removeFromRelation(OMObject, OMObject, OMObject)
+ deleteAllWithDomain(OMObject, OMObject, OMObject)
+ deleteFromRelation(OMObject, OMObject, OMObject)
Database classes

Figure 6.2: Data management layer

6.3.2 Application layer

This second layer provides the functionality for the application. It uses the classes from the data management layer to generate XML which is used by the visualization layer to render the client view.

Incoming request handling

Client requests are handled by a single servlet that forwards the requests to the appropriate class by examining the passed arguments. The class diagram in figure 6.3 shows the structure of the server-side component. The servlet handling all incoming requests is called `Application`. When a requests is received, it creates a new instance of the `ApplicationManager` class. The `ApplicationManager` class, through Java reflection, calls the appropriate child class of `ServletFunctionality`. The abstract `ServletFunctionality` class defines general functionality for all application classes. For example, when necessary, it checks whether the user is logged in, as well as whether all compulsory arguments have been provided. Only then, the `mainFunctionality` method of the application class is called. The application classes are all classes that provide

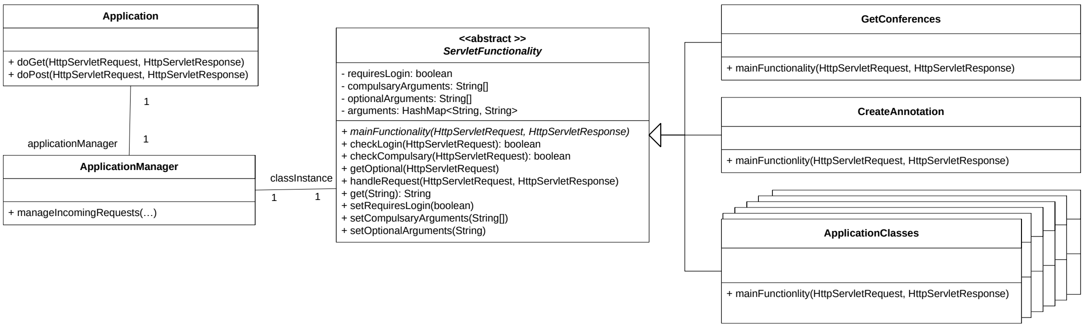

Application
+ doGet(HttpServletRequest, HttpServletResponse)
+ doPost(HttpServletRequest, HttpServletResponse)
1
applicationManager
1
ApplicationManager
+ manageIncomingRequests(…)
classInstance
1
1
<<abstract >>
ServletFunctionality
- requiresLogin: boolean
- compulsaryArguments: String[]
- optionalArguments: String[]
- arguments: HashMap<String, String>
+ mainFunctionality(HttpServletRequest, HttpServletResponse)
+ checkLogin(HttpServletRequest): boolean
+ checkCompulsary(HttpServletRequest): boolean
+ getOptional(HttpServletRequest)
+ handleRequest(HttpServletRequest, HttpServletResponse)
+ get(String): String
+ setRequiresLogin(boolean)
+ setCompulsaryArguments(String[])
+ setOptionalArguments(String)
GetConferences
+ mainFunctionality(HttpServletRequest, HttpServletResponse)
CreateAnnotation
+ mainFunctionlity(HttpServletRequest, HttpServletResponse)
ApplicationClasses
+ mainFunctionlity(HttpServletRequest, HttpServletResponse)

Figure 6.3: Application classes

some functionality requested by the client application. For example, one class gets all conferences, another creates an annotation, etc. All the application classes are situated in the `ch.ethz.globis.networking.application` package.

Report generation

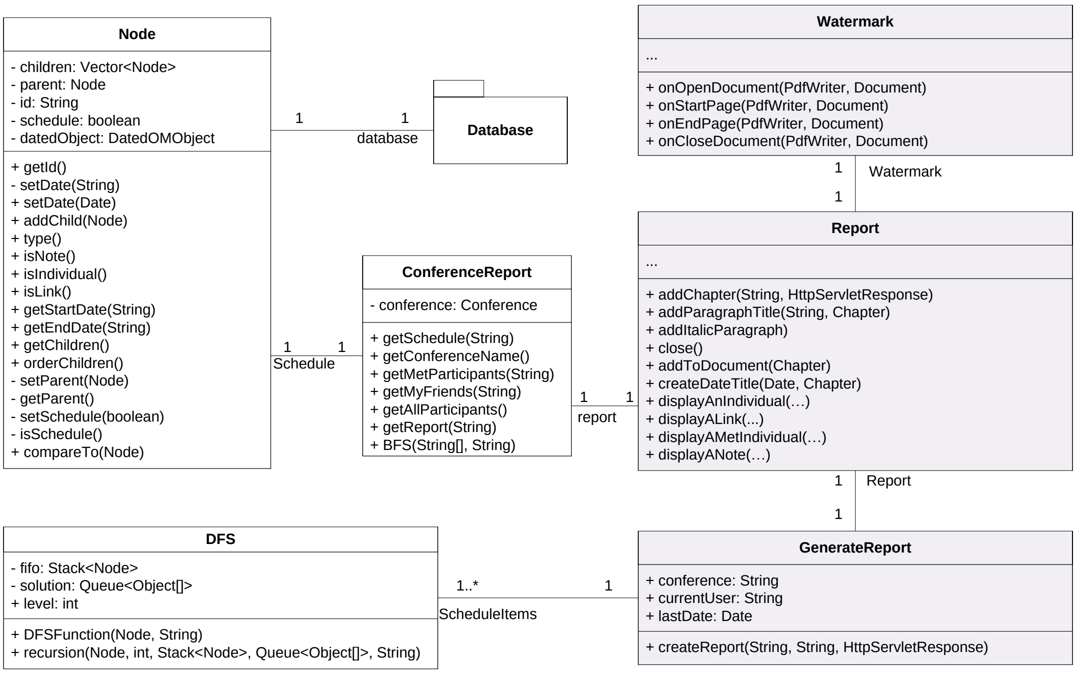


Figure 6.4: Conference report generation

The generation of a conference report is one of the core functionalities of our social networking application. The conference report contains several sections. One section is a page that gives a general description of the conference, another section shows the schedule of the participant. The schedule not only contains the sessions, meetings and events the participants have attended, but also their personal annotations. These annotations could be of any nature, but currently only personal notes, links and encounters with other people are displayed. There are two classes of annotations: public and private. Public annotations can only be made by the organizers of a certain event and will be displayed on the conference reports of all participants that attended the latter event. Public annotations could, for example, be links to slides or pictures of the event. Another section contains a list of the new acquaintances a participant made during the conference, followed by a list of all participants. The report is generated on the server-side and sent back to the client as a PDF. An example of such a report can be found in appendix D.

The report is generated as a PDF file with the help of a free Java library called iText[9]. This library allows to create PDFs directly from within a Java program and can then be sent to the client through HTTP. The class diagram in figure 6.4 shows on one side, in light purple, classes that are responsible for the generation of the document itself, and on the other side, supporting classes that retrieve the necessary information

Schedule The schedule is the central part of the conference report. In order to retrieve the necessary information from the database, we construct a tree containing that information. We developed our own data structure to support the creation of such a tree. The `Node` class provides all the necessary information to create a tree. In a first step, the tree is constructed with the help of an algorithm similar to breadth first search with cycle detection. The code can be seen in code listing 6.1. In summary, it constructs a tree from an initial collection (in our case, the schedule items), by looking for each elements children. An example of such a tree can be seen in figure 6.5. After the tree is constructed, we can see that the children of each node still need to be ordered. This is done recursively by the `DFS` class. The `DFS` class traverses the tree and while doing that orders the children of each node it encounters according to the nodes timestamps. At this point, we would have a tree similar to the one in figure 6.6 and the corresponding PDF can be easily rendered.

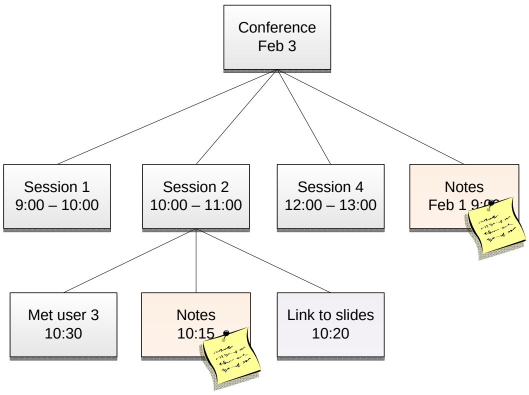


Figure 6.5: Tree with unordered children

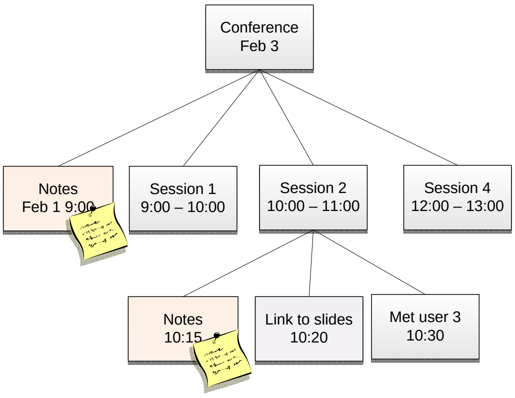


Figure 6.6: Tree with ordered children

[9] http://www.lowagie.com/iText/

```
public Node BFS(String[] initialCollection, String currentUser) {

      //first include the conference itself
      Node conf = new Node(conference.getObjectID(), true, currentUser);

      //initialization
      LinkedList<String> bfs = new LinkedList<String>();
      Vector<String> cycleCheck = new Vector<String>();
      HashMap<String, Node> nodes = new HashMap<String, Node>();

      //insert root element
      bfs.add(conference.getObjectID());
      cycleCheck.add(conference.getObjectID());
      nodes.put(conference.getObjectID(), conf);

      //retrieve the schedule and insert all items
      if(Utility.isDefined(initialCollection)) {
         //insert the schedule items
         for(String item: initialCollection) {
            bfs.add(item);
            cycleCheck.add(item);
            Classifiable itemC = new Classifiable(item);
            nodes.put(item, new Node(conf, itemC.getObjectID(), true,
            currentUser));
         }
      }

      while(!bfs.isEmpty()) {
         String currentNode = bfs.removeFirst();

         //get neighbours
         Classifiable current = new Classifiable(currentNode);
         Vector<String> neighbours = current.getLinkedTos(currentUser);
         Vector<String> publicNeighbours = current.getPublicLinkedTos();
         neighbours.addAll(publicNeighbours);
         neighbours = Utility.removeDuplicates(neighbours);

         //go through all neighbours
         if(neighbours != null && neighbours.size()>0) {
            for(String neighbour: neighbours) {
               if(!cycleCheck.contains(neighbour)) {
                  cycleCheck.add(neighbour);
                  bfs.add(neighbour);
               }
               Classifiable neigh = new Classifiable(neighbour);
               nodes.put(neighbour, new Node(nodes.get(currentNode),
               neigh.getObjectID(), false, currentUser));
            }
         }
      }
      return conf;
   }
}
```

Listing 6.1: Tree generation

Helper classes

In this section, we describe a few helper classes we created. These classes do not implement the functionality of the application, but are necessary for it to run efficiently.

utility package This package contains three classes: `Utility`, `UserLoginUtility` and `DateUtility`, it can be noted that all their methods that are public and static.

The first class is the `Utility` class. This class offers general methods such as transforming comma-separated strings to arrays, adding an element to an array or checking whether a string returned by the openlaszlo application is valid. The second class is the `UserLoginUtility` class. This class offers two methods: one checking whether a user is currently logged in and the other getting the id of the user that is currently logged in. Finally, the `DateUtility` class transforms dates into different formats. For example, it can transform a date object into a string of the following format: 10 Feb 2009. Furthermore, it contains methods for parsing strings into dates.

xml package The XML package contains classes that generate the XML that will be sent from the server to the openlaszlo client application. It is composed of four classes that can be seen in the UML diagram in figure 6.7. The `Generator` class is called directly by the servlets and generates the XML that is sent when a true or false response is generated. For example, when a user logs in, the application either sends back a message telling the application whether the login was successful or not. The XML generated in case of a success can be seen in figure 6.8, and the one generated in case of a failure in figure 6.9. As can be seen, in case of a failure, a message is sent. This message can then be displayed to the user.

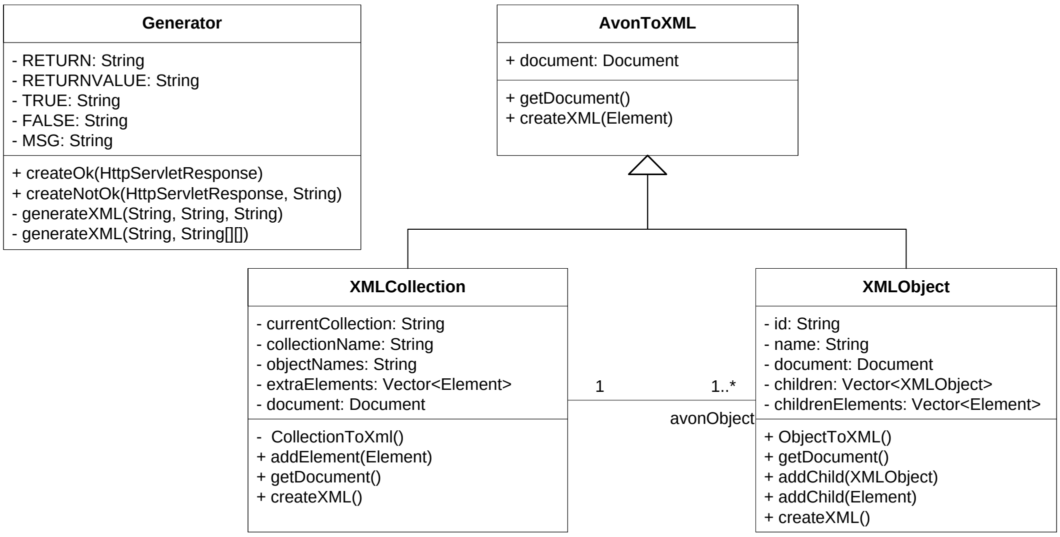
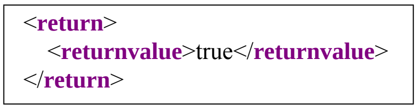
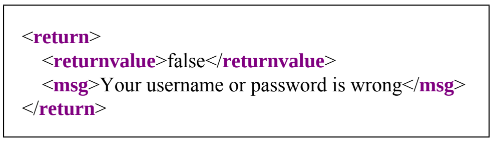


Figure 6.7: Classes used for the generation of xml



```
<return>
  <returnvalue>true</returnvalue>
</return>
```

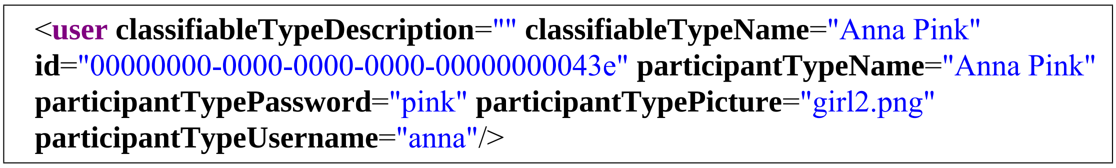
Figure 6.8: XML returned when an operation is successful



```
<return>
  <returnvalue>false</returnvalue>
  <msg>Your username or password is wrong</msg>
</return>
```

Figure 6.9: XML returned when an operation fails

The second kind of response that can be generated automatically is based on objects from the database. The main class defining that functionality is called `AvonToXml`. Currently, there are two implementations of automatic XML generation: collections and objects. The class implementing the serialization of an object is `XMLObject`. The `XMLObject` class automatically retrieves the types an object is dressed with as well as its attributes. As an example, figure 6.10 shows the XML generated for a user. As can be seen, the user is dressed with two types: `Classifiable` and `Participant`. The attribute names are prefixed with their type in order to prevent name clashes. If we take the attribute CLASSIFIABLETYPENAME, it means the attribute is called `Name` and the type `Classifiable`. Similarly, the generation of a collection of objects is done by the `XMLCollection` class. This class takes all objects in a collection and renders their xml representation with the help of the `XMLObject` class.



```
<user classifiableTypeDescription="" classifiableTypeName="Anna Pink"
id="00000000-0000-0000-0000-00000000043e" participantTypeName="Anna Pink"
participantTypePassword="pink" participantTypePicture="girl2.png"
participantTypeUsername="anna"/>
```

Figure 6.10: XML representation of a user

GeoIP package and locationLookup package Our application supports the creation of annotations. We saw in section 5.2.4 that annotations contain information about their creation time as well as their location. While the detection of the time of creation is trivial, it is more challenging to attach a meaningful location to an annotation. In the future, we might consider creating annotations with the help of sensors, but currently the only information we have about the user is his IP address. The `GeoIP` package contains the free localisation API provided by MaxMind[10]. This API, in combination with a database file situated in the resources folder, allows to recognize the country and city of a user's IP address. This information is then used for the creation of a location that can be attached to an annotation.

6.4 Summary: a data flow example

In order to summarize all concepts and classes we have seen in the last pages, figure 6.11 shows the example of a sequence diagram. It represents the method invocations necessary for retrieving all conferences from the database.

[10] http://www.maxmind.com/

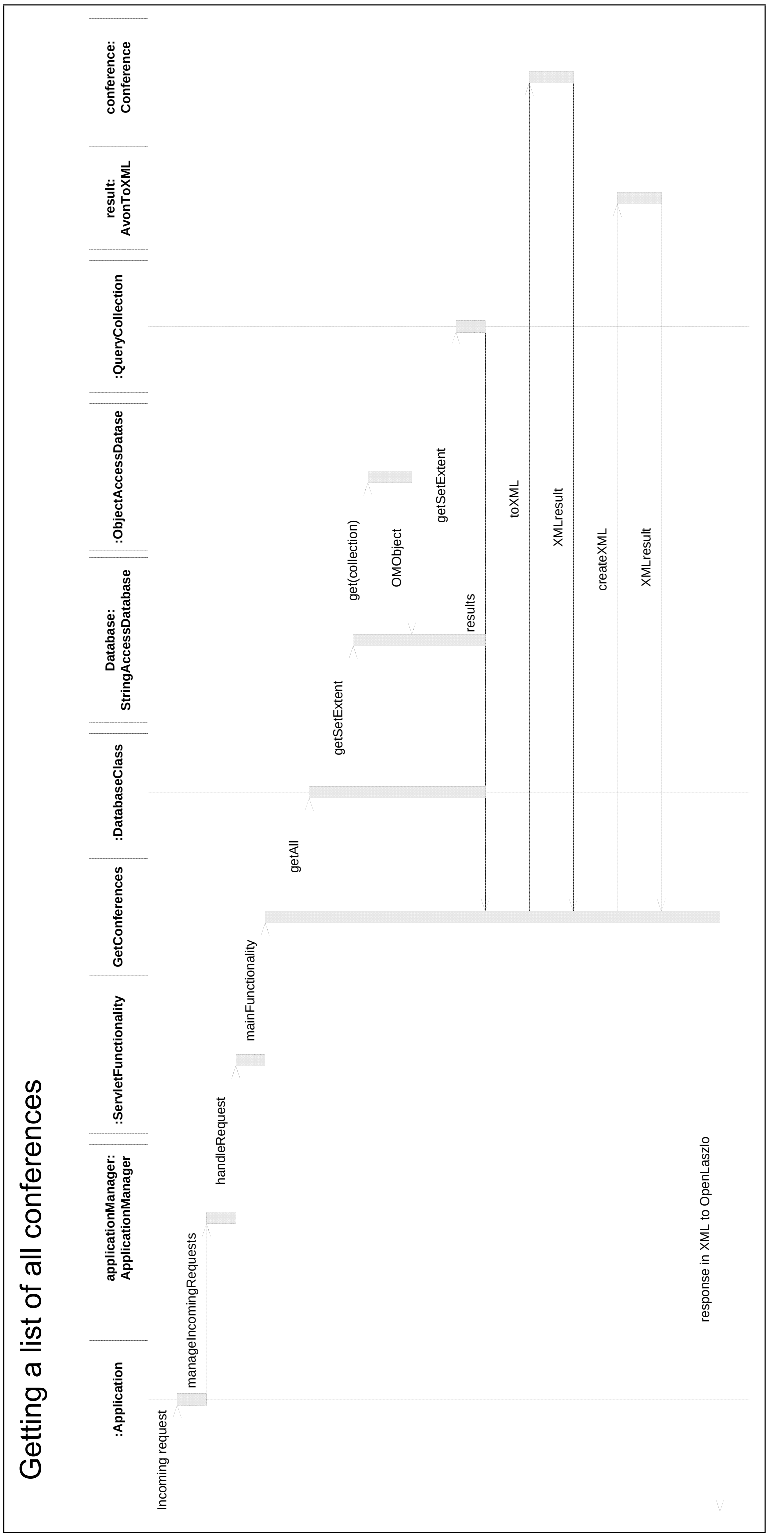


Figure 6.11: Sequence diagram

6.5 Client side component

The client side component is written in OpenLaszlo and then compiled into a flash application that can be embedded in any browser.

A web application written in OpenLaszlo should be seen as similar to a desktop application and not a typical website. Indeed, the application is not split up into pages as would be a website. The main application structure, as can be seen in figure 6.12, has the `Main.lzx` file as an entry point. This file includes the `Library.lzx` file which takes care of including all necessary classes. The classes included can be sub-divided into four categories: datasets, application classes, design classes and resources.

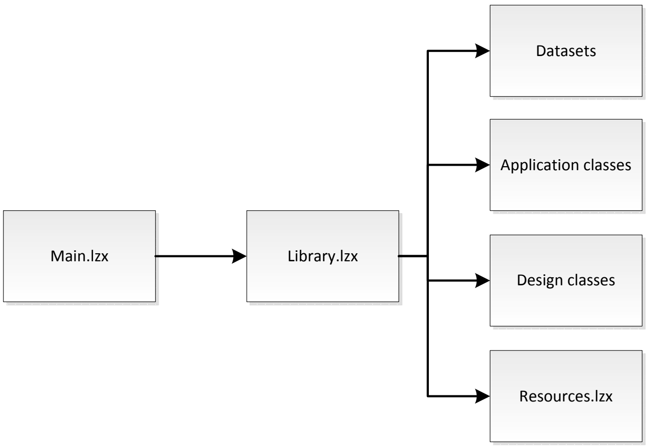


Figure 6.12: OpenLaszlo classes

In OpenLaszlo, a dataset is a reference to XML data. Generally, this data comes from the server. A dataset can be defined as follows:

```
<dataset name="GetReport" request="false"
src="http:../Application?function=GetConferenceReport" />
```

Listing 6.2: dataset definition

In order to update the data from the server, a request is done on a dataset. When something goes wrong, the operation times out or an error is thrown. In our application, we implemented a class, `myDataset`, which handles these cases. `myDataset` is an extension of the standard OpenLaszlo dataset and is defined as show in listing 6.3.

From this code snippet, we can see how classes in OpenLaszlo are similar to classes of any object-oriented language. The class tag defines a class with a certain name. In this case, the class tag also defines the class we want to extend. The `myDataset` contains a method and three event handlers. An event handler is a method that is called when a certain event occurs. Furthermore, it is possible to create event handlers for any class attribute. For example, if an attribute `height` is defined for a class, one could create the handler `onheight` that would be called automatically every time the height changes. Method behaviors are implemented

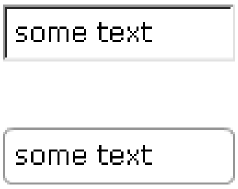
```
<class name="myDataset" extends="dataset">


            <!--- does the request -->
            <method name="doMyRequest">
                  this.doRequest();
                  loadingManager.start();
            </method>

            <!--- the request has successfully ended -->
            <handler name="ondata">
                  loadingManager.end();
            </handler>

            <!--- the request had en error -->
            <handler name="onerror">
                  canvas.datasetError.open();
                  Debug.write("error:␣"+this);
                  loadingManager.end();
            </handler>

            <!--- the request timed out -->
            <handler name="ontimeout">
                  canvas.datasetError.open();
                  Debug.write("timed␣out:␣"+this);
                  loadingManager.end();
            </handler>


      </class>
```

Listing 6.3: myDataset

in a language close to JavaScript. The main difference with JavaScript is the way objects are accessed. Indeed, `loadingManager.end()` would correspond to the following JavaScript statement: `document.getElementById('loadingManager').end()`. The functionality the `myDataset` class provides is the following: when a request is done (by calling the `doMyRequest` method), the request is done, and the `loadingManager` is started. The `loadingManager` is just a class with a count attribute. Every time a request is started, the count attribute is incremented and when the request is done, the count attribute is decremented. This allows us to show a loading symbol while data is loading from the server. All requests in OpenLaszlo are asynchronous, and trigger an event when they are finished. There are three event types that can be triggered by a request: `data`, `timeout` and `error`. The `myDataset` class catches these events and decrements the `loadingManager` count attribute while opening an error dialog if necessary. The `myDataset` class has been described in detail here. We will not do this for the other classes as it is not the goal to provide a tutorial for OpenLaszlo.

Application classes are classes that define the behavior of the application. For example, `Login.lzx` defines the login process. It not only creates the text fields for entering the user name and password, but also handles the data submission to server.

The third type of classes are design classes. These classes are situated in the `design` folder. They define the customized look and feel of components. For example, the `roundedittext` class defines an input field with rounded corners. Figure 6.13 shows a standard OpenLaszlo input field as well as the personalized used in our application.



some text

some text

Figure 6.13: Input field from OpenLaszlo (top) and personalized (bottom, with rounded corners)

6.5.1 Usage

This section is intended to give an overview of the final application. It will go through the implemented functionality step-by-step, illustrated with screen shots. The currently available actions are shown in figure 6.14.

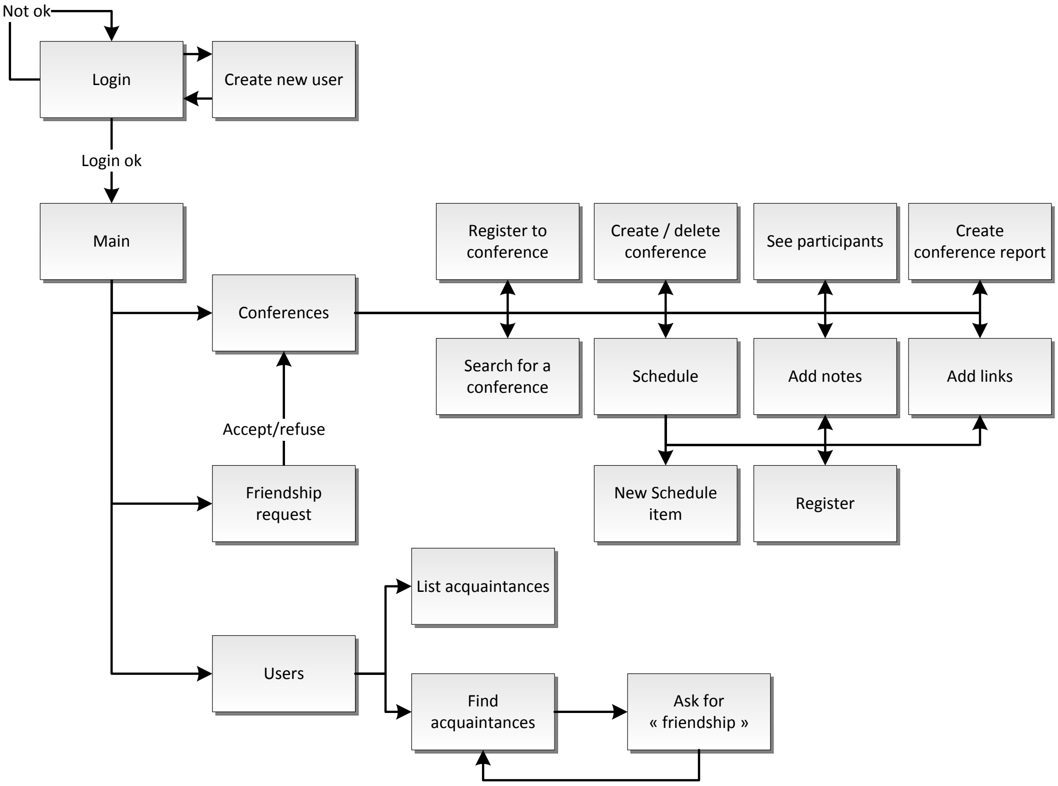


Figure 6.14: Currently implemented functionality

The entry point to the application is the login screen as shown in the upper part of figure 6.15. From here, it is possible to either create a new user account (figure 6.16) or for existing users to access to the main application.

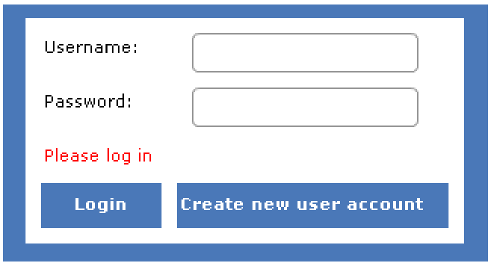


Figure 6.15: Login screen

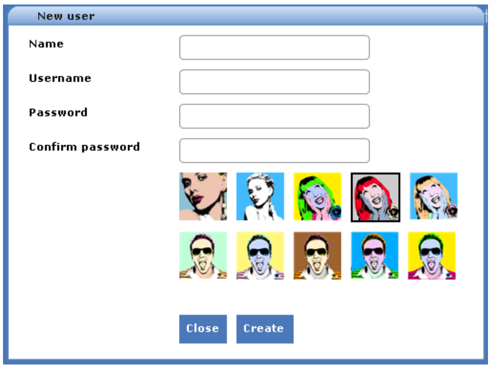
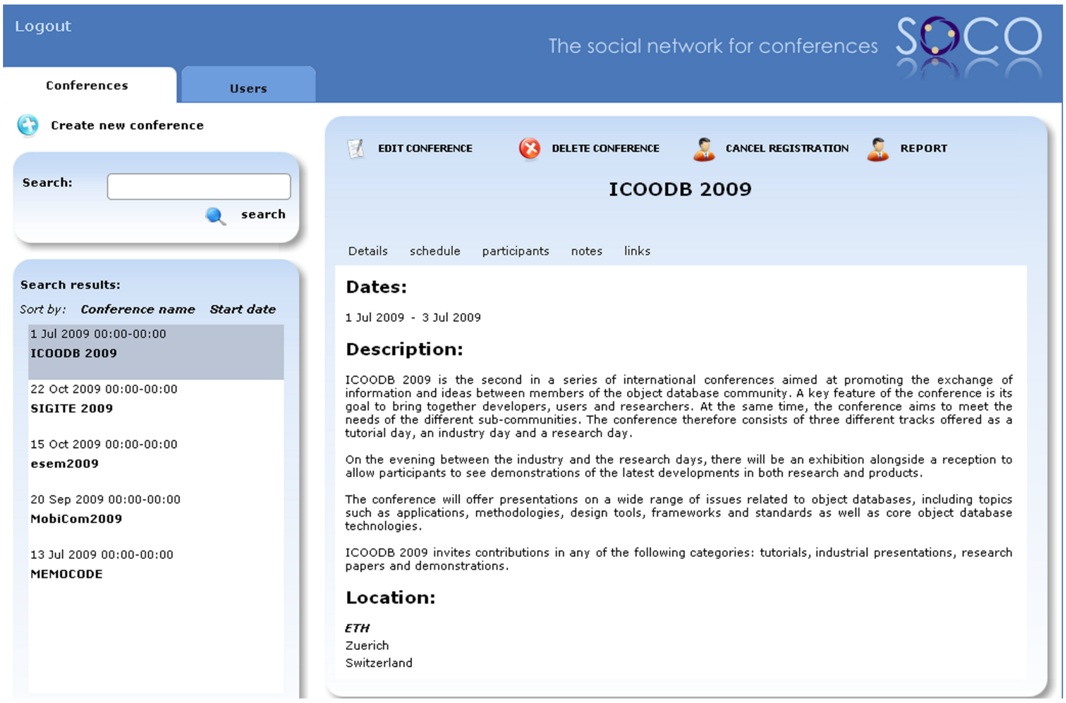


Figure 6.16: New User creation

Figure 6.17 shows the main screen of the application. On the top there is a series of tabs (currently only `Users` and `Conferences`) which allow users to access to the main parts of the application.



1 Jul 2009 00:00-00:00
ICOODB 2009

22 Oct 2009 00:00-00:00
SIGITE 2009

15 Oct 2009 00:00-00:00
esem2009

20 Sep 2009 00:00-00:00
MobiCom2009

13 Jul 2009 00:00-00:00
MEMOCODE

ICOODB 2009

Dates:

1 Jul 2009 - 3 Jul 2009

Description:

ICOODB 2009 is the second in a series of international conferences aimed at promoting the exchange of information and ideas between members of the object database community. A key feature of the conference is its goal to bring together developers, users and researchers. At the same time, the conference aims to meet the needs of the different sub-communities. The conference therefore consists of three different tracks offered as a tutorial day, an industry day and a research day.

On the evening between the industry and the research days, there will be an exhibition alongside a reception to allow participants to see demonstrations of the latest developments in both research and products.

The conference will offer presentations on a wide range of issues related to object databases, including topics such as applications, methodologies, design tools, frameworks and standards as well as core object database technologies.

ICOODB 2009 invites contributions in any of the following categories: tutorials, industrial presentations, research papers and demonstrations.

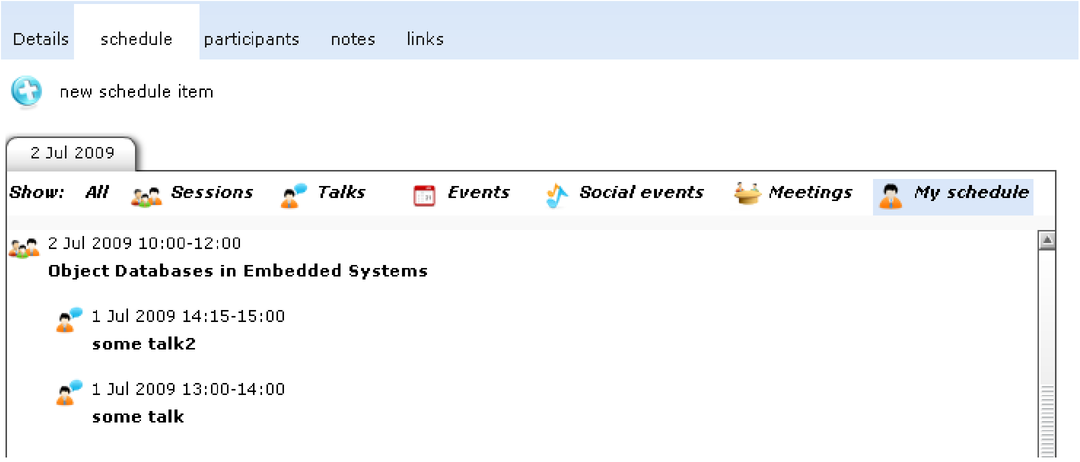
Location:

ETH
Zuerich
Switzerland

Figure 6.17: Main screen

Conferences tab

This first tab displays everything related to conferences. As can be seen in figure 6.17, there are two columns in this tab. The left column is used for searching and browsing through conferences, while the right column shows conference details. The conference shown on the screenshot was created by the user that is currently logged in, therefore he has access to functionality such as deleting and editing the conference. Furthermore, the user is currently registered to the conference and has a button for canceling this registration, as well as one to generate the conference report.



2 Jul 2009 10:00-12:00
Object Databases in Embedded Systems

1 Jul 2009 14:15-15:00
some talk2

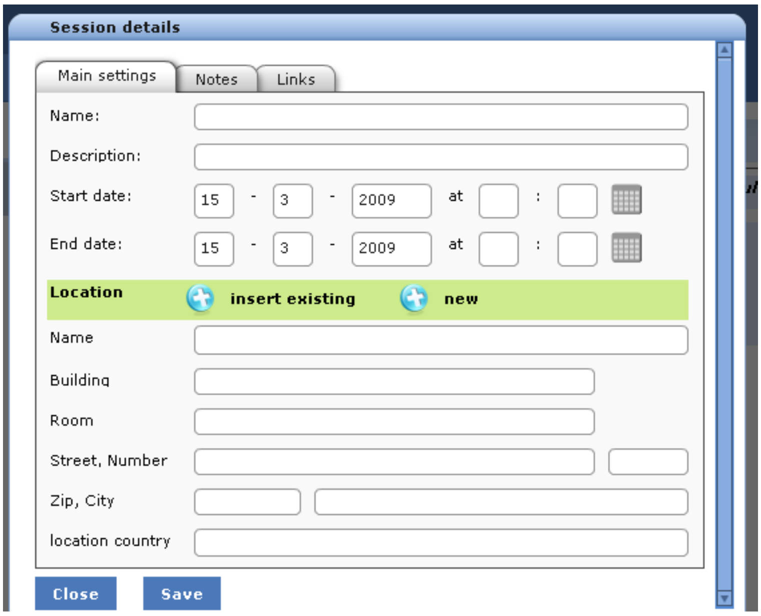
1 Jul 2009 13:00-14:00
some talk

Figure 6.18: Conference schedule

The detailed conference view contains several tabs: `details`, `schedule`, `participants`, `notes` and `links`. The details tab shows detailed information about the conference such as the location, dates and description. The schedule tab shows the conference schedule. An example of the schedule tab can be seen in figure 6.18. The conference organizer can add conference components through this interface, while participants can register for them. The items shown in dark blue are part of the participant's personal schedule. When a conference item is clicked, a window opens with details as well as the possibility to add notes and links. For example, a window for creating a new session is shown in figure 6.19.

The next tab is the participants tab. This tab just shows all participants that are registered for the conference.

The next two tabs; notes and links, allow users to create annotations for the conference. These annotations can also be added to conference components as we have seen in figure 6.19. Notes (figure 6.20) can be created by entering a note in the text field and submitting with the green arrow. All notes are personal and not shareable with other users. In contrast, links can be made personal by the conference organizer. The `public` checkbox on figure 6.21 will be only visible when the user that is currently logged in is the organizer.



Name:

Description:

Start date: 15 - 3 - 2009 at :

End date: 15 - 3 - 2009 at :

Name

Building

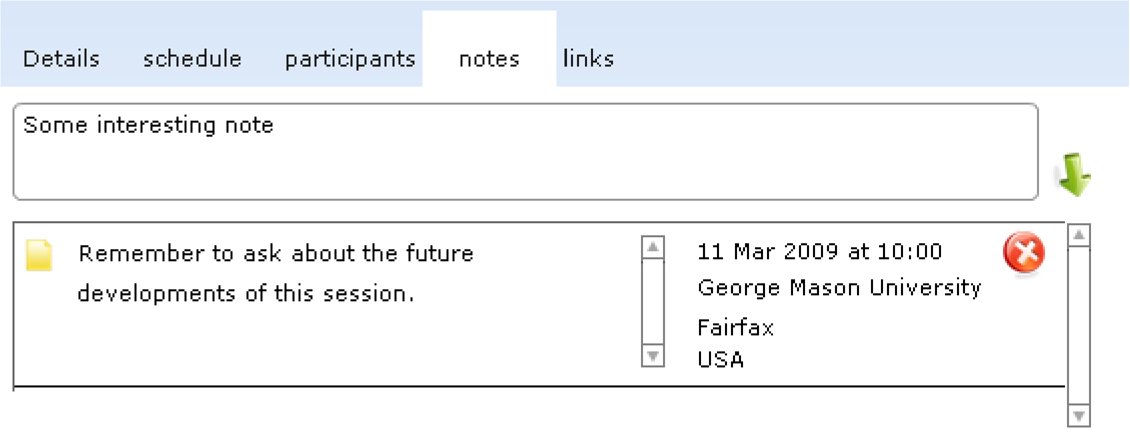
Room

Street, Number

Zip, City

location country

Figure 6.19: New session creation



Some interesting note

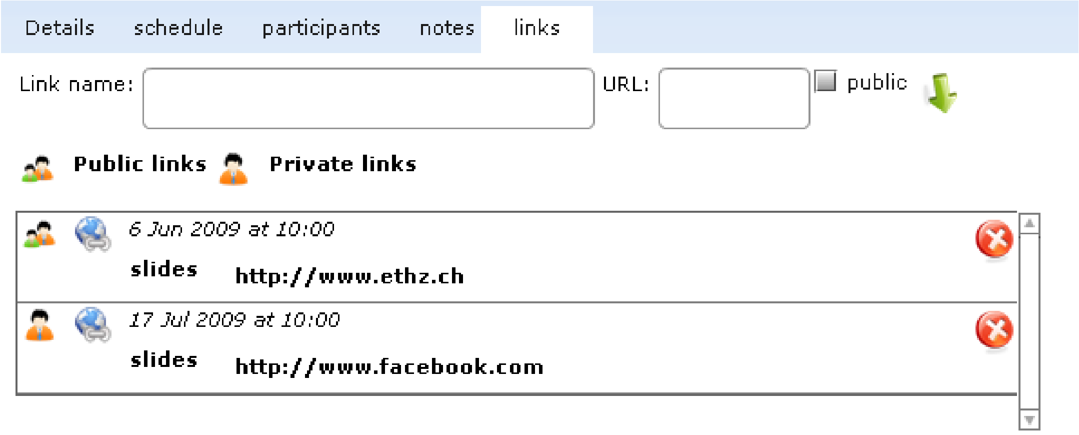
Remember to ask about the future developments of this session.

11 Mar 2009 at 10:00
George Mason University
Fairfax
USA

Figure 6.20: Manage notes

Users tab

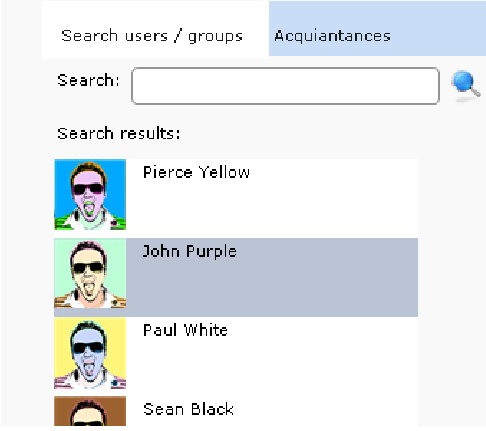
In the users tabs, one can find a list of all users (figure 6.22) in order to search for possible acquaintances. By clicking on a user in the list, a window pops up allowing to define the context in which the two users met. The system will automatically create a list of all events in common to the two users. An example can be seen on figure 6.23. Once a user requests another users, this latter user will see a notification on his page, as can be seen on figure 6.24. Finally, the user can decide to accept or refuse this new contact (figure 6.25).



6 Jun 2009 at 10:00
slides http://www.ethz.ch

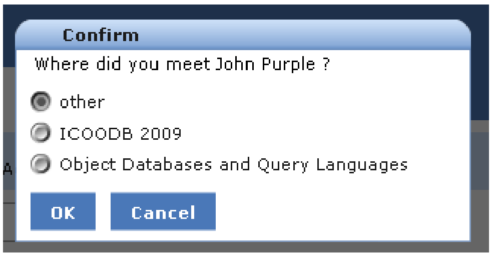
17 Jul 2009 at 10:00
slides http://www.facebook.com

Figure 6.21: Manage links



Pierce Yellow
John Purple
Paul White
Sean Black

Figure 6.22: Users list

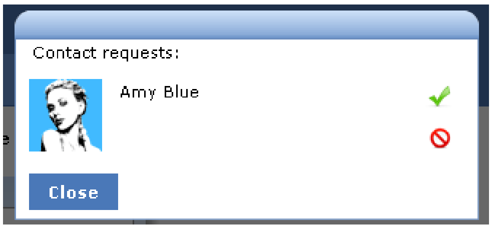


Where did you meet John Purple ?

● other
○ ICOODB 2009
○ Object Databases and Query Languages

Figure 6.23: Contact request

Figure 6.24: Notification



Amy Blue ✓ ⊘

Figure 6.25: Accept / refuse a new contact

7
Conclusion

This thesis had two main goals: on the one side determining the needs of professional computer scientists attending conferences and on the other hand, implementing a conference information system.

We started by doing a literature review in order to identify the research that has been done in this field as well as the existing commercial and non-commercial solutions. At the same time, we created a classification of different conference-related activities. We not only looked at activities of conference participants during a conference, but also investigated the pre- and post-conference phases. In a group discussion, we analyzed possible ideas for a new conference information system. We wanted to find an application that would accompany conference participants throughout their entire conference experience (before, during and after) as well as across all the conferences they attend. As a result, we decided to develop a social networking site for conferences. In order to validate our idea, we conducted an online survey amongst a group of computer science researchers and professionals. The survey confirmed the need for a social network for conference, as more than seventy percent of the respondents showed interest in our proposed system. An important result was that the generation of a conference report was the feature that was most requested. While systems such as nTag [2] provide the possibility to generate a so-called *trip report*, they are very dependent on the hardware that accompanies them. This means that in order to use this system, one has to equip all conference attendees with the nTag name badge.

In the second half of the thesis, we defined the requirements for a conference networking website, SoCo, and implemented the SoCo system. The particularity of our system is that it is a meta conference system. This means that a conference organizer can create their conference in the system and manage all conference-related information. Participants that are already member of SoCo can then just subscribe to a conference while always using the same platform. Over time, a user can have several roles, such as a conference participant, a conference organizer or a presenter of a talk. Depending on the role, users have access to a different set of actions. Furthermore, SoCo manages intra- and inter-conference information. For example, a schedule is attached to a single conference, while users' personal contacts will be managed across conferences. In the introduction, we defined the concept of scientific and complementary information. Scientific information will generally be managed in an intra-conference manner, while complementary information remains relevant throughout all conferences.

A very important feature is that the application is independent from any hardware for capturing events (where and when people meet, for example), but that it does not exclude the possibility to integrate such a system in the future. Currently, the main feature of the site is that it is able to generate automatically a personalized conference report. This report contains a participant's schedule, personal notes, links, other participants and new contacts that were made during the conference. This report can be used before a conference as it provides general information about the conference. During a conference, it provides a copy of the participants' schedules. After the conference, it is easy to go through notes and recall the whole conference. Thanks to the possibility to add links to conference components, it is possible to provide access to existing application. For example, Slideshare[1] could be used for posting slides, while Flickr[2] could be used for sharing pictures. This avoids redundant implementation of functionality.

7.1 Summary of contributions

Analysis of user needs for conference information systems This thesis provided a classification of activities related to conferences. Furthermore, we were able to identify the issues conference participants are confronted with, thanks to an online survey.

Design and implementation of a conference social network We developed a social networking system for conferences. We elaborated the specifications of such a system as well as a data model capable of supporting the latter specifications. In the implementation, we allowed users to automatically generate a conference report containing personalized, but also general information about the conference.

7.2 Future work

In the current state, the application provides administrator functionality for creating conferences and conference schedules as well as the generation of personalized conference reports. There are many extensions that could be made to the application.

[1] http://www.slideshare.net/

[2] http://www.flickr.com/

Currently, all annotations need to be made by hand. It would be interesting to develop a hardware infrastructure that would be capable of capturing events that would automatically be reflected within the application. For example, two people meeting each other could be captured with a device similar to the Pokens USB keys[3]. An alternative would be to provide a badge to all conference participants and sensors at different locations on the conference site. This would allow conference organizers to capture information such as who is asking a question during a question and answers session or what sessions a participant is attending. The introduction of such hardware could allow a lot of new functionality to be created that was discussed in the related work and user experience chapter.

The client-server architecture of our application allows for a lot of additional features. Indeed, communication with the server goes through HTTP requests and all server responses are XML. Therefore, it would be easy to augment the user interface with, for example, an interactive paper application. Interactive paper consists of normal paper that has a small, almost invisible pattern printed on it. A special pen that is equipped with an infrared camera allows to record all gestures coming from the pen. These gestures can then be recorded by a computer and used to trigger actions. This technology could be used for capturing handwritten notes. It is possible to imagine a system that captures handwritten notes as well as a corresponding timestamp. By cross-referencing this timestamp with the participants schedule, the notes could be attached to the session they correspond to.

As an elaboration to our social network, one could imagine integrating other social networks with SoCo. For example, it would be possible to import contacts from other social networks or export them.

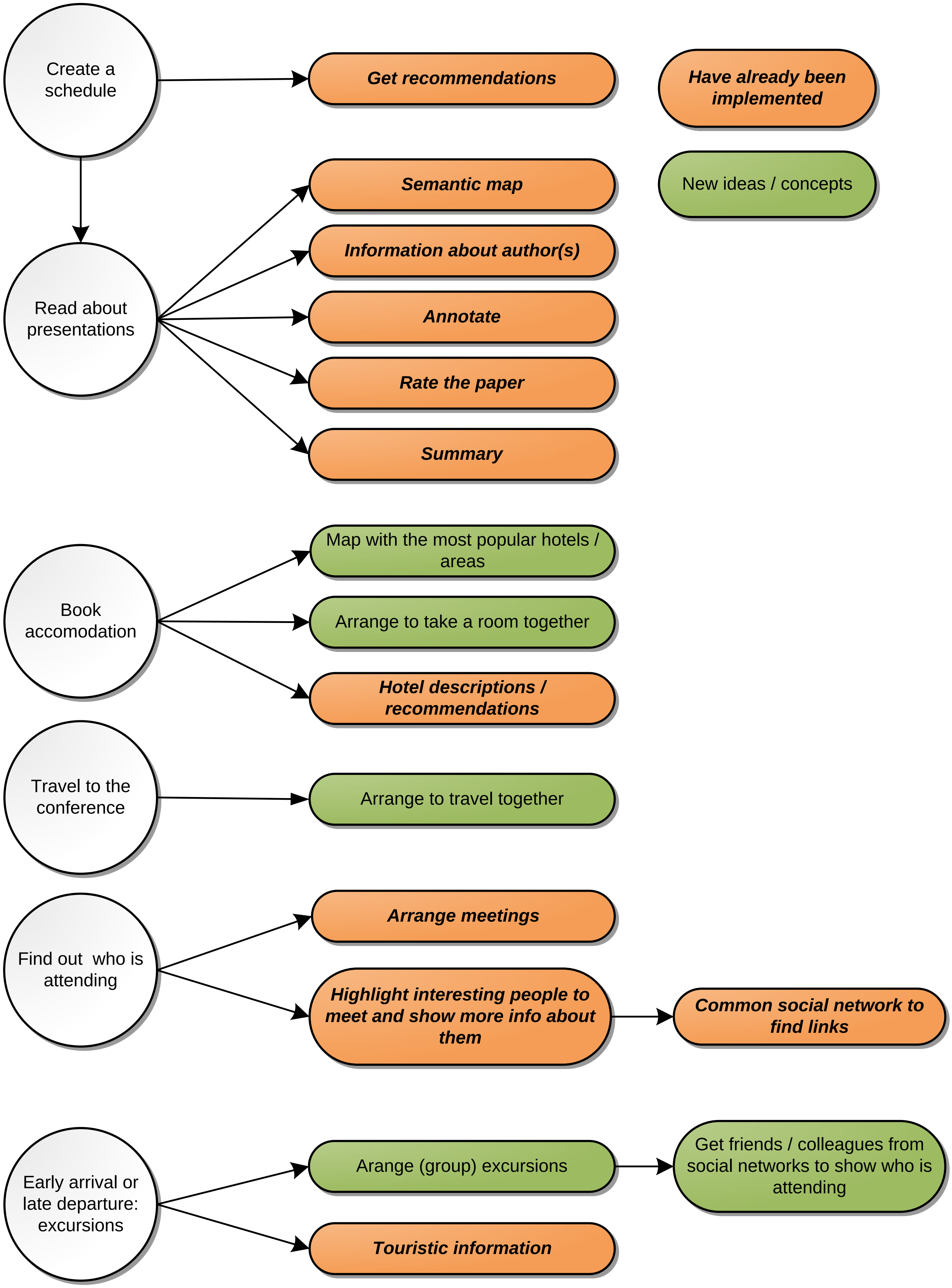
These were just a few possible extensions to our social network. Many of the functionalities described in the related work and user study chapters could be interesting to extend our application. Concerning the results of the user study, including the online survey, it would be interesting to extend it to other audiences and try to look at the differences and similarities between conferences in different fields.

[3] http://www.doyoupoken.com/

A

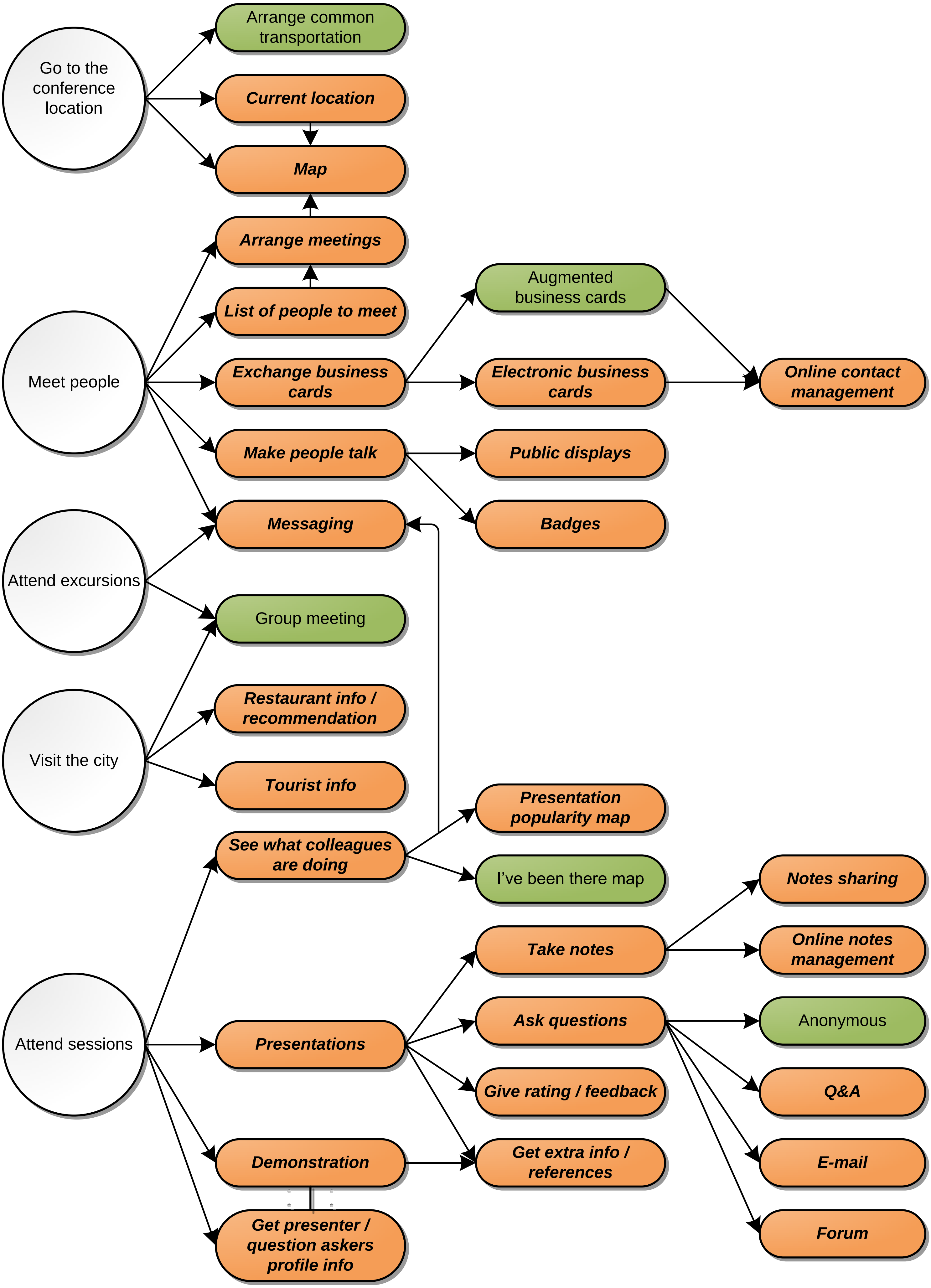
Typical conference activities

Before the conference: conference preparation



Create a schedule
Get recommendations
Have already been implemented
Semantic map
New ideas / concepts
Information about author(s)
Read about presentations
Annotate
Rate the paper
Summary
Map with the most popular hotels / areas
Book accomodation
Arrange to take a room together
Hotel descriptions / recommendations
Travel to the conference
Arrange to travel together
Arrange meetings
Find out who is attending
Highlight interesting people to meet and show more info about them
Common social network to find links
Get friends / colleagues from social networks to show who is attending
Early arrival or late departure: excursions
Arange (group) excursions
Touristic information

During the conference



Go to the conference location

Arrange common transportation

Current location

Map

Meet people

Arrange meetings

List of people to meet

Exchange business cards

Augmented business cards

Electronic business cards

Online contact management

Make people talk

Public displays

Badges

Messaging

Attend excursions

Group meeting

Visit the city

Restaurant info / recommendation

Tourist info

See what colleagues are doing

Presentation popularity map

I've been there map

Attend sessions

Presentations

Take notes

Notes sharing

Online notes management

Ask questions

Anonymous

Q&A

E-mail

Forum

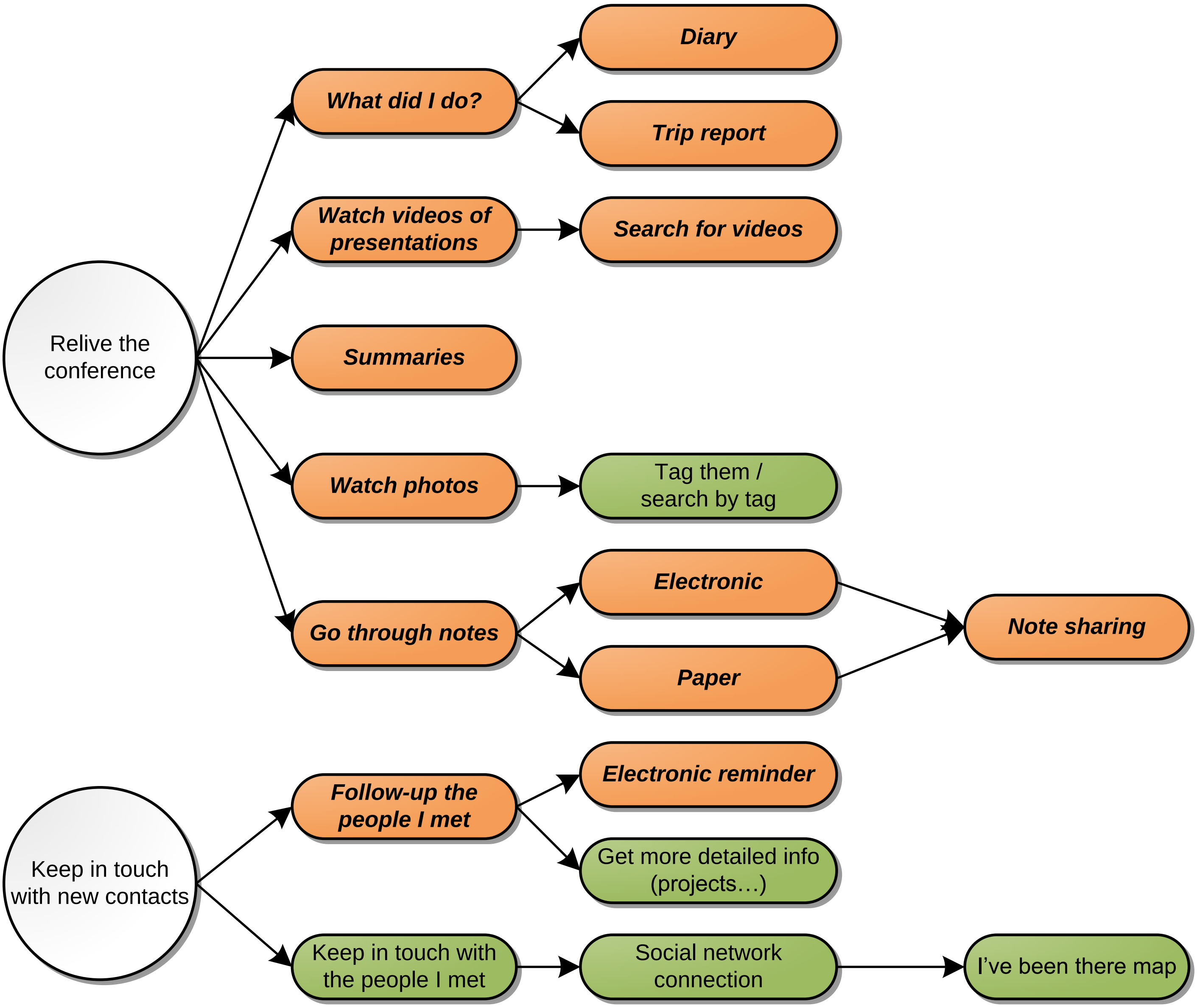
Give rating / feedback

Get extra info / references

Demonstration

Get presenter / question askers profile info

After the conference



B

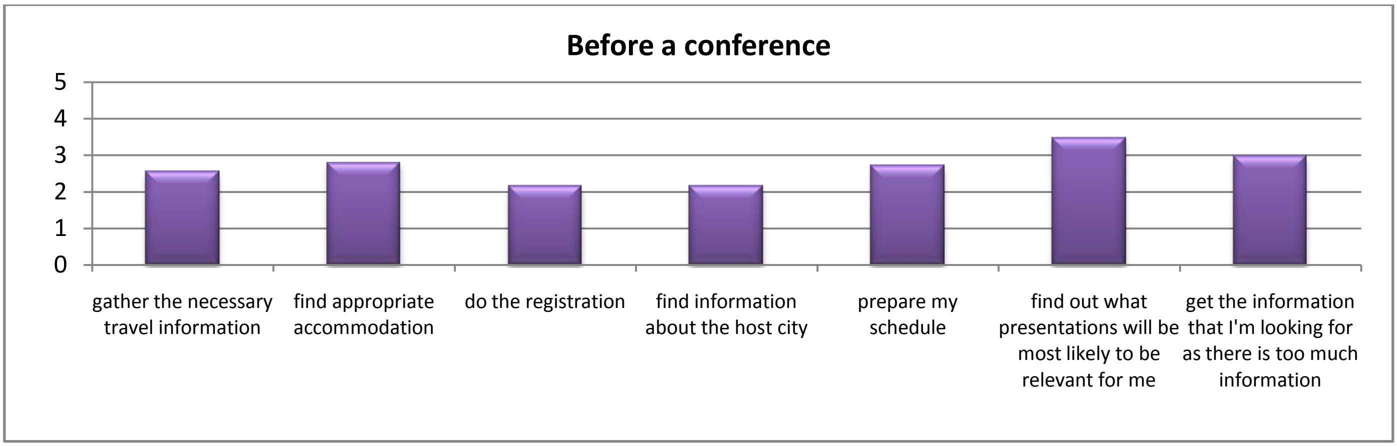
Interview Questions

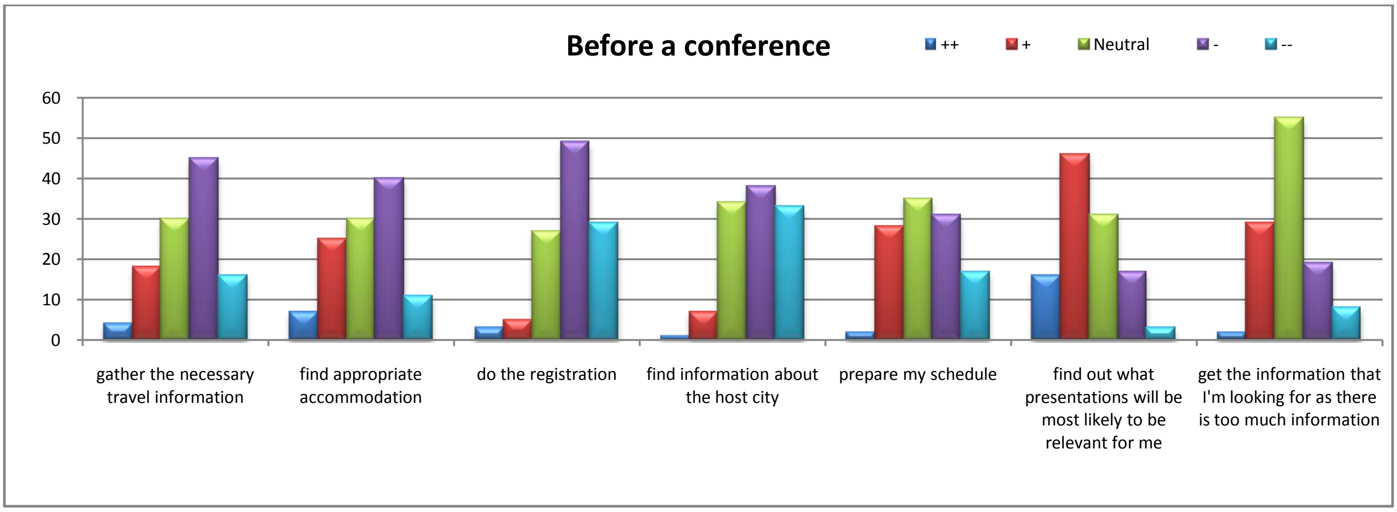
- What is your personal experience with conferences ?
- What are the issues you personally experienced during a conference that show the need to develop new conference assisting services ?
- Where did the idea for your project come from ?
- What audience does it target ? (academics, ..)
- What was your goal, what were the desired effects of the system ?
- What was the solution you deployed ?
- How did you manage the return of devices ?
- How did the system influence the quantity of paper distributed?
- What services were provided after the conference?
- What other solutions did you look at before choosing this one and why was this particular solution so attractive ?
- How big was the team ?
- How long did you need for the preparation ?
- How many people
- What was their first reaction ?
- Were there any unexpected uses or reactions to the system ?

- Were there any privacy concerns ?
- What issues did you not foresee while designing the solution ?
- What were the good points and what the bad ?
- Did the system solve the intended issues?
- If you'd have to do it again, what would you do differently ?

C
Survey results

Before a conference



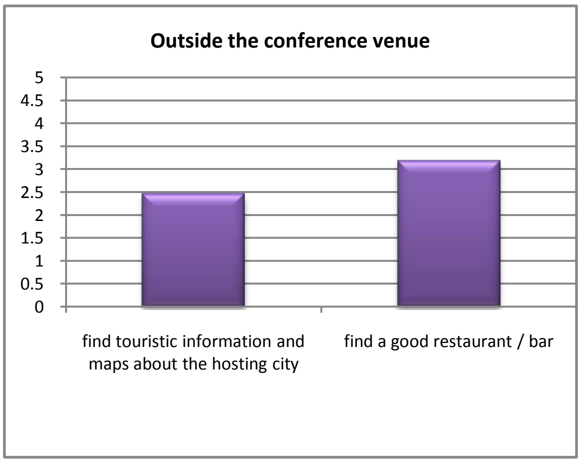


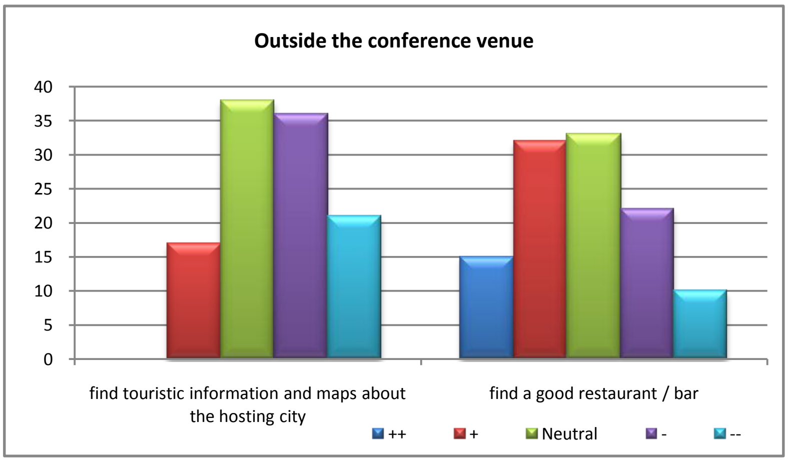
Remarks

- It’s not “hard” using Google, but I’m sure there are ways to make it easier...
- Conferences rarely offer the most inexpensive hotel options. I can often find something close that is much cheaper than what is offered through the conference.
- The tasks mentioned above are not hard but time consuming. On the other hand I do not use the suggested hotels. Usually I find cheaper hotels myself
- Costs & printout

During a conference

Outside the conference venue

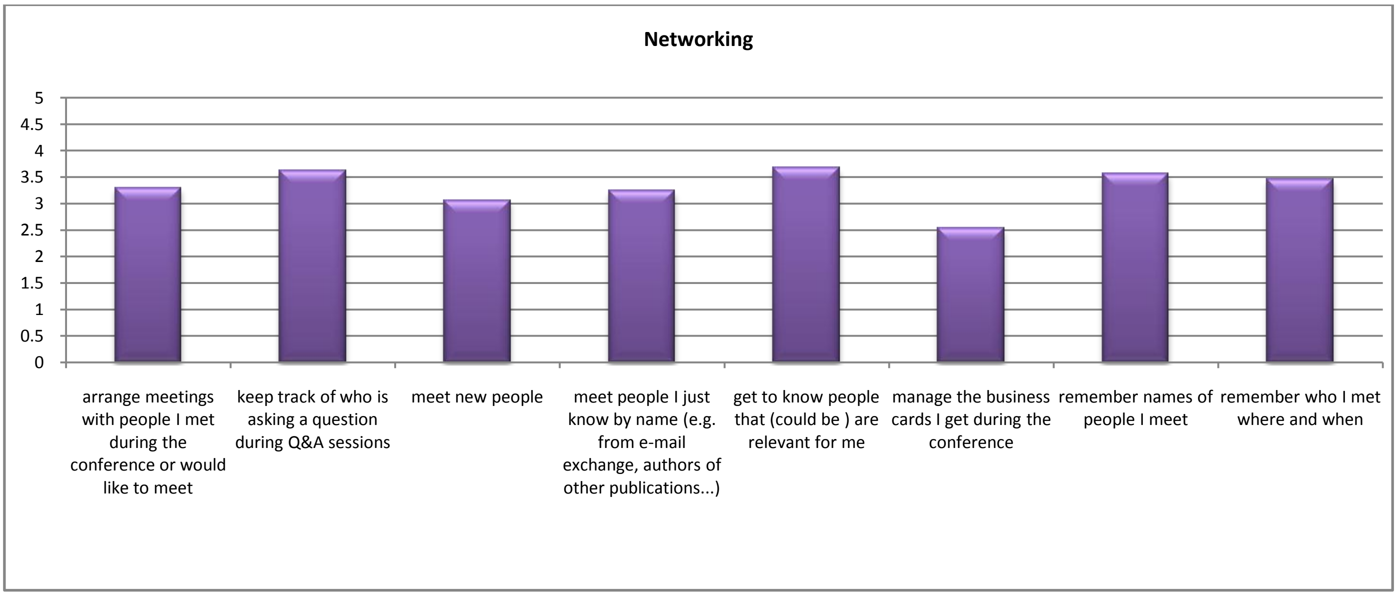


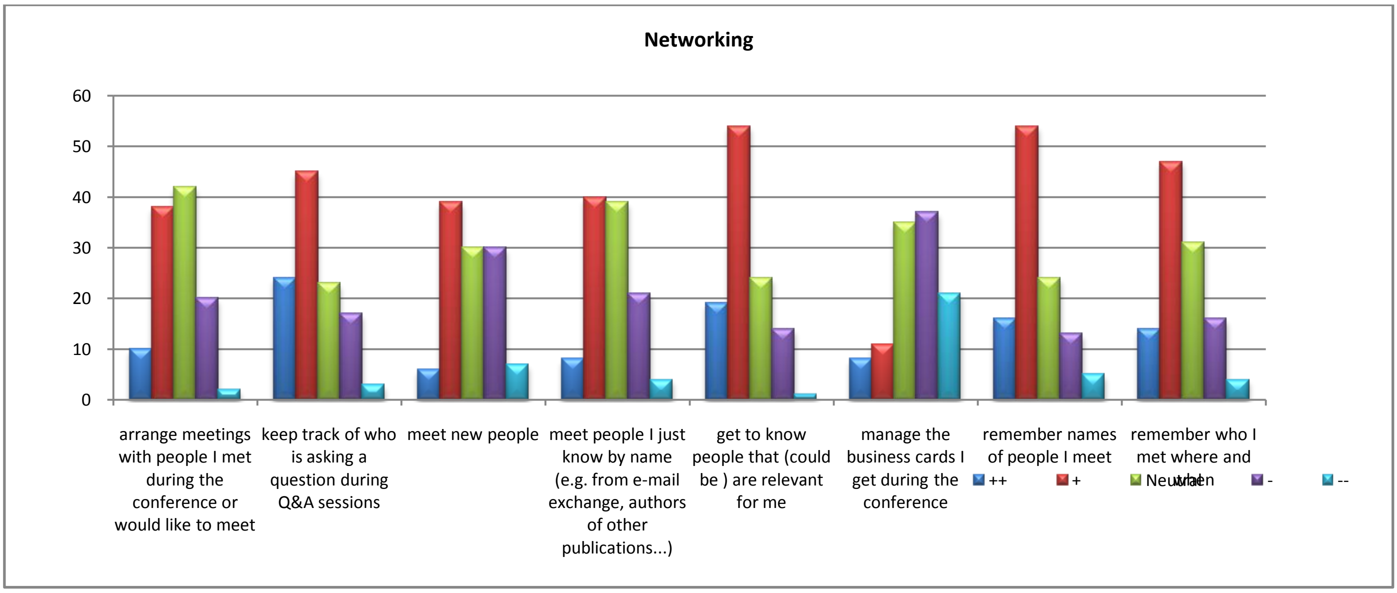


Remarks

- Usually conferences take place in fairly large cities and information about tourist attractions, restaurants, bars etc. can be easily obtained from the Internet. This is not something I expect to find on the conference web site.
- I find that usually you don't get recommendations about restaurants/bars, but would appreciate it though... with conferences taking place in hotels; they somehow expect u to dine there...
- Can find this information also in the web
- It would be nice to know who in the conference is planning to go in a restaurant. An empty good restaurant is not as interesting as an average restaurant with lot of people I know.

Networking

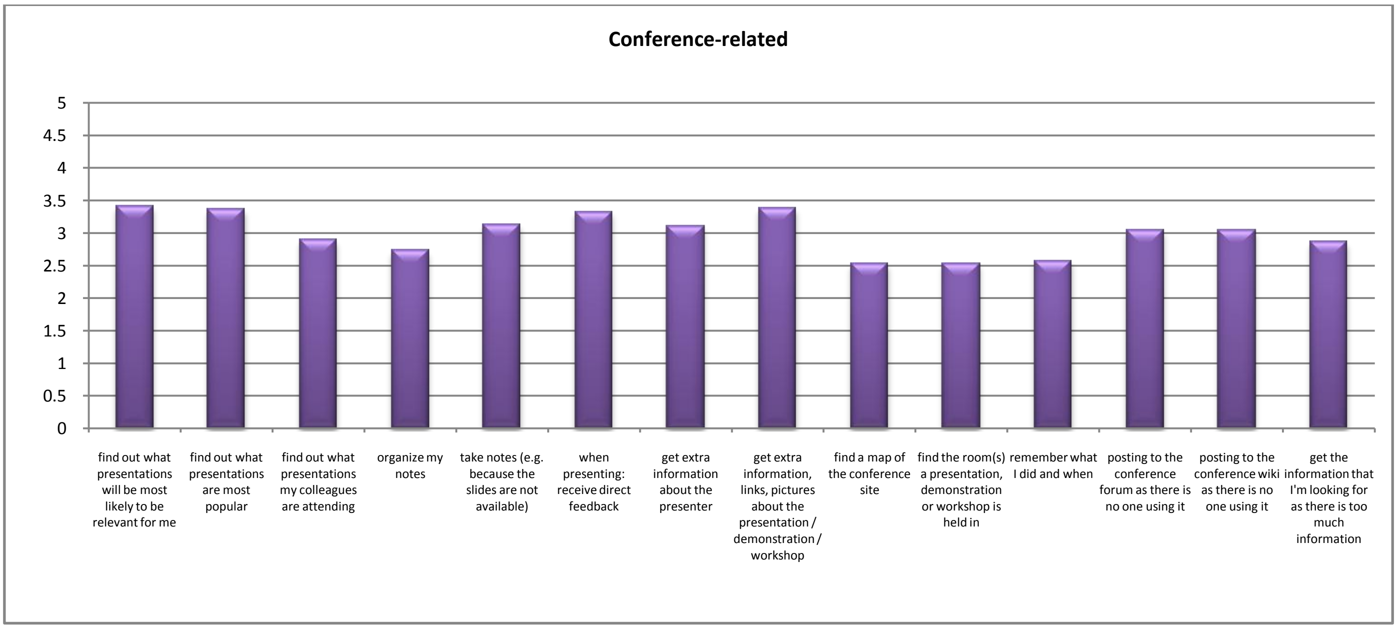


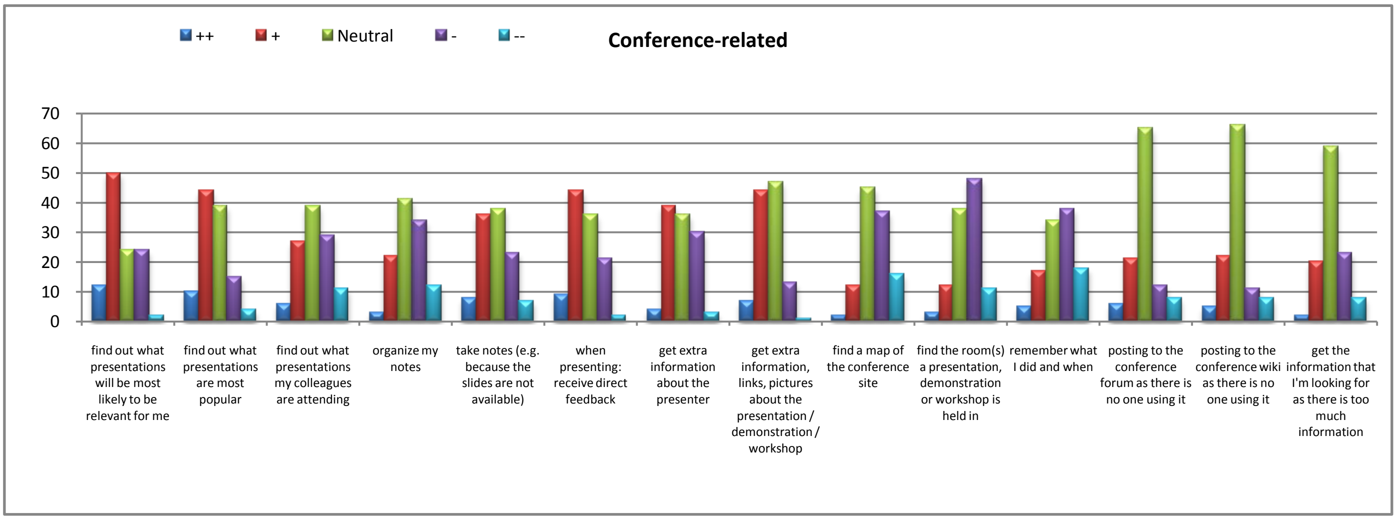


Remarks

- Everybody has their own way to manage their contacts. The big hurdle is to get to know the right people at the conference. Who works in your field but is too shy to approach you?
- That are more personal problems. I generally have difficulties remembering names and faces.
- Can do this also with my personal agenda
- To know who asked questions during the Q&A sessions might not be so important to me

Conference-related

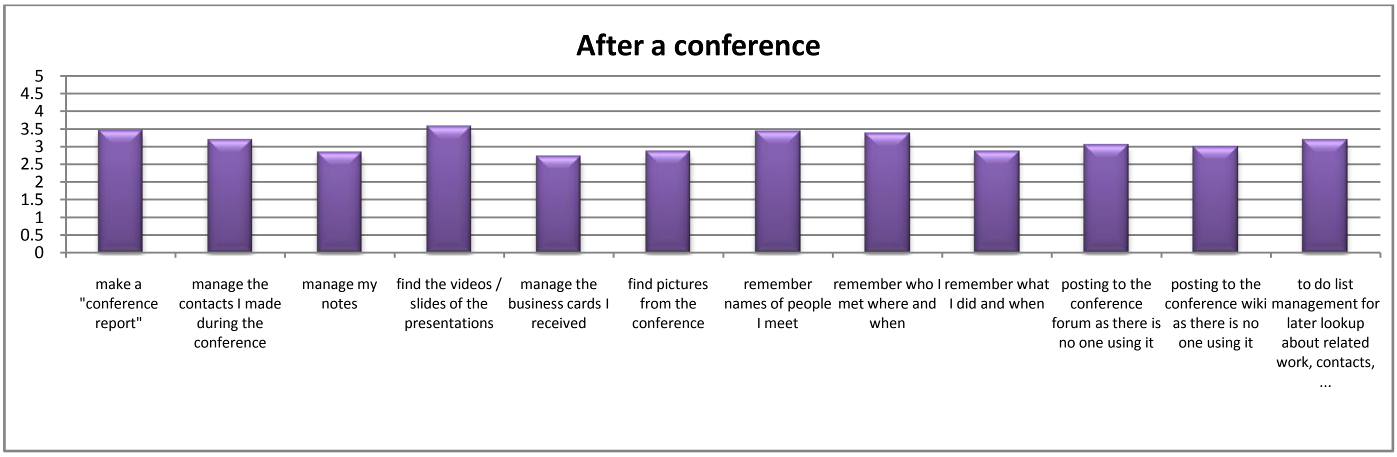


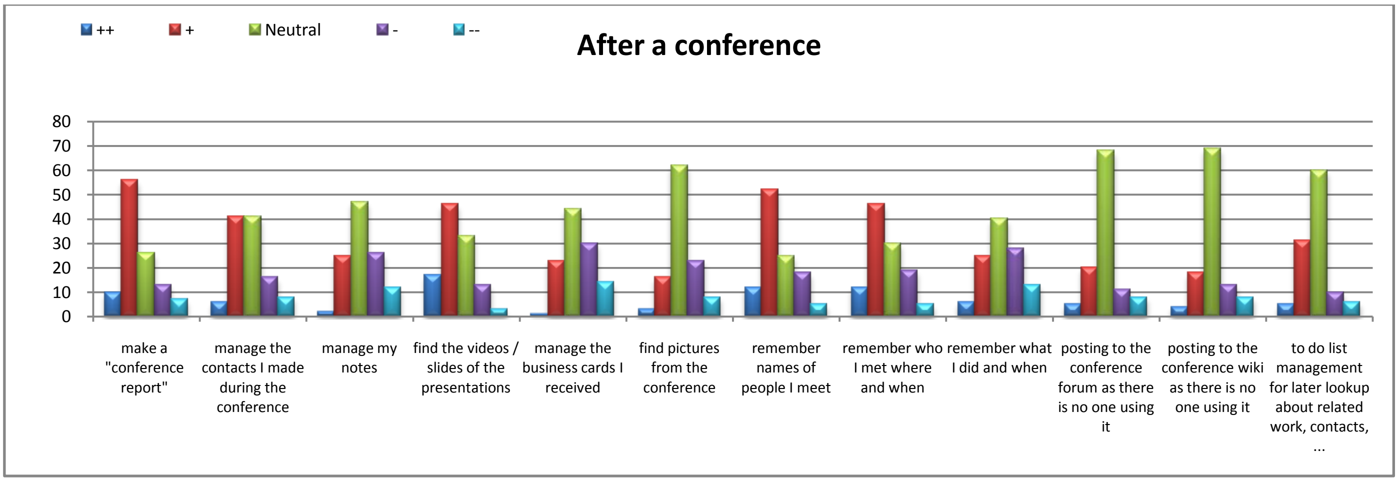


Remarks

- Wifi Internet access is usually available during the conference and so all information about presenters and relevant work can be gathered using one's preferred method. As for taking and organizing notes, each person has its own habits. Conference organizers should focus on what attendees cannot do by themselves.
- No forum/no wiki seen so far...
- I've never been at a conference with forum/wiki.
- Depends strongly on conference
- I would like to have access to slide and audio record of talks in the session I'm attending and the ones that are taking place in parallel.
- I don't see the relationship between the easiness of posting to a forum/wiki and the number of people who use it.
- get slides from presenters even after the presentation was very difficult, and often the organizer does not manage a site where you could get all stuff after the conference.

After a conference

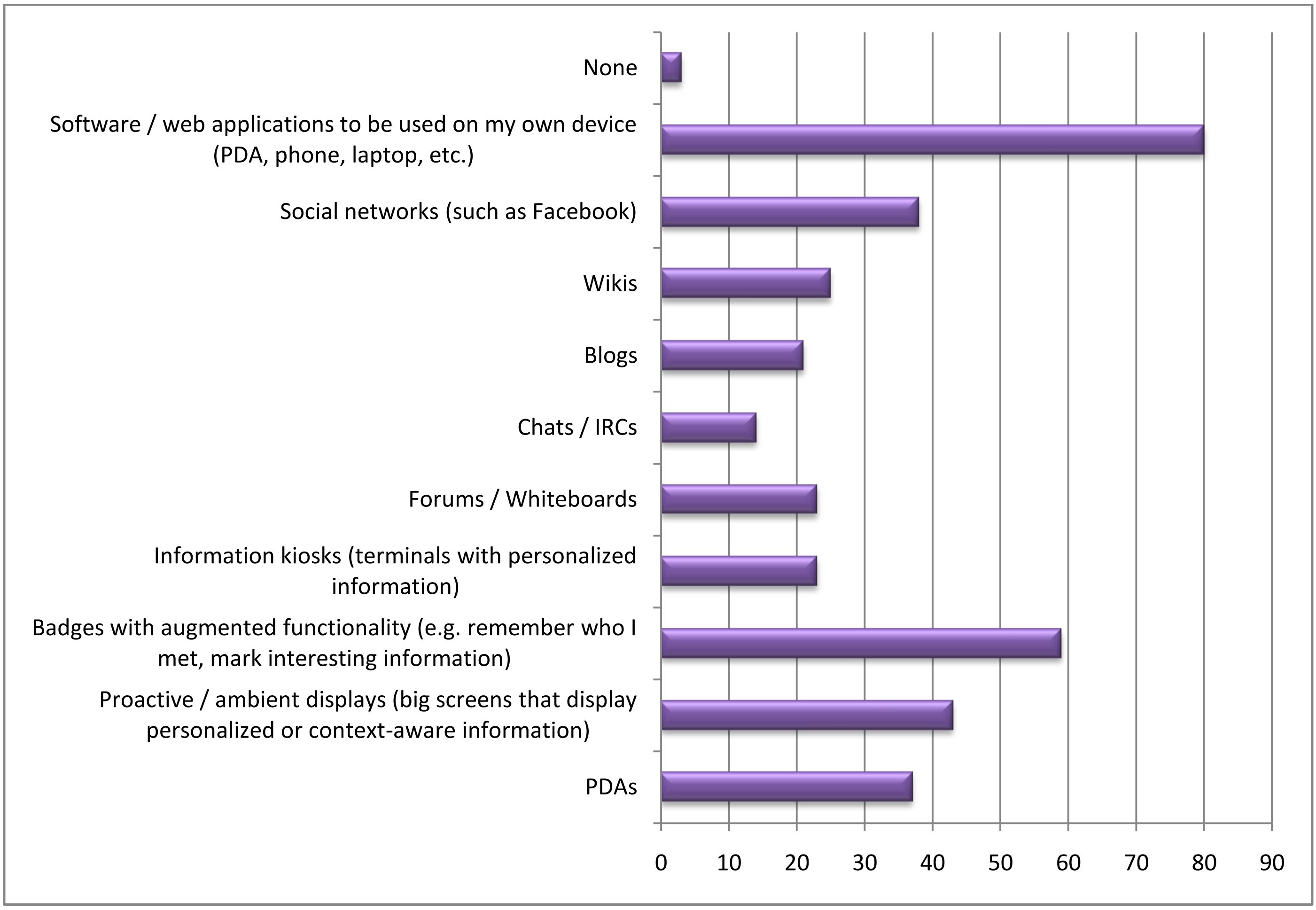




Remarks

- Making a conference report is generally a time consuming task.
- I'm not interested in doing most of the above (find pictures, remember what I did and when etc)

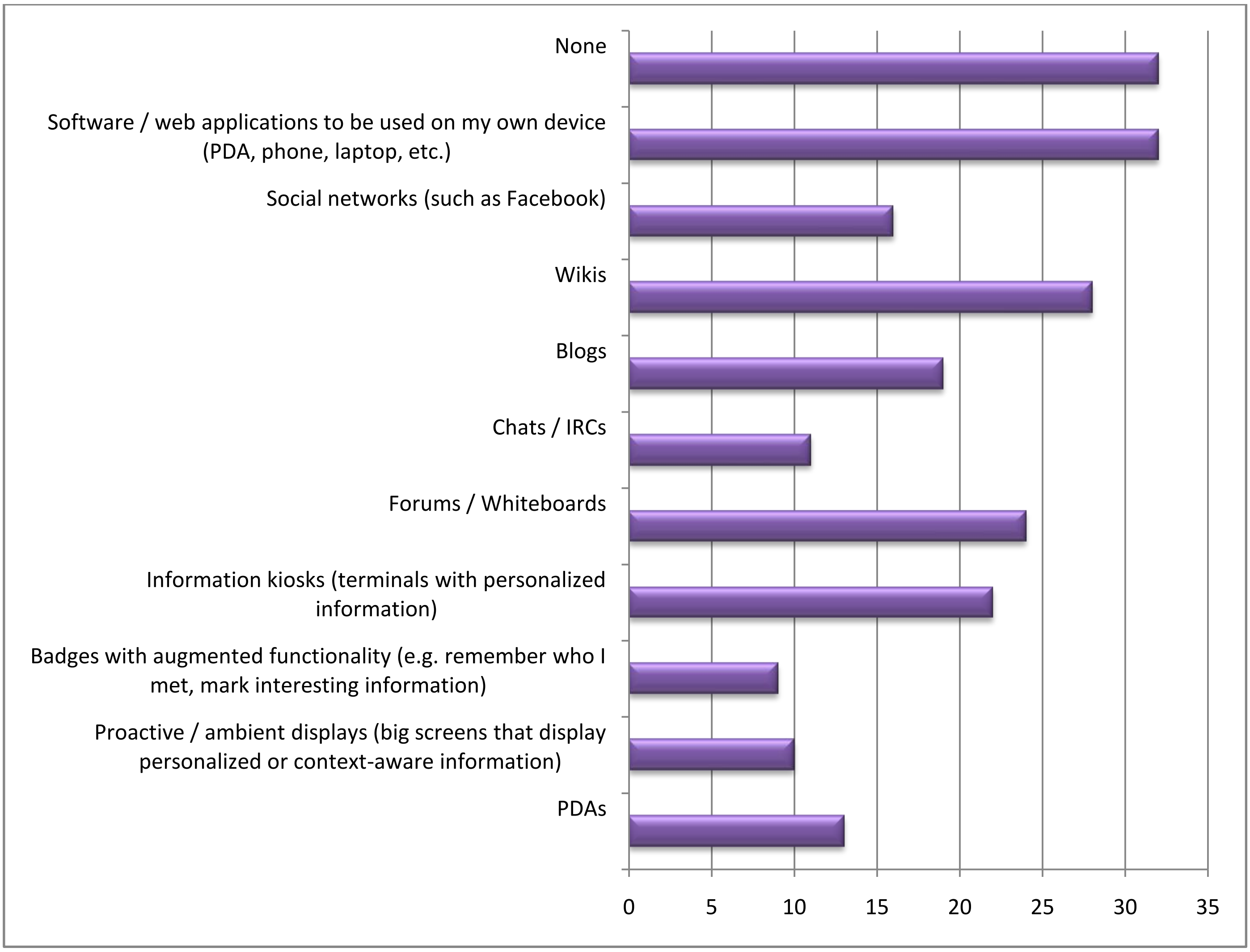
Systems, interested in



Remarks

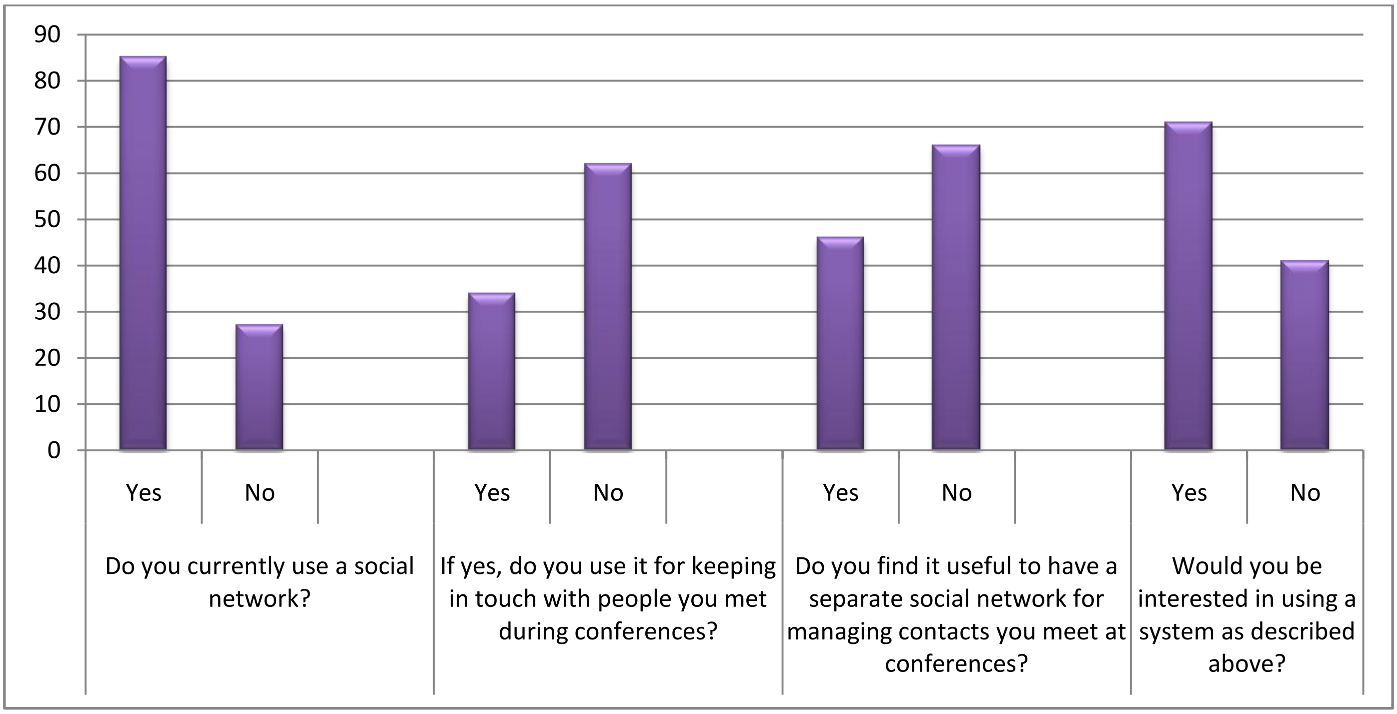
- In a multi-track conference, it would be convenient to have the possibility to quickly create and print out a customized schedule with presentations one wants to attend.
- More time after presentation with availability of presenter(s) would be helpful. Perhaps fewer presentations but at top-quality level might help. Less is more!! (in many cases, not always, of course)
- A simple possibility: download a file with dates and contacts for MS Outlook.
- Given the masses of smartphones out there and the app stores nowadays, this is a no-brainer: got to be apps for the usual phone OS, e.g. iPhone, Android, etc. Having to deal with another PDA on top of own devices? No, thanks!

Systems, used



Remarks

- Used many of them, but not in the context of a conference attendance
- It is not clear whether the question is meant in general or only during a conference.
- rather useless
- not very effective because seldom used
- Was evil (UPA Europe)
- The network was lame...
- I don't like the personalized information systems too much as I try to keep it, well personal, meaning to myself and only close selected contacts. I do like software and other systems that help me organizing during the conferences, such as electronic schedules and programs as well as calendar entries etc
- The internet connection should be available for everyone at all time.


90
80
70
60
50
40
30
20
10
0
Yes
No
Yes
No
Yes
No
Yes
No
Do you currently use a social network?
If yes, do you use it for keeping in touch with people you met during conferences?
Do you find it useful to have a separate social network for managing contacts you meet at conferences?
Would you be interested in using a system as described above?

D

Conference report

Conference information: ICOODB 2009

1 Jul 2009 00:00-00:00 ICOODB 2009:

ICOODB 2009 is the second in a series of international conferences aimed at promoting the exchange of information and ideas between members of the object database community. A key feature of the conference is its goal to bring together developers, users and researchers. At the same time, the conference aims to meet the needs of the different sub-communities. The conference therefore consists of three different tracks offered as a tutorial day, an industry day and a research day.

On the evening between the industry and the research days, there will be an exhibition alongside a reception to allow participants to see demonstrations of the latest developments in both research and products.

The conference will offer presentations on a wide range of issues related to object databases, including topics such as applications, methodologies, design tools, frameworks and standards as well as core object database technologies.

ICOODB 2009 invites contributions in any of the following categories: tutorials, industrial presentations, research papers and demonstrations.

Schedule

11 Mar 2009

10:00 Note:
Remember to ask about the future developments of this session.

6 Jun 2009

10:00 slides:
http://www.ethz.ch

1 Jul 2009

13:00-15:00 Object Database Products:
Description of a session on object database products.

> **13:00-14:00 Presentation of the OM Model:**
> *Talk on the OM model*
>
> **14:15-15:00 Presentation of Avon:**
> *Presentation of the Avon database*

2 Jul 2009

10:00-12:00 Object Databases and Query Languages:
Description of a session on Object Databases and Query Languages.

> **17 Mar 2009 at 10:00 slides:**
> *http://www.slideshare.net/targetsource/open-community-open-strategy*
>
> **09:55 Note:**
> *Introduced concepts that could be interesting for our project, contact Karl*
>
> **10:00 Met: Anna Pink**
>
> **17 Jul 2009 at 10:00 slides:**

http://www.google.com

15:00-17:00 Object Database Architectures and Patterns:

Description of a session on Object Database Architectures and Patterns.

15:00 Met: Clara Green

17 Jul 2009

10:00 slides:

http://www.facebook.com

People I met

- Jenny Orange at session: "Object Databases and Performance" (2 Jul 2009 09:00-11:00)
- Anna Pink at session: "Object Databases and Query Languages" (2 Jul 2009 10:00-12:00)
- Clara Green at session: "Object Database Architectures and Patterns" (2 Jul 2009 15:00-17:00)

Conference participants

Pierce Yellow

Anna Pink

John Purple

Paul White

Amy Blue

Clara Green

Sarah Red

Sean Black

Jenny Orange

Marc Gray

The participants with their names in bold and italic are part of your contact list

List of Figures

List of Tables

Acknowledgments

First of all, I would like to thank Stefania Leone and Beat Signer for supporting me throughout this thesis. I am grateful for the discussions we had and that helped me come forward, but also for the time they spent reviewing the thesis. I also want to thank them for helping me by sharing their contact network.

I want to thank the whole Globis group for the involvement in the group discussion as well as the help they provided for finding participants for my survey. Furthermore, I specifically want to thank Alexander de Spindler who helped me with the Avon database system.

I am grateful to Prof. Moira C. Norrie for giving me the opportunity to write a thesis with such a big degree of freedom.

I want to thank to Dr. Daniel Felix for the very interesting interview.

My thanks also go to Lukas Baenninger who helped me with creating my first survey.

I want to thank my family and Sébastien for motivating and supporting me throughout my whole studies.

Finally, I also want to thank the group in the lab, Christoph Z., Philip, Christoph L., Matthias and Arthur for the discussions and support they gave me.

Bibliography

[1] http://www.charmed.com/charmbadge.html.

[2] http://www.ntag.com/.

[3] http://www.spotme.com/.

[4] http://www.livescribe.com/.

[5] http://www.idemployee.id.tue.nl/g.w.m.rauterberg/conferences/INTERACT2003/proceedings.html.

[6] http://www.globis.ethz.ch/education/oodb/slides/10-omdatamodel.pdf.

[7] A. Amir, G. Ashour, and S. Srinivasan. Towards automatic real time preparation of online video proceedings for conference talks and presentations. *System Sciences, 2001. Proceedings of the 34th Annual Hawaii International Conference on*, pages 8 pp.–, Jan. 2001.

[8] Rudi Belotti, Corsin Decurtins, Moira C. Norrie, Beat Signer, and Ljiljana Vukelja. Edfest: Mobile information system for visitors to the edinburgh festivals.

[9] Richard Borovoy, Fred Martin, Sunil Vemuri, Mitchel Resnick, Brian Silverman, and Chris Hancock. Meme tags and community mirrors: moving from conferences to collaboration. In *CSCW '98: Proceedings of the 1998 ACM conference on Computer supported cooperative work*, pages 159–168, New York, NY, USA, 1998. ACM.

[10] Elizabeth Churchill, Andreas Girgensohn, Les Nelson, and Alison Lee. Blending digital and physical spaces for ubiquitous community participation. *Commun. ACM*, 47(2):38–44, 2004.

[11] I. Demeure, C. Faure, E. Lecolinet, J.-C. Moissinac, and S. Pook. Mobile computing to facilitate interaction in lectures and meetings. *Distributed Frameworks for Multimedia Applications, 2005. DFMA '05. First International Conference on*, pages 359–366, Feb. 2005.

[12] A.K. Dey, D. Salber, G.D. Abowd, and M. Futakawa. The conference assistant: combining context-awareness with wearable computing. *Wearable Computers, 1999. Digest of Papers. The Third International Symposium on*, pages 21–28, 1999.

[13] Volodymyr Kindratenko Donna Cox and David Pointer. Intellibadge tm : Towards providing location-aware value-added services at academic conferences. In *UbiComp 2003: Ubiquitous Computing*, pages 264–280. Springer Berlin / Heidelberg, 2003.

[14] Stephen Farrell, Christopher Campbell, and Suvda Myagmar. Relescope: an experiment in accelerating relationships. In *CHI '05: CHI '05 extended abstracts on Human factors in computing systems*, pages 1363–1366, New York, NY, USA, 2005. ACM.

[15] Rosta Farzan and Peter Brusilovsky. Community-based conference navigator.

[16] Tom Hope, Masahiro Hamasaki, Yutaka Matsuo, Yoshiyuki Nakamura, Noriyuki Fujimura, and Takuichi Nishimura. Co-construction of meaning and use with interactive information kiosks. In *UbiComp 2006: Ubiquitous Computing*, pages 387–403. Springer Berlin / Heidelberg, 2006.

[17] Min Huang, Yuwei Feng, and Bipin C Desai. Confsys: a web-based academic conference management system. In *C3S2E '08: Proceedings of the 2008 C3S2E conference*, pages 141–143, New York, NY, USA, 2008. ACM.

[18] Marije Kanis, Niall Winters, Stefan Agamanolis, Anna Gavin, and Cian Cullinan. Toward wearable social networking with iband. In *CHI '05: CHI '05 extended abstracts on Human factors in computing systems*, pages 1521–1524, New York, NY, USA, 2005. ACM.

[19] Doyoon Kim, Seungchul Shin, Cheolho Cheong, and Tack-Don Han. A usability test of exchanging context in a conference room via mobile device interactions. In *Human-Computer Interaction. Interaction Design and Usability*, pages 862–871. Springer Berlin / Heidelberg, 2007.

[20] Mik Lamming and Mike Flynn. ”forget-me-not” intimate computing in support of human memory. In *Tech. Rep. EPC-94-103*, pages 94–103, 1994.

[21] Joseph F. McCarthy and danah m. boyd. Digital backchannels in shared physical spaces: experiences at an academic conference. In *CHI '05: CHI '05 extended abstracts on Human factors in computing systems*, pages 1641–1644, New York, NY, USA, 2005. ACM.

[22] Joseph F. McCarthy, David W. McDonald, Suzanne Soroczak, David H. Nguyen, and Al M. Rashid. Augmenting the social space of an academic conference. In *CSCW '04: Proceedings of the 2004 ACM conference on Computer supported cooperative work*, pages 39–48, New York, NY, USA, 2004. ACM.

[23] Corsin Decurtins Alexandre de Spindler Andrei Vancea Moira C. Norrie, Michael Grossniklaus and Stefania Leone. Semantic data management for db4o. In *Proceedings of ICOODB 2008, 1st International Conference on Object Databases*, 2008.

[24] Kavitha Muthukrishnan, Nirvana Meratnia, Georgi Koprinkov, Maria Lijding, and Paul Havinga. Demonstrating flavour: Friendly location-aware conference assistant with privacy observant architecture.

[25] Yoshiyasu Nishibe, Hiroaki Waki, Ichiro Morihara, Fumio Hattori, Toru Ishida, Toshikazu Nishimura, Hirofumi Yamaki, Takaaki Komura, Nobuyasu Itoh, Tadahiro Gotoh, Toyoaki Nishida, Hideaki Takeda, Atsushi Sawada, Harumi Maeda, Masao Kajihara, , and Hidekazu Adachi. Mobile digital assistants for community support. *AI Magazine*, 19(2), 1998.

[26] Moira C. Norrie. An extended entity-relationship approach to data management in object-oriented systems. In *ER '93: Proceedings of the 12th International Conference on the Entity-Relationship Approach*, pages 390–401, London, UK, 1994. Springer-Verlag.

[27] Kosuke Numa, Toshiyuki Hirata, Ikki Ohmukai, Ryutaro Ichisea, and Hideaki Takeda. Action-oriented weblog to support academic conference participants. In *IADIS International Conference on Web Based Communities*, 2006.

[28] Gitta B. Salomon. Designing casual-user hypertext: the chi'89 infobooth. In *CHI '90: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 451–458, New York, NY, USA, 1990. ACM.

[29] Andrea Schweer and Annika Hinze. Augmented memory for conference attendees.

[30] Beat Signer and Moira C. Norrie. Paperpoint: a paper-based presentation and interactive paper prototyping tool. In *TEI '07: Proceedings of the 1st international conference on Tangible and embedded interaction*, pages 57–64, New York, NY, USA, 2007. ACM.

[31] Lisa J. Stifelman. Augmenting real-world objects: a paper-based audio notebook. In *CHI '96: Conference companion on Human factors in computing systems*, pages 199–200, New York, NY, USA, 1996. ACM.

[32] Yasuyuki Sumi and Kenji Mase. Conference assistant system for supporting knowledge sharing in academic communities. *Interacting with Computers*, 14(6):713–737, 2002.

[33] NORMAN KLEIN MAX CARLSON with GLENN MACEWEN. *Laszlo in Action*. Manning Publications Co., 2008.