Digital Facilitation of Distributed Workshops

Bjørn Lindeijer

July 29, 2007

Contents

List of Figures

List of Tables

1 Introduction

In this chapter I briefly introduce the context of my research and explain the motivation and research questions.

1.1 Context

One of the applications that are being developed at Fraunhofer IGD is "Digital Moderation". The goal of this application is to do digitally more efficient something that has been done for years on paper. It is about moderation in the sense of moderated meetings, which in this case are meetings driven by an agenda of activities and controlled by a moderator. Common activities seen in such meetings are brainstorming, ranking and answering a poll, all of which will be described later in more detail. Usually all of the participants of the meeting are involved in providing input during a certain activity.

The goal of doing a moderated meeting instead of an unmoderated meeting is to stimulate active participation from everybody and to increase efficiency. Also, for some things a poll or a ranking is a very efficient method of getting to a result or consensus. For example to find out how many people add milk to their coffee or whether a significant amount of employees prefer two small monitors over one big monitor.

There are several advantages in facilitating these workshops digitally. When the meeting is over, the results are immediately digitally available and can be distributed to all participants easily. Another advantage is that the computer can take over the storage, processing and displaying of the participant's input. This enhances the efficiency and causes the meeting to become much more scalable. As a result, hundreds of participants can participate in a meeting at the same time.

1.2 Motivation

This thesis presents the results of research at Fraunhofer IGD and is the fulfilment for the Master "Software and System Engineering" in Computer Science at the University of Groningen.

With the digitalization of moderation described above, an opportunity opens to use this technology over the Internet. It would allow the participants to be distributed instead of requiring them to be co-located. This is an interesting research area because the ability to hold meetings over the Internet can be a significant cost and time saver [BM01]. In the past it has been reported that there is an increasing interest in web-based meetings for this reason [Kha01].

Realizing that the developed application "Digital Moderation" could also be suitable for distributed web-based moderated meetings, the primary motivation for this research is to look into how the functional requirements of the system in this new context can be fulfilled. For a number of these requirements a solution will need to be developed from which experiences can be collected, so that conclusions about further development can be drawn.

1.3 Problem statement

The goal of this thesis is to improve the applicability of the software "Digital Moderation" developed at Fraunhofer IGD in a distributed web-based context. This is done by determining the functional requirements of such a system and developing the concept and design for fulfilling several of these, and testing a prototype implementation. This leads to the following problem statement:

> *Determine the functional requirements of a system allowing for web-based distributed moderated meetings and design and test solutions to the requirements that are essential in the context of "Digital Moderation".*

This problem statement leads to the following research questions:

1. What are the requirements for a system allowing for web-based distributed moderated meetings?
   a) In what kind of situations is a web-based solution attractive?
   b) How will such a system be used differently in this context than in regular moderated meetings?
2. Which of these requirements are vital and suitable for proof of concept implementation?
3. To what extend do the functional additions fulfil the functional requirements?

The answer to question 1 is determined in chapters 3 and 4. Chapter 7 provides an answer to question 2 and describes the concept and design of the functional additions I have made to "Digital Moderation". In chapter 8 these additions are tested in order to provide an answer to question 3.

2 Moderation

In this chapter I introduce the concept of moderation, in specific the "ModerationMethod". After that I talk about the software "Digital Moderation", which is mainly based on this method.

2.1 Moderated meetings

2.1.1 Introduction

Moderation is a very old concept, first developed in the sixties as the "Metaplan-Method" or "ModerationMethod" by the Quickborner Team (a consulting society) and their successors [Sei07]. Later this method was refined by Josef W. Seifert [Sei02] and Karin Klebert et al. [KSS02].

The purpose of moderating meetings is to actively involve participants with the matter being discussed. To make sure that everybody can take part and bring in his or her point of view, while minimizing the risk of disputes. With this goal, a moderator gives structure to the discussion and the activities involved. The "ModerationMethod" is characterized by:

- Specific seating arrangements to ensure that everybody can see each other and can communicate directly.
- Specific media and tools, like a pinboard, flip-chart, paper cards and pens.
- Specific methods for problem structurization.
- Ongoing visualization during the process, mainly for the purpose of information durability.
- A moderator with a neutral point of view, who should not be an expert on the treated issues. The use of certain asking techniques helps the moderator retain his neutrality.

2.1.2 Arrangement

A moderated meeting can take place with relatively few participants up to a few dozen, and can take from a few hours up to a day or two. As a rule of thumb the room needs an area of about 8 square meters per participant (ie. a room of 160 m^2 for 20 participants). There should be small tables for holding material and drinks, a sufficient

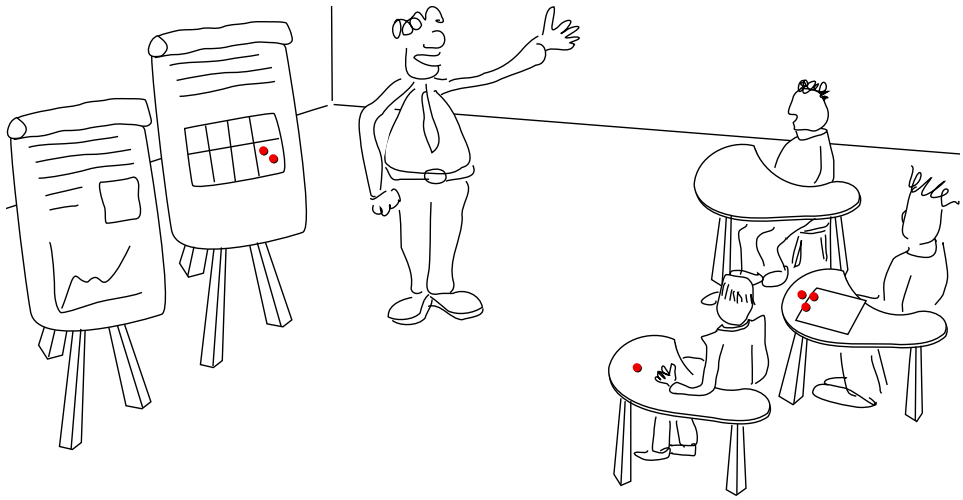

Figure 2.1: Sketch of moderation setup, showing the pinboards, moderator and participants.

amount of pinboards (usually one per participant), self-service arrangement for drinks and a stereo for music. Klebert further describes the importance of arranging the right colors, lighting, air, temperature and acoustics [KSS02].

Figure 2.1 shows an example session with the pinboards on the left, the participants on the right and the moderator in the middle.

2.1.3 Media and tools

Ongoing visualization is realized by the use of pinboards on which the participant contributions are pinned up using paper cards or sticky dots. Writing on the cards and on the flip-chart is done using thick pens to ensure readability from a distance.

The cards are about 10 by 21 cm and come in four colors (white, green, orange and yellow). Aside cards, discs of 20 or 10 cm diameter are also used, which come in the same four colors. Felt markers, again available in four colors (black, red, green and blue) are used to write on the cards and discs. Finally, sticky dots (in yellow, red and green) are used [KSS02].

2.1.4 Methods

The "ModerationMethod" describes several methods, or ways of using the above mentioned media and tools. The generally easiest to understand method is the *single-dot-question*. Here, a question is written on a pinboard and a bar of horizontally arranged boxes is displayed below. Each box can have a label or they can be numbered. Every participant will place a sticky dot in one of the boxes. Generally the box where a participant places his dot indicates something like his agreement with the question, his level of interest or his opinion on a certain issue. This method is very flexible, where the horizontally arranged boxes can also make place for a line with labels at both ends,

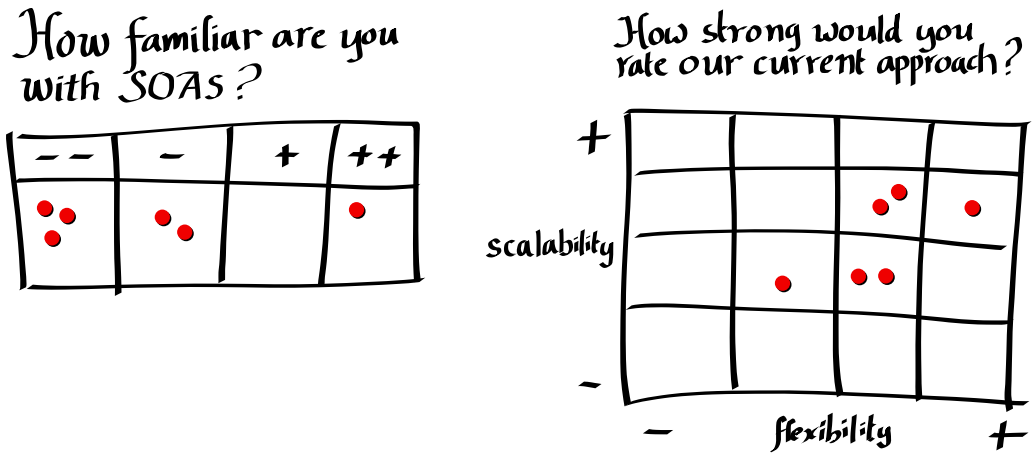

Figure 2.2: Single-dot-question (left) and 2D-single-dot-question (right).

or for any pictures or graphs that represent what is being answered. At the end, the distribution of the dots usually gives a quick impression of the group's position on the given statement or question.

The single-dot-question can easily be extended to a *multi-dot-question*, where multiple questions are answered at the same time or one question is answered with multiple possible answers. The question can also be extended to a *2D-single-dot-question*, where the answer to two related questions is given on a 2D grid of boxes or a graph with two axis. Figure 2.2 shows two single-dot-question variations.

The sticky dots only provide limited freedom for the participant. When the participants are asked to define a problem, come up with solutions or comment on ideas, the paper cards are used. The participants can write down their answers on the cards, which can then be stuck up on a pin board. The methods used here include a *brainstorming*, where participants are asked to contribute answers to an open question, or giving *recommendations*, where reactions are given to a previous brainstorming session. After either session, a *clustering* can be performed, where the cards are grouped together in discussion with the participants. Using the sticky dots, it is also possibly to perform a *ranking* on contributed answers or clusters, to get an idea of how important or valued the individual answers or clusters are considered by the group.

2.1.5 The moderator

Central to the moderated meeting is the moderator. The opinion of the moderator about the subject matter however, is of least importance. Aside introducing the tools and methods to the participants and telling them what to do, the moderator needs to pay attention that everybody has an equal chance to voice his opinion. He has to ask questions in a way that makes the participants think, but that do not direct them to any specific opinion or conclusion. Staying neutral is easiest for the moderator when he himself is not concerned with the matter being discussed. Hence, the task of the moderator is sometimes fulfilled by a "neutral third person" [Sei07].

In conclusion, the "ModerationMethod" is more than just which tools to use and what to do with them. It's a way of creating an atmosphere in which everybody is comfortable to come out with their opinion, to create the possibility for everybody to do so and finally to make sure their opinion doesn't get overlooked in the discussion.

2.2 Digital moderation

2.2.1 Introduction

In recent years, ways have been sought to digitize the practice of moderation described in section 2.1 to reduce the amount of paperwork and to efficiently create and distribute the results. This has led to the development of Digital Moderation, an application specifically designed for this purpose. This application is designed to be used in a similar setting as the "ModerationMethod", with all participants together in a single room. The participants enter their contributions using laptops, and a projector provides a view on the results. The moderator can control the process through a specialized application running on his laptop.

2.2.2 Server and clients

The Digital Moderation software consists of a server, which keeps track of the current meeting state and the results, a number of different clients which connect to this server, and an editor for configuring a meeting. The *participant client* runs on the laptops which the participants use, and allows them to submit their contributions. The *stage client* runs on the machine connected to the projector, and is used both for explaining how things work before the participants submit their contributions as well as showing the results. Finally, the *facilitator client*[1] allows the moderator to control the meeting state and provides him with information about the connected participant clients.

The participant and facilitator clients of the Digital Moderation software are named after their user. In this thesis, I will always refer to these clients explicitly. For example the participant uses the participant client. Whenever I mention just the participant, I am not talking about the client but about the person using the client.

Figure 2.3 gives an overview of a typical network setup and the most typical data flow during the action phase. Answers messages go from the participant clients to the server. The server uses the answers to augment the result data. The result data is then sent to both the stage and the facilitator client. The stage client usually displays the result on a projector, when all answers are in. The facilitator client shows the result as answers come in, which is useful for the moderator. An arrow goes back from the facilitator client to the server to indicate the high level of control that the moderator has over the meeting through the facilitator client.

[1] It is not clear from existing literature whether the one who leads a moderated meeting should be called a *moderator* or a *facilitator*. In this thesis I've decided to call this person the moderator, while the Digital Moderation software refers to the facilitator.

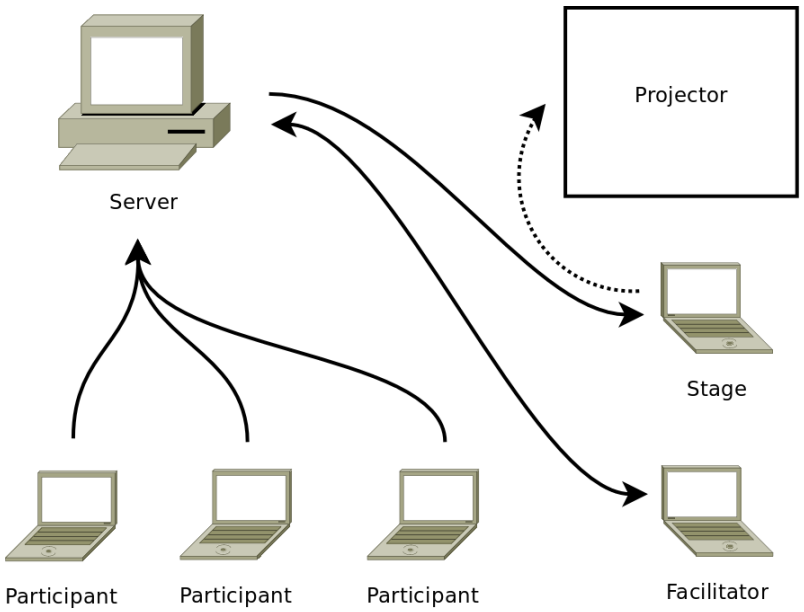

Figure 2.3: Typical network setup and data flow during the action phase.

2.2.3 Tools and phases

In the Digital Moderation software, a distinction is being made between tools and phases. The tools are functionally equivalent to the methods described as part of the "ModerationMethod". They are normally split up into three phases: an *explain phase*, in which the moderator explains the participants how to use the tool, an *action phase*, in which the participants make their contributions through their participant client, and a *result phase*, in which the results are shown to the participants via the projector and can be discussed.

Most of the methods described in section 2.1.4 are present as tools in Digital Moderation. These tools work similar to how the media and tools described in section 2.1.3 are used, where virtual sticky dots can be used to place votes and text input fields replace the paper cards. The *brainstorming* tool can be used to let the participants submit arbitrary answers, which consist of a summary and a detailed description. These submissions can be grouped together in the *clustering* tool, which displays the ideas on the stage client and allows them to be put together in clusters by the moderator. Either the brainstorming submissions or the created clusters can be ranked using the *ranking* tool, which allows the participants to place sticky dots next to the entries in one or two columns, representing different criteria. The same sticky dots are also used in the *single-dot-question*, *multi-dot-question*, and *2d-single-dot-question* tools, which provide the equivalently named methods of the "ModerationMethod".

Other tools available in Digital Moderation are the *action item* tool, which allows tabular data to be entered on the stage client (like filling a table with data on a flip-chart), and the *portfolio* tool. The portfolio tool has no equivalent in the "ModerationMethod". It allows the participants to rate a number of alternatives on up to three criteria. It

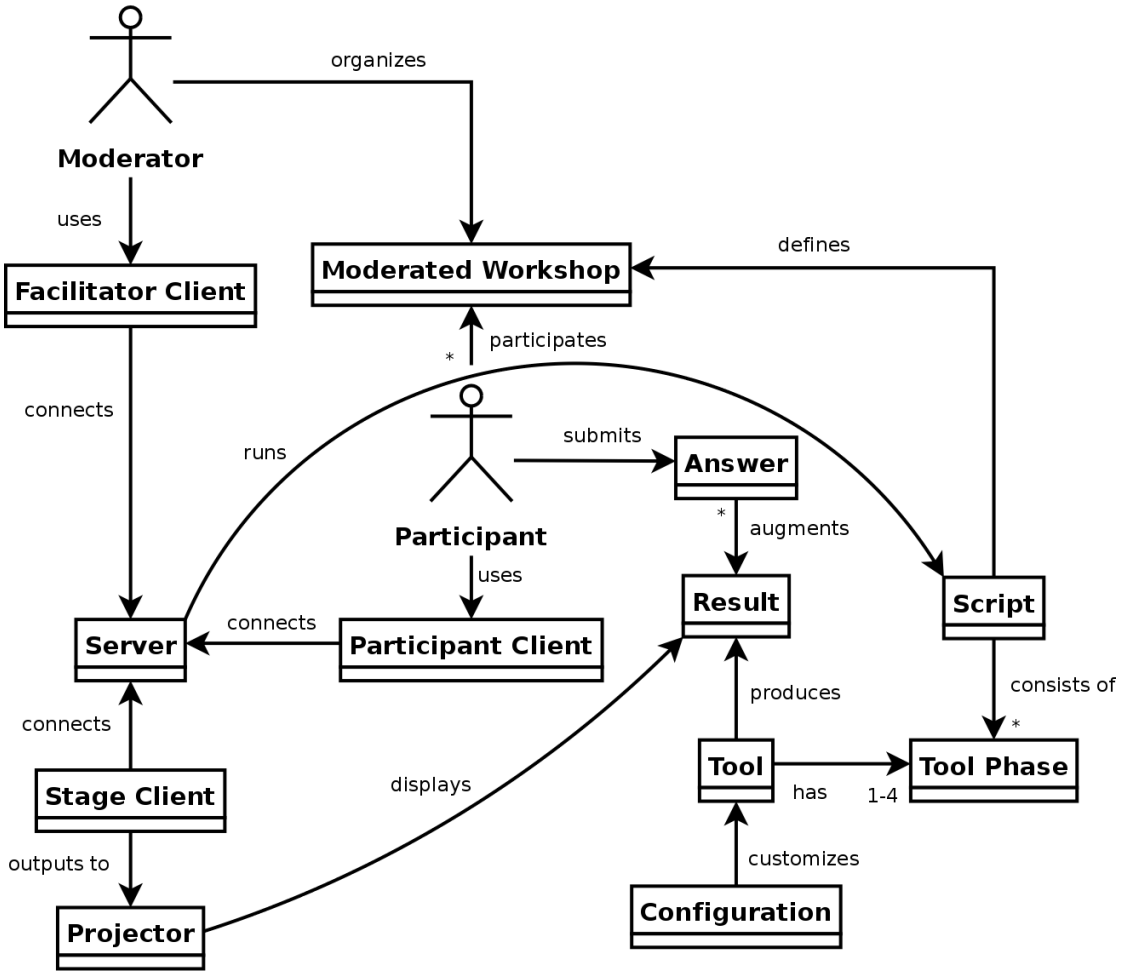

Figure 2.4: Domain model of Digital Moderation.

uses the answers to draw either a bubble diagram or a bar chart which shows for each alternative the average rating.

2.2.4 Domain Model

The domain model [Lar04] shown in figure 2.4 illustrates the various relationships between the tools and phases we just mentioned and the server and clients mentioned earlier. It also shows how these things relate to the moderator and participants in the moderated workshop.

In the next chapter I will take a look at the problems that come up when the moderated workshop is distributed and performed over the Internet. The domain model will stay largely the same in this scenario, but an alternative will have to be found for the projector and communication between actors will no longer be implicitly possible.

3 Problem analysis

Now I describe several new scenarios in which software for moderated meetings could be applied, in particular the software "Digital Moderation". The scenarios have in common that the participants of the meeting are in some way distributed across multiple locations.

3.1 Scenarios

In order to more precisely identify the functional requirements for the facilitation of distributed meetings, we look at several common scenarios. A meeting can be distributed in a number of ways. The first case I discuss is the situation where one or more participants are participating remotely. I then discuss the situation where two or more teams are doing a meeting together, while only being site-wise together in the same room. Finally I discuss the situation where all participants in the meeting are distributed individually, which is a special case of the first case.

3.1.1 Remote participants

Imagine a meeting is being held by a company, but a few important participants are unable to attend locally, for example because they are teleworking. This scenario is illustrated in figure 3.1. In section 1.2 I have pointed out that this is an increasingly common situation because of financial benefits. One possible solution is to have these participants log in remotely to the server. Since they won't be present in the room, they won't be able to talk to people from the rest of the group, and their status won't be directly visible to the moderator. Also, the common stage view is not visible to the remote participants. Minimally, our solution will need to allow communication between the remote participants and the moderator, provide access to the results in the result phase and provide some way for the moderator to know more about the status of the remote participants. A login facility using some form of authentication is used in order to identify the remote participants.

Communication by remote participants with the rest of the group is desirable, but voice or video communication is not always convenient and requires the use of special hardware. One unobtrusive way of introducing this is with so called backchannels [CFKW01, YMW+05]. They provide a way of using chat in the background, potentially separated (for example on topic) across several channels and potentially allowing private communication.

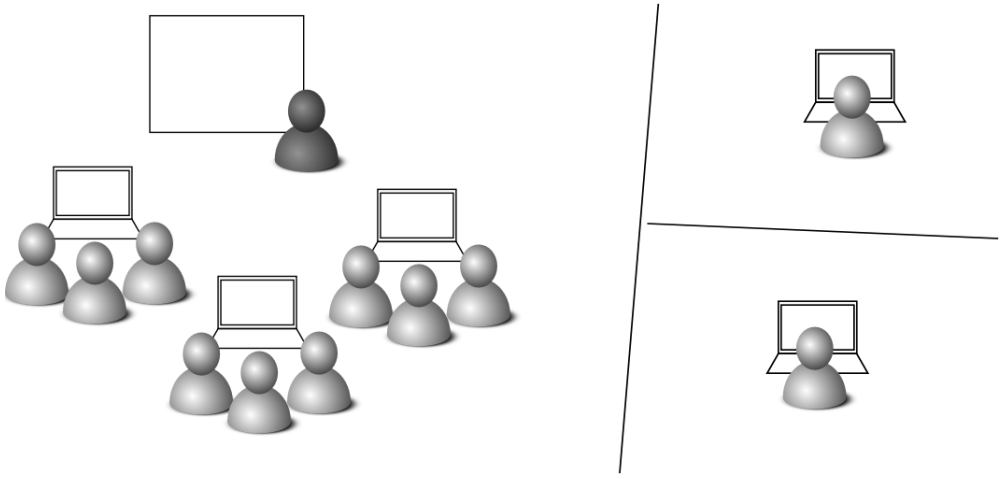

Figure 3.1: Scenario with remote participants.

3.1.2 Multiple teams

Another common situation is when two or more teams have to work together in order to agree about certain decisions. For companies that are distributed across several offices, it can be costly and inconvenient to bring everybody together at the same location. One possible scenario is to bring only the teams together and connect them via the Internet, as illustrated in figure 3.2. It would be convenient if a moderator (or moderator assistant) was part of every team. Our solution has to allow communication between these moderators, and to allow both a facilitator client as well as a stage client to be present at each location. We also need to take into account that the participants may want to see both the global results as well as individual team results.

As with the remote participants, communication between participants in different teams will be difficult, but may be done on a backchannel.

3.1.3 Completely distributed group

As a final example of a distributed scenario, we imagine all of the participants are on their individual workstations. This is illustrated in figure 3.3. In contrast to the other two scenarios, communication among participants happens exclusively over the network. The moderator needs to have enough status information available to have a good idea of what's going on at each participant, but the participants also need to know some status information about each other. Finally, all participants will need to have access to the results locally.

3.2 Problems

In the above scenarios, a few common problem areas can be identified. In the first place, a communication channel will need to be provided so that the participants can

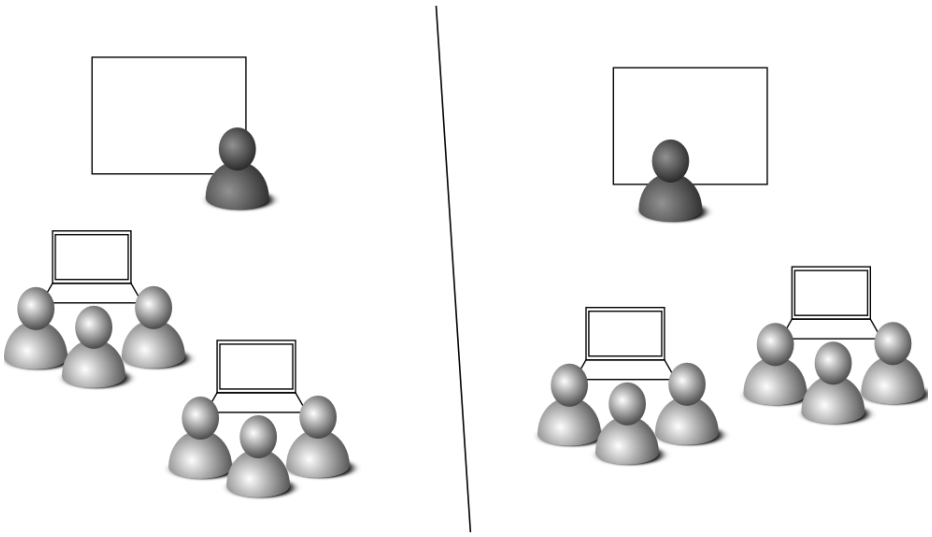

Figure 3.2: Scenario with multiple teams.

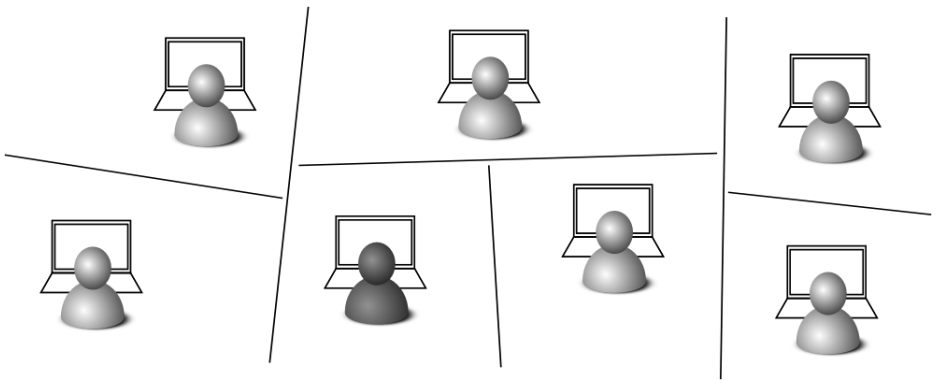

Figure 3.3: Scenario with a completely distributed group.

still communicate with each other and with the moderator. Second, the moderator will no longer be aware of participant activity if not additional awareness is added by the system. Finally, remote or distributed participants will need to be able to access the results normally shown on the stage client.

3.2.1 Communication

When participants can not communicate with each other, there cannot be discussion and there can hardly be cooperation. In the workshops that the software is meant to facilitate, discussion and cooperation are often very important.

There are three common ways of communicating over the Internet. They are chat, voice and video. In chapter 6 I will look at both chat and voice communication as possible options for doing a distributed workshop.

3.2.2 Awareness

Awareness is a very broad term that is confusing when not specified more specifically [Sch02]. A lot of work has been done into researching the addition of various types of awareness to workshops where participants are expected to cooperate [GG99, GG02, KPVB02, GG98, GRG96, GPS04].

There are two kinds of awareness that are important in our case. The first is inherently important for communication, namely to be aware of whether the participants you're talking to are paying attention and to be aware when somebody is talking to you. Second, it's important for the moderator of the workshop to be aware of the activity of the participants. This includes knowing whether they are working on an answer or pausing for an extended period of time.

In chapter 6, the first kind of awareness is to be provided by the addition of a web conference tool. The second kind of awareness is added to Digital Moderation and tested later in chapter 8.

3.2.3 Accessibility of results

For remote participants, a projector can't provide a common view on the results. Two options are tested in chapter 6. First, participants can run a stage client locally, since Digital Moderation allows any number of stage clients to be active. Second, a single stage client is shared via application sharing.

Neither of these solutions seems entirely optimal, so an optional result view is also added to the Digital Moderation client as described in section 7.2, which is tested in chapter 8.

Before I start to address this and the above two problems, I have first made an effort to more clearly specify the new requirements of the system and taken a look at related work, in respectively chapter 4 and 5.

4 Requirements analysis

In this chapter, use-cases are extracted from the scenarios described in the previous chapter in order to compose a list of functional requirements. The non-functional requirements are also mentioned.

4.1 Use cases

As we have seen when going through the scenarios, the completely distributed meeting includes many of the requirements that are part of the other two scenarios. Running additional stage or facilitator clients is already supported by the Digital Moderation system. Hence, the use cases will primarily tackle the third scenario.

Figure 4.1 shows an overview of several common tasks that I will cover in detail below. Some of them require participation of both the moderator and the participant(s) at the same time. First the moderator sets up the digital moderation, after which the participant connects to this moderation session. For each new tool that is used during the meeting, the moderator explains the usage of the tool to the participants during the explain phase. After this follows the action phase during which the participant submits one or more answers. Meanwhile, the moderator monitors participant activity and keeps an eye on potential problems. Finally, the participants look at the results in the result phase.

In the summary of each of these use cases I also point out the difference with a non-distributed moderated workshop based on Digital Moderation.

4.1.1 The moderator sets up a distributed moderation

Summary

Before a distributed moderation can start, the moderator has to set up the Digital Moderation server. The difference to keep in mind in relation to a non-distributed moderation is that the server is generally not running on the same network as all the participant clients. Also, additional communication software will be necessary, although it's not possible to describe this in detail as long as we haven't specified what kind of software to use.

Precondition

The moderator has installed the Digital Moderation software.

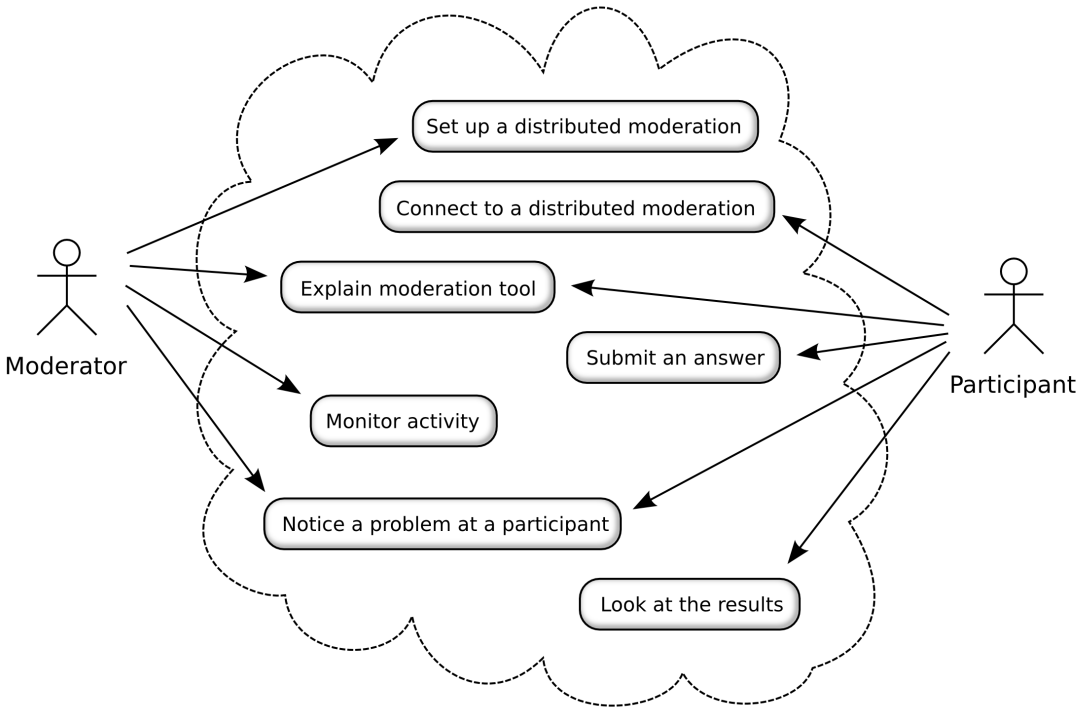

Figure 4.1: Overview of use cases.

Basic course of events

1. An administrator prepares a dedicated machine on which the Digital Moderation server is launched, with the address and port which Digital Moderation clients use to connect to the server reachable from wherever is necessary.

2. The moderator launches the Digital Moderation editor on his machine and opens or creates the script he wants to use.

3. The moderator clicks the button to start the meeting, specifying the address of the server he wants to run the meeting on.

4. The moderator launches the Digital Moderation facilitator client via the server's web interface, and verifies the active meeting script.

5. The moderator passes instructions about how to connect on to the participants. The system can automate this when the moderator has previously configured the list of participants and their email addresses.

Postcondition

The participants can begin to connect to the distributed moderation.

4.1.2 A remote participant connects to a distributed moderation

Summary

In a non-distributed moderation, the participant clients are usually launched before the moderation starts. In a completely distributed moderation, each participant launches a Digital Moderation participant client individually. Depending on the communication software used, this process could be automated. However, we will first assume this is not the case and describe the necessary steps without automation.

Precondition

The moderator has set up a distributed moderation and the participants have received the instructions about how to connect to the moderation (see 4.1.1).

Basic course of events

1. The participant opens the received web address in a web browser.
2. The system prompts the participant for his name and topic area, if this information wasn't included in the web address.
3. When prompted in step 2, the participant enters his name and his topic area, and clicks the button to launch a Digital Moderation participant client.
4. A Digital Moderation participant client launches, which connects to the Digital Moderation server.
5. The Digital Moderation participant client communicates the participant name and topic area to the Digital Moderation server.
6. The participant can verify his name and topic area from the Digital Moderation participant client window.

Postcondition

The participant is ready to participate in the meeting.

4.1.3 The moderator explains the tool

Summary

In this phase the moderator explains the participants of the meeting how the next tool that is being used works. It is important that all participants pay attention to the moderator and that they can see the Digital Moderation stage client, which is where an example is shown. The difference with a non-distributed moderation is that the stage client isn't visible on a projector and communication goes through a different medium.

Precondition

The moderator has entered the explain phase of a tool.

Basic course of events

1. The Digital Moderation stage client shows an example of the action and result phases of the tool.
2. The moderator uses a real time communication medium to provide explanation to the participants.
3. The participants listen to the moderator while they look at the stage client.

Postcondition

The participants have a good idea about how to use the tool.

4.1.4 A remote participant submits an answer

Summary

As with a non-distributed moderation, the participants send in their answers in the action phase for every tool. The difference in a distributed moderation is the use of a separate communication medium.

Precondition

The moderator has entered an action phase of a tool that allows participants to submit answers.

Basic course of events

1. The participant switches to the Digital Moderation participant client window. If the participant client was minimized, it will pop up automatically when the action phase is entered.
2. The participant enters his answer in the Digital Moderation participant client.
3. The modarator notices the participants' activity in the Digital Moderation facilitator client.
4. The participant clicks the button to submit the answer, which causes the Digital Moderation participant client to send the answer to the Digital Moderation server.
5. The moderator notices an answer has been submitted in the Digital Moderation facilitator client.

6. The participant switches back to the communication software. If only one answer can be submitted, the participant client minimizes automatically, causing focus to go back to the communication software.

Postcondition

The answer of the remote participant has been submitted to the system.

4.1.5 The moderator monitors participant activity

Summary

During the action phase in which participants submit their answers, it is important for the moderator to be aware of the activity of the participants and the progress towards the end of the phase. In a non-distributed moderation, the moderator can gain a lot of this information by simply looking around.

Precondition

The meeting has reached an action phase.

Basic course of events

1. After the action phase has begun, the participants start making their contributions using the Digital Moderation participant client. This causes the activity status of the client to change.

2. The Digital Moderation participant client sends status updates to the Digital Moderation server whenever the status changes.

3. The Digital Moderation server sends status updates to all Digital Moderation facilitator clients.

4. The moderator quickly sees who has started making a contribution in the Digital Moderation facilitator client.

5. Some participants pause while reconsidering their contribution or while being temporarily distracted. Others finish submitting their entry.

6. As with steps 1-3, this leads to status updates received at all Digital Moderation facilitator clients.

7. On the Digital Moderation facilitator client, the moderator sees some participants are no longer actively working on their contribution, while others have finished submitting their contribution.

8. The Digital Moderation facilitator client also displays the percentage of participants that finished submitting their contribution, so that the moderator has an idea about the progress towards the end of the phase.

Postcondition

The moderator is well aware about progress and participant activity.

4.1.6 The moderator notices a problem with a participant client

Summary

It is likely that problems will occur during a distributed workshop. Common examples are connection issues and participants accidentally closing their Digital Moderation participant client. It is important that the moderator is aware of these problems, so that he may be able to take action quickly. If a problem occurs in a non-distributed moderation, usually all the participant needs to do is raise a hand.

Precondition

A problem occured which prevents a participant from participating.

Basic course of events

1. When the participant accidentally closed his client, the Digital Moderation server is notified about the disconnection event immediately.
2. When a connection problem occurs, the Digital Moderation server will notice this after a short amount of time.
3. Both of the above problems cause the client's connection status to change.
4. The Digital Moderation server communicates the change in connection status to the Digital Moderation facilitator client.
5. The Digital Moderation facilitator client displays the client connection status together with the name of the client in the client status view.
6. The moderator notices the problem and sees immediately which client is having a problem, and the nature of the problem.
7. The moderator communicates with the participant in question and tries to resolve the issue.

Postcondition

The problem has been identified and resolved when possible.

4.1.7 The participants look at the results

Summary

When all participants have submitted their answers, the result phase of the tool usually follows. It is important that all participants can have a good look at the results. The difference with a non-distributed moderation is that the Digital Moderation stage client isn't visible on a projector and that a separate communication medium is used. Application sharing can be used to make sure that the results shown on the stage client are visible to the participants.

Precondition

The moderator has entered the result phase of a tool.

Basic course of events

1. The Digital Moderation stage client displays the results.
2. The moderator discusses the results with the participants.
3. The participants look at the results shown on the stage client.

Postcondition

All participants have been able to see the results and follow the discussion.

4.2 Functional requirements

Now that the core use cases have been described we can derive the functional requirements for the system. These requirements are listed in table 4.1 on the following page. Here follows a short explanation about each of them.

The use cases described in 4.1.3, 4.1.6 and 4.1.7 have indicated that real time communication between the moderator and the participants is necessary. Especially in 4.1.7 and when participants work together, it is also important that participants can communicate with each other. With communication comes the demand for identification and thus also authentication. This brings us to requirements R1, R2 and R3.

With communication also comes a demand for awareness. Especially the moderator needs to be well aware of participant activity or connection problems, as we can conclude from use cases 4.1.5 and 4.1.6. Thus follow requirements R4 and R5.

In 4.1.7, the participants look at the results. From this follows requirement R6. Since the Digital Moderation software also supports dividing the participants in topic area groups, also the topic-area specific results should be available, hence R7.

R1	Participants are required to identify and authenticate themselves via a login process. The system must not allow participation of unauthorized participants.
R2	The system must provide a means of communication between the remote participants and the moderator.
R3	The system must provide a means of communication between the participants.
R4	The facilitator client must provide detailed information about the status of remote participants. This includes displaying when a participant is typing, whether there is an answer in progress and whether a participant has been idle for an extended period of time.
R5	The system must provide participants information about the status of each other. This includes information on whether somebody is paying attention or composing an answer.
R6	The participants must have local access to the results. This can be provided either by the participant client or on the stage client via application sharing.
R7	When topic areas are used, the participant client must provide access to the results of the topic area in addition to the global results.

Table 4.1: The functional requirements.

4.3 Non-functional requirements

Many of the non-functional requirements of the Digital Moderation software also apply in the new context. Hence, the following list of non-functional requirements are mostly taken from the Digital Moderation software specification [DT04].

Stability

1. The program must not crash. This means, a best effort is made to minimize the risk on crashes.

2. The program must not become unstable when other applications fail. This includes failing of the connection to the server.

3. When the server crashes, it recovers to the previous state automatically after a restart.

4. When a client crashes, it reconnects to the server after a restart and requests the current state.

5. When a client can't reach the server or loses the connection, it will keep trying to connect at 10 second time intervals.

Serviceability

1. The system must provide information about the state of all clients (stable, crashed).
2. The user (also the participant) will be informed when his computer can't be used at the moment.

Extensibility

1. The system must be implemented in such a way, that additional tools can be implemented without making changes to the base of the system.

Scalability

1. The system must support up to 100 participant clients.
2. The system must support up to 700 participants. They are spread over 3 to 8 topic areas.
3. The time between a command performed by the moderator and the completion of the execution of this command must not take longer than 10 seconds.
4. The time between sending an answer from a participant client and the completion of processing this answer by the system must not take longer than 60 seconds.
5. The time between sending an answer from a participant client and the visual reaction on the local machine must not take longer than 2 seconds.

Now that both the functional and the non-functional requirements are defined, I will take a look at the related work in the next chapter.

5 Related work

In this chapter I present related work in this new application area for computer-supported moderation. In particular I look at support for web based meetings in competing moderation software, and discuss the experience with web conference tools and whether they fulfil the requirements set up in chapter 4.

5.1 Moderation software

Digital Moderation is not the only moderation software on the market. Similar products include the HELIUS® method from ConnectInc and NextModerator from NextPractice. As with Digital Moderation, these applications have been developed for co-located medium to large moderated meetings.

5.1.1 HELIUS® method

The HELIUS® method is based on the "ModerationMethod". Similarly to Digital Moderation, it replaces the sticky dots, pin boards and cards with an application running on several laptops.

The method is based around five primary phases [Con06]. First the warming up phase, where teams are formed and the teams choose their name. Then the collection phase, where the teams suggest topics. These topics are submitted to a central team of editors, where they are grouped together. The results are visualized on a projector. The next phase is the prioritization phase, where through placing dots the topics are rated. This allows the top topics to be defined within a few minutes. The 4th phase involves working the top topics into actions to take or projects to start. Again, the suggestions are grouped together by the team of directors as they come in. Through HELIUS®, the participants can at any time look back at information gained in earlier phases. Different topics can be discussed at the same time in parallel and in the same room. When project groups form, the participants can register these in the system. Finally, the closing phase is for reflection, evaluation and feedback. The gathered information is available electronically, and can be handed out to the participants directly after the meeting.

The HELIUS® method is marketed as a complete solution for moderated meetings, of which the software is only part. It is at the moment only applied to co-located meetings and does not provide for participants joining over the Internet, or organizing complete meetings online.

5.1.2 NextModerator

NextModerator is a computer-supported moderation method from NextPractice that applies the techniques and methods of small group moderation over a computer network (LAN) [nG06]. Up to one thousand individuals can participate at the same time, and the software supports methods like brainstorming, ranking and recommendations. The modular design of the software allows for numerous event structures.

The brainstorming and ranking phases are integrated, in the way that new ideas are visible to everybody and can be positively acknowledged with a mouse click. Positively acknowledged ideas will stay at the top of the list, while others drop down.

In the recommendation phase, responses can be added to the ideas, which are developed in small individual groups. Afterwards, each recommendation is rated on feasibility and importance, which allows them to be visualized on a 2D grid to quickly see which ones are the most interesting to deal with first.

Through the NextModerator software, the moderator has a good idea about the progress in a certain phase and can see potential problems in order to offer assistance when necessary.

Similar to HELIUS® and Digital Moderation, NextModerator is not used over the Internet and focuses on co-located meetings. NextModerator is particularly suitable for very large events.

5.2 Web conference tools

The moderation software we looked at does not provide for Internet based moderated meetings. Now we will have a look at some popular web conferencing tools, and in particular determine whether any moderation methods are already supported.

5.2.1 WebEx Meeting Center

WebEx[1] is an advanced web conferencing application. It allows users to share applications and files, and to communicate using chat, voice over IP, video or phone. The client is available for Windows and MacOS X. Once installed, joining a meeting can be done by clicking a link on a website or in an email. For communicating by phone, toll-free numbers are provided for many countries.

WebEx focuses on integration with existing applications, in particular office applications. It allows people to collaborate effectively or to hand over control to somebody at a technical helpdesk. It has a poll feature which can be used for a single-dot-question, but other than that it does not have any moderation methods built in. There is however the option of splitting up the group in smaller teams, so that they may handle certain tasks independently after which the group can be merged together again.

[1] WebEx website: http://www.webex.com/

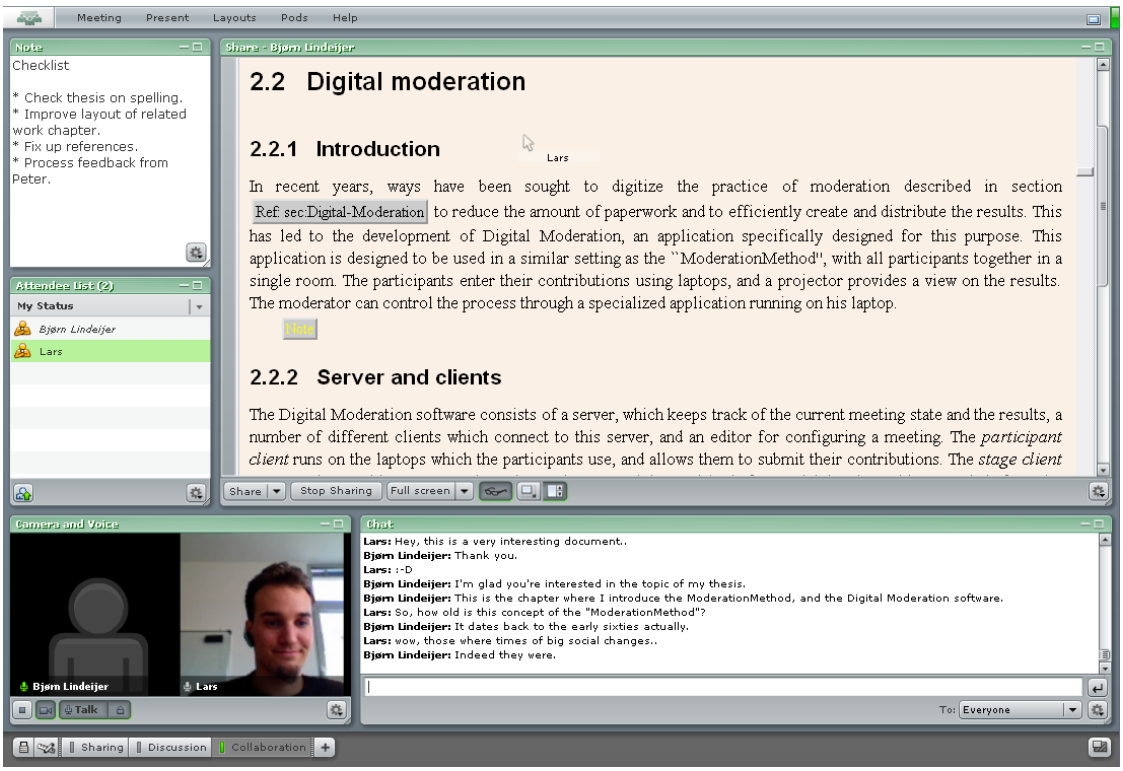

Figure 5.1: Acrobat Connect Professional during a test session.

5.2.2 Acrobat Connect Professional

Acrobat Connect Professional is a service from Adobe[2]. The application is almost completely implemented in Flash, so aside Flash no install procedure is necessary to join a meeting. However, an add-in does need to be installed for sharing files or applications. Acrobat Connect supports chat, audio and video communication, and sharing of desktop, applications and files. Users are able to see the position of each other's mouse cursors. The only moderation method supported by this tool is the single-dot-question, which can be simulated using a poll.

Figure 5.1 shows a test session I did together with a colleague. It demonstrates the chat and video communication and the application sharing feature allows us to discuss this thesis while it is being written. The mouse cursor of Lars is visible, which allows him to point at what he is talking about.

5.2.3 FastViewer

FastViewer[3] is primarily a desktop and application sharing tool, which also allows primitive chat communication and file sharing. The chat window has no support for typing awareness or notifying the user when being addressed personally. The file sharing feature

[2]Acrobat Connect Professional website: http://www.adobe.com/products/acrobatconnectpro/
[3]FastViewer website: http://www.fastviewer.com/

Figure 5.2: The web-based communication software vitero.

only allows you to hand over full access to your local filesystem to others, so requires that the participants can fully trust each other. Also in this application, telepointers allow the users to see the position of each other's mouse cursors. There is no support for any moderation methods.

5.2.4 Unyte (Skype)

Skype is a well known VOIP application which allows people to communicate over the Internet using chat and audio, and also integrates with the phone network. Unyte[4] integrates with Skype and offers desktop and application sharing. More features are provided by Unyte Meeting, supporting voice conferencing for over 500 users. Finally Unyte Events also allows event management and participant registration. Unyte doesn't add support for any moderation methods.

5.2.5 Vitero

Vitero stands for Virtual Team Room [vG06]. Its primary feature is to allow voice communication for small to medium groups and give the participant the impression of sitting together at a table, as shown in figure 5.2. It also offers other ways of communication in the form of text balloons and symbols. The participants are visualized

[4]Unyte website: http://www.unyte.net/

using avatars, usually a prepared picture of the participant. On the table in the middle sits a viewport that can provide application sharing or show a PowerPoint presentation. Vitero has primitive support for some moderation methods, including brainstorming and subsequent clustering of the ideas.

5.3 Conclusion

The direct competition of Digital Moderation does not yet allow for meetings to be organized on the Internet. However, the web conferencing tools that we've seen do not provide the moderation features that moderators would expect when considering to organize a moderated meeting over the Internet. However, they do fulfil many of the requirements I have set up in section 4.2. For all conferencing tools, user identification (R1) is achieved by having the participants choose a name when they join the meeting. Some, like vitero and Unyte, also require user authentication. Requirements R2 (communication with the facilitator), R3 (communication among participants) and R5 (status information) are all provided for to some degree, since all conference tools we looked at provide one or more ways of communication and different levels of awareness (typing awareness, telepointer, away status, etc.). Finally, local access to the results (R6) could be implemented based on application sharing. The fulfilment of these requirements makes it interesting to look into integrating Digital Moderation with a web conferencing application, in order to determine the usability of Digital Moderation is in this context.

As a first test, I've organized a meeting where Digital Moderation was not integrated into the communication medium. For communication a simple chat application was run on the side. This test is described in detail in section 6.1. After that, another test meeting was organized where Digital Moderation was integrated with vitero (see 5.2.5). This test is described in section 6.2.

6 Current experience

In this chapter I describe two test sessions where Digital Moderation is used for a distributed meeting in combination with two web-based conference tools, in order to find out the remaining gaps in the fulfillment of the functional requirements. These remaining gaps identify areas that still need work.

6.1 Experience with non-integrated chat

In order to determine the usability of Digital Moderation in a distributed meeting, I have tested the experience with using Digital Moderation in combination with a non-integrated chat based application. This also gives more insight about to which degree the functional requirements are satisfied by such software as well as which of the remaining functional requirements are the most important ones to satisfy.

From the requirements we can conclude that one of the biggest functional gaps in making Digital Moderation usable in distributed settings is communication (R2 and R3) and related awareness (R4 and R5). Both communication between the moderator and the participants as well as communication among the participants is necessary. A chat application running in addition to Digital Moderation provides for this. This is easy to set up and requires no extra hardware like head- or microphones. The test was done among colleagues who generally were familiar with the Digital Moderation software.

Another vital requirement is for the participants to have local access to the results, since there is no stage client visible to all (R6). To solve this, each participant runs his or her own stage client.

6.1.1 Setup

The subject of the meeting was to try coming up with a nice idea the T-shirt design of our Triathlon team. The agenda of the meeting started with a brainstorming session followed by a clustering in order to get rid of duplicate ideas (see section 2.2.3 for a short introduction about these tools). On the resulting set of ideas a ranking was performed, after which additional comments could be added. Finally, actions were defined as well as their deadlines and who is responsible for them.

After the meeting a portfolio was made to compare different aspects of traditional moderation, digital moderation and distributed moderation. Also a brainstorming was performed in order to gather ideas for improving the distributed moderation experience. In this document we won't talk about the actual results produced from the meeting, but

rather the experience of doing a distributed session using Digital Moderation, and the feedback gained afterwards.

6.1.2 Experience

Because no integration existed between the chat application and Digital Moderation, the start of the meeting was chaotic. Problems included difficulties in making sure everybody was available and technical problems with running the software. The participants had to launch three separate applications: the Digital Moderation participant and stage clients, and the chat application. Clearly, the setup was left too much configuration up to the users.

Once everybody had joined the chatroom and launched their participant and stage client as instructed, still a lot of communication problems remained. The agenda of the meeting wasn't clear, and no convenient way to communicate this to the participants existed. As such the participants were often confused about what they should do. Also, the participants didn't fully realize that it was important to pay attention to the moderator of the meeting. They often gave no response even when personally addressed, at times introducing the need for somebody to actually walk over to their office to see what the problem might be. Usually there wasn't really any problem, but the participant would simply be confused, might not have noticed the message or was not aware that a response was actually expected. A lot of this could be blamed on the need to handle multiple windows at once without any notifications, which easily causes something important to be missed.

Sometimes these problems were caused by participants doing multiple things at the same time or they might have simply left their office without notice, to make a small chat with another colleague participating in the meeting. It's part of a vicious circle: a chaotic meeting experience causes people to lose attention and wander off, and this in turn makes for an even more chaotic meeting experience. However, clearly the software was lacking proper display of a participant's status. It would have been very useful when the system would indicate whether somebody is away or busy with something else, so that the moderator and the other participants can take this into account.

6.1.3 Results

After the meeting all participants rated the general impression about *comfort*, *productivity* and *convenience* for three different situations: *traditional moderation*, *digital moderation* and *distributed moderation*. While only a distributed moderation was performed, the participants could be assumed to have at least some knowledge about the other two forms.

Surprisingly, traditional moderations were thought to be most comfortable and even most productive. Doing a digital moderation was only judged to be more convenient. Distributed moderation was judged worst for all criteria, indicating that there is a lot of room for improvement.

The brainstorming at the end yielded a lot of suggestions. A very popular suggestion was that integration was needed between Digital Moderation and the chat application. This would allow the two windows to be merged into one and make it easier to refer to submissions while reducing the amount of task switching. Improvements to the chat application were also suggested, for example only notifying about important messages (from the moderator or personally addressed messages) and having the ability to directly communicate to other participants. Voice communication was also requested, which could greatly improve the attention level of the participants.

Aside improvements to the communication among participants, a few other useful additions came up. A common view on the agenda of the meeting and the current status would be nice. Also, some participants would have liked to be able to go back to look at the submitted ideas. It would also have been better when submitted ideas would not have been anonymous. On the stage client, a telepointer might make it easier for the moderator to indicate what he is talking about.

6.1.4 Conclusion

While a lot of work can be done to enhance the comfortability, productivity and convenience of using Digital Moderation system in a distributed session, this test has shown that it is already possible to achieve a workable distributed meeting by adding simple chat based communication. However, the requirements I have set up earlier were not met to a satisfactory level, in particular the general awareness about the status and activity of other participants was not sufficient.

Obviously a lot of improvement could be made towards integration of the different applications. However, looking at the experience and the suggestions, it seems that chat as such is insufficient as a step-in communication method. While it is easy to set up, it requires some experience to use effectively because it doesn't offer any awareness beyond what participants actively make each other aware of by typing something. Hence, we did a similar test meeting using the audio based tool vitero (described in section 5.2.5). This experience is described below.

6.2 Experience with vitero, partly integrated

We had the opportunity of working together with vitero GmbH, making it possible to partly integrate Digital Moderation with their application, vitero. The integration included showing the Digital Moderation stage view to all participants using application sharing, and the ability for the moderator to launch the Digital Moderation participant client automatically for all participants with the click of a button. To make this possible, the participant client was bundled as a stand-alone application with the vitero client software. Vitero passes the correct server settings on to the client on launch, using command line parameters just like is usually done when the Java WebStart client is used. Figure 6.1 shows how the Digital Moderation stage client was integrated into vitero.

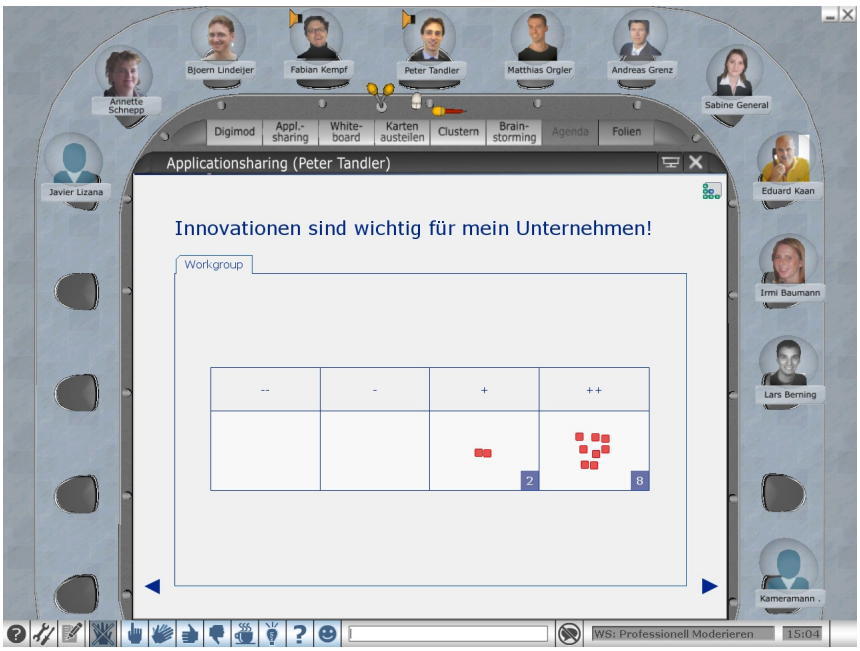

Figure 6.1: The Digital Moderation stage client embedded into the web-based communication software vitero.

6.2.1 Experience

One day in advance of the meeting, the vitero client bundled with the Digital Moderation participant client was sent to all participants. It offered for each participant an empty room to practice in, and they could already see the avatar image derived from the picture they submitted earlier. This made sure that once the meeting actually started, the participants were already somewhat familiar with the software and the meeting could start relatively soon after everyone had arrived.

Compared to the chat based meeting described in section 6.1, this voice based meeting felt a lot more effective. The participants were less likely to do other things in parallel and when a participant would switch to another task, vitero visualized this by putting a monitor in front of the participant's avatar.

The audio quality was good in general, except for a few participants who had bad microphones. This usually didn't bother since participants have to press a button to unmute their audio while talking, unless the single mobile virtual microphone is moved in front of their avatar by the moderator.

During the clustering that was part of the meeting, the text balloon feature proved very useful to come to a quick consensus about the names for new clusters.

6.2.2 Results

As with the previous distributed Digital Moderation meeting, a brainstorming was performed to gather feedback from the participants about what they thought about it and what could be improved. Contrary to the results from the chat supported session, this

time the suggestions were mostly cosmetic or functionality tweaks. This included remarks about the amount of whitespace used, scrollbars not drawing properly and no sufficient indication that an answer had been submitted.

However, also some important functionalities were still considered missing. This was not so much about the communication, in which area vitero did very well, but more about some functionalities that participants were missing in Digital Moderation.

The primary request from the participants was to have access to browse the results individually, including the ability to go back to previous phases in the moderation. It is not convenient to have to ask the moderator to do this in the common view, in which case all other participants are forced to look at the same. In this distributed setting, it would be very helpful if Digital Moderation provided a way to look back at previous brainstorming or ranking results, for example. It would also be nice to be able to see the contents of an arbitrary cluster or scroll up in a list of brainstorming ideas, without disturbing others.

Especially for the clustering tool, the participants would have liked to have better awareness about what is changing. Since the creation of a new cluster or the assignment of an idea to a cluster happens instantly, it is easy to miss during the blink of an eye or when you're temporarily distracted. This could be solved by having new clusters appear in a different colour for a while, as is done in the brainstorming result view. Also when a new cluster appears out of view, the view should automatically scroll to the new cluster.

Finally two integration issues between vitero and Digital Moderation are worth mentioning. In the Digital Moderation facilitator client, participant clients are identified with table numbers. In this setting however, it would be much more convenient when these table numbers would match the names of the participants. That way the moderator can easily see who has finished submitting his or her answer. Another feature the moderator would have liked would be to have a special status icon to be shown when a participant is busy typing something in the Digital Moderation client.

6.2.3 Conclusion

While the test meeting can be called a success, there are a few non-trivial features missing which can improve the efficiency and convenience of using Digital Moderation in a distributed session combined with vitero. While vitero added user authentication, the moderator was still not able to effectively identify the participants in the facilitator client because the participant name wasn't visible there. This means that R1 (identification) was not sufficiently met. R6, local access to the results, was expected to be met using vitero's application sharing feature. However as we have experienced in this test session the participants would prefer local control over the view on the results as well. Finally, R4 (activity awareness) wasn't met since vitero does not provide the moderator with any information about participant activity while using the participant client.

6.3 Remaining gaps

After an examination of currently available computer-supported moderation software and web conferencing tools in sections 5.1 and 5.2, we have seen that the combination of these two areas hasn't really been addressed yet. This has brought us to look at the current state of the art when web communication software is combined with moderation software, in particular when Digital Moderation is combined with either a chat application running on the side, or partly integrated with vitero.

From these tests I have concluded that many of the requirements set up in chapter 4 can be fulfilled in this way. Requirements on communication between different people in the meeting (R2 and R3) have been sufficiently met by using vitero, for example. I have also concluded, however, that a number of requirements, in particular R1 (identification), R4 (activity awareness) and R6 (local access to the results) are still not addressed fully. In the next chapter I describe the design and implementation of the improvements I have made to Digital Moderation in order to address these remaining gaps. In chapter 8 I evaluate the effect of these additions in another combined Digital Moderation and vitero meeting.

7 Design and implementation

I have now looked at related work and evaluated the experience with two different applications offering web-based communication. Then I have set up the remaining gaps compared to the functional requirements determined in section 4.2. In this chapter I describe the concept, design and implementation of the solutions that I have developed in order to fill these remaining gaps. Table 7.1 shows an overview of which of the functional requirements are addressed in which section.

7.1 Participant identification

7.1.1 Concept

One of the primary requirements to fulfil is that the participants can be quickly and reliably identified by the moderator while monitoring the activity of the group and paying attention to possible problems (R1 in section 4.2 on page 20). In our primary usage scenario, the participants are distributed: every participant has his own workstation and is connected via the web. Also, the participants are able to communicate with the moderator as well as with other participants, via a web-based communication solution such as vitero.

As can be concluded from the experience gained in the first session with vitero, described in section 6.2.2 on page 32, the most convenient thing to do would be to use the same name for the Digital Moderation participant client in the Digital Moderation facilitator client, as is used to identify the participant in vitero. That way, when a problem occurs at a participant or when a participant seems inactive for an extended period of time, the moderator directly knows who to address.

Requirement	Fulfillment by
R1	Web conference tool and changes in section 7.1
R2	Web conference tool
R3	Web conference tool
R4	Changes in section 7.3
R5	Web conference tool
R6	Changes in section 7.2
R7	Not addressed

Table 7.1: Overview of fulfillment of functional requirements

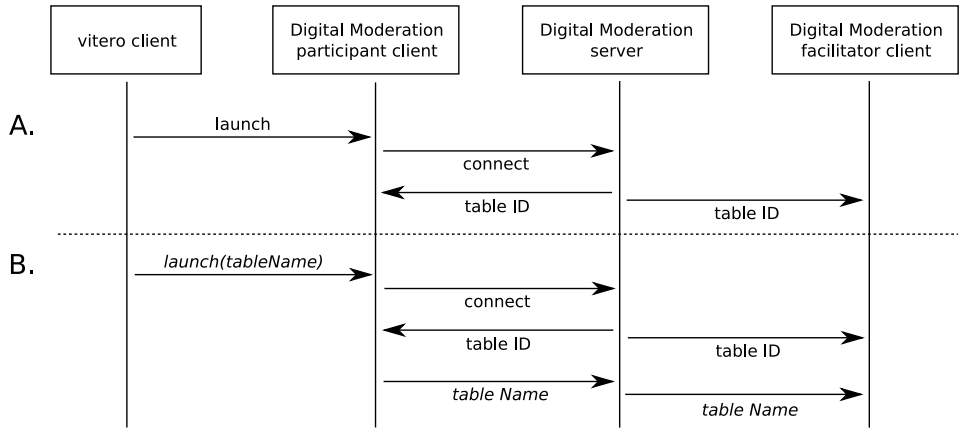

Figure 7.1: Sequence diagram of vitero client launching the Digital Moderation client. Situation B shows the addition of table name communication.

7.1.2 Design

In the test session with vitero described in section 6.2, the integration was solely client-side. In this situation it makes sense to transfer the participant name from the communication tool to the Digital Moderation participant client. I have done this by using a system property, which can be passed as a command line parameter. After the client connects to the server, the value of this variable is sent as the *table name*. The server passes this on to the facilitator client as part of the client status information.

The situation before this change is visible in part *A* of the sequence diagram in figure 7.1. Part *B* shows the display name being passed from vitero to the participant client that it launches. The addition of the display name communication changes nothing in the existing connection sequence. This approach is safe and effective. It avoids modifying the *table ID*, which is also used during communication between peers and hence must be unique.

Now the Digital Moderation facilitator client can display the participant's name in the clients table, giving the moderator a means of identifying which participant belongs to each client. This is shown in figure 7.5 on page 39. Finally, the Digital Moderation participant client displays the participant's name in the top right instead of the table ID, as shown in figure 7.2. This has the added advantage of confirming towards the participant that he is being identified correctly.

Figure 7.2: The participant client header, showing the name of the participant.

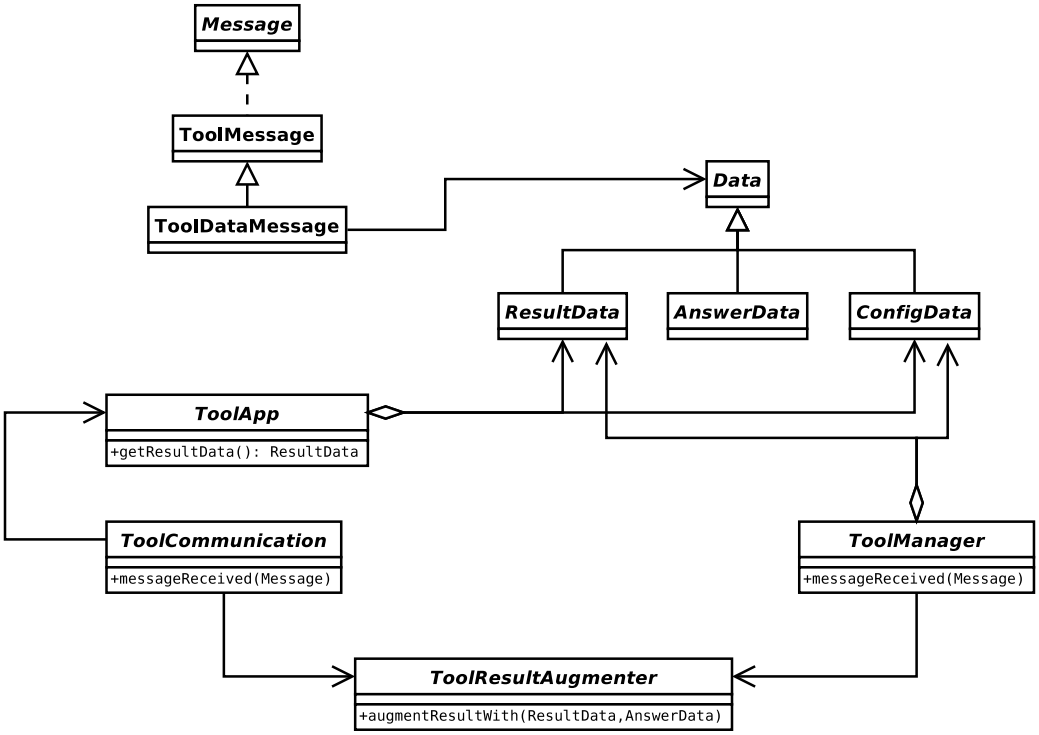

Figure 7.3: UML class diagram around tool communication layer.

7.2 Access to results on participant client

7.2.1 Concept

From the feedback from participants obtained during the first session with vitero described in section 6.2.2 on page 32, it can be concluded that the shared view on the stage client offered by vitero was not satisfying all users as a means of giving access to the results locally. This means that requirement R6 (providing local access to the results) was not fulfilled.

For most tools, the Digital Moderation stage client features a view on the results during the result phase, while the Digital Moderation participant clients are locked. To give the participants access to the results locally, it would be enough to replace the locked screen on the Digital Moderation participant client with the same view on the results as is used in the Digital Moderation stage client. This would work for most tools, except for the clustering and the action item tools, since their action phase is performed on the Digital Moderation stage client. For these tools, a special read-only result view needs to be implemented which can be shown on the participant client during the action phase.

7.2.2 Design

Before the Digital Moderation participant client can display anything meaningful, the information needs to be available. For most tools this is already the case, because the result data of a tool can be used by subsequent tools and thus it needs to be available on the participant client. However, tools like the stage clustering and action item work during the action phase, and communication happens only between the stage client and the server.

The design of the tool communication layer is shown in figure 7.3. On the client side, tool specific implementations of *ToolApp* and *ToolCommunication* perform respectively the logic and the communication for a tool in the meeting script. The tool application keeps an instance of a *ResultData* derivative, which stores the complete result for a certain tool. On the server side, an implementation of *ToolManager* performs a similar task. Both sides use an instance of a *ToolResultAugmenter* derivative to augment an existing *ResultData* instance with an incoming answer. Also, both sides have access to the tool's configuration through an instance of a *ConfigData* derivative.

I have introduced an option in the tool configuration data telling whether participant clients should be displaying the results. When this option is enabled, the tool manager forwards changes during stage clustering and setting up action items as they happen, to allow a client-side result component to display the progress.

On the clients, all tool GUI components are obtained from a tool component factory based on the station and phase. For most tools it was straight forward to have their result component created on the participant client in the result phase instead of the locked component. For the stage clustering tool and the action item tool, I have implemented additional read-only result components. Figure 7.4 shows the new situation during stage clustering, with the stage client on the left and a participant client on the right.

Since the option to show the results on the participant introduces additional communication towards all clients, care should be taken that the scalability demands set in section 4.3 are still met. However, the amount of additional communication for the stage cluster and action item tools is not expected to exceed the level of communication of any of the other tools. The system already has to cope with messages coming from all participant clients, which are broadcasted to all clients (for example during a brainstorming), while in this new situation only the changes made on the stage client are broadcasted.

7.3 Activity awareness

7.3.1 Concept

Both the experience gained from the session with a non-integrated chat application, as well as the session performed in combination with vitero showed the need for better awareness of participant activity. As per R4 (activity awareness, see 4.2), the moderator needs to have a good idea about the activity of the participants and know about any problems to perform his task effectively. In neither of the two test sessions was this

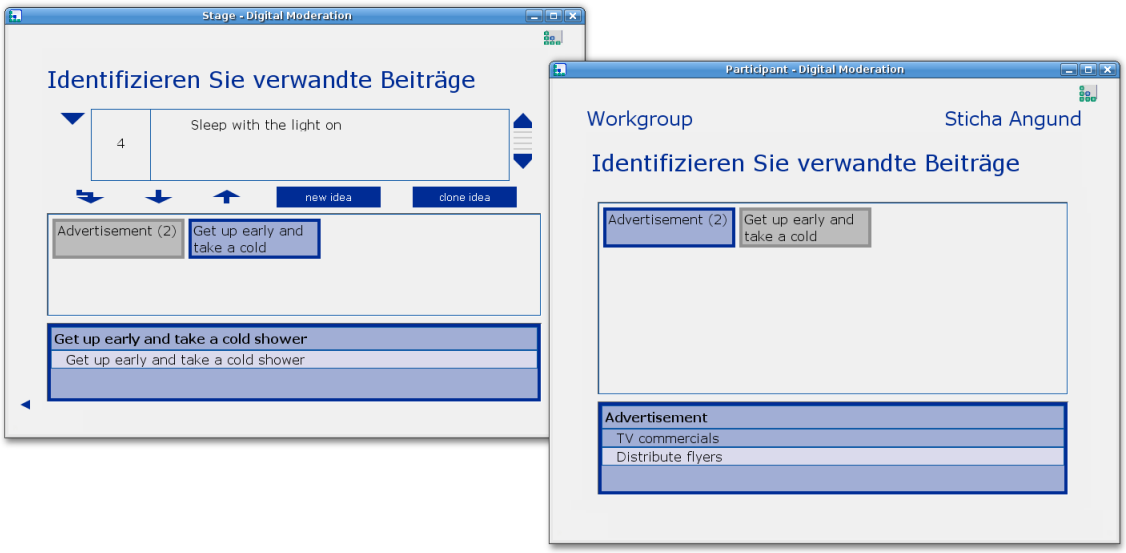

Figure 7.4: The stage client (left) and the participant client (right) during a clustering session. Notice that the participant is looking at a different cluster than shown on the stage.

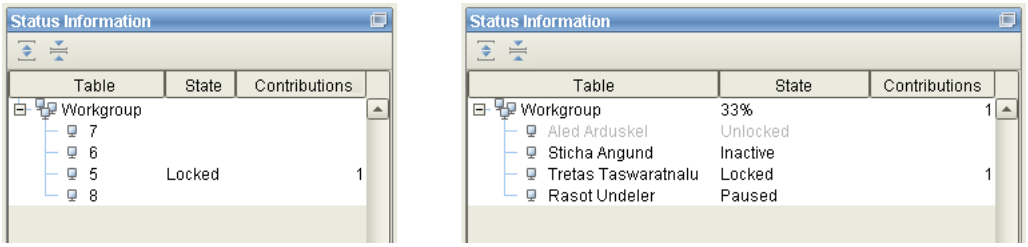

Figure 7.5: The status information displayed in the facilitator client previously (left) and the same clients with their names and more status information added (right).

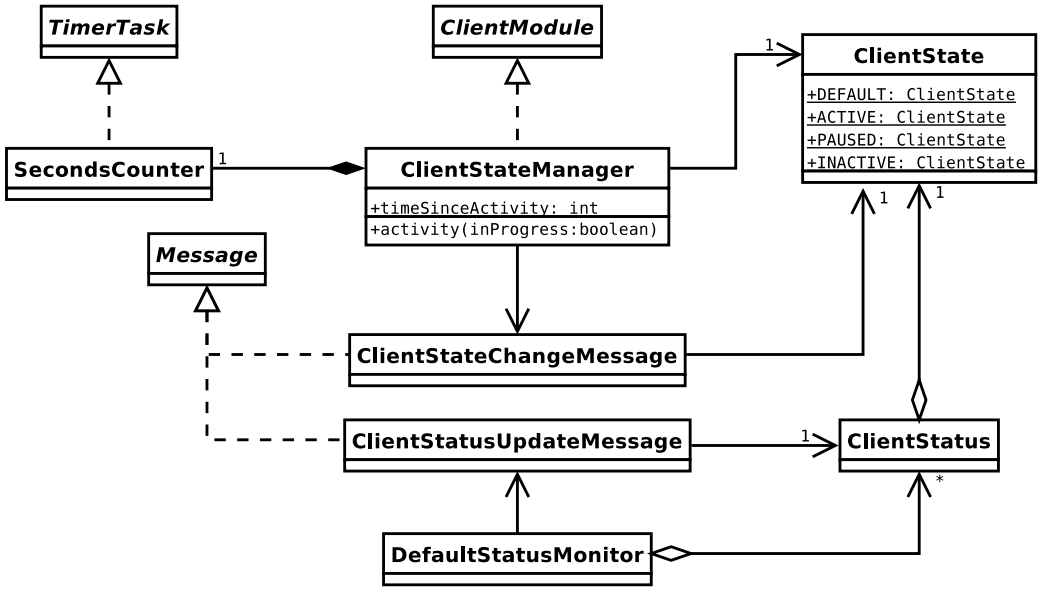

Figure 7.6: UML class diagram around the client state manager and server status monitor.

requirement sufficiently addressed. In the design of this feature we assume that the problem of participant identification is taken care of by implementation of the solution for R1 described in section 7.1 on page 35.

The nature of this feature asks for direct integration into the Digital Moderation system. This is because the Digital Moderation server keeps track of client connection status, and because any information about activity will originate from the Digital Moderation participant client. The Digital Moderation participant client will directly monitor the usage of certain input elements by the participant, in order to be able to tell whether the participant is active and whether there's an answer being composed. If there is no answer being composed, then there is no longer any relevant activity as far as the moderator is concerned. The activity of a participant can be described as a state; there are several states a participant may be in. In the current system these states are *Locked* and *Unlocked*. New states have to include information about activity, for example how long the participant was inactive and whether an answer is being composed. One can argue that the time of inactivity is only relevant when an answer is being composed. Also, the state *Disconnected* is introduced and displayed in a special way on the Digital Moderation facilitator client.

The complete set of states and the transitions between them in the new situation is visualized in figure 7.7. Before choosing these new states, I have looked at an extension of the XMPP protocol[1], in particular XEP-0085: Chat State Notifications [SAS06]. While there is no direct match between the states and transitions in this protocol, the separation between *Paused* and *Inactive* states was taken from there.

[1]Extensible Messaging and Presence Protocol - http://www.xmpp.org/

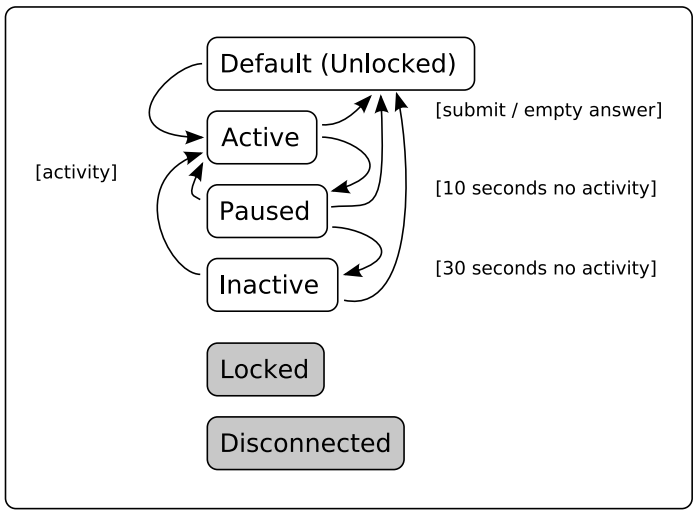

Figure 7.7: Client state diagram.

7.3.2 Design

While it would make sense to integrate the new states with the existing client module that takes care of the *Locked* status, I have implemented the activity awareness in a separate module. This avoids some of the technical challenges which were not relevant in the context of this thesis. For example the client has sole control over his activity status, while the locked status can be controlled from the server. Also, the *Locked* and *Unlocked* states affect the local client GUI, while the activity state does not.

The separate module is called the client state manager. The related classes and their relationships are shown in figure 7.6. Each client module is able to send and receive messages. The *ClientStateManager* only sends messages of type *ClientStateChangeMessage*. These messages are only sent when the state changes, and they are received on the server by the *DefaultStatusMonitor*. A change of client state will cause the status monitor to send a *ClientStatusUpdateMessage* to the facilitator client. The client state manager has one important public method, *activity(inProgress:boolean)*, which is called from the GUI on user activity. The *inProgress* parameter indicates whether the input field are empty or whether an answer is being composed.

For each tool I have added listeners to the input fields in their answer component, that notify the client state manager about any activity and also tell whether there is an answer in progress. Activity can cause there to be no longer an answer in progress when the last character in an input field is erased, for example.

The client state manager counts the seconds in a separate thread running parallel to the rest, using the *SecondsCounter*. When 10 seconds have passed without any activity, with an answer in progress, the state switches to *Paused*. When after 30 seconds still nothing happened, the state is changed to *Inactive*. On any activity the state switches back to *Active*, unless the activity caused the answer to be empty, in which case the state changes to *Unlocked*. Finally, the state also changes to *Unlocked* when an answer is submitted. All this is illustrated in figure 7.7.

Lastly, since activity awareness also includes to be aware of the progress made by the participants, the facilitator client now includes for each workgroup the percentage of clients who are done with submitting their contributions, as shown in figure 7.5. This works based on the *Locked* status, so it only works with tools where the client is locked once the answer is submitted. This feature is especially convenient once the group gets larger or when there are multiple topic areas.

As with the enhancement made in section 7.2, this change adds to the amount of communication between the clients and server. While it is possible for a client to deliberately cause frequent state changes, this is not a realistic scenario. However, it is expected that for each answer that a client submits, its state changes at least once to *Active* (on starting to compose an answer) and at least once to *Unlocked* (when submitting the answer). Since the submission of answers make up most of the communication during a meeting, we can safely say that the amount of messages sent will at least triple. However, since the state change and status update messages are very small, hardly require any processing and do not need to be made persistent, this is not expected to affect the performance or scalability of the system significantly.

Now that I have described the concept and design behind each of the solutions that I have implemented, in the next chapter I will verify whether they indeed provide the functionality required in order to fill the remaining gaps.

8 Evaluation

In this chapter I describe the experience of using the Digital Moderation with the functional additions described in chapter 7. In this meeting the partially integrated voice based communication software vitero was used again, because that turned out to yield the best results in section 6.2.

8.1 Introduction

A distributed workshop was held among 11 participants. The topic of the workshop was to determine the problem areas in innovation by companies. As with the previous vitero sessions, the participants were sent their client a few days in advance so that they could get familiar with vitero before the meeting started. The meeting was recorded on video with the consent of the participants.

As per convention, the participants were introduced to each Digital Moderation tool right before first use. The meeting consisted of a single dot question, followed by a brainstorming session, a clustering of the generated ideas, a ranking of these clusters, adding recommendations to these clusters, ranking the recommendations to the ranked clusters and finally adding additional notes to the ranked recommendations.

One of the first questions the group was asked was how much experience they had with Moderation (based on the ModerationMethod, described in section 2.1). The question was asked using the 2D single dot question tool (described in section 2.2.3), so that participants had the ability to answer the question for four different kinds of Moderation: Workshop Moderation, Moderation over the internet (for example as done in this test), large events without the use of computers and large events with the use of computers. The results are shown in figure 8.1. It shows that among this group there is moderate familiarity with Workshop Moderation and its use in large events, but very little experience with its use over the internet and large computer supported events.

8.2 Observations

During the meeting I have paid special attention to the use of the newly added functionalities. Questions to ask in this respect are:

- Can the moderator quickly identify the participants in the facilitator client?
- Is the moderator well aware of participant activity and potential problems?
- Do the participants appreciate the local view on the results?

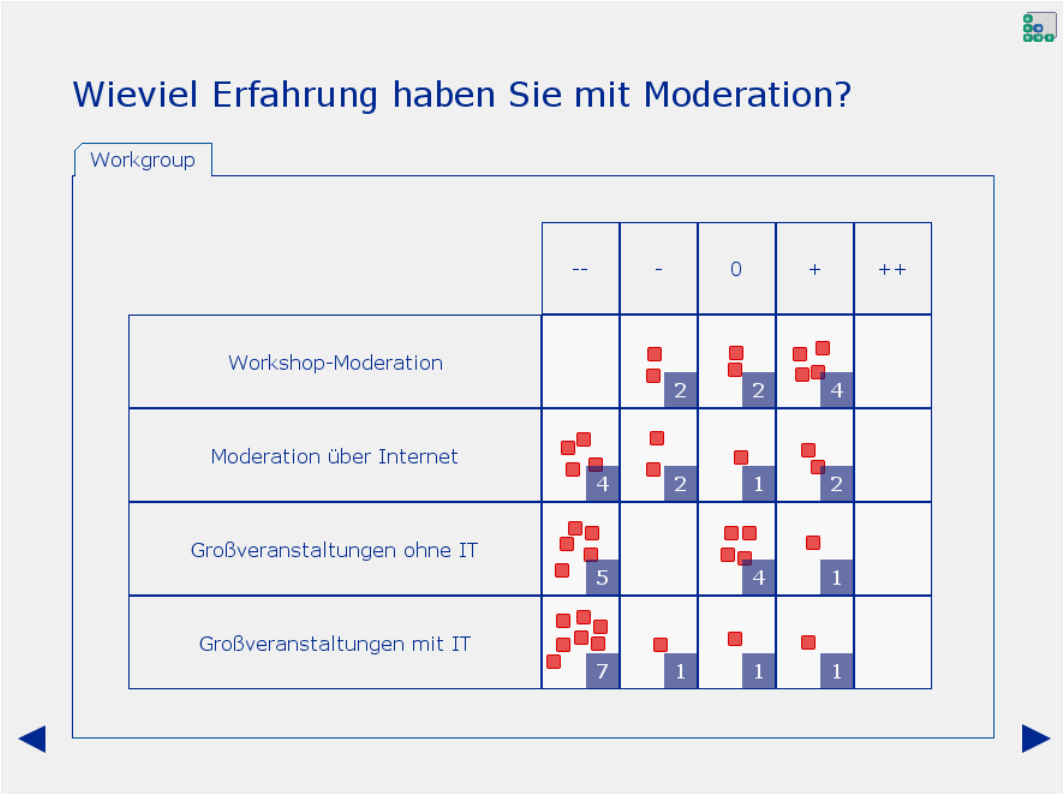

Figure 8.1: Results of the 2D-single-dot-question tool, used to poll the amount of experience with Moderation among the participants.

#	Time elapsed	Observation
1	00:03:30	A participant complains about the process of recovering the minimized window.
2	00:09:48	The moderator notes that he can see three participants have already submitted their answer.
3	00:24:08	A participant closes her client. The moderator notices this and restarts her client.
4	00:25:18	Some participants indicate scrolling performance is very bad (scrolling happened here for the first time).
5	00:34:55	Shortly after the clustering has started, the moderator notes to the participants that they can browse the clusters on their local clients.
6	00:40:23	The moderator notes again to the participants that they can browse the clusters locally.
7	00:46:36	A participant asks whether she has already submitted. The moderator can immediately answer than she didn't submit yet.
8	00:48:51	A participant accidentally closed her client. The moderator restarts it for her. She thanks the moderator.
9	01:09:23	The moderator notes that he can see submissions coming in. A participant asks whether he can also see who answers what. The moderator says the system is anonymous by design and that it only displays whether there is activity or not.

Table 8.1: Observations made during the second test session using Digital Moderation in combination with vitero.

Table 8.1 shows my observations during the meeting. From observations 3, 7 and 8 we can conclude that the moderator had no problems to quickly match the participants in the meeting with the client displayed in the Digital Moderation facilitator client. This is sufficient evidence that the user identification part of requirement R1 has been fulfilled by displaying the same participant names in the facilitator client as displayed in the communication tool.

Unfortunately there is little evidence of participants using their local results view. At observation 4 the participants indicated the scrolling performance of the shared view on the Digital Moderation stage client was bad. This is partly because the application sharing technology is unaware of the scrolling, and sends the complete screen over the network for each small scrolling step. A local view on the results would not suffer from these problems, but unfortunately in this state the participants were not yet able to see the results on their own client because it was displaying the answer component, which allows them to submit their ideas. Later on, during the clustering, the moderator noted twice to the participants that they had the ability to browse the created clusters locally (5 and 6). Unfortunately it was technically not possible to see whether the participants were using this possibility.

Observations 2 and 7 show that the moderator is aware of whether participants have submitted their answer. Then, at observation 8, the moderator's appreciation of being able to see the activity of the participants in detail becomes apparent. There was an immediate worry from the participants though, about the anonymity of their contributions. This could be related to the slightly ambiguous formulation by the moderator, who couldn't actually see the submissions that were coming in.

8.3 Conclusion

From the observations described in the previous section I conclude that no serious problems remained. It was evident that user identification (R1) and activity awareness (R4) were sufficiently addressed and their solutions appreciated by the users. Having local access to the results (R6) didn't have any observable impact, but users no longer asked for this like they had in the previous test described in section 6.2.2 on page 32.

9 Conclusions

In this final chapter I summarize the findings in this thesis and their implications. I also make suggestions for further research.

9.1 Summary

I started out with the objective to determine the functional requirements of a system allowing for web-based distributed moderated meetings and to provide a solution that fulfils the most important of these requirements. Subsequently, I have described the moderation technique in some detail and introduced the Digital Moderation software developed at Fraunhofer IPSI & IGD in chapter 2, which formed the basis of my solution.

In chapter 3, I have analyzed the problem area by defining three usage scenarios, which led to the use cases which helped to define the functional requirements in chapter 4. These formed the answer to the first research question, defined in section 1.3.

Knowing the requirements for the system I had to realize, I analyzed related work in chapter 5 in order to determine which requirements were already fulfilled by current software and which requirements still had to be fulfilled. In chapter 6, two tests were performed in which Digital Moderation was combined with web-based communication software, which gave a good impression about what was lacking. This provided the answer to the second research question, namely which of the functional requirements were vital and suitable for proof of concept implementation. These requirements were user identification (R1), activity awareness (R4) and local access to the results (R6), which were defined in chapter 4.

I then designed and implemented solutions to these primary requirements. Fulfilling these requirements required making enhancements to the Digital Moderation software. I have described the design and implementation of each solution in chapter 7, also discussing the effect these changes could have on the non-functional requirements.

Finally, I have tested whether the enhancements to Digital Moderation indeed fulfilled the requirements in chapter 8. This was done by combining the improved Digital Moderation with the vitero web-based communication software, and observing the differences with the earlier session. This provided the answer to the third research question, which was to what extend the functional additions fulfilled the functional requirements. In section 8.3, I have concluded that the functional additions have fulfilled the three vital functional requirements which they were meant to address.

9.2 Discussion

The improvements I have made to the Digital Moderation software based on the functional requirements determined earlier are not new in the area of web-based communication software. The ability to identify another user properly is imperative, for without knowing who to address, all you can do to get a message across is to address everybody. Knowing something about the activity status of another user is not a requirement for communication, but is useful enough to be included in most of today's instant messaging protocols (for example in XMPP). Finally, having some local control over certain shared data is common in for example tools supporting shared whiteboards.

However, what I have found in chapter 6, is that such functionality doesn't automatically extend to the complete system when a web-based conferencing tool is combined with an application like Digital Moderation. The enhancements described in chapter 7 were necessary to fulfil the requirements for the combined system. As such, they do represent a step forward in the context of distributed web-based workshops.

What I have shown in chapter 8, aside the level of fulfillment of the functional requirements, is that it is now indeed very feasible to hold distributed moderated workshops like the ones envisioned in the introduction. Finally, I believe bringing moderated workshops to the web is only the beginning.

9.3 Future work

Completing the functional requirements

While Digital Moderation can now be used properly for a distributed moderated meeting in combination with vitero, two functional requirements were not fulfilled and still need to be handled for completion. The first of these is the authentication part of R1 (user identification). As it is now, the system is not secure and anybody who can figure out which parameter to use can join a meeting with a name of their choosing.

The second unfulfilled requirement is R7, support for topic areas in locally shown results. Topic areas allow for dividing the participants into groups, and normally the results for each group are visible in addition to the complete results. The Digital Moderation participant client should be able to show not only the complete results, but also those of the topic area.

Alternative conference tools

For our tests we have chosen to combine Digital Moderation with vitero. However, a number of alternatives exist and it would be interesting to see how well Digital Moderation can integrate into other conference tools and whether this changes anything in regards to the requirements or the expectations from the users.

Large scale meetings

In the tests performed in this thesis, the distributed meetings were relatively small scale compared to most colocated meetings, which typically host from two dozen participants up to 500. Future research can be done towards the additional requirements and effects on performance of holding larger scale meetings over the internet. Vitero supports up to 40 participants and additionally 200 listeners, who cannot talk.

New interactive ways of cooperation

Looking at the way distributed moderation is currently taking place, it becomes clear that it is basically just digital mirror of what we are used to in the real world. However, real-time data communication, automatic data processing and new ways of presenting information can all contribute to new methods of cooperation. An example of this is the integration of tagging into moderation software like Digital Moderation [Ber07].

Further research on awareness

Further research can be done towards ways to enhance awareness of the meeting progress, of the information flow and of other participants in the meeting. The awareness features that were implemented were generally regarded as very useful and enhanced the productivity.

Bibliography

[Ber07] Lars Berning. A tagging based tool for clustering of large idea collections in computer supported facilitation. Master's thesis, Fraunhofer IPSI & IGD, 2007.

[BM01] G. Bafoutsou and G. Mentzas. A comparative analysis of web-based collaborative systems. In *Proceedings of the 12th International Workshop on Database and Expert Systems Applications*, pages 496–500, 2001.

[CFKW01] S. Cogdill, T. L. Fanderclai, J. Kilborn, and M. G. Williams. Backchannel: whispering in digital conversation. In *Proceedings of the 34th Annual Hawaii International Conference on System Sciences*, pages 8 pp.+, 2001.

[Con06] ConnectInc. Digital moderation: http://www.connectinc.biz/, 2006.

[DT04] Laura Dietz and Peter Tandler. Digimod specification v1.0: Software requirements. Fraunhofer internal document, June 2004.

[GG98] Carl Gutwin and Saul Greenberg. Effects of awareness support on groupware usability. In *CHI '98: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 511–518, New York, NY, USA, 1998. ACM Press/Addison-Wesley Publishing Co.

[GG99] Carl Gutwin and Saul Greenberg. The effects of workspace awareness support on the usability of real-time distributed groupware. *ACM Trans. Comput.-Hum. Interact.*, 6(3):243–281, September 1999.

[GG02] Carl Gutwin and Saul Greenberg. A descriptive framework of workspace awareness for real-time groupware. *Computer Supported Cooperative Work (CSCW)*, 11(3 - 4):411–446, 2002.

[GPS04] Carl Gutwin, Reagan Penner, and Kevin Schneider. Group awareness in distributed software development. In *CSCW '04: Proceedings of the 2004 ACM conference on Computer supported cooperative work*, pages 72–81, New York, NY, USA, 2004. ACM Press.

[GRG96] Carl Gutwin, Mark Roseman, and Saul Greenberg. A usability study of awareness widgets in a shared workspace groupware system. In *CSCW '96: Proceedings of the 1996 ACM conference on Computer supported cooperative work*, pages 258–267, New York, NY, USA, 1996. ACM Press.

[Kha01] Salina Khan. Electronic meetings increase as travel eases. *USA TODAY*, May 2001.

[KPVB02] M. Kirsch-Pinheiro, Valdeni, and M. R. S. Borges. A framework for awareness support in groupware systems. In *The 7th International Conference on Computer Supported Cooperative Work in Design, 2002.*, pages 13–18, 2002.

[KSS02] Karin Klebert, Einhard Schrader, and Walter G. Straub. *ModerationsMethode. Das Standardwerk*. Windmühle Verlag GmbH, 2002.

[Lar04] Craig Larman. *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development (3rd Edition)*. Prentice Hall PTR, October 2004.

[nG06] nextpractice GmbH. nextmoderator: http://www.next-practice.com/tools/nextmoderator/, 2006.

[SAS06] Peter Saint-Andre and Dave Smith. Xep-0085: Chat state notifications: http://www.xmpp.org/extensions/xep-0085.html, 2006.

[Sch02] Kjeld Schmidt. The problem with "awareness": Introductory remarks on "awareness in cscw". *Computer Supported Cooperative Work (CSCW). The Journal of Collaborative Computing*, 11(3-4):285–298, 2002.

[Sei02] Josef W. Seifert. *Visualization, Presentation, Moderation*. John Wiley & Sons, July 2002.

[Sei07] Josef W. Seifert. *Moderation: What is it?* Seifert & Partner GbR, 2007.

[vG06] vitero GmbH. vitero - virtual team room: http://www.vitero.de/english/, 2006.

[YMW+05] Nicole Yankelovich, Jen Mcginn, Mike Wessler, Jonathan Kaplan, Joe Provino, and Harold Fox. Private communications in public meetings. In *CHI '05: CHI '05 extended abstracts on Human factors in computing systems*, pages 1873–1876, New York, NY, USA, 2005. ACM Press.