Journal of Universal Computer Science, vol. 15, no. 16 (2009), 3139-3157
submitted: 30/5/09, accepted: 29/9/09, appeared: 1/10/09 © J.UCS

Diminishing Chat Confusion by Multiple Visualizations

Torsten Holmer
(Upper Austria University of Applied Sciences, Hagenberg, Austria
torsten.holmer@fh-hagenberg.at)

Stephan Lukosch
(Delft University of Technology, Delft, The Netherlands
s.g.lukosch@tudelft.nl)

Verena Kunz
(FernUniversität in Hagen, Hagen, Germany
verena.kunz@fernuni-hagen.de)

Abstract: In this article, we address the problem of confusion and co-text-loss in chat communication, identify requirements for a solution, discuss related work and present a new approach for addressing co-text loss in text-based chats. We report about first experiences with our solution and give an outlook on future work directions. The core idea of our solution MuViChat (multiple-visualization chat) is to support multiple visualizations of referenced chat transcripts in which users can choose their preferred view. The multiple visualizations offer different possibilities to follow and understand a communication and thereby diminish chat confusion which often occurs in standard chat systems. By enabling the recording and replaying of chat discussions and an extensible modular architecture we are supporting evaluation and further integration of advanced visualization concepts.

Keywords: Chat tool, multiple visualizations, threading, chat transcript
Categories: H.4.3, H.5.3

1 Introduction

Chat systems like IRC or web-based chats support written communication of multiple participants. They are widely accepted by users as they have low requirements concerning computing power or network bandwidth. Recently, text chats are often mixed with audio-chats. This leads to new problems and questions that have to be considered when communicating with each other, e.g. what has to be written and what has to be said or how is it possible to link textual information with audio information. However, this article focuses on text-based chat systems and users that use a text-based chat system to communicate with each other. Hence, we refer to text-based chat systems when talking about chat systems.

As chat systems enable computer-mediated interaction and communication, chat systems are part of the interdisciplinary research domain of computer-supported cooperative work (CSCW) [Liu et al., 2001] and cooperative learning (CSCL) [Münzer and Xiao, 2005]. Additionally, chat systems are a research subject for communication science as well as linguistics. Therefore, there are several studies that deal with the linguistical particularities of chat communication and the special role of chat communication in the field between speech and writing (cf. [Garcia and Jacobs,

1999], [Werry, 1996]). Chat systems are used as a single application or in combination with other tools in order to support more complex cooperation processes [Kuo et al., 2001] [Geyer et al., 2008].

There are several advantages of using chat systems for communicating with each other. Chat systems do not depend on a specific location. From a cost perspective, chat systems are a cheap form for communicating with each other. As users are not aware of what the communication partners are doing, it is not considered as impolite to deal with other things in addition to chat with each other. The latter includes the participation in several chat-based communications. For a lot of users, communicating via chat systems is attractive because of the humorous and playful speech. Finally, compared to oral communication, most chat systems allow users to refer to previous communications and check the own or the communication partner's statements [Herring, 1999].

However, there are also disadvantages for communicating via chat systems. Smith et al. [Smith et al., 2000] identify five major problems for communication via chat systems:

(P1) Lack of links between people and what they say
(P2) No visibility of listening-in-progress
(P3) Lack of visibility of turns-in-progress
(P4) Lack of control over turn positioning (co-text loss)
(P5) Lack of useful recordings and social context

To what degree these problems occur or influence a chat-based communication is highly dependent on the characteristics and offered functionalities of the used chat system, P1, P2, and P3 mainly influence the communication dynamics. P4 and P5 mainly influence the presentation of the chat-based communication.

Current chat systems do not sufficiently address the co-text loss, i.e. P4. In this article, we will therefore especially tackle this problem by introducing a new chat system that offers different visualizations to overcome co-text loss. We first identify the requirements for such a chat system. For that purpose, we take a closer look on an example scenario and define our understanding of co-text loss. Based on the requirements, we review the state-of-the-art. Then, we present our chat-system MuViChat (multiple-visualization chat) which fulfils the identified requirements to tackle the co-text loss problem. Before concluding and giving an outlook on future work directions, we report on experiences and a preliminary evaluation of MuViChat.

2 Requirements Analysis

Standard chat systems visualize chat messages in the order in which they arrive at the server. As all chatters can simultaneously write and send messages all have the right of speech. Hence, the server defines the order of the messages as they appear in the chat window [Garcia and Jacobs, 1999]. As result, the shown conversation is not linear, i.e. a chat message may not refer to the previous one but to another which has been posted earlier. This results in parallel intertwined conversation threads which may even discuss different topics. Due to these parallel threads a chat participant might not know to which previous chat messages a message refers and thus might not be able to follow and understand the conversation. Pimentel et al. [Pimentel et al.,

2003] describe such a situation as co-text-loss. To illustrate this table 1 gives a prominent example for co-text-loss.

Green:	Did you see that new Mel Gibson movie – I think it is called "Payback"?
Blue:	I saw the academy awards last night. Did you watch it?
Blue:	yep.
Blue:	It was very violent, but funny.
Green:	You saw it? You liked it?
Green:	How did it end up – who won?
Yellow:	I heard it was good.
Blue:	It was OK. At least Titanic didn't win everything.
Green:	I guess you can only be king of the world once.

Table 1 Well-known example for co-text loss [Vronay et al., 1999]

Linguistics considers "co-text" as text which has been written immediately before and after a message and which is helpful or necessary to understand a message. In chat systems the messages around a message are often not the intended co-text of the message, so the user has to find it in order to understand the message, e.g. very often a question is not directly followed by the intended answer. In table 1 the message by the user Yellow is obviously not the answer to the message before it, so the user has to read backwards in order to find the best match. According to Pimentel et al. [Pimentel et al., 2003], users which are looking for the co-text are distracted and cannot follow the "conversation rhythm". Pimentel et al. [Pimentel et al., 2003] distinguish four different actions once a user detects co-text loss:

1. The chat participant searches for the co-text in the previous messages. If the co-text is quickly identified the conversation continues.
2. If the chat participant cannot quickly identify the co-text and continues the search, this takes time and causes the loss of "conversation rhythm" while the conversation continues.
3. If the chat participant stops searching and does not declare the co-text-loss, the participant might not understand the conversation anymore.
4. If the participant states co-text loss another participant may step in and help to understand the conversation. When the help of this participant is successful, the other participant may declare his/her understanding of the conversation.

Especially, when a lot of users participate in a chat, these users need a high degree of concentration and a high reaction rate, to disentangle the different conversation threads, avoid co-text loss, and to follow the conversation. Identifying the co-text loss in such a setting gets even more complicated when there is a huge time gap between the messages which belong together. When it is unclear which messages belong together, co-text loss can lead to ambiguities. Chat messages which simply consist of an answer like "yes" or "no" even increase the possibility of co-text loss.

Co-text loss is often also considered as “intention confusion” [Vronay et al., 1999] when the recipient of a chat message is unclear or as “thread confusion” when it is unclear to which conversation thread a message belongs. Herring [Herring, 1999] considers co-text loss as “interactional coherence” while Pimentel et al. also call it “chat confusion” [Fuks et al., 2006]. All agree that co-text loss occurs when it is not possible to identify the message to which another refers. A chat system should therefore help users to identify the messages which belong together and thereby allow users to follow the conversation without co-text loss. For that purpose, users must be able to reference another chat message when posting a new chat message. This leads to the following requirement:

(R1) A chat system has to support users to reference a chat message when posting a new chat message.

The chat messages and the references between chat messages have to be visualized so that users can easily identify the messages which belong together. The visualization has the main goal to reduce to probability of co-text loss. The visualization has to take care that the relation between a chat message and the user which has posted the chat message is still obvious (cf. P1). Users should be able to choose between different visualization forms so that they can select the visualization form which suits them best in a certain context (discussing with one person vs. in a big group or arguing vs. brainstorming). This requirement is supported by a study who found that users seemed to like the traditional chat interface more than the more efficient threaded chat interface [Smith et al., 2000]. Users liked to review the discussion in the threaded visualization, but were more convenient with the linear interface while chatting. The authors of the study mentioned as future work alternative views which should highlight other dimensions like the temporal order. By applying different visualizations in different contexts user studies can show which visualization overcomes the problem of co-text less most efficiently. This leads to the following requirement a chat system has to address when tackling the co-text loss problem:

(R2) A chat system has to offer multiple visualization forms for displaying chat messages and the references between chat messages.

In some cases, co-text loss might occur even when the requirement R1 and R2 are met by a chat system. To resolve the co-text loss subsequently to a chat conversation, users must be able to store the conversation and to replay the conversation. When replaying a conversation users should as well have the possibility to choose between the different available visualization forms. By offering such a replay mode, the chat system also addresses the problem of unfeasible chat protocols (cf. P5). For storing a chat conversation, a chat system should use a format which can also be read by the users and which can easily be used to import chat conversations which have been stored otherwise. This broadens the usability of the chat system as it then can be used to comprehend other chat conversations. Summing up, this leads to the following requirement.

(R3) A chat system has to allow users to store as well as replay chat conversations.

3 Related Work

In order to improve the interface of chat systems a lot of tools have been developed but most of them neglected the problem of co-text loss. In the following we will only mention those approaches which have addressed the requirements (R1)-(R3). ThreadedChat [Smith et al., 2000] is an experimental version of a tree-based chat tool, which was used in a small group decision task with 3-4 people for research purposes. HyperDialog [Pimentel et al., 2003, Fuks et al., 2006] was also developed for research purposes and was used in groups up to eleven people. ThreadChat [Holmer and Wessner, 2004] was part of learning environment and intended to be used for small groups only. All of these tool provide referencing functionality and visualize the chat transcript as an indented tree. New contributions are integrated into the tree structure and can appear in different areas of the screen especially when participants are discussing in parallel threads. The structure is clear and the co-text of a message is easily identifiable but in the process of communication some problems are likely to occur: if a message is not referenced by accident, it will be sorted into a new thread and messages can be missed because new messages appear in areas which are not visible on the screen. Parallel threads are shown as two indented lists which results in an increasing gap between the messages. This visualization needs more screen space and makes it difficult to follow more than one thread at a time. In addition, it is difficult to identify messages, which are produced at the same time, because they may appear at totally different parts even if they belong to the same thread.

These problems are tackled in systems like KOLUMBUS [Holmer et al., 2006] and ConcertChat [Mühlpfordt and Wessner, 2005] by showing the transcript in linear order but providing a reference indicator at each message which shows a pointer to the co-text when selected. Thereby it is clear where new messages will appear and how the co-text of a message could be inferred. The drawback of these approaches is that they do not provide means for getting a visual overview of the discussion and its structure. It is awkward to find out how many parallel threads exist or how many branches are in a thread.

The system factChat [Harnoncourt et al., 2005] allows users to put the messages on a two-dimensional surface in proximity to their co-text. In addition it is possible to create references to messages which are not visible anymore on the screen (older messages fade out and seem to disappear in the background). Although is possible to review the transcript by using a timeline it is not possible to get a visual representation of the entire structure.

In summary there are many different approaches which address partially some problems related to co-text loss but none of them addresses all of the identified requirements, esp. multiple visualizations (R2).

4 Approach

Our approach aims at diminishing co-text loss in text-based chats. Therefore, our tool MuViChat has in the first case to offer standard text chat functionality. Then, MuViChat has to offer functionality to include references between different chat messages (R1) and support multiple visualizations of these references (R2). Finally, in

order to resolve co-text loss once a discussion is over, MuViChat has to support a replay of the text-based communication (R3). In the following, we will present MuViChat and how it addresses the identified requirements. MuViChat is based on XMPP and therefore can be used with huge variety of different chat servers already supporting this standard.

4.1 Referencing chat messages (R1)

At the content level, a chat message can refer to none previous messages, exactly one, or even multiple ones. A chat message does not refer to another chat message if the author starts a completely new topic. When a chat message references exactly one message, it is often an answer or comment to a previously posted question. Multiple references may exist, when a user, e.g., summarizes previous chat messages. The latter occurs quite rarely and would make the visualization and the comprehension of the visualized references more difficult. ConcertChat [Mühlpfordt and Wessner, 2005] is a chat system which allows users to refer to multiple messages but up to now there are no reported experiences if and how users used this feature. Like other chat systems that offer referencing functionality [Smith et al., 2000, Holmer and Wessner, 2004, Holmer et al., 2006, Harnoncourt et al., 2005], we support in MuViChat references between two different chat messages only.

Apart from the number of messages which can be referenced, it is also important to decide whether users can post new messages without referencing another chat message. The experiences with the chat system HyperDialog show [Pimentel et al., 2003, Fuks et al., 2006] that especially missing and wrong references are the reason for not being able to overcome the co-text loss by referencing functionality. Additionally, it confines the conversation possibilities when forcing users to reference other chat messages. Experiences with the chat system KOLUMBUS [Holmer et al., 2006] have shown that referencing is used if the message cannot be placed directly after the referred message. Users are willing to use the functionality if the value is directly visible. In most of our provided visualizations the benefit of referencing is very clear and thereby should lead to increased usage of the referencing functionality.

Due to this, we do not force users in MuViChat to reference another chat message. Instead, we allow users to reference a chat message before or while creating a new chat message by selecting the referenced message from the displayed chat messages.

4.2 Multiple visualization forms for chat messages and the references between chat messages (R2)

We designed MuViChat so that users can select between different forms of visualizing chat messages as well as the references between chat messages. This allows users independently from each other to choose their favourite visualization alternative. Furthermore, all visualization alternatives offer further configuration possibilities to tailor the visualization even more to the users' preferences. There are many possibilities to visualize connected objects and it is still a research topic which visualization alternatives are the most effective ones for specific purposes. We decided not to implement a fixed set of alternatives but to provide a framework for developing and integrating visualizations into MuViChat. Thereby we can use

MuViChat as a platform for testing different visualization alternatives in the same environment. The following list summarizes our current visualization alternatives:

1. Classical visualization
2. List view with highlighting
3. Simple tree view
4. Tree view with highlighting time by fading colours
5. Tree view with highlighting time by layout
6. Sequential tree view

Of course, this list is not comprehensive. Other visualization alternatives are could be e.g. mindmap-like structures or visualizations that make use of three dimensions. But in the beginning we concentrate on the classical forms of message and thread presentation modes in order to investigate their usefulness and to be able to test them against each other. In the following section, we will describe the different visualization forms in more detail. At the end of the section, we will then discuss the different visualization forms in general.

4.2.1 Classical visualization

Our classical visualization alternative corresponds to the visualization of all major chat clients in which chat messages are displayed in a chronologically ordered list. We choose to integrate this variant for future experiments and evaluation because this variant allows us to compare the new forms of visualization with the standard case. Nevertheless, users can configure the font size, the font colour, and the background colour. For group awareness purposes, we integrated a USER LIST [Schümmer and Lukosch, 2007] which list the users who are currently participating in the chat. This USER LIST is available in all visualization forms (cf. Figure 1).

4.2.2 List view with highlighting

The list view with highlighting represents a first variant of the classical view. Here, chat messages are displayed in a list and are chronologically ordered. The newest chat message is highlighted in bold font. If this message references another message, this message is displayed in bold font as well. If this message references another one, this also highlighted etc. By this all referenced messages are highlighted on a path from the most actual message to the root message, Figure 1 illustrates this visualization form. The list view with highlighting is a modified visualization form of the reference indicator approach in KOLUMBUS [Holmer et al., 2006] and ConcertChat [Mühlpfordt and Wessner, 2005]. However, compared to these systems MuViChat highlights the whole path instead of the first direct reference only.

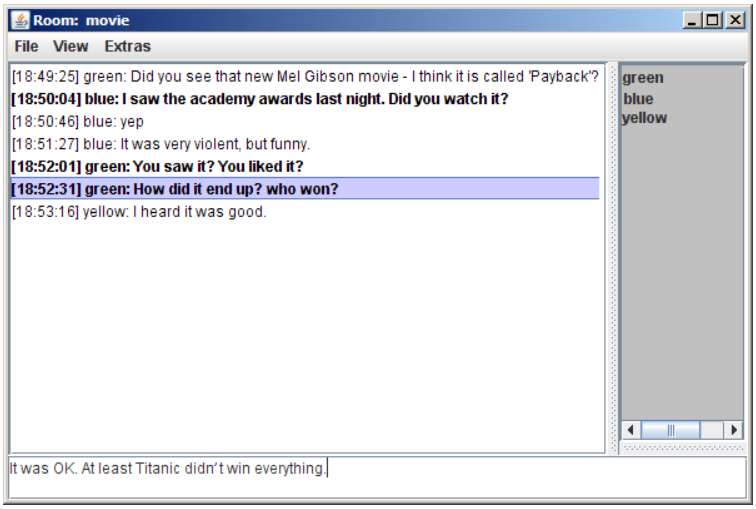

Figure 1: List view with highlighting

4.2.3 Tree-based visualizations

Apart from the two textual visualization alternatives our chat systems offers four different tree-based visualizations. We have tried to resolve the disadvantages of current tree views. For that purpose, all of our tree-based visualizations have in common that

- chat messages are shown as coloured rectangles,
- references are shown as lines between these rectangles,
- each user is represented by a different colour,
- chat messages can be selected by clicking on them with the left mouse button,
- selected chat messages are highlighted, and
- the size of the chat window can be changed arbitrarily.

Users can choose whether the view is automatically focussed on the most recent chat message. As the tree visualizations require more visualization space than the list-based alternatives, we included a bird's view which shows the complete chat conversation and highlights the part that is currently shown in the chat window.

Simple tree view

The simple tree view ignores the time factor and simply orders the chat messages which reference each other in a tree (cf. Figure 2). This visualization has compared to typical indented trees some advantages. The direction of the tree can be rotated in four ways (left-right, right-left, top-down, bottom-up), whereas especially the top-down view has the advantage against the typical left-right view of being more chat-like than newsgroup-like. In most chat systems, new messages appear at the bottom of the screen and users expect this to happen in other visualizations, too. Most other tree-

view chat-systems (e.g. [Smith et al., 2000] [Pimentel et al., 2003]) also provide single line text only, which results sometimes in messages, which have to be horizontally scrolled in order to read them. With the configurable multi-line text each message could be read in a glance which supports reading and getting a quicker overview.

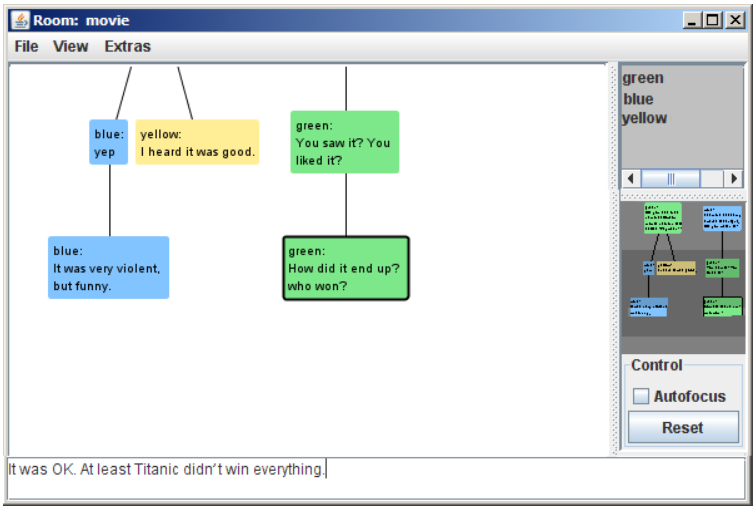

Figure 2: Simple tree view

Tree view with highlighting time by fading colours

This visualization alternative considers the time at which a message was created by fading the colour of the rectangle for older messages while new messages are displayed using a bright colour (cf. Figure 3). With this feature we address the problem of missing the newest messages especially if they occur in different parts of the screen. It is configurable how long it takes for the colour to fade out, so the user can set his according to his information needs. One drawback of this view is that older messages all have the same colour and it is more difficult to identify older messages from a specific author. Thereby it is most useful for the observation of active chats and replays but not for scanning the whole transcript as a document.

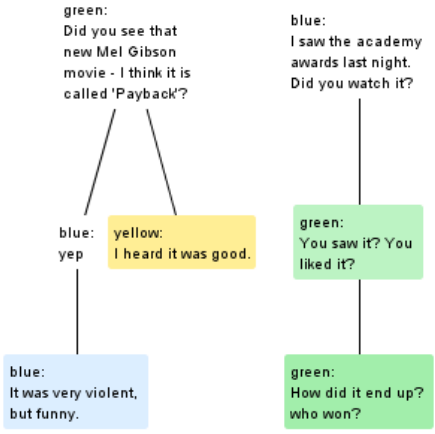

Figure 3: Tree view with highlighting time fading colours

Tree view with highlighting time by layout

Smith et al., 2000 mentioned as part of their future work, that temporal order can also be a dimension of conversation which could be used for visualization. Our variant increases the distance between two messages the more time has passed between them. In this visualization alternative, the y-axis corresponds to the time (cf. Figure 4). This view is especially useful to identify nearly simultaneous messages as well as pauses in the communication flow.

In Figure 4 is a relatively long break in the middle of the conversation followed by a burst of activity and some nearly simultaneous messages. In larger conversations this can help latecomers or evaluators to identify the flow of conversation and to get an insight in the dynamics of the ongoing discourse.

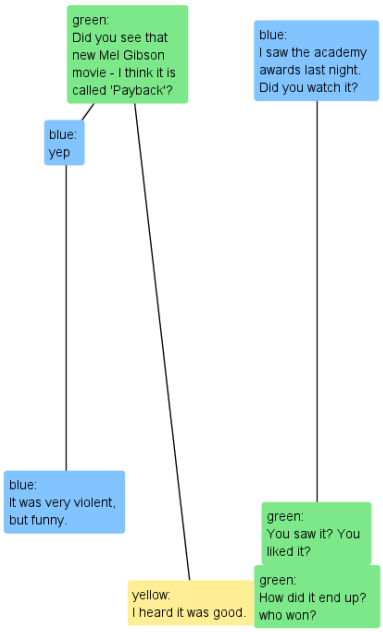

Figure 4: Tree view with highlighting time by layout

Sequential tree view

When linearly mapping the time to the y-axis, a lot of visualization space can be wasted if there are long breaks (minutes to hours) in the communication. On the other hand, if messages are produced by many users at the same time, the distance between them is very small and visualization could be too dense. This effect can be seen in Figure 4 by the gap between the messages in the middle and very small distance between the messages at the bottom. The sequential tree view which has been used effectively for reading and analyzing chat logs [Holmer, 2008] considers this observation and uses the y-axis to visualize the order of the messages. The distance between two messages can be configured by the user (cf. Figure 5).

Figure 5: Sequential tree view

This visualization tries to satisfy the combination of needs for getting a quick impression of the structure, representing aspects of the chronological order and provide a readable and compact representation. Thereby, we minimized the trade-off between co-text distance and support for communication. As another advantage of this visualization, it is simpler to detect messages which are erroneously not referencing another message. In the simple tree view, a message without reference creates a new thread and will be placed at the end of the tree starting positions. This message will have a big distance to the intended message and the co-text is very difficult to find. In contrast, the sequential order creates nearness by time and the chance is higher to be placed more near to the intended message even if the reference is missing.

4.2.4 Discussion

Providing different visualizations gives users the flexibility to explore them and to identify the one which suits them best for a certain task. We expect that the simple tree view is especially useful in situations where users want to see the structure of the discussion and to look for, e.g., the most complex threads or the longest linear sequence. If the discussion is still ongoing, users can see the most actual messages and current threads by visualizing time by lighter colours (cf. Figure 3). If users want to look at parts of the discussion in which the frequency of interaction has been very intense or in which have been pauses in the discussion, they can use the view with highlighting time by layout (cf. Figure 4). This allows users to understand the dynamics of the ongoing discussion. In our opinion, the sequential tree view (cf. Figure 5) is the most convenient one for actively participating in discussions because

it shows the most actual messages at the bottom of the screen and makes it easier to follow multiple parallel discussions.

The list view with highlighting (cf. Figure 1) can be used as "power user mode" because it has the same look & feel as traditional chat interfaces but users can reference and can see referenced messages inside their classical list view. This mode is very similar to other tools like ConcertChat [Mühlpfordt and Wessner, 2005] and KOLUMBUS [Holmer et al., 2006] and allows direct comparisons with other visualizations instead of comparing the tools.

4.3 Persistence and Replay of Chat Conversations (R3)

To reconsider chat conversations, users are able to store chat conservations in the XML-based format TreeML[1]. Thus, a conversation stored by our chat system can be read by users and as TreeML is based on XML with a publicly available DTD it is also simple to implement import filters for other formats. Once a conversation has been stored in this format, it can be imported via our chat system and replayed. Thereby, we implement the REPLAY pattern [Schümmer and Lukosch, 2007]. A chat conversation replay does not need any input fields or a user list. Instead the user interface for the replay contains a control panel for the replay (cf. Figure 6). In this control panel, users have the possibility

- to start the replay,
- to pause the replay,
- to stop the replay, and
- to vary the speed of replay between real time and a faster replay.

Figure 6: Replay control panel

The above functionality further allows users to process stored chat conversations with other software programs. ChatLine [Holmer, 2008] is able to interpret the data structures and apply the discourse structure analysis method in order to compute measures like thread structures, multitasking behavior, participation rates and social network structures. Thereby, chat discussions can be evaluated with quantitative criteria. Such an evaluation can give interesting hints about differences between chat conversations. A group can use this to understand their communication behavior and look for improvements, e.g. is the communication equally distributed or are all participants responding to each other?

The other way around chatLine can convert chat conversations from other chat tools into TreeML. Such transformed conversations can then be replayed within MuViChat. Even when references are not present, users can get an impression how

[1] TreeML is an XML-based format that can be used to store tree-based structures. It was developed for the IEEE InfoVis Contest 2003. The document type definition is available at: http://www.nomencurator.org/InfoVis2003/download/treeml.dtd

their conversation would looked like when using MuViChat. Again, this can lead to important insights about the visualization demands of different groups and improve the understanding of a chat conversation.

5 First Experiences

We split up the evaluation of MuViChat in three phases. In the first phase, we have conducted functional tests. These functional tests concentrated on validating the intended interaction possibilities and checking if our requirements are met. These tests showed that all three requirements are fulfilled.

In order to gain experience with the new interface and the supported interaction we conducted, we evaluated MuViChat in the second phase as follows:

1. Pilot study based on online discussions
2. Pilot study based on the replay functionality
3. Expert interviews

The following sections report on our experiences in the second evaluation phase and summarize the feedback which we have got for MuViChat. We will use the insights from this second evaluation phase to improve the design of MuViChat and add additional functionality. Thereby, we employ an iterative and end-user centred development process. This process is inspired by the *Oregon Software Development Process (OSDP)* [Schümmer et al., 2006] for groupware. The improved MuViChat will then be used in the third evaluation phase. For the third evaluation phase, we plan to employ MuViChat in a broader setting that will allow us to perform qualitative evaluation of the MuViChat functionality.

5.1 Pilot study based on online discussions

For the pilot study based on online discussions, we conducted informal experiments with students at the university campus in Hagenberg, Austria. By using MuViChat in 90-minute discussions with 18 active participants we could show that the concept worked for this group size (see Figure 7). The referencing concept was immediately understood and used correctly. Participants tried the different visualizations and developed their own strategy to use them for their purpose. We observed the following user behaviours: In order to browse the discussion, participants used the simple tree visualization (cf. Figure 2) or the tree variant shown in Figure 3a. In this visualization the distance between references is the smallest and therefore the understanding of the conversation is simplified the most. For participating actively in the conversation and following ongoing threads the participants used the sequential tree (cf. Figure 5). Current messages appear at the bottom of the screen and make it easier to follow the discussion in one thread. These preliminary results show that participants embraced the different visualizations and used them for their own purposes. They could also imagine using the tool for learning and working e.g. structured brainstorming sessions with given topics, language learning, planning meetings and distribution of tasks. In order to create a discussion protocol they demanded an “export to mindmap” feature which should be able to export selected trees. In general the students would recommend using the tool in smaller groups from

six to ten participants and the open new rooms or channels if the group size gets bigger.

In these first trials, we also identified some problems which have to be fixed: the user colour palette is different for each user so that user A is may red for user B but green for user C. This sometimes caused irritations and can be fixed by representing the user colour in the USER LIST [Schümmer and Lukosch, 2007] which can also present additional awareness information. One feature of this USER LIST could be to disable or highlight the messages of specific users in order to focus on specific topics. The users also proposed that the autofocus function which jumps to the newest message should be temporarily disabled when typing a message. They also proposed to think more about an intelligent autofocus which ignores messages in threads which does not interest the current user (except new threads).

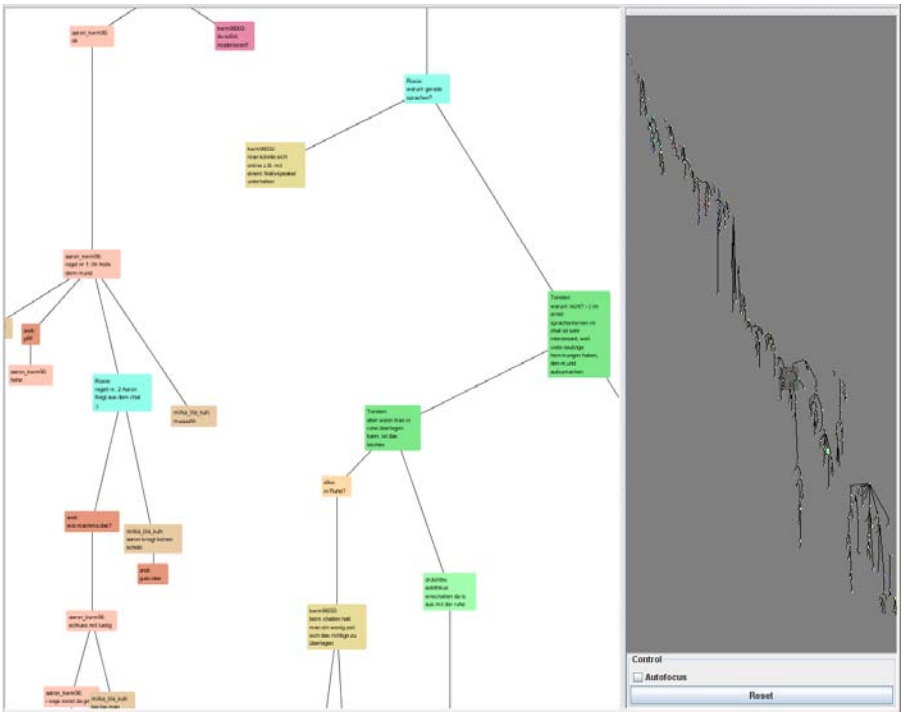

Figure 7: Tree view with highlighting time by layout of a chat transcript with 18 participants

5.2 Pilot study based on the replay functionality

In a further pilot study, we used the replay function for testing co-text loss while reading chat transcripts. In a 30-minute session the same transcript was replayed to groups of three individuals which had to read them on the screen and answer questions in between regarding references between messages. Correctness of answers was recorded by a questionnaire and the speed of response was observed directly.

While with the tree visualization all three participants could answers all questions correctly, using the classical list visualization participants in two out of nine cases had problems to identify the references and needed more time to identify them.

Although the identification of references seems to be better supported in tree visualizations the pilot study showed some drawbacks. Similar to the study of Smith et al. [Smith et al., 2000] the participants sometimes ignored messages because they appeared in different locations or the auto-focus function focused the view area on newest messages faster than participants were able to read the current message. These problems will be addressed in future studies and should be addressed in the next extensions of MuViChat.

5.3 Expert interviews

In order to gain further experiences about the usability of MuViChat, we demonstrated the use of MuViChat to about 30 experts in the domain of synchronous textual communication during an international research conference. Afterwards some experts tested MuViChat during a short informal discussion.

After demonstrating the basic functionality of MuViChat and letting the experts use the system, we conducted unstructured interviews to collect the experts' feedback on MuViChat. In summary, the feedback was positive. The experts especially appreciated the various visualization options and the possibility to switch between the various options at runtime. The replay feature was also highly appreciated, as it allowed them to review former discussions and resolve misunderstandings. However, the experts also suggested various ideas for improving the functionality of MuViChat. The following list summarizes the major recommendations:

- An important topic which is often neglected in chat tools is group or workspace awareness. Workspace awareness will give users an understanding of their activities as well other users' activities in their current work context [Dourish and Bellotti, 1992, Gutwin et al., 1996]. In the current version, MuViChat only offers a USER LIST [Schümmer and Lukosch, 2007] for providing awareness. Experts suggested offering more functionality for workspace awareness. They suggested making users aware of what the typing activity of other users before their messages are posted into the chat transcript (cf. P3). Regarding the graphical structure of MuviChat we plan to integrate an ACTIVITY INDICATOR [Schümmer and Lukosch, 2007] which appears when a user is referencing a message and thereby shows that there will be a reply to that message in the future. This helps to get an overview about the concurring activities in the chat room.
- In the current implementation messages which were not referenced by accident could not be linked afterwards. In some cases, the experts would have liked to include such links afterwards to better structure the discussion and to explicitly address co-text loss once it has occurred. In order to give better support for structuring this feature, we will implement this feature in a future release of MuViChat.
- Up to now the spatial layout of threads is the same for all participants although threads in which a user is actively participating should be more important than other threads. The experts recommended bringing active threads to the centre of the screen and let older threads disappear into the

background. Such kind of personalized views could help to stay focused on the subjective most interesting parts of a chat discussion. Additionally, we intend to integrate a CHANGE INDICATOR [Schümmer and Lukosch, 2007] which highlights the most recent changes within the bird view of the tree visualizations. Such a CHANGE INDICATOR will also improve the group awareness.

- When chat contributions get rather long, the experts experienced difficulties in keeping the overview when using a tree view. This effect was mainly caused by the nodes which got to big when displaying the whole text of the contribution. Some experts suggested using a fixed size for each node in the tree and only displaying the full text on a mouse hover. Another suggestion was to display the classical view in parallel to the tree view. Thereby, the tree view would show the references whereas the classical view could be used to read and follow the discussion.

6 Next Steps

The above sections show that MuViChat effectively can be used to address co-text loss in synchronous text-based discussions. However, our experiences also show that there are still various areas where MuViChat can be improved. We plan to tackle most of the above issues and integrate our ideas in the next release of MuViChat. Then, we will start the third evaluation phase to test our ideas in broader user studies. Such studies will allow us to conduct qualitative evaluations of the offered functionality and reveal the role of our approach in referenced chat communication. Currently, we are setting up a lab in order to investigate the effects of synchronous communication and cooperation. We will use this lab to evaluate the various ideas for improving MuViChat. Additionally, MuViChat will be one of the first tools we will use for studying effects of group size and awareness support in text-based synchronous communication. We will use eye-tracking systems and screen recording software in order to analyze crucial aspects of the different visualization alternatives and other interface features. In which way users are scanning the screen and look for the co-text of messages? How long does it take and what are the actions of people when the co-text is difficult to find? How do people read complex graphical thread structures and which strategies do they use in order to follow multiple discussions? Which role do the enhanced awareness functions play when people are actively engaged in conversations? In order to answer these questions we will carry out single-user studies with the replay functionality as well as task-based group experiments.

7 Conclusions

MuViChat is both an innovative chat tool and an environment for experimenting and evaluating new variants of chat visualizations. By using widespread standards (Java and Jabber) MuViChat can be used on every java-capable computer with internet connection. Storing the log files in XML-format is useful for visualizing them in many different ways as well as using them for visualizations which we will be integrated in the future. Thereby it is possible to make controlled experiments

between different visualizations with least effort in order to check their usefulness and readability. Because of the open source license and the modular architecture of MuViChat interested researchers can develop their own visualizations and integrate them in order to share their ideas.

In future, we will work on improving MuViChat and add some of the functionality suggested by the users in the pilot studies as well as the experts. This new version of MuViChat will then be subject to broader user studies in our third evaluation phase. Furthermore, we will use MuViChat for conducting experiments regarding the usefulness of different visualizations for reading chat transcripts as well as for supporting different kinds of discussions. These experiments will clarify under which conditions the provided functions are useful, in which way successful discussions can profit from the functionality and which new visualizations have to be developed to overcome the identified deficits and dangers of co-text loss.

References

[Dourish and Bellotti, 1992] Dourish, P. and Bellotti, V. (1992). Awareness and coordination in shared workspaces. pages 107–114, Toronto, Canada.

[Fuks et al., 2006] Fuks, H., Pimentel, M., and de Lucena, C. J. P. (2006). R-u-typing-2-me? evolving a chat tool to increase understanding in learning activities. *International Journal of Computer-Supported Collaborative Learning*, 1(1):117–142.

[Garcia and Jacobs, 1999] Garcia, A. and Jacobs, J. (1999). The eyes of the beholder: understanding the turn taking system in quasi-synchronous computer mediated communication. *Research on Language and Social Interaction*, 32(4):337–367.

[Geyer et al., 2008] Geyer, W., Filho, S. R. S., Brownholtz, B., and Redmiles, D. F. (2008). The trade-offs of blending synchronous and asynchronous communication services to support contextual collaboration. *Journal of Universal Computer Science*, 14(1):4–26.

[Gutwin et al., 1996] Gutwin, C., Greenberg, S., and Roseman, M. (1996). Workspace awareness in real-time distributed groupware: Framework, widgets, and evaluation. In Sasse, M., Cunningham, R., and Winder, R., editors, *People and Computers XI (Proceedings of the HCI'96)*, pages 281–298, Imperial College, London, UK. Springer-Verlag.

[Harnoncourt et al., 2005] Harnoncourt, M., Holzhauser, A., Seethaler, U., and Meinl, P. (2005). Referenzierbarkeit als Schlüssel zum effizienten Chat. In Beißwenger, M. and Storrer, A., editors, *Chat-Kommunikation in Beruf, Bildung und Medien: Konzepte - Werkzeuge - Anwendungsfelder*, pages 161–179.

[Herring, 1999] Herring, S. (1999). Interactional coherence in CMC. *Journal of Computer-Mediated Communication*, 4(4).

[Holmer, 2008] Holmer, T. (2008). Discourse structure analysis of chat communication. *Language@Internet*, 5.

[Holmer et al., 2006] Holmer, T., Kienle, A., and Wessner, M. (2006). Explicit referencing in learning chats: Needs and acceptance. In Nejdl, W. and Tochtermann, K., editors, *Innovative Approaches for Learning and Knowledge Sharing*, pages 170–184. Springer Berlin.

[Holmer and Wessner, 2004] Holmer, T. and Wessner, M. (2004). Tools for cooperative learning in l^3. In Ehlers, U., Gerteis, W., Holmer, T., and Jung, H., editors, *E-Learning*

Services in the Crossfire. Pedagogy, Economy and Technology, pages 138–152. Bertelsmann, Bielefeld.

[Kuo et al., 2001] Kuo, C. H., Wible, D., and Chou, C. L. (2001). A synchronous efl writing environment for the internet. *Journal of Universal Computer Science*, 7(3):240–253.

[Liu et al., 2001] Liu, Y., Ginther, D., and Zelhart, P. (2001). How do frequency and duration of messaging affect impression development in computer-mediated communication? *Journal of Universal Computer Science*, 7(10):893–914.

[Mühlpfordt and Wessner, 2005] Mühlpfordt, M. and Wessner, M. (2005). Explicit referencing in chat supports collaborative learning. In Koschmann, T., Suthers, D. D., and Chan, T.-W., editors, *Computer Supported Collaborative Learning 2005: The Next 10 Years!*, pages 460–469. Lawrence Erlbaum Associates.

[Münzer and Xiao, 2005] Münzer, S. and Xiao, B. (2005). Small groups learning synchronously online at the workplace: The interaction of factors determining outcome and acceptance. *Journal of Universal Computer Science*, 11(3):378–393.

[Pimentel et al., 2003] Pimentel, M. G., Fuks, H., and de Lucena, C. J. P. (2003). Co-text loss in textual chat tools. In *Proceedings of Fourth International and Interdisciplinary Conference on Modeling and Using Context (CONTEXT 2003)*, LNAI 2680, pages 483–490. Springer-Verlag Berlin Heidelberg.

[Schümmer and Lukosch, 2007] Schümmer, T. and Lukosch, S. (2007). *Patterns for Computer-Mediated Interaction*. John Wiley & Sons, Ltd.

[Schümmer et al., 2006] Schümmer, T., Lukosch, S., and Slagter, R. (2006). Using patterns to empower end-users – The Oregon software development process for groupware. *International Journal of Cooperative Information Systems, Special Issue on '11th International Workshop on Groupware (CRIWG'05)'*, 15(2):259–288.

[Smith et al., 2000] Smith, M., Cadiz, J. J., and Burkhalter, B. (2000). Conversation trees and threaded chats. In *Proceedings of the 2000 ACM Conference on Computer supported cooperative work*, pages 97–105. ACM Press, New York, NY, USA.

[Vronay et al., 1999] Vronay, D., Smith, M., and Drucker, S. (1999). Alternative interfaces for chat. In *Proceedings of the 12th annual ACM symposium on User interface software and technology*, pages 19–26. ACM Press, New York, NY, USA.

[Werry, 1996] Werry, C. (1996). Linguistic and interactional features of internet relay chat. In Herring, S., editor, *Computer-Mediated Communication: Linguistic, Social and Cross-Cultural Perspectives*, pages 47–63. John Benjamins, Amsterdam.