A generic layout-tool for summaries of meetings in a constraint-based approach

Sandro Castronovo, Jochen Frey, and Peter Poller

Deutsches **F**orschungszentrum für **K**ünstliche **I**ntelligenz GmbH
Stuhlsatzenhausweg 3, 66123 Saarbrücken, Germany,
`sandro.castronovo[at]dfki[dot]de`
`jochen.frey[at]itemis[dot]de`
`peter.poller[at]dfki[dot]de`

Abstract. We present SuVi - Summary Visualizer -, a generic layout-tool which displays a multimodal summary of a meeting in either a story-board style or newspaper-style. The system relies on constraint solving techniques in such a way that the two layouts have been extensively modeled in a series of constraints representing the underlying design knowledge. While the story-board aims to give the reader an overview of the chronological sequence of the meeting, the newspaper-layout focuses on presenting the topics of a meeting depending on their relevance. We also show two methods for connecting the whole AMI meeting corpus as a large input resource for the story-board part of SuVi and present a first end-to-end implementation of our system.

1 Introduction

One of the main goals of the AMIDA project[1] is the automatic generation of multimodal meeting summaries. Apart from their generation, there are many ways of presenting these summaries to the user. A very appealing presentation style has been realized by the SuVi-tool (Summary Visualizer). Based on actual audio and video data it generates a story-board layout or displays the meeting in the style of a newspaper.

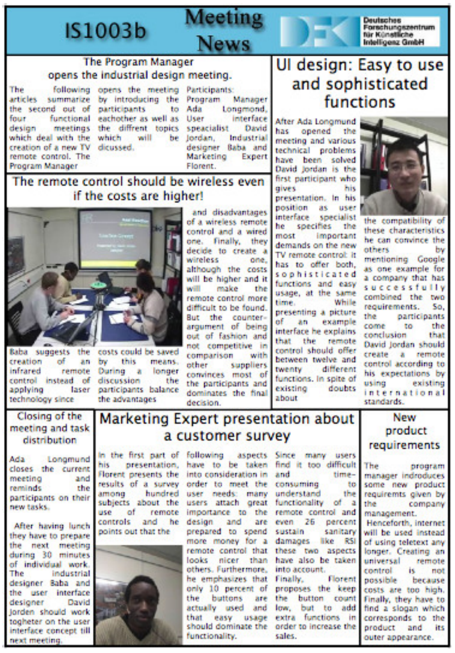
The major task of SuVi was the modeling of the necessary design knowledge for the layout process of the respective output presentation style. Consequently, we implemented SuVi based on hand-generated input data for just a single meeting. The primary goal of the development was to show the general feasibility of a constraint-based layout approach. Later on, in order to greatly extend the number of available input resources, we developed M2SuVi, a generic interface to the whole AMI meeting corpus. In the compound system it is even possible to implement different ways of filling the story-board layout with content very quickly giving us the flexibility to adapt the system to completely new domains in a very short amount of time.

[1] Augmented Multiparty Interaction with Distance Access (AMIDA) is an Integrated Project funded by the ECs 6th Framework Program FP6-0033812, Publication ID - AMIDA-26, jointly managed by IDIAP (CH) and the University of Edinburgh (UK).

This paper presents first the SuVi system in section 2 and the layout constraints used for the story-board part of SuVi in section 3. We elaborate the generic interface which makes the whole AMI meeting corpus accessible for SuVi in section 4 and show the robustness of our implementation by a batch-run over the whole AMI corpus in section 5. Our first online end-to-end implementation of the compound system which was configured to generate a story-board layout of a selected part of the first AMIDA review meeting is presented in section 6. Finally, section 7 lists related work and gives an overview of the future work in this context.

2 SuVi

SuVi - Summary Visualizer is a constraint based layout system for the automatic visualization of multimodal meeting summaries either in a multimedia story-board style or in the style of a newspaper. While the newspaper component focuses on the hierarchical topic presentation, e.g., showing more relevant topics more prominently on the page, the story-board layout aims to represent the chronological sequence of a meeting. Figure 1 depicts the resulting layout for the newspaper style and an example of the story-board layout is shown in figure 4. In this section we give a general introduction to our constraint-based layout approach and explain the story-board layout component of SuVi.



IS1003b Meeting News

Deutsches Forschungszentrum für Künstliche Intelligenz GmbH

The Program Manager opens the industrial design meeting.

The following articles summarize the second out of four functional design meetings which deal with the creation of a new TV remote control. The Program Manager opens the meeting by introducing the participants to eachother as well as the diffrent topics which will be dicussed.

Participants: Program Manager Ada Longmond, User interface speacialist David Jordan, Industrial designer Baba and Marketing Expert Florent.

The remote control should be wireless even if the costs are higher!

Baba suggests the creation of an infrared remote control instead of applying laser technology since costs could be saved by this means. During a longer discussion the participants balance the advantages and disadvantages of a wireless remote control and a wired one. Finally, they decide to create a wireless one, although the costs will be higher and it will make the remote control more difficult to be found. But the counter-argument of being out of fashion and not competitive in comparison with other suppliers convinces most of the participants and dominates the final decision.

UI design: Easy to use and sophisticated functions

After Ada Longmund has opened the meeting and various technical problems have been solved David Jordan is the first participant who gives his presentation. In his position as user interface specialist he specifies the most important demands on the new TV remote control: it has to offer both, sophisticated functions and easy usage, at the same time. While presenting a picture of an example interface he explains that the remote control should offer between twelve and twenty different functions. In spite of existing doubts about the compatibility of these characteristics he can convince the others by mentioning Google as one example for a company that has successfully combined the two requirements. So, the participants come to the conclusion that David Jordan should create a remote control according to his expectations by using existing international standards.

Closing of the meeting and task distribution

Ada Longmund closes the current meeting and reminds the participants on their new tasks.

After having lunch they have to prepare the next meeting during 30 minutes of individual work. The industrial designer Baba and the user interface designer David Jorden should work togheter on the user interface concept till next meeting.

Marketing Expert presentation about a customer survey

In the first part of his presentation, Florent presents the results of a survey among hundred subjects about the use of remote controls and he points out that the following aspects have to be taken into consideration in order to meet the user needs: many users attach great importance to the design and are prepared to spend more money for a remote control that looks nicer than others. Furthermore, he emphasizes that only 10 percent of the buttons are actually used and that easy usage should dominate the functionality. Since many users find it too difficult and time-consuming to understand the functionality of a remote control and even 26 percent sustain sanitary damages like RSI these two aspects have also be taken into account. Finally, Florent proposes the keep the button count low, but to add extra functions in order to increase the sales.

New product requirements

The program manager indroduces some new product requiremts given by the company management.

Henceforth, internet will be used instead of using teletext any longer. Creating an universal remote control is not possible because costs are too high. Finally, they have to find a slogan which corresponds to the product and its outer appearance.

Fig. 1. Example of a newspaper layout

2.1 Architecture of SuVi

From a functional point of view, the constraint-based layout-system SuVi is a self-contained, generic and parameterizable layout generator for multimodal meeting summaries. It realizes a broad spectrum of functionalities:

- full automatic layout generation of meeting summaries
- API for different layout-specifications
- user adaptivity by offering a variety of parameters for layout generation

Figure 2 shows the architecture of SuVi from the story-board point of view. SuVi consists of three main components which are coordinated by a central distributor component. The central architecture is the same as for the newspaper layout. The major difference is the constraint based modeling of the target domain, the different layout objects and the varying layout manager.

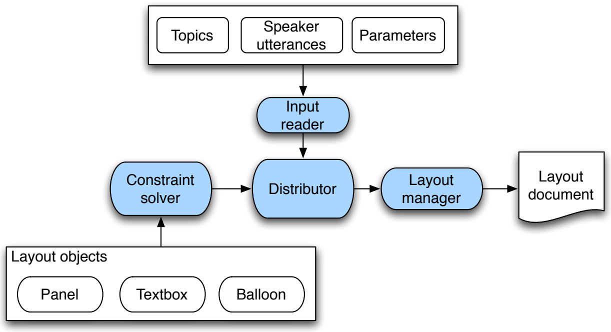


Fig. 2. Architecture of SuVi

The input data consists of meeting topics, speaker utterances and parameters for the layout generation. The user can exert an influence on the topics used in the layout and the parameters of the generation, e.g., the number of panels on a page. Speaker utterances are used to fill the layout objects, namely text-boxes and balloons. All of this data is analyzed and appropriately represented by the **input reader**. Based on that representation, the **constraint solver** derives the necessary constraint variables which are stored within corresponding layout objects. These are are shown in the resulting layout in figure 4. The layout objects that are used in the story-board implementation of SuVi are balloons, panels and text-boxes. Like the input data for the layout objects, the background images were hand-set in SuVi.

The constraint solver is comprised of the mechanisms needed to process the layout knowledge by initially producing and then solving appropriately defined general constraints for all layout objects. More details of these different constraints are given in the following section.

Finally, the **layout manager** component creates a corresponding layout representation in XML-format from the instantiated layout objects. This format is then processed by Comiclife©, a commercial software which is used by SuVi to render the resulting story-board layout.

3 Constraints

Story-board generation by SuVi distinguishes between two types of layout constraints: Page layout constraints and panel layout constraints. Furthermore, all constraints are split into hard and soft constraints. The former have to be fulfilled by the constraint solver in order to find a solution at all. The latter represent optimizations but do not necessarily have to be fulfilled. They are used to optimize found solutions with respect to the story-board specific design knowledge, e.g., how it's layout elements are organized in general. For example, be aware of the effect that appropriate locations of the balloons in a panel must consider the fact, that these locations "imply" a reading order for them, e.g., left to right vs. top down.
The system relies on Choco[2] , a constraint solver implemented in the programming language Java. Below we describe the most important constraints in more detail.

3.1 Hard page layout constraints

These constraints model the alignment of the story-board panels on a single page.

Page border constraints: A panel must not poke out the page margins.

Beginning constraint: The first panel is always placed on the top left position of a page.

Panel width constraint: The panel width is determined by the panel type,e.g., the width of a panorama panel is larger than a portrait panel, because the first one shows the whole meeting room, while the latter shows a close-up of a participant.

Brick wall constraint: A brick wall is the typical layout of a story-board and modeled by this constraint. Its purpose is to emphasize the reading direction, e.g., from left to right. Thus the leftmost panels in subsequent rows must not have the same width. Figure 3 shows a typical brick wall layout.

Maximum balloon constraint: Every panel must not contain more than five balloons.

[2] http://choco-solver.net

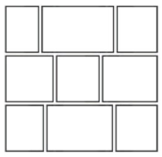


Fig. 3. A typical brick wall layout

3.2 Hard panel layout constraints

These constraints control the positioning of text-boxes and speech balloons inside a single panel.

Balloon border constraint: Balloons must not exceed the borders of a panel.

Text-box placement constraint: The alignment of the speech balloons must reflect the natural reading direction of the user, e.g., from the top left corner to the bottom right corner.

3.3 Soft constraints

Soft constraints are variables defined over a range of values. Their optimization is used to find the best layout solution out of a number of valid solutions.

End of line constraint: In order to exploit the available space of each line, the horizontal gap between the last panel of a line and the right border of the page should be minimized.

End of page constraint: The vertical gap between the last panel (i.e., last line) of a page and the bottom of that page should also be minimized. This constraint is used for for the optimal utilization of the available space on a page.

3.4 Resulting layout

The story-board that results from the application of all the constraints shown above on the hand-generated input is shown in figure 4. Text-boxes are rendered in green in the upper left corner of the panels which fulfills the text-box placement constraint. The panels are positioned according to the brick wall constraint, e.g., the first two panels in the first row have different widths unlike the first two panels in the second row. Also, all balloons fit into the panels as required by the balloon border constraint. The tails of all balloons are statically set in dependence of the panel type. The problem of overing the faces of the speakers is addressed in section 7.

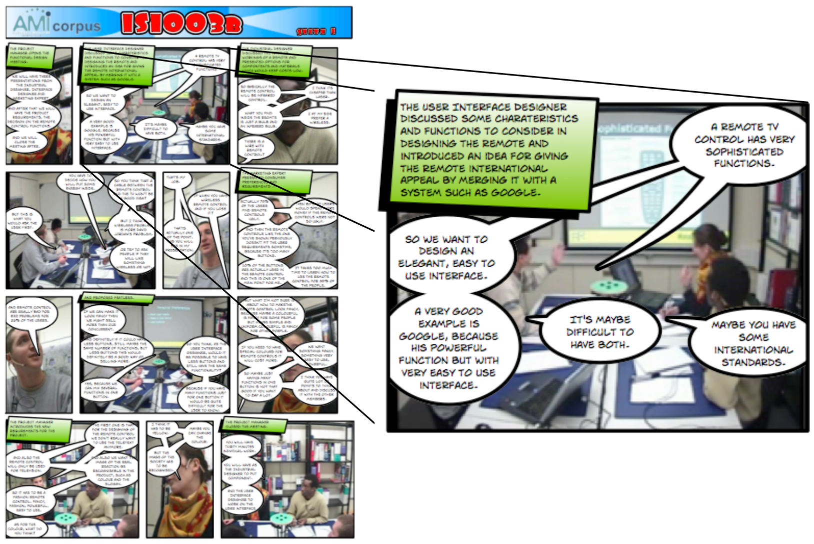

AMI corpus
IS1003b
THE USER INTERFACE DESIGNER DISCUSSED SOME CHARATERISTICS AND FUNCTIONS TO CONSIDER IN DESIGNING THE REMOTE AND INTRODUCED AN IDEA FOR GIVING THE REMOTE INTERNATIONAL APPEAL BY MERGING IT WITH A SYSTEM SUCH AS GOOGLE.
A REMOTE TV CONTROL HAS VERY SOPHISTICATED FUNCTIONS.
SO WE WANT TO DESIGN AN ELEGANT, EASY TO USE INTERFACE.
A VERY GOOD EXAMPLE IS GOOGLE, BECAUSE HIS POWERFUL FUNCTION BUT WITH VERY EASY TO USE INTERFACE.
IT'S MAYBE DIFFICULT TO HAVE BOTH.
MAYBE YOU HAVE SOME INTERNATIONAL STANDARDS.

Fig. 4. Story-board layout of meeting IS1003b

4 M2SuVi

The AMI meeting corpus offers a huge amount of annotated meetings and several video streams for each meeting but not explicitly contains story-board annotations. In order to use the data as input for SuVi, we need to find a mapping between the available corpus resources and the layout objects we described in the previous chapter. Therefore, we developed M2SuVi - meeting to SuVi. The following layout elements need to be filled automatically:

– The content of the text-boxes
– The content of the balloons
– Background images for the panels

SuVi needs the content for the balloons and text-boxes as complete sentences which are not available in the AMI corpus. Second, SuVi uses all sentences which are fed into the system for the layout. It is impossible to use all the data in a meeting and therefore a selection of relevant data for a story-board layout has to be made. We elaborate and implement two promising approaches for filling the layout objects with content taken from the AMI corpus.

First, we give an overview of the general system architecture in section 4.1. Then we describe the two approaches to make AMI corpus data available for automatic summary generation in section 4.2 and 4.3, respectively.

4.1 Architecture of M2SuVi

Figure 5 depicts the architecture of M2SuVi. The Input Reader of the system needs appropriate input data in the Nxt-Format[3] from the AMI corpus. Depending on the particular Content Creator the data are then extracted and converted into a special internal content representation which is depicted in figure 6. The linked Hash-map shown on the left contains the content for each topic. Every topic in turn comprises the contents for the text-box shown in green in figure 4 and the content for the assigned balloons. We additionally store the speaker, time and topic for each balloon to preserve the reading order. For filling the panels with appropriate pictures the Image Extractor uses the available video streams of a meeting and take time synchronized still-pictures. Otherwise, it uses a default picture. As a fallback for cases in which the required video stream is not available. Section 4.4 describes this extraction process in more detail. Finally, the Output Writer uses the resulting data to output the story-board file which is understood by SuVi.

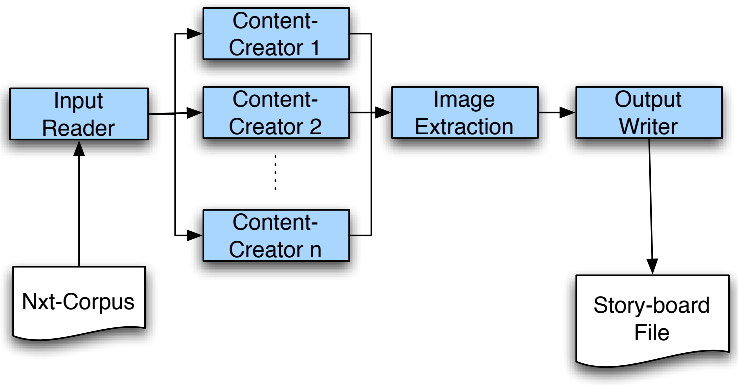


Fig. 5. Architecture of M2SuVi

4.2 Content creation with abstract summaries

Our first implementation gathers input data for the story-board generation from the AMI-corpus based on the abstract summaries of a meeting. Abstract in that sense, that they contain a hand written summary of the whole meeting. They are divided in the sections "abstract", "decisions", "action points" and "problems". In order to cover the whole meeting, we use the abstract-section of a summary for our approach. In the corpus, each sentence of the abstract summary is linked to a series of actually uttered sentences in the meeting. Moreover, it summarizes

[3] http://www.ltg.ed.ac.uk/NITE/index.html

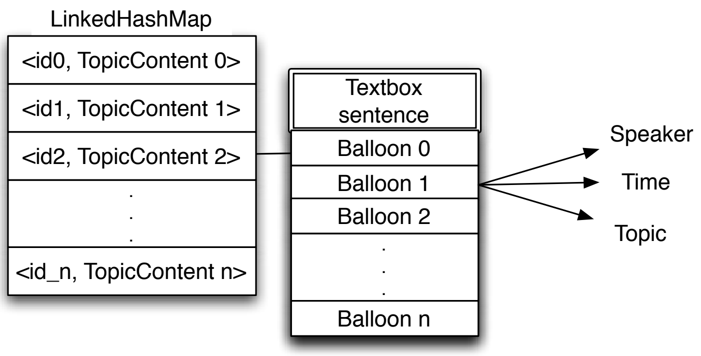


Fig. 6. Internal representation of the content

all these linked sentences.
The idea is to transform this type of annotation into a story-board layout by taking the sentence from the abstractive summary for a text-box while all linked sentences are taken for filling the balloons. Figure 7 illustrates the relation between a summary link from the abstractive summary, its linked sentences and the role they play in the story-board layout. In order to restrict the number of balloons for each summary sentence, the maximum number of links to follow can simply be set by a system parameter.

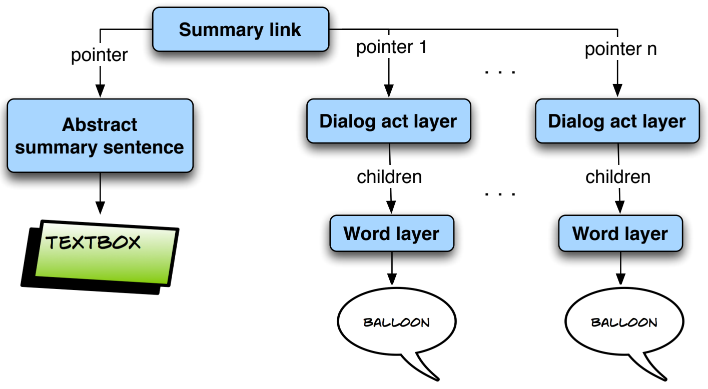


Fig. 7. Following the links in the abstractive summaries

4.3 Content creation with extractive summaries

Our second implementation of a story-board generation out of the AMI-corpus is based on the extractive summaries and the topic segmentation. The advan-

tage of this approach is that there are already automatic tools that produce this kind of data. However, this implementation is more complex because the extractive summaries are not linked in the corpus. Therefore, we cannot benefit from existing corpus structures. Instead, we have to compute the relations of the story-board elements to each other, e.g., which topic a sentence of an extractive summary belongs to.
Figure 8 shows the strategy for filling the balloons and text-boxes in this case. Every word in a meeting is assigned to a topic and stored in an appropriate data-structure. The corpus contains a set of default-topics, which are used throughout for annotation. We use these default-topics in the order in which they occur in a meeting and map every word to the topic it belongs to (see the left part of figure 8).
Every balloon is filled by one sentence of the extractive summary. Again, every sentence of the summary is used for the content of the story-board but in this case every extractive summary sentence fills one balloon. Remember that in the previous section we used the sentences of the abstract summary for the text-boxes, not the balloons. Since we have to build a mapping from every word to a topic, we can in turn assign a topic to every sentence because topics do not change within a sentence.
To fill the text-boxes with content we simply take the topic description as it exists for each topic of the AMI corpus. To augment the content of the text-boxes, one could simply extend these topic descriptions manually by providing more elaborated sentences.

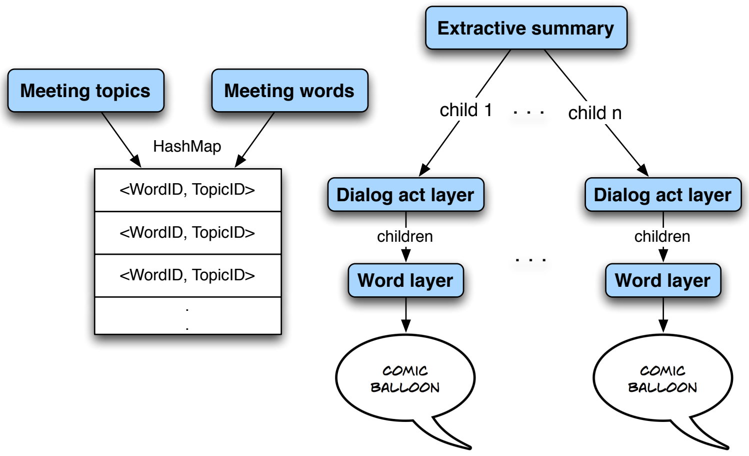


Fig. 8. Creating story-board content with extractive summaries

4.4 Still-pictures extraction

The extraction of still pictures is a procedure running independently of the content extraction step. There are two types of panels in SuVi: Portrait-panels and Panorama-panels The first type of panel is a close-up of a single speaker. It is chosen if SuVi identifies a dominant speaker for one panel (i.e., the speaker who made most of the utterances in a panel). In the corpus, there is an individual video stream available for each speaker from which we extract a time synchronous still-picture. Depending on the dominant speaker of a panel, the appropriate video stream is selected for still-picture extraction. The second type of panel is used if no dominant speaker can be identified. We then choose the video stream which shows the general view, e.g., the meeting room and all participants. If one of the video files cannot be found, e.g., if a reduced version of the system is running offline on a small mobile device, we implemented a simple fallback and use previously stored default pictures for the missing stream.

5 Batch-run over the AMI-corpus

In order to prove the robustness of the two components of our system, we did a batch-run over all available meetings in the AMI-meeting corpus. We did this for the two strategies we described in the previous two sections (extractive and abstractive summaries).
M2SuVi generated at most six topics per page and used a maximum of eight balloons per topic. For reasons of simplicity, we used default pictures for the layout. The batch-run was done on an Intel Core2Duo processor running at 2.2 GHz with 2 GB of working memory.
Table 5 shows the results for both of our components and demonstrates that a solution is possible for nearly every input as shown in the sixth column. There were a few meetings, however, for which no layout was possible because of errors in the corpus, e.g., dead links. Furthermore, an abstract summary is not available for every meeting, which explains the difference of total meetings in the fifth column. The average runtime in seconds of the constraint solver is shown in the second column.

Mode	avg. runtime (s)	result	no result	total	successful layouts in %
Extractive	48.8	120	3	123	97.6
Abstractive	86.1	86	12	98	87.8

Table 1. Batch-run results

6 Automatic end-to-end story-board generation

In order to show the applicability of our generic approach in an end-to-end implementation for a completely different domain and in a different use case, we tested the system on a selected part of the first AMIDA review meeting. Here is a short description of this special approach:

In contrast to the data in the AMI-corpus, the review meeting data was completely processed by automatic tools from The University of Sheffield (ASR), The University of Edinburgh (topic segmentation & labeling) and The IDIAP Research Institute in Martigny (recordings). An ASR engine provided the raw data and the necessary foundation for further processing. After that, the ASR output was automatically converted into the required NXT-format and segmented into topics.

Both components of our system, SuVi and M2SuVi, proved to be very robust under the new input data and in a completely new domain. Although the data was previously unknown, after resolving some minor formating issues it was possible to do a provisional story-board layout of the review meeting. However, we realized, that the balloons got too large and covered the whole panel. A newly invented "spurts"-layer linked far more words per comic-balloon than the original data, i.e. the links in the abstractive and extractive summaries. We quickly limited the content-size of the comic-balloons and added "..." at the end of each comic-balloon to indicate that there is more text available. Due to technical problems, there were no video streams available. However, we could benefit from our fallback strategy here by simply using default pictures which were taken during the meeting with a digital camera. Figure 9 shows the resulting story-board layout of the first AMIDA review meeting.

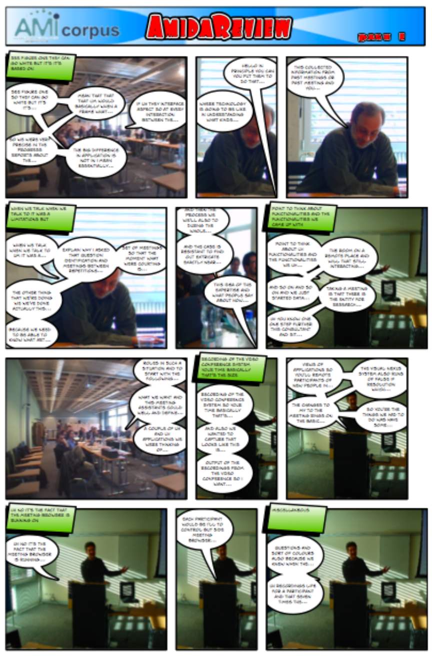


Fig. 9. Story-board layout of the first AMIDA review meeting

7 Conclusion & future work

In this paper we described SuVi, an automatic layout tool for meeting summaries. Due to the initial restrictions of the input for this tool, M2SuVi was developed which made the complete AMI meeting corpus accessible for story-board generation.

The generation of layouts is robust and works fairly well, even in a completely new domain as described in section 5 and 6. But too often the faces of the speakers are superposed by the balloons. In the next version of SuVi, we will make use of image processing techniques to detect faces and set dynamic constraints in such a way, that the faces are kept free of balloons. The detection of the face positions will also give us the possibility to optimize the positioning of the balloon tails and adjust them to the mouth of the person. Actually, they are statically set simply depending on the panel type. SuVi now uses a one page layout by default. Depending on the number of how many topics/balloons available, we will implement a multi page layout automatically.

With our project partner Philips we are currently implementing a server-version of our system which takes the user input from a web page, generates the story-board in the background and sends back the result as a Scalable Vector Graphic (SVG) using a newly developed proprietary display component. The http-client will run on an embedded platform making it necessary to replace the proprietary ComicLife©-format. The final version will feature output formats in jpeg, png, pdf and plain svg.

We are also in negotiation with industrial partners for commercial use of this technology.

References

1. Frey, J.: "Constraint-basierte Generierung parametrisierbarer, multimodaler Comic-Layouts für verlaufsorientierte Meeting-Zusammenfassungen", Master's thesis, Saarbrücken, 2007
2. Lang, B.: "Parameterisierbares Layout inhaltsorientierter, multimodaler Zusammenfassungen von Meetings anhand der Zeitungsmetapher in einem Constraint-basierten Ansatz", Master's thesis, Saarbrücken, 2007
3. Kleinbauer, T.; Becker, S.; Becker T.: "Indicative Abstractive Summaries of Meetings", Saarbrücken, 2007
4. Rendering software Comiclife© by plasq, http://plasq.com/comiclife/
5. Carletta, J., Ashby, S., Bourban, S., Flynn, M., et al: The AMI Meeting Corpus In: Proceedings of the Measuring Behavior 2005 symposium on "Annotating and measuring Meeting Behaviour", 2005
6. AMIDA: "Augmented Multiparty Interaction with Distance Access", Deliverable D5.2: Report on multimodal content abstraction. Technical report, Brno University of Technology, DFKI, ICSI, IDIAP, TNO, University of Edinburgh, University of Twente and University of Sheffield, 2007
7. Carletta, J., S. Evert, U. Heid und J. Kilgour: "The NITE XML Toolkit: data model and query language" Language Resources and Evaluation Journal, 2006.
8. Apt, K. R.: "Principles of Constraint Programming", Cambridge University Press, 2003.