# Interactors: operators to automatically generate interactive multimedia documents from captured media

Didier A. Vega-Oliveros, Diogo S. Martins, Maria da Graça C. Pimentel
Universidade de São Paulo – São Carlos, SP - Brazil
{davo,diogosm,mgp}@icmc.usp.br

## ABSTRACT

In some scenarios, it is paramount that collaborative synchronous sessions be recorded for later review. Particularly in the case of webconferencing tools, the approach usually adopted for recording a meeting is to generate a linear video with the content of the exchanged media. Such approach limits the review of the meeting to the user watching a video using the traditional timeline-based video controls. In this work we advocate that synchronous communication tools generate interactive multimedia documents as a result of a session. For this purpose we detail the generation of an interactive multimedia document by means of user-media operators (e.g. slide changes, chat messages, ink-based interactions, audio events) that we call Interactors, enabling users to browse the generated document by navigating points of interest in the captured media. We define the Interactors approach and demonstrate it in the context of a tool in use.

## Categories and Subject Descriptors

I.7.2 [**Document and Text Processing**]: Document Preparation, Format and notation, hypertext/hypermedia, Languages and Systems, Multi/mixed media.; H.5.1 [**Information Interfaces and Presentations**]: Multimedia Information Systems - Audio, Video.

## General Terms

Documentation, Human Factors.

## Keywords

Interactive Video, Document Engineering, Automatic Authoring, NCLua.

## 1. INTRODUCTION

In remote work sessions such as meetings and distance education lectures, synchronous communication tools enable collaboration by exchanging text, images, documents, audio or video streams. In some scenarios, it is paramount that collaborative synchronous sessions be recorded for later review. Main reasons for reviewing a recorded session include to keep accurate records, to revisit portions of the session which were misunderstood, to obtain proofs and to recall certain ideas [9]. Particularly in the case of webconferencing tools, the approach usually adopted for recording a session is to generate a linear video with the content of the exchanged media. Such approach makes reviewing the session a linear and time consuming process, specially if the user only count on traditional VCR-like operators (e.g. play, pause, forward, rewind) available in media players.

In favor of circumventing the potential inability to lookup specific information or to efficiently recall the overview of a session, researches related to capture & access systems [20] and smart meetings rooms [23] have been concerned with the development of better methods for reviewing sessions. These researches generally tackle the issue of developing indices so that users can jump to relevant points of interest within the session. Such non-linear access is mediated by specialized tools called browsers (commonly regarded as meeting browsers or meeting players). Features for building indices range from intentional user annotations to events derived directly from media elements via content-based analysis [14]. Browsers for recorded sessions are generally focused on specific types of indexed features, such as audio, video, discourse and artifacts [22].

Current approaches for developing meeting browsers have their drawbacks. These browsers are commonly built using *ad-hoc* development frameworks that entail tight coupling among the capture environment and the browser tool [4], thus reducing reuse of the same tool to access sessions gathered from different environments (e.g. meeting rooms and webconferencing systems). Moreover, these browsers are usually specialized toward one or two types of indices [21], impairing the richness of random-access operators users can apply while reviewing the session.

In this paper we advocate that browsers be built through the automatic generation of an interactive multimedia documents (iMMD) from the captured session. The generated iMMD is enriched with several types of timestamped operators (e.g. slide changes, chat messages, ink-based interactions, audio events) called Interactors, enabling users to browse the generated document by navigating points of interest in a timeline. In order to enforce loose coupling among the capture environment and the iMMD, the session information is exported from the environment into an XML-based interchange document which encapsulates cap-

tured media (e.g. slides, video, chat conversations) alongside the operators. Via document transformations the session information is converted into an iMMD that synchronizes the streams of captured media and provides timeline-based access operations for browsing. We demonstrate our approach through a case study concerning remote collaboration and illustrate the generation of an iMMD in NCL with procedural objects in Lua [18].

The remaining of this paper is organized as follows: theoretical background on indexing and browsing captured media is presented in Section 2; Section 3 details our approach for generating an interactive multimedia document from a recorded session, explaining the data interchange document, the document generation process and the characteristics of the autogenerated interactive multimedia document. A case study demonstrates a proof-of-concept prototype in Section 4. Related work is analyzed in Section 5. Finally, Section 6 concludes this paper and points out future research efforts.

## 2. THEORETICAL BACKGROUND

A classification of indices for captured media is proposed by Minneman et al. [14] who categorize indices into four broad classes: intentional annotations, performed explicitly by participants while the meeting is happening; side-effect indices, produced by capturing user-media or user-equipment interactions, such as slide changes and microphone activation; derived indices, automatically obtained by content-based analysis, for instance to identify speakers or detect moments of user-user interaction [13]; and post-hoc indices, consisting of user-media interactions performed during review of the meeting recordings. An extension of this taxonomy is proposed by Geyer et al. [11] who define online and offline indices, built during and after the meeting respectively; they define also explicit and derived indices, built by users and obtained from analysis of media elements, respectively. Similar developments have been proposed by Bouamrane and Luz [4] which formalize user-media interactions yielding indices at production-time or at consumption-time.

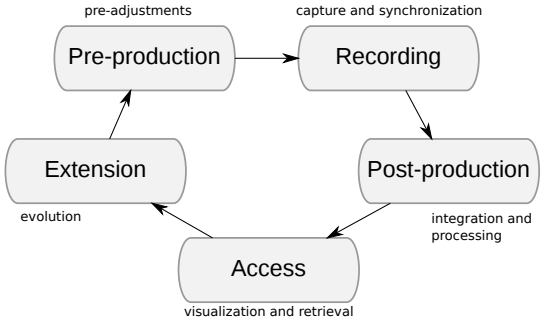


**Figure 1: Lifecicle of captured media. Adapted from Abowd et al. [1] and Pimentel et al. [17]**

From these categorizations we notice that central dimensions are the types of indexed interactions (user-media, user-equipment or user-user) and the moments at which these interactions take place. In regard to moments for capturing interactions, Abowd et al. [1] and later Pimentel et al. [17] propose a model of the lifecycle of captured media (Figure 1). Each phase of the lifecycle presents opportunities to build indices based on different types of interactions. In the pre-production phase offline intentional annotations can be applied to perform content segmentation, for instance. During recording, online annotations and side-effect indices are commonly built [5]. Offline indexing can occur in the post-production phase, such as the generation of derived indices [2]. Review-time explicit indices obtained from user-media interaction can be built while the user is accessing captured media elements (by means of annotations [15] or discrimination of moments [19], for instance). Such user-media interactions can be used to enrich and generate new versions of the original media elements [7, 6] in the extension phase.

Meeting browsers exploit indices to assist users through access interfaces to navigate recordings. Whittaker et al. [22] categorize meeting browsers according to the types of indices available. Audio-based browsers generally focus on events such as speaker turns, pause detection and emphasis; video browsers exploit keyframes and observable participant behavior; artifact browsers enable navigation by means of user-media interaction such as slide changes, ink-based notes and document sharing; and discourse-based browsers concentrate on navigation of speech transcripts, named entities and perceivable emotions. Tucker and Whittaker [21] point out several limitations of meeting browsers, among them impaired browsing on devices with limited resources (such as mobile devices and TV set-top boxes) and filtered presentation (browsing through summarized versions of the captured media).

In this paper we tackle the issue of automatically generating document-based browsers by means of several types of indices: online side-effect indices (slide transitions and chat messages), intentional annotations (ink-based interactions), and offline derived indices (observable participant behavior from audio). Such indices are captured during recording and post-production phases and are used to provide an interface focused on navigation of artifacts and audio events via timeline-based operators called Interactors. Our document-centric approach tackles challenges for meeting browsers: in particular, the approach enables the efficient review of meeting recordings via a constrained device such as a TV set-top box. Moreover, adopting structured multimedia documents opens up benefits for content enrichment, filtering and sharing, as advocated by Cesar et al. [8].

## 3. INTERACTORS: PROPOSED MODEL AND IMPLEMENTATION

In this section it is presented the proposed model of operators to generate interactive multimedia documents (iMMD) from captured media. The iMMD is intended to be generated from a XML-based interchange document that aggregates the captured media alongside logged interaction events (e.g. slide changes, ink-based events, chat messages) that we call Interactors. Adopting a XML-based interchange document enables the proposed model to be instantiated to different capture environments, regardless of particularities of their implementations. We demonstrate our approach considering that the automatically generated document is reviewed in interactive TV clients whose primary interaction mechanism is via remote controls. As such the generated iMMD follows design guidelines and interaction mechanisms specially suited to these devices.

An interactor is defined as an operator that is applicable

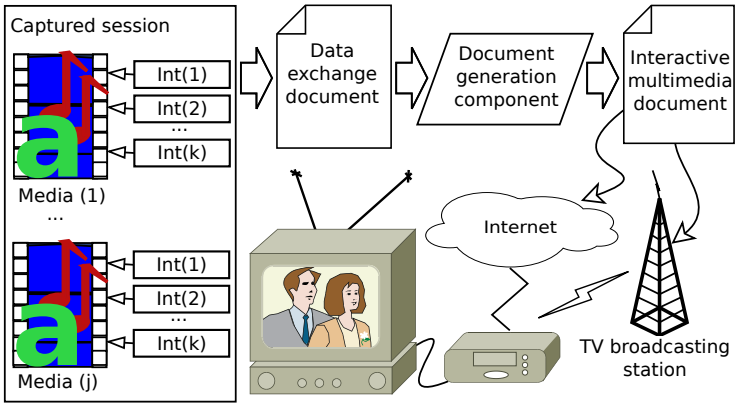


**Figure 2: Interactors model for generation of interactive multimedia documents**

to a specific type of medium. In particular, in this work we deal with whiteboard interactors (slide transitions and ink-based annotations), chat-based interactors (messaging and posting of URLs) and audio-based interactors (detection of moments of silence). The proposed model is presented in Figure 2. The session data captured by the environment is exported into a data interchange document comprising links to the several media elements and logged interactions for each media element. The exported interchange document feeds the document generation component which transforms it into an iMMD. After these steps, the automatically generated iMMD can be transmitted (e.g. via TV broadcast or return channel) to client set-top boxes and be reviewed by users in interactive TV clients.

Considering that a captured session describes $j$ media elements and $k$ logged interaction events, we define an interactor as a mapping $TL(a, b)$ where $a \leq j$ is a captured media element and $b$ is a logged interaction event. By definition, each interactor can map an unlimited number of interaction events. Thus for a specific media element $i$, the mapping $TL(i, b)$, $b \geq 0$, represents the full set of interaction events for the media element $i$. Additionally, there is the constraint that an interactor may not be applicable to every type of medium (e.g. slide changes are not applicable to audio).

In the following sections it is detailed the main issues of the proposed model: Section 3.1 details the data exchange document. The document generation component is explained is Section 3.2. Section 3.3 reports the characteristics of the generated iMMD, including its declarative and procedural features. Finally, Section 3.4 illustrates the proposed model by definition and demonstration of offline derived operators called audioInteractors.

## 3.1 Data exchange document

The data exchange model standardizes: *i*) the description of the several media elements comprising a session; and *ii*) the description of the logged interaction events captured by the environment. This model is defined by means of a XML Schema and can be used by different environments to export captured data, enforcing loose coupling among the underlying session storage format and the generated iMMD. Moreover, a structured document facilitates transformation-based approaches to automatically generate iMMDs.

A data exchange document (Figure 3) has a primary element `player` which comprises several media elements captured by the meeting environment. For illustration purposes, the document in Figure 3 describes slides, chat conversations, video and audio (lines 3-12, 14-22, 24-30, 32-38, respectively); but it is worth stressing that the underlying XML schema foresees other types of medium not represented in this document. Each media element has attributes to specify its source, which may be local or remote. A media element is local when a bundle is retrieved from the capture environment, containing the data exchange document and all captured media; remote media is relevant when the exchange document is obtained as response from a web service, for instance, thus requiring that the media be retrieved later by their URLs.

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <player author="..." >
3    <slideSet>
4      <local path="player/slides/">
5        <interactor description="slide change">
6          <image file="1.png" begin_ss="58" end_ss="76"/>
7          <image file="2.png" begin_ss="76" end_ss="94"/>
8          ...
9          <image file="23.png" begin_ss="425" end_ss="425"/>
10       </interactor>
11     </local>
12   </slideSet>
13   ...
14   <chatSession>
15     <local path="player/chatLog.html">
16       <interactor description="someone wrote">
17         <chat id="chat0" begin_ss="54" end_ss="56"/>
18         ...
19         <chat id="chat13" begin_ss="438" end_ss="441"/>
20       </interactor>
21     </local>
22   </chatSession>
23   ...
24   <video>
25     <server url="http://...">
26       <interactor description="participant interview">
27         <area begin_ss="10" end_ss="385"/>
28       </interactor>
29     </server>
30   </video>
31   ...
32   <audio>
33     <local filePath="file://...">
34       <interactor description="participant interview">
35         <area begin_ss="0" end_ss="385"/>
36       </interactor>
37     </local>
38   </audio>
39     ...
40 </player>
```

**Figure 3: Excerpt illustrating the global structure of the data exchange document**

Additionally, each media element has a set of interactors associated with it. For instance, slides can have a series of interactors of type `slide change` associated with them (lines 5-9), while chat sessions can have interactors of type `someone wrote` (lines 17-19). For each interactor, the schema provides specific identification and timing attributes. Interactors related to continuous media elements (such as streams of audio and video) are identified by filename and timed by their start and end moments. On the other hand, interactors related to discrete media elements (captured from whiteboards, chats, photographic cameras, scanners, etc.) are identified by filename or surrogate and are timed by the period between two capture events (attributes `begin_ss` and `end_ss`): for instance, the period among two slide changes or two chat messages. All these time markings are relative to the beginning of the session and are used for synchronization purposes when the iMMD is generated.

## 3.2 Document generation component

In order to create an iMMD from a captured session, the document generation component performs transformations on the exported data interchange document. This component clusters the interactors by type (e.g. slide changes, ink annotations, chat conversations, etc.) and for each cluster it builds a main timeline, decorated with its interactors. The generated timelines are assembled in the iMMD with the media elements. Consequently, the interactors act as operators that enable users to navigate points of interest — such navigation is possible due to activation of a procedural document (more details on section 3.3) which guarantees that all media elements are synchronized when an operator is issued.

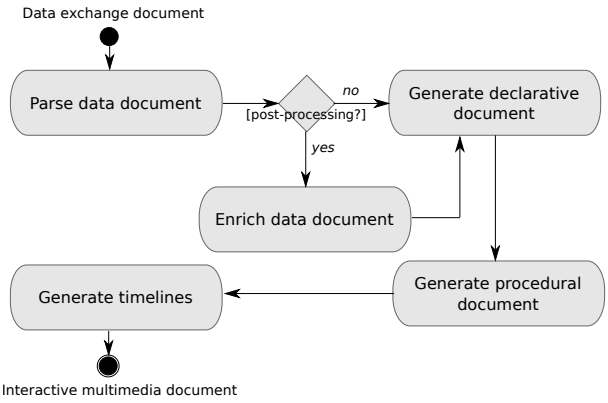


**Figure 4: Activity model for the document generation component**

The document generation component (Figure 4) first performs parsing of the data exchange model. If postprocessing is available (for example to build offline derived indices) the resulting DOM is enriched with additional interactors. The next activity is concerned with the generation of a declarative multimedia document which acts as the user interface for reviewing the session. This declarative document describes the attributes of the media elements (such as medium type, source location, etc.), position and size of each media element on the screen, inter-media connections and synchronization according to the interactors. Additionally, the declarative document includes the timeline that is generated in a later activity. In particular, this timeline is the element connecting the declarative document with the procedural document generated afterward.

Even though in this work we are concerned with interactors obtained in the recording and post-processing phases (refer to Figure 1), it is worth mentioning that enabling enrichment of the data document opens up possibilities for exploring interactors in the access and extension phases also. Once review-time interactors for media editing and bookmarking are provided, the data document can be enriched with user-media interactions that could yield the generation of new, user-tailored versions of the original session.

## 3.3 Interactive multimedia document

The interactive multimedia document is the element that allows access and review of a captured session. The autogenerated iMMD is composed of two distinct documents: the declarative review document and the procedural controller document. Built in NCL, the declarative document is responsible for rendering and synchronizing the media elements as well as configuring the layout of the multimedia presentation; built in Lua, the procedural document receives navigation events via remote control and in response activates the appropriate anchors in the declarative document.

From a presentation perspective, it is possible to define different layouts for the declarative document according to the media elements described in the data exchange document. Different layouts are appropriate for different sets of captured media: for instance, a session composed of three streams of video, one chat conversation and a set of slides may require a different layout than a session with two streams of video and a slide set. Therefore, the approach of defining a declarative document for tackling presentation-only issues promotes separation of view-related concerns from data-handling and application logic concerns.

```
1  <ncl xmlns="http://www.ncl.org.br/NCL3.0/EDTVProfile">
2      ...
3      <media descriptor="video2_Descriptor" id="video2small"
4                  src="ds3.avi">
5          <area begin="2s" id="v2t2s"/>
6          <area begin="4s" id="v2t4s"/>
7          ...
8          <area begin="384s" id="v2t384s"/>
9      </media>
10     ...
11     <media descriptor="stop_Descriptor" id="btnStop"
12                 src="stop24.gif" type="image/gif">
13         <!-- sync discrete media -->
14         <area begin="2s" id="bt2s"/>
15         <area begin="4s" id="bt4s"/>
16         ...
17         <area begin="384s" id="bt384s"/>
18     </media>
19     <media descriptor="timeLine_Descriptor" id="timeLine"
20                 src="tl.lua" type="application/x-ginga-NCLua">
21         <area id="t2s"/>
22         <area id="t4s"/>
23         ...
24         <area id="t384s"/>
25     </media>
26     ...
27 </ncl>
```

**Figure 5: Excerpt of the autogenerated declarative document focusing on anchors**

Structurally, the declarative document defines a set of anchors that represent all available operators contained in the document timeline (Figure 5) — operators which are used to synchronize the presentation. All discrete media like slides and chat messages are grouped in chronological order into a single NCL object (lines 11-18) which connects and orches-

trates their proper synchronization. In the case of continuous media, such as audio and video, a separate anchor is defined (lines 3-9) for each media element. Additionally, an anchor is defined for the timeline (lines 19-25).

Imported by the declarative document (lines 19-20), the procedural document defines an object in Lua that takes care of handling navigation events issued by the user. Upon receiving these events, this object operates over the declarative document to jump to the requested point of interest. This action entails resyncing and playing the media element via a signal (Lua event) that activates one of the available anchors in the declarative document. The procedural document defines two elements: *i*) the timeline model, that maps navigation events (e.g. user-initiated events, such as pressing a remote control button) to interaction events (e.g. points of interest) within the media element; and *ii*) the timeline controller, that handles navigation events and queries the timeline model to update the state of the presentation.

```
1  /* FIRST STRUCTURE */
2  local tam = E   //number of Interactive Events
3  local frame={}
4  frame[1]={instant='t1s',button='RED', dist= 0, definition= 1}
5  ...
6  frame[E]={instant='tTs', button='B', dist= X, definition= D}
7
8  /* SECOND STRUCTURE */
9  local mTam = N //number of Interactors
10 local menu={}
11 menu[1]={text='Midia(1)_Interactor(1)', button='RED',
   definition= 1}
12 ...
13 menu[N]={text='Midia(J)_Interactor(K)', button='B',  definition=
   D}
```

**Figure 6: Excerpt of the autogenerated procedural document focusing in the timeline model**

The mappings established by the timeline model (Figure 6) are registered in the structure `frame` by means of records storing the moment (`instant`) of the point of interest, navigation event (`button`) issued by the user, normalized offset position (`dist`), in dots or pixels, of the point of interest in the visual representation of the timeline and a sequential identifier (`definition`) for the point of interest.

The total number of interactors is stored by the variable `mTam`, being `mTam` $\leq$ `tam` because each interactor is a non-empty set of interaction events. The structure `menu` defines the underlying data structure used for generating the interactive timeline to navigate the session. Each record of this structure is logically linked to the `frame` structure by a pair $(B, def)$ where $B$ is a button and $def$ is the identifier of an interaction event.

Both data structures are employed by the timeline controller to activate specific anchors of the declarative document in response to a navigation event issued by the user. Main methods used by the timeline controller are defined below.

- `Redraw()`: takes care of updating the visual appearance of the timeline in the declarative document. Its responsibilities include the placement of tags in the timeline corresponding to the interaction events.
- `Stop()` and `Start()`: these functions are responsible for triggering the resynchronization of an specific anchor in the declarative document
- `Handler(evt)`: receives as a parameter a user navigation event (issued via remote control) and reacts according to a mapping that relates an event to its corresponding action. Beyond controlling the access to specific points of interest, this function concerns also other interface-related issues.
- `openMenu()` and `closeMenu()`: receives events for showing/hiding the decorated timeline when a button in the remote control is pressed.

In summary, the timeline controller is responsible for receiving events from the user remote control, querying the timeline model for the anchor that corresponds to the point of interest and in response activating the returned anchor in the declarative document to update the presentation state.

## 3.4 AudioInteractors

An important characteristic of our approach is that the interchange document can be enriched (refer to Figure 4) with additional interactors prior the generation of the iMMD. In this section we demonstrate such enrichment by means of derived indices called AudioInteractors: a special type of interactors obtained by content-based analysis of voices in the captured session.

AudioInteractors can be used in many phases of the media lifecycle, but in this work we are concerned with their use in the post-production (offline) phase, which does not entail intrusive interventions in the capture environment. These interactors are divided in two categories:

*Time-based.* Obtained by monitoring user-microphone interaction (capture phase) or detecting patterns of digital audio (post-production phase). They can be used to select moments or intervals in the audio in which a specific voice behavior applies.

- silenceMoments(time $Tmin$): moments in which there were no voices during $T$ ($T > Tmin$) units of time.
- spokenMoments(time $Tmin$): returns the moments before which someone has spoken for at least $Tmin$ units of time.

*Attribute-based.* There are digital audio attributes (e.g. frequency, pitch, noise, amplitude, etc.) that can be exploited to detect points of interest within the media element.

- voiceIncrease(): returns moments in which there was a consistent increase in speech volume.
- conversation(interval $T$): in reference to an interval $T$ in the audio, returns the potential number of participants who spoke during this interval.
- outstandingMoments(): return the moments in time when there were outstanding moments in the media element. We consider outstanding moments as those in which several people are talking at the same time with consistent increase in the volume of their voices.

Various content-based methods can be employed to derive audioInteractors. For demonstration purposes, Figure 7 illustrates the derivation of the time-based operator *SilenceMoments()* using wavelet transformations [12], in particular the Haar transformation. The adopted strategy consisted on applying the discrete wavelet transform using filters $h[.]$ and $g[.]$. The curve (a) in Figure 7 corresponds to the low-pass haar filter of the original signal. The result of the filter is grouped in chunks of 0.5 sec and for each chunk is counted the number of samples that are in the center line of $h[.]$ whose value in $g[.]$ is greater than a threshold continuously updated at runtime. For each chunk is calculated the

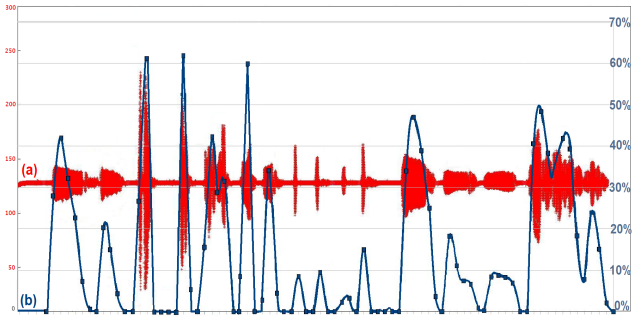


**Figure 7: Audio analysis to detect silence. Curve (a) represents low-pass filter haar h[.]; curve (b) represents detected silence**

percentage of samples that were not classified as silence, as shown in the curve (b). The chunk with value equal to 0% represents a moment of silence.

The result of this processing determines the moments of silence. Those moments are then included in the interchange document during the enrichment activity (refer to Figure 4) of the document generation component. After this post-processing step, the component can proceed with the generation of the interactive multimedia document as illustrated in Section 3.3.

# 4. CASE STUDY

In order to illustrate the approach for automatic generation of interactive multimedia documents using interactors, this section reports a case study conducted with a capture environment in use. For the case study, a proof-of-concept prototype was developed in NCLua. The capture environment used in the case study is presented in Section 4.1 and Section 4.2 demonstrate the interactive multimedia document obtained with data from this environment.

## 4.1 Capture environment and scenario

DiGaE (*Distributed Gathering Environment*) is a capture environment for collaborative meetings that can be used both in instrumented rooms or in webconferencing mode. When used in instrumented rooms, DiGaE has support for synchronization of video cameras, microphones, electronic whiteboards and projectors; RFID readers can be used for participant identification. When used for webconferencing purposes, a special configuration called DiGaE Home provides a web-based tool to capture audio and video streams from desktops and laptops, as well as other communication tools such as instant text-based messaging and synchronous software-based whiteboard.

In this case study, we employed DiGaE Home to capture a sample meeting with three remote participants that used a whiteboard software to load a set of slides and make annotations, a chat session to provide complementary textual content (such as URLs) and video capture from webcams. The dynamics of the meeting required that participants took turns performing their oral presentations, making ink-based annotations over the slides and posting messages in the chat. The recorded session was exported to a XML data interchange document that was post-processed and enriched with the audioInteractor *spokenMoments()* over each captured video.

## 4.2 Automatically generated document

From the resulting data interchange document an interactive multimedia document was automatically generated and is detailed in the following figures.


**Figure 8: Screenshot depicting the structure of the generated document**

Figure 8 depicts the general layout of the generated document. This figure illustrates the main regions of the interface, namely: *1*) the whiteboard region; *2*) the video region; *3*) the chat region; and *4*) the timeline region. The timeline enables users to navigate different points of interest within the media element. Interactive digital TV remote controls present colored buttons specially suited for interactivity. In the generated document in particular, those buttons can be used to toggle media-specific interactors to decorate the timeline (e.g. red button enables ink-based interactors, yellow button enables video-related interactors, and so on).


**Figure 9: Screenshot illustrating the timeline decorated with ink-based interactors**

Figure 9 depicts the decoration of timeline with ink-based interactors after the red button is pressed. As a result, the points of interest populate the timeline with icons resembling the button that was pressed.

To navigate to specific points of interest the user employs


**Figure 10: Screenshot depicting the presentation state when video-based interactors are activated**

the right/left navigation buttons of the remote control to change focus and pressing the *ok* button when the desired point is reached. Figure 10 depicts the state of the presentation after video-based interactors were activated (by pressing the yellow button on the remote). As a consequence, the timeline is decorated with an interactor related to this media element, in particular the audio-based *spokenMoments* interactor corresponding to the first participant (video at the top) — this interactor in particular can be filtered to specific users. The right/left navigation keys were employed to focus on the second point of interest available.


**Figure 11: Screenshot depicting the activation of an alternate interactor for the same media element**

Additionally, a label is included above the timeline to indicate the specific interactor that is active; if other interactors are available for the same media element, both up/down navigation keys, and also a numerical code, can be used to select them. In the screenshot depicted in Figure 11 the up/down keys were used to activate the *spokenMoments* interactor for the third user (bottom video); consequently the presentation is updated to change the label of the timeline and also the corresponding points of interest.

# 5. RELATED WORK

Much research on meeting browsers have focused on navigation of audio recordings alone. Ehlen et al. [10] describes an interface for timeline-based meeting review and user feedback that concerns only navigation of speech transcripts via events of interest (e.g. decisions, minutes). Bouamrane and Luz [3] provide a system to record user-media interactions (editing operations) that are used to build a browser interface with tree-like operators to navigate audio recordings. Although very recurrent in the literature, such exclusive focus on just one type of medium hinders applicability of a meeting browser to environments that can record heterogeneous media sources, such as webconferencing systems and smart meeting rooms.

Similar limitations are observable on video-only browsers: Yu et al. [24] presents a system to navigate segmented video recordings through events of user-user interaction (e.g. propose, request information, acknowledge, etc.). Another video-only approach is reported by Behera et al. [2], which provides a browser to query a repository of SMIL documents generated from recordings of projected slides: using the SMIL browser, users can navigate segments of the retrieved documents. Even though an approach to generate interactive multimedia documents is advocated, it focuses only on one type of medium.

In respect to discrete media elements, Cattelan et al. [7] provides ink-based operators related to slide change, pen strokes, etc. Based on these operators, it is possible to filter or expand a set of slides. Branham et al. [5] also presents a system for indexing operations performed over a whiteboard. A limitation of these researches is the restriction to a single tool, hindering their generalization to environments with additional synchronous collaboration tools.

Pimentel et al. [16] presents an architecture for capture and access systems. To index the contents, various indices are built and a XML-based interchange document can be exported and transformed to an access interface. But the research does not explain how an access interface can be effectively generated from the interchange document or how such an interactive document could be built.

# 6. CONCLUSIONS AND FUTURE WORK

We have presented an approach to automatically generate interactive multimedia documents from media captured in collaborative activities. The approach defines several operators called Interactors that can be used to index points of interest in a recorded synchronous session. Advantages of adopting a structured document to export data from the capture environment were highlighted, such as loose coupling and applicability of transformations. For demonstration purposes, we have experimented with a proof-of-concept prototype, developed in NCLua, that was fed with data exported from a capture environment in use.

As far as future work is concerned we point out the formalization of more operators and the development of methods to effectively combine them while navigating the document. Such a development could enable users to generate new versions of the multimedia document by combining a series of operators that yields a summarized version of the original media. We plan also to improve the review interface and the underlying interaction mechanisms by performing inspections and user tests.

**Acknowledgments**. We thank FAPESP, CNPq, CAPES, FINEP, MCT; and professor Rodrigo Fernandes de Mello, who greatly contributed to the advancement of the audioInteractor analysis module.

## References

[1] G. D. Abowd, C. G. Atkeson, J. Brotherton, T. Enqvist, P. Gulley, and J. LeMon. Investigating the capture, integration and access problem of ubiquitous computing in an educational setting. In *Proc. of the International Conference on Human Factors in Computing Systems*, pages 440–447, 1998.

[2] A. Behera, D. Lalanne, and R. Ingold. DocMIR: An automatic document-based indexing system for meeting retrieval. *Multimedia Tools and Applications*, 37 (2):135–167, June 2007.

[3] M. Bouamrane and S. Luz. Navigating multimodal meeting recordings with the meeting miner. In *Proc. of the 7th International Conference on Flexible Query Answering Systems*, pages 356–367, 2006.

[4] M.-M. Bouamrane and S. Luz. Meeting browsing. *Multimedia Systems*, 12(4-5):439–457, October 2006.

[5] S. Branham, G. Golovchinsky, S. Carter, and J. T. Biehl. Let's go from the whiteboard: supporting transitions in work through whiteboard capture and reuse. In *Proc. of International Conference on Human Factors in Computing Systems*, pages 75–84, 2010.

[6] R. G. Cattelan, C. Teixeira, R. Goularte, and M. D. Pimentel. Watch-and-comment as a paradigm toward ubiquitous interactive video editing. *ACM Trans. Multimedia Comput. Commun. Appl.*, 4(4):1–24, 2008.

[7] R. G. Cattelan, C. Teixeira, H. Ribas, E. Munson, and M. Pimentel. Inkteractors: interacting with digital ink. In *Proc. of the 2008 ACM Symposium on Applied Computing*, pages 1246–1251, 2008.

[8] P. Cesar, D. C. A. Bulterman, and A. J. Jansen. Benefits of structured multimedia documents in IDTV. In *Proc. of the ACM Symposium on Document engineering*, pages 176–178, 2006.

[9] M. Czerwinski, D. W. Gage, J. Gemmell, C. C. Marshall, M. A. Pérez-Quiñones, M. M. Skeels, and T. Catarci. Digital memories in an era of ubiquitous computing and abundant storage. *Communications of the ACM*, 49(1):44, January 2006.

[10] P. Ehlen, M. Purver, J. Niekrasz, K. Lee, and S. Peters. Meeting adjourned: off-line learning interfaces for automatic meeting understanding. In *Proc. of the 13th international conference on Intelligent user interfaces*, pages 276–284, 2008.

[11] W. Geyer, H. Richter, and G. D. Abowd. Towards a Smarter Meeting Record-Capture and Access of Meetings Revisited. *Multimedia Tools and Applications*, 27 (3):393–410, December 2005.

[12] R. C. Guido, J. F. W. Slaets, R. Köberle, L. O. B. Almeida, and J. C. Pereira. A new technique to construct a wavelet transform matching a specified signal with applications to digital, real time, spike, and overlap pattern recognition. *Digital Signal Processing*, 16 (1):24–44, 2006.

[13] S. Junuzovic, R. Hegde, Z. Zhang, P. A. Chou, Z. Liu, and C. Zhang. Requirements and recommendations for an enhanced meeting viewing experience. In *Proc. of the 16th ACM international conference on Multimedia*, pages 539 – 548, 2008.

[14] S. Minneman, S. Harrison, B. Janssen, G. Kurtenbach, T. Moran, I. Smith, and B. van Melle. A confederation of tools for capturing and accessing collaborative activity. In *Proc. of the third ACM international conference on Multimedia*, pages 523–534, 1995.

[15] K. Ntalianis, A. Doulamis, N. Tsapatsoulis, and N. Doulamis. Human action annotation, modeling and analysis based on implicit user interaction. *Multimedia Tools and Applications*, (online first), October 2009.

[16] M. Pimentel, L. A. Baldochi-Jr, and R. G. Cattelan. Prototyping Applications to Document Human Experiences. *IEEE Pervasive Computing*, 6(2):93–100, 2007.

[17] M. G. C. Pimentel, Y. Ishiguro, B. Kerimbaev, G. Abowd, and M. Guzdial. Supporting educational activities through dynamic web interfaces. *Interacting with Computers*, pages 353–374, February 2001.

[18] F. Sant'Anna, R. Cerqueira, and L. F. G. Soares. NCLua: objetos imperativos Lua na linguagem declarativa NCL. In *Proceedings of the 14th Brazilian Symposium on Multimedia and the Web*, pages 83–90, 2008.

[19] C. A. C. Teixeira, G. B. Freitas, and M. Pimentel. Distributed discrimination of media moments and media intervals: a watch-and-comment approach. In *Proceedings of the 2010 ACM Symposium on Applied Computing*, pages 1929–1935, 2010.

[20] K. N. Truong and G. R. Hayes. Ubiquitous computing for capture and access. *Found. Trends Hum.-Comput. Interact.*, 2(2):95–171, 2009.

[21] S. Tucker and S. Whittaker. Accessing Multimodal Meeting Data: Systems, Problems and Possibilities. In *Proc. of the Workshop on Machine Learning for Multimodal Interaction*, pages 1 – 11, 2004.

[22] S. Whittaker, S. Tucker, K. Swampillai, and R. Laban. Design and evaluation of systems to support interaction capture and retrieval. *Personal and Ubiquitous Computing*, 12(3):197–221, 2007.

[23] Z. Yu and Y. Nakamura. Smart meeting systems: A survey of state-of-the-art and open issues. *ACM Computing Surveys*, 42(2):1–20, 2010.

[24] Z. Yu, Z. Yu, H. Aoyama, M. Ozeki, and Y. Nakamura. Social interaction detection and browsing in meetings. In *Proc. of the 10th International Conference on Ubiquitous Computing*, pages 40–41, 2008.