Enhancing Media Enrichment by Semantic Extraction

Michael Krug
Department of Computer Science
Technische Universität Chemnitz
Chemnitz, Germany
michael.krug@informatik.tu-chemnitz.de

Fabian Wiedemann
Department of Computer Science
Technische Universität Chemnitz
Chemnitz, Germany
fabian.wiedemann@informatik.tu-chemnitz.de

Martin Gaedke
Department of Computer Science
Technische Universität Chemnitz
Chemnitz, Germany
martin.gaedke@informatik.tu-chemnitz.de

ABSTRACT
The opportunities of the Internet combined with new devices and technologies change the end users' habits in media consumption. While end users often search for related information to the currently watched TV show by themselves, we propose to improve this user experience by automatically enriching media using semantic extraction. In our recent work we focused on how to apply media enrichment to distributed screens. Based on the findings we made from our recent prototype we identify several problems and describe how we deal with them. We illustrate a way to achieve cross-platform real-time synchronization using several transport protocols. We propose the usage of sessions to handle multi-user, multi-screen scenarios and introduce techniques for new interaction and customization patterns. We extend our recent approach with the extraction of keywords from given subtitles by utilizing statistical algorithms and natural language processing technologies, which are then used to discover and display related content from the Web. The prototype presented in this paper reflects the improvements of our work. We discuss next research steps and define challenges for further research.

Categories and Subject Descriptors
C.0 [**General**]: System architectures; C.2.4 [**Computer-Communication Networks**]: Distributed Systems - *Client/server, distributed applications*; H.3.4 [**Information Storage and Retrieval**]: Systems and Software - *Distributed systems*

Keywords
Media enrichment, distributed display environment, mashups, semantic extraction, HTML5

1. INTRODUCTION
The advancement of the Internet combined with new devices and technologies change the end users' habits in media consumption. Nowadays end users often search for related information on their mobile devices using the Internet while watching a television show [9]. This additional information about the currently watched program can be, i.e. a list of actors, statistics related to a sportscast or social comments. Providing the users directly with related information instead of having them search for it on their phone or tablet is called media enrichment [6].

WWW'14 Companion, April 7–11, 2014, Seoul, Korea.
ACM 978-1-4503-2745-9/14/04.
http://dx.doi.org/10.1145/2567948.2577025

One way to discover related information is to use annotations. These annotations can be subtitles as well as metadata. Currently most annotation has to be done manually by a human. With the improvement of natural language processing tools the extraction of semantic keywords from given subtitles facilitates an automatic annotation process of videos with metadata. Furthermore, the automatic subtitling based on speech recognition done by video platforms, such as YouTube, enables the media enrichment of a wide range of videos. These two opportunities can be used to improve the process and quality of media enrichment.

The purpose of this paper is to demonstrate how media can be enriched with additional related content automatically. We base this demonstration on our previous work - a component-based framework for media enrichment [6]. This framework already utilizes the second screen approach to present information on multiple screens, such as TVs, smartphones, tablets or desktop PCs. Thus, we support scenarios like: users interacting with additional information on their tablet, while watching a video on a large screen like a TV. This paper presents the extension of our approach by semantic extraction and automatic captioning, as well as enhancements we achieved in interaction and customization to improve the user experience.

The rest of this paper is structured as follows: We discuss our previous work about media enrichment in Section 2. Our latest enhancements are presented in Section 3. In Section 4 we describe our planned demonstration. We discuss related work in Section 5. Finally we conclude the paper and provide an outlook to further developments.

2. SMARTCOMPOSITION APPROACH
In our previous work we created a prototype that demonstrates our approach for media enrichment by synchronizing a video with additional related content from the Web. This component-based prototype uses standard Web technologies and runs as a Web application in the browser. As described in our recent publications [6, 7] we use annotated video files to dynamically retrieve and display different kinds of information from various Web sources, i.e. maps, images, tweets or articles as well as Web services like translation or geocoding. Our approach consists of three components implemented as JavaScript classes: SmartScreen, MessageCenter and SmartTile. A SmartScreen is an abstract representation of a browser window. It is the host for multiple SmartTiles and contains a MessageCenter for communication purposes. The SmartScreen provides a mashup environment for SmartTiles, which are widgets that can process and display a certain kind of information. To distribute the events across our components we are using a modified publish/subscribe mechanism. We extended this prototype to support multiple screens by introducing a multi-device synchronization mechanism

that utilizes WebSockets, which is presented in [6]. During some testing we faced several problems like the lack of support of new HTML5 features, such as the TextTrack- [5] or WebSockets-API [11], in some browsers. Furthermore, the process of manually annotating videos requires a lot of effort and should be eased in some way.

In the following chapter we show how we solved these problems and describe new extensions and enhancements of our media enrichment prototype.

3. NEW CHALLENGES SOLVED

Based on our recent prototype we now present several enhancements that, on the one hand, will achieve cross-platform support for real-time synchronization using several transport protocols, on the other hand, will provide the users with an automatic enrichment process and new features for interaction and customization for a better user experience.

Unfortunately not all recent browsers support the TextTrack-API of the HTML5 video element; this includes i.e. Firefox and SmartTV browsers. Thus, our prototype was limited to a smaller number of browsers. Therefore, we now use our own parser written in JavaScript as a fallback solution for browsers without TextTrack-API support. This parser processes the WebVTT [12] files, which are used by the HTML5 video element for time-based information, like subtitles or metadata, and provides the same behaviour as the API. Using this fallback mechanism a larger number of browsers can be used for our approach.

Another problem we targeted was the synchronization technology. As mentioned before, we used WebSockets as the only communication protocol. This was another reason that our prototype was limited to a smaller number of user agents. To handle this issue we integrate a new synchronization component, which is written in JavaScript and uses the SockJS library [10]. This library provides a WebSocket-like object that gives us a coherent, cross-browser JavaScript API, which creates a low latency, full duplex, cross-domain communication channel between the browser and the web server. It automatically uses fallback transports like XHR streaming, XHR polling or iFrames. Since SockJS offers the same API on client-side as the WebSockets-API, we did not have to change our client-side code. Thus, all major browsers, including the ones that do not offer the WebSockets-API, are capable of synchronizing events.

The most significant enhancement of our approach is the integration of a SmartTile that extracts keywords from subtitles using text analysis to automatically enrich videos without manual annotation. By utilizing the AlchemyAPI, which "provides advanced cloud-based and on-premise text analysis infrastructure" [1], we are able to use the transcripts or subtitles of videos to provide the user with related information from the Web. In particularly we are using the entity extraction endpoint of the service, which "[...] is capable of identifying people, companies, organizations, cities, geographic features, and other typed entities within your HTML, text, or web-based content" [2]. The AlchemyAPI uses "[...] sophisticated statistical algorithms and natural language processing technology to analyze [...] information, extracting the semantic richness embedded within" [3]. The code in Listing 1 shows an example query to the Entity Extraction API and the corresponding result. We are using an extended form of publish/subscribe, where information gets published on topics. The semantic extraction SmartTile processes the results and generates a filtered output in the form of a publication with the entity as keyword and a matching topic. The topic is selected by mapping the type property to a predefined list of topics. Currently we are only using the types and names of the extracted entities to generate the output. Figure 1 displays a schematic overview of this process.

Since our framework is already capable of handling single keywords on different topics, we just map the entities, which are the result of the text analysis, to these topics. Using this approach, the existing SmartTiles can use the extracted keywords to present related information in form of images, maps, articles or tweets.

Query sent to the Entity Extraction API:

```
Endpoint: http://access.alchemyapi.com/calls/text/
TextGetRankedNamedEntities

Parameter:
text="Portugal now wants to apply for EU financial
aid."
```

Answer (shortened):

```
[...]
  "entities": [{
       "type": "Country",
       "relevance": "0.33",
       "count": "1",
       "text": "Portugal",
       "disambiguated": {
         "subType": [
           "Organization",
           "Location",
           "CompanyDivision",
           "GovernmentalJurisdiction",
           "Airline"
         ],
         "name": "Portugal",
         "geo": "38.7 -9.183333333333334",
         "website":
"http://www.portugal.gov.pt/Portal/EN/",
         "dbpedia":
"http://dbpedia.org/resource/Portugal",
[...]
```

Listing 1. Example query and result of the AlchemyAPI entity extraction API

As we target mostly English content but not all videos have English subtitles, we offer the option to chain a translation SmartTile in this operation to translate foreign subtitles to English ones, which then can be again be processed by the semantic extraction tile.

Another extension we made is the support for session management. In our recent prototype we faced the problem that all connected screens or devices shared the same workspace. This caused problems while testing as well as on real scenarios. Especially in multi-user, multi-screen scenarios there is a need for grouping screens and separating user groups, because users do not want to have all their devices in one workspace. Furthermore, it was not possible to enrich multiple videos at the same time without influencing the other ones. Therefore, we introduced a mechanism to divide groups of participants into sessions (distributed workspaces). A freely chosen string identifies a session. This session identifier can be changed by the user at any time and is represented in the URL of the application as the fragment identifier. This enables the users to easily change their session or join another one by changing the fragment identifier. For a comfortable way of adding smartphones and tablets to a session we provide a QR code that leads directly to the corresponding URL.

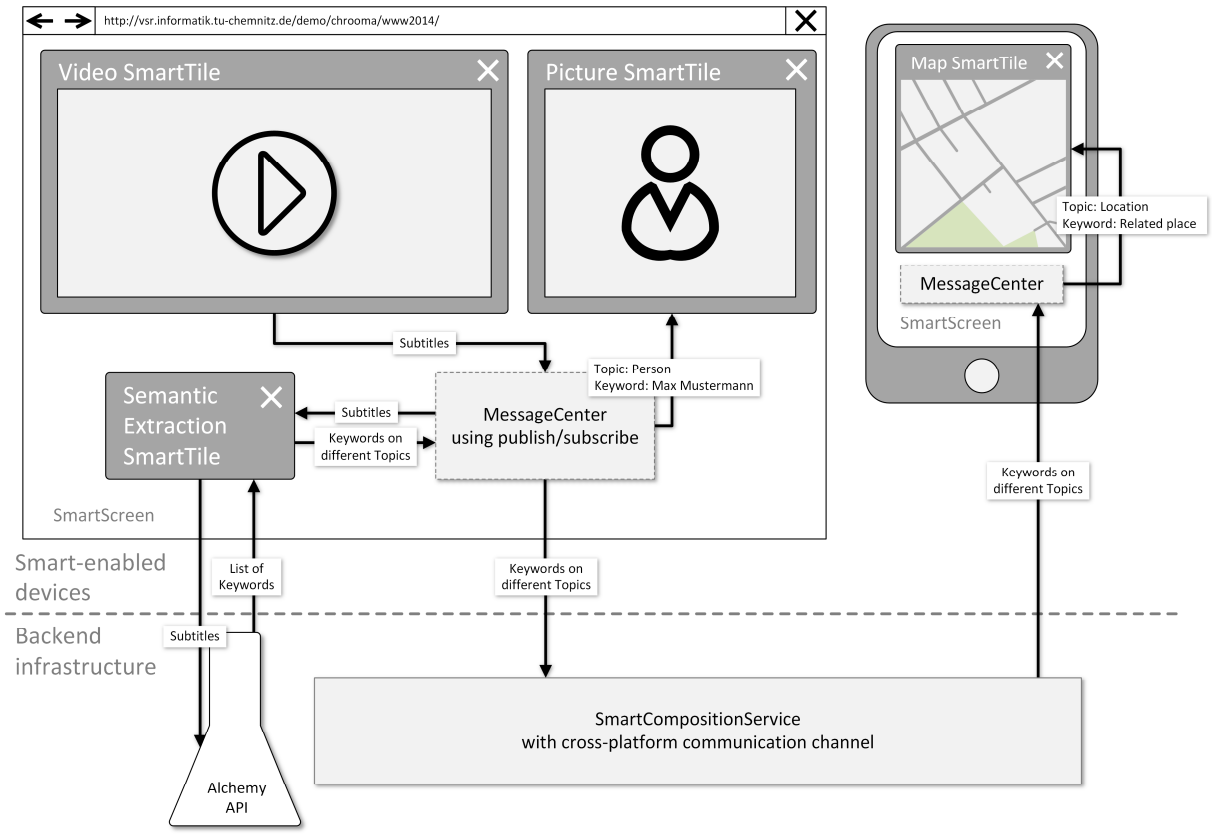

Figure 1. Media enrichment on multiple screens utilizing semantic extraction

Each participant is presented a list of screens that are active in the current session. This list is also used for the next enhancement we made: the exchange of SmartTiles between screens. Users are able to move SmartTiles from one screen to another by dragging and dropping a SmartTile on the name of the screen they want the tile to appear on. This operation removes the tile from the current screen and tries to add it on the target screen. If the target screen provides enough free space, the SmartTile will be added and the content and functionality of it will be restored. To give an example: a user has added a video and a map SmartTile on her large desktop PC screen and wants to interact with the map on her mobile device while watching the video on the large monitor. She can move the map tile to the connected smartphone by dropping it on the corresponding device name and the currently shown map will be available on the smartphone. Furthermore, the tile will receive following events from the video and will show new related information. The exchange is done using a special protocol message that is sent from one screen to the other and contains all necessary data – such as the type and information to display.

Using the new HTML5 Fullscreen API [4] we offer the users to maximize SmartTiles. This is particularly useful, i.e. if one screen is only showing a video or a small device, like a smartphone, is showing information like a map. The content of the SmartTile gets expanded to the whole screen, while hiding all window frames and controls.

To provide the users with more flexibility while creating their workspace, we made it possible to resize SmartTiles. With respect to a grid, the users cannot only move tiles to their desired location; they can also change the size of them by direct interaction. We support interaction with mouse as well as with touches on mobile devices. This functionality is achieved using the jQuery-UI library.

Another problem we dealt with was that users were unaware of the availability of annotations that are used for enrichment. To handle this, we provide custom controls for our video SmartTile. Those custom controls offer the same functionality as the standard controls but are extended with a visualization of annotation segments. Each segment is represented as a thin yellow line under the seeking bar. This enables the users to easily identify how many and how long additional information is available.

In the next chapter we describe the demonstration we will present.

4. DEMONSTRATION

Our demo presents an example workspace for automatically enrich media using semantic extraction. We show a video tile that plays a German newscast. The video has German transcriptions in the WebVTT format attached. Next to the video tile there are a few other kinds of SmartTiles that will present related information from the Web as well as some tiles that are used as intermediate components in the enrichment process. Since the video provides German subtitles and we currently support English text only, the subtitles get translated by a translate tile, which will then publish the text in the translated form. The English text is received and processed by the semantic extraction SmartTile. The results of the semantic extraction - keywords that are published on different topics - are displayed in different representations, as there are SmartTiles for displaying maps, images, tweets or articles. The

user can rearrange the workspace, add or remove SmartTiles and can join multiple screens to a shared session. Within the session the events are synchronized and the user can exchange SmartTiles between the screens. An architectural overview of this demonstration is depicted in Figure 1.

To show that and how our approach and demonstrator works with different videos and subtitles, we integrate a special YouTube connection. The user is able to add a video from YouTube using the URL or ID of the video. The ID is used to get the URLs of the actual video streams, which are added as the source attribute for the HTML5 video tag. Since we rely on given subtitles or annotation, the YouTube video has to have closed captions. Those videos can be found by setting a filter on the YouTube search page. Our approach uses WebVTT files to synchronize the annotations with the playing video. YouTube does not offer the subtitles in this format. Therefore, we provide a Web service that transforms the closed captions from YouTube into the standardized WebVTT format. It uses a simple grouping algorithm to join single text lines into larger blocks. The service is also capable of the automatic translation of the subtitles. Since the service is directly responding the WebVTT structure, we can use the URL of the service as the source attribute of the text track element in the video tag. After the video is added to the SmartScreen and the metadata is loaded, the semantic extraction process described above is performed.

Demonstration: The prototype presented in this paper is available for testing at:
http://vsr.informatik.tu-chemnitz.de/demo/chrooma/www14demo/

5. RELATED WORK

Closely related to enriching videos with additional content is the HTML5 media framework Popcorn.js [8]. This framework is "written in JavaScript for filmmakers, web developers, and anyone who wants to create time-based interactive media on the web" [8]. The framework allows one to attach time-based events to web videos and audio files. The events can trigger various actions, which are defined in plugins. For example you can display simple text, a map, images, tweets or information from Wikipedia. Most plugins display this information in a defined HTML container element. As mentioned before, adding new plugins, which can make use of other Web services, can extend this functionality. The framework also provides parsers for multiple subtitle formats. Thus, all those formats can be used to populate a subtitle track. Popcorn.js is a client-side only framework. There are no server-side components. The layout is fixed and cannot be modified by the end-user. While our approach enables enriching videos by displaying related content on multiple screens, Popcorn.js does not cover any multi-screen aspects as well as communication between plugins. Furthermore, there is no semantic extraction available.

6. LESSONS LEAREND AND OUTLOOK

The presented prototype enhances the approach of media enrichment by using semantic extraction. We extend our previous work to extract keywords from given subtitles using the AlchemyAPI. Our approach uses extracted named entities to enrich a given video with additional related content. We reduce the limitations of our recent work by implementing fallback mechanisms to support more platforms and browsers. We use our own WebVTT parser in JavaScript for browsers that do not yet support new HTML5 APIs. Furthermore, the usage of SockJS offers multiple fallback transports for the cross-screen communication if the user agent does not support the WebSockets API. New features for interaction and customization improve the user experience and enable multi-user, multi-screen scenarios.

Our future work will focus on how to present information on mobile devices, such as smartphones and tablets, with respect to their screen size and resolution to improve the usability. Especially the research of the human perception of information on mobile devices would be challenging. Furthermore, based on the semantic extraction of keywords and entities, we want to use Semantic Web technologies to discover more related and interlinked content. The AlchemyAPI already offers Linked Data for their extracted keywords and entities, which can be exploited.

7. ACKNOWLEDGMENTS

This work was supported by the Sächsische Aufbaubank within the European Social Fund in the Free State of Saxony, Germany (Project Chrooma+).

8. REFERENCES

[1] AlchemyAPI | Transforming text into knowledge: *http://www.alchemyapi.com/*. Accessed: 2013-10-08.

[2] Entity Extraction | AlchemyAPI: *http://www.alchemyapi.com/products/features/entity-extraction/*. Accessed: 2013-10-08.

[3] Entity Extraction API Documentation | AlchemyAPI: *http://www.alchemyapi.com/api/entity-extraction/*. Accessed: 2013-10-08.

[4] Fullscreen API: *http://fullscreen.spec.whatwg.org/*. Accessed: 2013-10-08.

[5] HTML Standard | The video element: *http://www.whatwg.org/specs/web-apps/current-work/multipage/the-video-element.html#timed-text-tracks*. Accessed: 2013-10-08.

[6] Krug, M. et al. 2013. Media Enrichment on Distributed Displays by Selective Information Presentation: A First Prototype. *Paper presented at the 5th International Workshop on Lightweight Integration on the Web (ComposableWeb 2013)* (2013).

[7] Oehme, P. et al. 2013. The Chrooma + Approach to Enrich Video Content using HTML5. *Proceedings of the 22nd international conference on World Wide Web companion (WWW '13 Companion)* (2013), 479–480.

[8] Popcorn.js | The HTML5 Media Framework: *http://popcornjs.org/*. Accessed: 2013-10-08.

[9] Smith, A. and Boyles, J. 2012. *The Rise of the "Connected Viewer."*

[10] SockJS | WebSocket emulation: *http://sockjs.org/*. Accessed: 2013-10-08.

[11] The WebSockets API: *http://w3.org/TR/websockets/*. Accessed: 2013-10-08.

[12] WebVTT: The Web Video Text Tracks Format: *http://dev.w3.org/html5/webvtt/*. Accessed: 2013-10-08.