D5.4 Complete set of final prototypes for professional and consumer media search environments, with interfaces, documentation and APIs

Due date of deliverable: 30.06.2009
Actual submission date: 03.09.2009

<table>
<thead>
<tr>
<th>D5.4</th>
<th>Complete set of final prototypes for professional and consumer media search environments, with interfaces, documentation and APIs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project Acronym :</td>
<td>SEMEDIA</td>
</tr>
<tr>
<td>Contract No :</td>
<td>FP6 – 045032</td>
</tr>
<tr>
<td>Due Date :</td>
<td>30.06.2008</td>
</tr>
<tr>
<td>Reply To:</td>
<td>Wolfgang Schmidt, <a href="mailto:schmidt@dvs.de">schmidt@dvs.de</a></td>
</tr>
<tr>
<td>Actual date of delivery:</td>
<td>03.09.2009</td>
</tr>
<tr>
<td>Project ref. no.</td>
<td>FP6-045032</td>
</tr>
<tr>
<td>--------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>Project acronym</td>
<td>SEMEDIA</td>
</tr>
<tr>
<td>Project full title</td>
<td>Search Environments For Media</td>
</tr>
<tr>
<td>Security (distribution level)</td>
<td>Public (PU)</td>
</tr>
<tr>
<td>Contractual date of delivery</td>
<td>Month 30, 30.06.2009</td>
</tr>
<tr>
<td>Actual date of delivery</td>
<td>Month 30+3, 03.09.2009</td>
</tr>
<tr>
<td>Deliverable number</td>
<td>D5.4</td>
</tr>
<tr>
<td>Deliverable name</td>
<td>Complete set of final prototypes for professional and consumer media search environments, with interfaces, documentation and APIs</td>
</tr>
<tr>
<td>Type</td>
<td>Demonstrator + Report</td>
</tr>
<tr>
<td>Status &amp; version</td>
<td>Final 1.0</td>
</tr>
<tr>
<td>Number of pages</td>
<td>37</td>
</tr>
<tr>
<td>WP / Task responsible</td>
<td>WP5 / DVS</td>
</tr>
<tr>
<td>Other contributors</td>
<td>Yahoo!Research, Yahoo!, Smoke&amp;Mirrors, CCMA, JRS, UG, FBM-UPF</td>
</tr>
<tr>
<td>Author(s)</td>
<td>Martin Halvey (UG), Gert Kienast (JRS), Lluis Garcia (Yahoo!), Irene Zeller Sancho (CCMA), Ernesto Arroyo (FBM-UPF), Wolfgang Schmidt (DVS)</td>
</tr>
<tr>
<td>Internal Reviewer</td>
<td></td>
</tr>
<tr>
<td>EC Project Officer</td>
<td>Antonio Puente Rodero</td>
</tr>
<tr>
<td>Abstract</td>
<td>This deliverable describes the final state of the main prototypes developed within SEMEDIA, six research prototypes and four integrated prototypes. Additionally the APIs and interfaces used for the integration of the different SEMEDIA tools are documented.</td>
</tr>
</tbody>
</table>
Executive Summary

Introduction

The SEMEDIA project’s research and development efforts have produced a set of tools that can be used in various video environments, such as audiovisual production, film industry postproduction and online communications media. These SEMEDIA tools have been integrated into multiple prototype applications. In this deliverable we describe the six major research applications, the four integrated applications and the interfaces and APIs that have been developed with the aim to integrate SEMEDIA tools into useful prototypes for the three SEMEDIA target domains broadcast, postproduction and Web.

Research Prototypes

The Group Interface (UG) is an attempt to overcome many of the problems associated with video search. We present a novel video retrieval system that allows users to create semantic groups to help conceptualise and organise their results for complex video search tasks. The interface allows users to organise their results into semantic groups and to share results between multiple groups, these groups can be used to launch new searches thus allowing users to view multiple result sets simultaneously.

The Video Browsing Tool (JRS) is an application for browsing in a video collection by clustering and filtering video segments based on content-based features. The central component of the user interface is a light table view which shows the current content set and cluster structure using a number of representative frames for each of the clusters. The clusters are visualised by coloured areas around the keyframes. A typical browsing session consists of repeated clustering and reducing content by selecting individual clusters.

TagExplorer (Yahoo!) is a tool for browsing Flickr photos using textual keywords -- a.k.a. tags. The TagExplorer helps the user browsing the collection by generating a cloud of words related to the user's query. The user can use the suggested words to further refine her query or to browse related topics. To make the tag clouds more functional and easy to use the TagExplorer organizes the tag cloud by grouping together words with similar meaning.

The VideoTagGame (Yahoo!) is a multi-player game for collecting time-based annotations over video. The players watch a video simultaneously and type keywords that describe the corresponding moment in the video. If the players agree on keywords in a small timeframe they receive points. Ultimately, the data collected by the game will enable the retrieval of relevant parts in a video, rather than returning the entire video as unit of retrieval.

PhotoSearch (Yahoo!) is an application for searching photos. It is a keyword-based search application that searches Flickr photos based on metadata information added by the photo owners. The main features of PhotoSearch are various tools for assisting the user in the search process. The tools allow the user to disambiguate ambiguous queries, expand queries with related terms, and to get metadata-similar images for an example image.

The Concept-based Video Explorer (FBM-UPF) is geared towards divergent content exploration. A web-based tool allows people to browse multimedia repositories by exploring the conceptual relationships that exist between them. Concepts and their relationships to an initial search concept are organized visually using a bubble
metaphor. The interface uses drag-n-drop to refine each query to obtain visual representations within a topic domain or for searching for specific imagery.

Integrated Prototypes

VideoSearch (Yahoo!) is a web-based prototype for searching video. It is currently applied to the broadcasting scenario, searching videos from CCMA. The main novelty of VideoSearch compared to state of the art video search tools is the ability to retrieve relevant fragments of videos as opposed to full videos. VideoSearch also provides search assistance tools such as query completion and query refinement suggestions.

Digimedia (CCMA) is a small and modified version of Digition, CCMA's own Media Asset Management, which has been transformed in order to be able to cope with SEMEDIA metadata, technologies and new user requirements. It is mainly used to show professional broadcast users the potential of SEMEDIA research, and to be able to gather their feedback in respect to the technologies. This feedback lets the research focus on real needs of the Broadcast scenario, one of the three real scenarios under SEMEDIA's scope.

SpycerWeb (DVS) is a Web Interface for DVS's content management software Spycer(TM) that is capable of searching for a company's media files through the network or the internet. The user can view and edit content metadata and preview video files via an integrated streaming player. SpycerWeb is a Flex application that runs in a web browser with a Flash Player plugged in and is geared towards film and video postproduction environments.

CMD (Smoke&Mirrors) is web browser based allowing user access from any location. It is designed for S&M's in-house use, as well as to be used by external clients. S&M will use it as a direct replacement for Cakes managing media assets, but its use will also be extended into a full workflow tool – planning and scheduling work to be done, recording work that has been done, allowing clients to approve work remotely, flowing through to despatch and invoicing.

Interfaces and APIs

For efficient handling of metadata in all stages of a media workflow tools need to be available that allow easy read/write access to the metadata. SEMEDIA has chosen the ISO Standard MPEG-7 as its metadata format. To enable application developers to access the metadata without needing to have a deep understanding of all the details of MPEG-7 the MPEG-7 Utility library has been used and extended.

The SEMEDIA Framework is based on the low-level process of the Spycer software and encapsulates a set of tools from other SEMEDIA partners that generate metadata in multiple ways. Those tools are loosely “plugged” into the framework using an open-standard web services architecture. The Framework itself is also accessible via this open approach using SOAP methods. Therefore the Framework on the one hand acts as a “client” using the video processing “server” services internally and on the other hand it is also a “server” for 3rd party “client” applications that access the framework’s functionalities.

Conclusion

During the lifetime of the project the SEMEDIA partners developed ten prototypes, each with a slightly different emphasis.
The six research prototype applications represent individual developments by single partners that were started or enhanced during SEMEDIA's activity with the main goal to create test environments for the tools. These could already be used in all phases of tool development, but also for some of the final evaluation tests.

The four integrated applications combine SEMEDIA technologies from different partners with real-world applications for specific use cases from the target domains broadcast, postproduction and Web. At current state some of these applications are already publicly deployed like the Video Search tool from Yahoo! or even in daily production use as the CMD asset management tool from Smoke&Mirrors.

The use of standards like MPEG-7 and a framework with a plug-in like architecture, connected via state of the art Web service technology, provides a simple connection to third-party products and allows the use of SEMEDIA tools in a wide range of search environments and systems.
Contents

Executive Summary ........................................................................................................ 3
1 Introduction .......................................................................................................... 7
2 Research Prototypes ....................................................................................... 7
  2.1 Group Interface (UG) ................................................................................ 7
  2.2 Video Browsing Tool (JRS) ....................................................................... 8
  2.3 TagExplorer (Yahoo!) ............................................................................. 10
  2.4 VideoTagGame (Yahoo!) ....................................................................... 12
  2.5 PhotoSearch (Yahoo!) .......................................................................... 14
  2.6 Concept-based Video Explorer (FBM-UPF) ......................................... 15
3 Integrated Prototypes ..................................................................................... 18
  3.1 Video Search (Yahoo!) ........................................................................... 18
  3.2 Digimedia (CCMA) ............................................................................... 20
  3.3 SpycerWeb (DVS) .................................................................................. 22
  3.4 CMD (Smoke&Mirrors) ......................................................................... 23
4 The SEMEDIA Web Demonstrator .......................................................... 25
5 Interfaces and APIs ....................................................................................... 26
  5.1 MPEG-7 Utility Library ......................................................................... 26
     5.1.1 Creating MPEG-7 Documents ...................................................... 26
     5.1.2 Loading, Saving and Closing MPEG-7 Documents ...................... 26
     5.1.3 Setting and Getting Metadata .................................................... 27
     5.1.4 Time-Based Metadata .................................................................. 27
     5.1.5 Getting Shot and Keyframe Information ...................................... 27
     5.1.6 Getting Stripe Image Information ............................................... 28
     5.1.7 Getting Camera Motion Information ........................................... 28
     5.1.8 Getting Visual Activity Information ........................................... 28
  5.2 The SEMEDIA Framework ..................................................................... 28
     5.2.1 Video Processing SOAP Server .................................................... 29
     5.2.2 Framework Access SOAP Server ............................................... 32
     5.2.3 Framework Indexing vs. Base System Indexing ........................... 34
     5.2.4 Implementation of JRS’ Mpeg7UtilLib ......................................... 35
     5.2.5 General Workflow of the SEMEDIA Framework ....................... 35
     5.2.6 The Framework’s search functionality ....................................... 37
6 Conclusion ........................................................................................................ 37
1 Introduction

The prototypes of the SEMEDA project have been planned for the purpose of producing useful and working technology, to highlight research results, to demonstrate integration and especially to perform tests under realistic conditions in real-live use cases.

By the end of the project the SEMEDIA research and development efforts have produced a set of tools for search and retrieval of media files that can be used in a wide range of applications, such as audiovisual production, film industry postproduction and online communications media. These SEMEDIA tools have been integrated into multiple prototype applications. In the first part of this document we describe the six major research applications and the four integrated applications that have been produced with the aim to integrate SEMEDIA tools into useful prototypes for the three SEMEDIA target domains broadcast, postproduction and Web.

The integration of the SEMEDIA tools is based on the MPEG-7 standard for metadata management and on a framework with a plug-in like architecture and state of the art Web service technology. This framework together with its interfaces and APIs is described in the second part of the deliverable.

The prototype development started already in the first project year since the goal was to get feedback from prospective end-users at an early stage and to use this feedback to further improve the tools and to adjust the directions of further research and development. The SEMEDIA workshops together with individual internal evaluation activities of the partners provided this feedback and led to a 2 ½-cycle process in which the first cycle produced initial early prototypes at T15 and the second delivered more significant ones at T27. The last half-cycle was used for some final refinements in the last three months.

2 Research Prototypes

2.1 Group Interface (UG)

ViGOR is an attempt to overcome many of the problems associated with video search. ViGOR is a novel video retrieval system that allows users to create semantic groups to help conceptualise and organise their results for complex video search tasks. The interface allows users to organise their results into semantic groups and to share results between multiple groups. These groups can be used to launch new searches thus allowing users to view multiple result sets simultaneously. There are a number of potential advantages to using ViGOR. First, interactive grouping is a flexible means to communicate numerous information needs. Second, the semantic gap is narrowed by the abstraction to high-level semantic groupings. Finally, the user leaves trails of their actions behind that not only the system can exploit for adaptation but also that other people can trace.

ViGOR (see Figure 1) is comprised of a search panel (A), results display area (B), workspace (C) and playback panel (D). These facilities enable the user to both search and organise results effectively. The users enter a text based query in the search panel to begin their search. The result panel is where users can view the search results (a). Additional information about each video shot can be easily retrieved by placing the mouse cursor over a video keyframe for longer than 1.5 seconds, which will result in any text associated with that video being displayed to the user (we will henceforth refer to this action as tooltip) (e). If a user clicks on the play button the highlighted video shot will play in the playback panel. Users can play, pause, stop and navigate through the
video as they can on a normal media player (D). The main component of ViGOR is the provision of a workspace (C). Groups can be created by clicking on the create group button. Users must then select a textual label for the group and can potentially add any number of annotations to the group, but each group must have at least one annotation.

![Figure 1: Screen shot of ViGOR interface for YouTube.](image)

Drag-and-drop techniques allow the user to drag videos into a group or reposition the group in the workspace (b). Groups can be deleted, minimised and moved around the workspace using a number of buttons (f). It should be noted that any video can belong to multiple groups simultaneously. The workspace is designed to accommodate a large number of groups. Each group can also be used as a starting point for further search queries. Users can select particular videos and can choose to view an expansion of the group that contains similar videos based a number of different features (c, d). The description above describes the basic functionality of ViGOR; two slightly different versions of ViGOR were used for development on two different datasets - YouTube and TRECVID. The YouTube version offers three expansion options for each group (see Figure 1 (d)): 1) related videos; 2) videos from the same user 3) and text expansion which is the result of a new search using text extracted from the selected videos. The TRECVID interface offers three expansion options for each group (see Figure 1 (d)): 1) similar colour; 2) similar shapes, this was retrieved using edge histograms 3) and similar homogenous texture.

### 2.2 Video Browsing Tool (JRS)

The Video Browsing Tool is an application for browsing in a video collection by clustering and filtering video segments by content based features. The features include camera motion, visual activity, colour layout, settings and retakes.

Figure 2 shows the user interface of the tool which consists of four main parts:

- The central component is a **light table view** (red) which shows the current content set and cluster structure using a number of representative frames for each of the clusters.
The clusters are visualised by coloured areas around the keyframes, with the cluster label written above the first two images of the cluster.

The size of the displayed keyframes in the light table view can be changed with the slider at the bottom of the window. The user can choose between the level of detail and the number of keyframes visible without scrolling.

A cluster can be selected by clicking on an arbitrary keyframe in this cluster. The light table view allows selection of multiple clusters by holding down the Ctrl-key when clicking on the keyframes.

On the left of the application window the history (yellow) and the result set (green) are displayed. The history of the user interactions with the light table view (clustering, reducing selection, show all segments) is visualized in the history window on the left side of the window. At any time the user can drag relevant representative frames into the result set, thus adding the corresponding segment of the content to the result set.

![Image](image.png)

**Figure 2: The user interface of the Video Browsing Tool**

The browsing toolbar (blue) provides options for interacting with the dataset. The main interaction steps are clustering filtering (reducing) the results.

**Clustering:**

Clustering is a way of grouping items together which are similar in a certain way. Video clips can be clustered (grouped) according to different (mostly visual) features. These features are:
• **Camera motion:** Clusters are derived from the dominant camera motions (pan left/right, zoom in/out, tilt up/down, roll clockwise/anticlockwise and static camera).

• **Camera motion advanced:** Similar to “Camera motion” but the strength of the camera motion is also considered. Leads to a higher number of clusters than “Camera motion”.

• **Motion activity:** The amount of motion activity in the video. Clusters range from “low activity” (static shots) to “high activity” (e.g. action scenes).

• **Colour layout:** Keyframes with similar colour layout are grouped in the clusters.

• **Faces:** This leads to two clusters. One cluster with the keyframes where faces are shown and another cluster without visible faces. This can be used to find material with close-ups of people (e.g. interview scenes).

• **Settings:** Groups the keyframes according to the location (setting) where they have been recorded. Keyframes with a similar background fall in one cluster.

• **Retakes:** When shooting raw footage some scenes have to be recorded more than once. This option groups all takes for the same scene recorded with the same camera in one cluster.

• **Media Item:** Keyframes are grouped by the video they are from.

After performing a clustering the keyframes are arranged accordingly. Each cluster is highlighted with a different background colour. Above the first two keyframes of each cluster a label describing the cluster is displayed.

The history of the user interactions with the light table view (clustering, reducing selection, show all segments) is visualized in the history window on the left side of the window. Any relevant keyframes can be added to the result list at any time by dragging them from the light table view to the result list in the lower left part of the window. The results can be exported as an Edit Decision List for reuse in other applications.

### 2.3 TagExplorer (Yahoo!)

TagExplorer is a tool for browsing Flickr photos using textual keywords -- a.k.a. tags. The TagExplorer helps the user browsing the collection by generating a cloud of words related to the user's query. The user can use the suggested words to further refine her query or to browse related topics. To make the tag clouds more functional and easy to use the TagExplorer organizes the tag cloud by grouping together words with similar meaning.

The TagExplorer is a Web based application written in PHP. Figure 3 shows a screenshot of the system. The system UI has 6 main parts:

A. A query input box where the user can post queries

B. A query action box where the current query is shown and the user can remove query terms (by clicking the x sign).

C. A cloud of terms where the user can choose query terms to add to her query (clicking the + sign) or post a new query using the term (clicking the term itself)

D. Help on how to use the system

E. Photo results – top relevant photos for the query
F. Photo details panel where the user can view details of a particular photo

Figure 3 - TagExplorer browsing interface

Figure 4 - TagExplorer components

Given the tag-cloud of related tags the user can continue her browsing by choosing to add to her query a term from the tag cloud (click on the respective green plus) or start a new query based on one of the tags in the cloud (click on the tag). Given the new query
a new tag-cloud is generated and the photo list is updated. For example, suppose the 
user chooses to add the tag Thames to her London query. The tag will now show 
different tag-cloud relevant to the new query, including the tags Tower Bridge, bridge, 
river, South Bank, etc. Now the user can again choose to add tags to her query or 
browse other tags. This process can then -- in principle -- continue forever.

2.4 VideoTagGame (Yahoo!)

Annotating web-pages, photos, and now also short videos is common practice for 
those that share their content on social media sites, such as Delicious, Flickr, or 
Yahoo! Video. The massive scale at which Yahoo!'s users are annotating media is 
impressive and enables effective retrieval of web-pages, images, and video at large, 
which was believed to be out of scope for a long time.

![VideoTagGame](image)

**Figure 5 - VideoTagGame, game screen**

With the introduction of the VideoTagGame we try to push the boundaries at bit further. 
The objective of the VideoTagGame is to collect time-based annotations over video in
the context of a multi-player game. Ultimately, this will enable the retrieval of relevant parts in a video, rather than returning the entire video as unit of retrieval.

To play the game, simply sign in with your Yahoo! ID, and read the how-to-play instructions, before joining a new game. You will be synchronised with other players before the game starts. There will always be at least three players in a game. A three second countdown signals the start of the game, after which the video starts playing immediately. Make sure that you are ready to enter the tags (keywords) as soon as the video starts playing. To submit a tag, you need to hit enter after each tag. If two players agree on a tag, each player receives points. The points that are awarded are proportional to the time difference between the two players submitting their tags; the shorter the time difference, the more points are rewarded. But you'll get no time to relax, as the video keeps streaming and new tags need to be entered as quickly as possible, which can be quite a challenge. One last hint, entering the same tag within a few seconds will not reward you additional points, in fact, another player in the game might benefit, and receive more points for his tag, due to a smaller time difference.

![Videotaggame screenshot](image)

Time Remaining: 0:56
Score: 0

<table>
<thead>
<tr>
<th>Player</th>
<th>Tags</th>
</tr>
</thead>
<tbody>
<tr>
<td>242</td>
<td>robcap 27</td>
</tr>
<tr>
<td>249</td>
<td>clips 25</td>
</tr>
<tr>
<td>292</td>
<td>deepblue 25</td>
</tr>
<tr>
<td>0</td>
<td>Isaac 0</td>
</tr>
</tbody>
</table>

Figure 6 - Videotaggame, tags used by users are shown at the game, synchronized with the video
Once the video has finished playing, you'll automatically go to the next screen, and you can sit back and relax while watching the video again. This time with the tags that were entered during the game are projected over the video. Hint, if you click on the video, the tag stream will split in multiple columns, so that you can see who entered which tag.

1. Front page of the VideoTagGame
2. Before the game starts, there is a synchronization phase with other players
3. Tagging the video during the game.
4. After the game, the tag-density bar is shown below the video.

In summer 2007 we ran a first experiment with the VideoTagGame during the Yahoo! party at SIGIR in Amsterdam. Using a game room set-up, we had 57 participants playing in total 26 games with on average 5 players per game during a 3 hour time window. Each game lasted 2 minutes and at the end of the event a total of 5890 tags had been entered. In 57% of the cases players agreed on a tag. As part of our SEMEDIA activities, we have now made the VideoTagGame publicly available. This will allows us to collect time-based video annotations for a larger set of videos and tapping into Yahoo!'s extensive user base.

The VideoTagGame runs on top of a Red5 server, which is responsible for the communication with the players in the game. Once the Flex client is loaded in the browser, a player can choose to join a game. This will start a synchronisation process with other players that want to play, and a new video is selected from the pool of videos. In this experiment we'll only show movie trailers. A three second countdown starts as soon as all players are ready to go. During the game each tag is time-stamped and communicated with the server, which controls the scoreboard and notifies the clients of updates. After the end of the game you can relax and watch the video again to see what tags have been entered by you and the other players in the game.

2.5 PhotoSearch (Yahoo!)

PhotoSearch is an application for searching photos. It is a keyword-based search application that searches Flickr photos based on metadata information added by the photo owners. The main features of PhotoSearch are various tools for assisting the user in the search process. The tools allow the user to disambiguate ambiguous queries, expand queries with related terms, and to get metadata-similar images for an example image.

The PhotoSearch application has the following main features:

- Query disambiguation: Given an ambiguous query -- e.g., Washington -- PhotoSearch will suggest disambiguation options -- e.g., D.C. or Seattle.
- Query refinement: Given a query -- e.g., Barcelona -- PhotoSearch will suggest query refinement terms for either making the query more general -- e.g. Spain or Catalonia -- or more specific -- e.g., Sagrada Familia or Barcelona architecture.
- In-context exploration of search results: For the query refinement/disambiguation terms, PhotoSearch helps the user to visualise the effect of the refinement by highlighting the search results that match the refinement criteria.
- Query by example: Given a query and a photo: PhotoSearch will suggest a query for retrieving similar photos.
The search functionality of PhotoSearch is powered by the Flickr API.

Figure 7 - Photosearch, improved version

Given the tag-cloud of related tags the user can continue her browsing by choosing to add to her query a term from the tag cloud (click on the respective green plus) or start a new query based on one of the tags in the cloud (click on the tag). Given the new query a new tag-cloud is generated and the photo list is updated. For example, suppose the user chooses to add the tag Thames to her London query. The tag will now show different tag-cloud relevant to the new query, including the tags Tower Bridge, bridge, river, South Bank, etc. Now the user can again choose to add tags to her query or browse other tags. This process can then -- in principle -- continue forever.

2.6 Concept-based Video Explorer (FBM-UPF)

Marketing and post-production agencies have seen a large increase in the creativity levels required in order to succeed in today's market environment. Nevertheless, the creative process has not changed, requiring a lot of human effort in order to stay competitive. Furthermore, people and the processes are the most important assets in modern enterprises [Kao, J, 1996]. That is, there is a growing need to improve existing procedures and tools to support the creative process, ideation, and communication.

Despite new advances in information visualization, existing tools do not consider the needs from the post-production and broadcasting environments. That is, there are no tools that allow people to easily communicate ideas or concepts between each other. For example, in broadcasting environments, journalist might ask expert media achievers to find and retrieve content that expresses "sorrow and devastation from an earthquake"1. They have a clear idea (in their minds) about what they need, but they fail to express it in words. In post-production environments, marketing campaigns are

1 Observations from BBC Workshop conducted on February 2009
defined based on a briefing document and a short briefing session. The ideas are expressed by a few paragraphs and in the best cases, using a slideshow presentation.

The concept-based video explorer project is an attempt to solve the communication problem by providing a visual common language for communicating ideas. This language is based on imagery and conceptual associations that represent an idea space. The end goal is to support journalists and marketing professionals' ideation process by providing tools for media access and exploration. The project exploits advances in data mining and indexing in order to create a repository of conceptually related media assets based on their metadata.

Descriptive representation

In marketing and post-production environments it is common to use images and videos to communicate a message effectively. Multimedia content has several connotations associated to it and can be used as an inspiration source. Conceptual meaning can be literal or associative. This conceptual meaning is automatically extracted from metadata and labels into a textual description. Extracted information includes time, place, visual elements, objects, and conceptual relationships [concepNet][WordNet].

Interface

Marketing and post-production professionals informed a novel interface design and provided feedback for its development. This application is geared towards divergent content exploration. A web-based tool allows people to browse multimedia repositories by exploring the conceptual relationships that exist between them. Concepts and their relationships to an initial search concept are organized visually using a bubble
metaphor. Each concept has imagery associated with it, while selecting a concept shows its relationships with other, as well as the type of relationship based on marketing templates. The interface uses drag-n-drop to refine each query to obtain visual representations within a topic domain or for searching for specific imagery.

Figure 9 – Visual representation for concepts related to “romance”
3 Integrated Prototypes

3.1 Video Search (Yahoo!)

Videosearch is a new interface to search for videos in the Web. It uses textual and video data from the SEMEDIA partner CCMA. The novelty of the project is to present a video search engine capable to search inside videos, rather than looking in the general title or description.

To do that video the needs to be partitioned in segments. Professional users at CCMA annotate videos daily at a segment level, which means that they associate textual information with starting and ending time that tries to explain what is happening in the scenes. This provides information to know the content of that segment with no need of watching it.

With VideoSearch this data has been indexed and retrieval models are used to get a ranked list of videos given a query. The ranked list takes into account both complete videos (using global video fields) and segments of it (using annotated data at a segment level). A segment of video will be relevant to a certain query if the text that describes the segment contains the query terms.

Besides the retrieval of these parts, segments are shown to users in several ways.

Figure 10 - VideoSearch results page
VideoSearch presents the ranked list and, for each one of the results, it shows all the segments for that video and remarks the relevant ones. Users can hereby see which part of the video is relevant, and jump directly to that part with no need to watch the previous non-relevant scenes.

Regarding to the interface, VideoSearch shows many frames for each video. The frame selection is done using visual difference. Users don’t need to see repeated frames and can jump between scenes easily and fast.

Besides the result list, a single result can be shown in a more detailed way with the full screen feature. This new perspective displays all meta data, all segment text and frames related to these segments and a visual table of contents in the bottom right corner, showing as much frames as possible giving priority for those that have relevant information.

Figure 11 - VideoSearch result detail screen

The second version of the prototype was deployed publicly on December 2008 and evaluated by professional users at CCMA. After the evaluation, several improvements were carried out as a result of the feedback and usability test scores obtained. The modifications performed to the application are summarized in the following table:
### Table 1 - VideoSearch improvements after evaluation

<table>
<thead>
<tr>
<th>Improvement</th>
<th>Before</th>
<th>After</th>
</tr>
</thead>
<tbody>
<tr>
<td>Underline query terms</td>
<td>Texts in the results shown with no extra decoration</td>
<td>Query terms displayed with in bold, underscore and blue color in the search results</td>
</tr>
<tr>
<td>Visual Table of Contents</td>
<td>Only showing frames, located according to time distance</td>
<td>Segments included, relevant segments have priority, then non relevant segments and, finally, frames. For the segments, the description is shown as tooltip, for the relevant ones a green lines substitutes the default red line</td>
</tr>
<tr>
<td>Video Representative</td>
<td>The frame representing the video is the 1st frame</td>
<td>The frame representing the video is the one corresponding to the first relevant segment. Otherwise the 1st frame is used</td>
</tr>
<tr>
<td>Field indexing</td>
<td>4 fields indexed</td>
<td>14 fields indexed</td>
</tr>
<tr>
<td>Advanced Search</td>
<td>No advanced search available</td>
<td>Advanced search using 14 fields plus 2 date fields available. Date fields used in a post retrieval step</td>
</tr>
<tr>
<td>Pagination</td>
<td>No pagination available. Only 10 results per query</td>
<td>Pagination available, all results available in groups of 10</td>
</tr>
</tbody>
</table>

Among all suggestions and comments gathered during the evaluation, those with more user impact and reasonable technical effort were chosen to be applied. The modifications done can be checked in the online version available from the SEMEDIA Demonstrator web site.

#### 3.2 Digimedia (CCMA)

Digimedia is CCMA’s broadcast prototype and it has been developed as an enhancement of Digition, CCMA’s Media Asset Management system.

Media asset management systems are computer software and hardware systems that aid in the process of digital asset management mainly for audio and video content. It also refers to the protocol for downloading, renaming, backing up, archiving, optimizing, maintaining and exporting files. The management and retrieval of content are quite complicated because of the huge amount of digital content that the management systems usually contain.

The broadcast prototype of CCMA has been improved with the integration of tools developed in the SEMEDIA project. These new tools give users new options to carry out their searches and other actions of their daily work more efficiently and also make the management of content easier.
The selection of these integrated technologies was carried out following the trend revealed in the initial user case document (D2.1), whose aim was to identify the requirements of the users, and the tools that would possibly help to fulfill users’ needs and improve their daily work.

Two different types of tools have been integrated in Digimedia: text and video tools. We used different approaches for their integration. Specifically, the different integration approaches are the following:

- The integration of the text tools was originally and continues to be focused on a REST protocol approach. It is a simple and fast way to connect the technologies to the broadcast prototype.
- In contrast, the way in which we connected the video tools to the prototype has changed as the project progressed. In the first stage of integration, the SEMEDIA framework, developed by DVS, was used to connect the technologies to Digimedia. Other requirements emerged during the process of integration (the assets workflow was not as fast as desired). As a consequence, a specific framework was designed in order to connect Digimedia to the video tools. This last approach uses the SOAP protocol, and directly connects the video module of JRS to Digimedia contents. It is installed and currently running at CCMA’s facilities.

The integration design has been based on the user requirements. The front-end interface that shows the technologies to the end users has been developed on a co-design basis. This interface has let Digimedia users interact with the integrated SEMEDIA tools, and has let them check if the technologies could ease their daily work. Consequently CCMA developers have put a lot of effort into verifying that the way they have integrated the tools is clear and intuitive for the final users.

The tools integrated in CCMA’s prototype and their main features are the following:

- **Stripe Images**: only one image summarizes the whole video. They let users clearly identify the shot boundaries and the different sequences in a video. It is a new way of navigating through the video.
- **The Tag Suggestion Tool**: this tool suggests some words when a user carries out a search in Digimedia. These words allow users to sharpen the search and to retrieve better contents.
- **Key Frames**: it shows the main static frames that summarize a video. A new frame is generated at every shot boundary; the more motion there is, the more frames are generated. It gives users a quick idea of the video, and also eases the navigation through the video.
- **Camera motion**: the system automatically identifies the camera motion of the video. So the user can use the camera motion information to sharpen the searches of sequences, and retrieve them by the type of camera motion. This is something they could not do before unless camera motion had been manually annotated.
- **Motion activity**: the aim of this technology is to give information about the amount of movement that a video contains. This information is showed in Digimedia as a data chart, where a user could see the amount of motion of the video related to time code of the video.
- **Color Layout**: this technology gives information about the predominant colour of the videos. Digimedia shows the main colours of the different key frames of the videos and the users may search videos by primary colours.
- **Ranking**: It is another way to present the results of the searches in the system. It shows the results by importance and appearance probability. The search system of Digimedia shows the results ordered by creation date.
- **Xtreme News**: This tool has been designed to help journalists. It is not directly integrated in Digimedia, although it is linked to it. Its aim is simple but powerful:
thanks to Xtreme News, journalists will be able to look up external information related to an asset, i.e., information from the web, not from Digition. The users will double-click on an asset, and the Xtreme News interface will be opened with information that is related to the title of the asset, and that has been obtained from a selected range of web pages (Reuters, APTN, Vilaweb, El Periódico, etc).

The deliverable 5.1.2 “Second version of Prototypes of modular tools in Broadcast Environments” explains in more detail this prototype and the different integrated tools. It also deals with the stages of CCMA tools integration. It includes the main steps the users follow to use these tools, and highlights how Digition has been enhanced by the SEMEDIA technologies.

![Figure 12: Digimedia Screenshots](image)

### 3.3 SpycerWeb (DVS)

SpycerWeb is a Web Interface for DVS's content management software Spycer(TM) that is capable of searching for a company's media files through the network or the internet.

DVS Spycer is a distributed content management application, which provides a solution for dealing with large amounts of video data and its accompanying metadata. It has been designed as an open platform that supports existing heterogeneous IT infrastructures and workflows in typical postproduction houses.

With SpycerWeb the user can view and edit content metadata and preview video files via an integrated streaming player. It is a Flex application that runs in a web browser with a Flash Player plugged in and is geared towards film and video postproduction environments.

The SpycerWeb is a platform- and location-independent user interface that makes use of the SEMEDIA Framework (an enhanced Spycer installation sharing some functionality through SOAP) for which it also acts as a reference implementation. The main functionalities of the SpycerWeb are:

- Simple and extended textual search for media files
- Viewing and editing of metadata (textual metadata, camera motion, keyframes and stripe images)
- Previewing of streaming video files
Figure 13: The SpycerWeb GUI with metadata details, Video player, search dialog and results list

The following SEMEDIA tools are integrated: shot boundary detection, key frames, stripe images, camera motion and visual activity.

For searching, the user can choose between a simple or extended search mechanism. The search results are shown in a table or image view. By clicking on a result item, the user can access related metadata in form of textual metadata (file and production metadata) or image representations like keyframes and stripe images. In the timeline of the video player one can also find segments of camera motion (camera pan, camera tilt) and visual activity. Some metadata called “production metadata” can be edited and searched afterwards. One can also search for camera motion and visual activity as well as navigate intuitively by clicking on the keyframes.

3.4 CMD (Smoke&Mirrors)

Over the past two years, the Smoke and Mirrors software development team have been busy developing an updated version of Cakes, S&M’s in-house asset management system. The new system has been re-branded as CMD, which stands for Create Manage Deliver.

As CMD is web browser based it allows user access from any location. It is designed for S&M’s in-house use, as well as to be used by clients. S&M use it as a direct replacement for the predecessor system Cakes managing their media assets, but its use has also been extended into a full workflow tool – planning and scheduling work to be done, recording work that has been done, allowing our clients to approve work remotely, flowing through to despatch and invoicing.
There is multi-level access, such that in-house, S&M are able to view all media, but clients can only view their own assets – thus maintaining confidentiality for, what can be, very sensitive material.

The following SEMEDIA tools have been implemented: extraction of technical metadata, shot boundary detection, key-frame extraction, stripe image generation, camera motion extraction, visual activity detection.

[Image: An asset browser interface showing keyframe and stripe image layout in CMD asset browser]

**Figure 14: Keyframe / stripe image layout in CMD asset browser**

Under the Cakes system all meta-data was manually input leaving scope for human error. The algorithms developed by JRS automatically extract meta-data from each clip and self-populates the related information fields. Whilst file name is still a manual input, clip length, file format, creation date, aspect ratio, timecodes, frame rates and television standards are now automatic leading to huge time savings and greatly minimising the effect of human error on input and retrieval mistakes.

With increased content and accuracy of the database, the search process has become faster and more efficient. This has been achieved both within the visual image based environment and for text based content. The use of key frame or storyboards and stripe image viewing has enabled users to immediately see the content of the media, without the need to view the whole clip.

A group of 25 people was involved in the evaluation of the CMD system. In summary the CMD system has been well received by both clients and staff. The visual displays have been well received and the ordering process has been greatly streamlined.
4 The SEMEDIA Web Demonstrator

The SEMEDIA Web demonstrator aims at showing the tools and applications in an interactive way. It provides video captures and additionally lets users play around with the applications preferably in real-time. More information is given in the form of continuous text or links to documents (PDFs, http links, etc.). This demonstrator has become an integral part of the SEMEDIA web site where there are now two web demonstrations available:

- an enhancement to the semedia.org website in form of a new section called “showcase”
- an interactive stand-alone demonstrator using Adobe Flash that also runs in the web browser

The reason for having these two approaches is to fulfill two different user requirements:

- on the one hand an interactive and impressive experience with Flash
- on the other hand an ordinary text-oriented information with HTML with improved usability navigation

![Figure 15: Start page of the SEMEDIA showcase](image-url)
5 Interfaces and APIs

5.1 MPEG-7 Utility Library

For efficient handling of metadata in all stages of a media workflow tools need to be available that allow easy read/write access to the metadata. SEMEDIA has chosen the ISO Standard MPEG-7 as its metadata format. To enable application developers to access the metadata without needing to have a deep understanding of all the details of MPEG-7 the MPEG-7 Utility library has been described.

This library provides a utility class. Its functionalities are described in the following sections:

5.1.1 Creating MPEG-7 Documents

If no document exists for the clip, a new document will be created. The information that needs to be passed to the function is:

- Clip path and name
- Clip UUID
- Clip start/end time code (optional)

5.1.2 Loading, Saving and Closing MPEG-7 Documents

The load method takes the file name of the document as argument, parses the files and performs the necessary initializations of internal data structures.
The save method serializes the MPEG-7 document and writes it to a file. If a filename is passed, the document is serialized to this file. If no filename is passed, the document is serialized to the file from which the document has been loaded. Saving the file does not remove the document from memory. When saving the document, the last modification data will be set in the document’s description metadata.

The close method removes the document from memory. If it has not been saved before, the changes are lost.

5.1.3 Setting and Getting Metadata

Metadata will be read and written using a flexible mechanism that allows to modify the mapping of arbitrarily named metadata elements to the MPEG-7 document. This mapping is performed in two steps: An arbitrarily named metadata element (“key”, name of the metadata property in Spycer) is mapped by a configurable dictionary to an xPath statement. The xPath statement (relative to a video segment) is used to address the appropriate node in the MPEG-7 document, whose value is set to the passed metadata property. The value is passed as a string (which needs to be parsed/serialized in the utility library).

- As the MPEG-7 location may be invalid or the data types could mismatch, the getter/setter throws an exception in such cases specifying the cause.
- The change of the path or clip name is also done using the global metadata setter.
- Elements for which no MPEG-7 locator is given will be treated as user-defined metadata.

This mechanism can be used for all metadata that is constant throughout a segment. For global metadata, no further information has to be specified. Optionally global metadata can be treated as segment based metadata, with specification of the time range of the whole clip.

In order to set metadata with a certain time range, the start time point (inclusive) and end time point (exclusive) has to be specified. If no segment exists for this time range, a new one will be created and the metadata property will be set.

For reading segment-based metadata, two options methods will be provided:

- If a single time point is specified, the value of the property for that time point will be returned, along with the start and end time of the segment.
- If a time range is specified (or no time range to indicate the whole clip), a list of segments within this time range (specified by value, start time, end time) is returned. The lists shall be implemented as wrapped Qt containers.

5.1.4 Time-Based Metadata

Any kind of metadata can be set globally on the MPEG-7 document, as well as time based. Time based metadata is organized in Segments that can be structured hierarchically.

5.1.5 Getting Shot and Keyframe Information

In the case that content analysis has to be performed, shot and key frame information can be retrieved using

`ShotInfoList getShotInfo()`, which returns the list of shot identifiers, the associated start and end times,
KeyframeInfoList getKeyframeInfo (optional shotid), which returns the list of key frame paths within the specified shot, as well as the key frame's time points.

The lists shall be implemented as wrapped Qt containers.

5.1.6 Getting Stripe Image Information
In the case that content analysis has to be performed, stripe images information can be retrieved using

StripeimageInfoList getStripeimageInfo (), which returns the list of stripe image identifiers, the associated start and end times and the stripe image file names.

The lists shall be implemented as wrapped Qt containers.

5.1.7 Getting Camera Motion Information
In the case that content analysis has to be performed, camera motion information can be retrieved using

CameraMotionInfoList getCameraMotionInfo (optional shotid) which returns a list of camera motion segment start and end times and a pan, tilt and zoom value for each of the segments.

The functionality is available both on per shot and per clip level.

5.1.8 Getting Visual Activity Information
In the case that content analysis has to be performed, visual activity information can be retrieved using VisualActivityInfoList getVisualActivityInfo (optional statisticalMeasureType, optional shotid) which returns the average activity value per shot.

The functionality is available both on per shot and per clip level. In both cases it returns a list of activity values with start and end time (containing only one element if called with a specific shot id).

5.2 The SEMEDIA Framework
Although the SEMEDIA Framework was already covered in detail in deliverable D5.1 this document gives a short overview and adds some enhancements that were made since then. As said in the introduction, the SEMEDIA Framework encapsulates a set of tools that generate metadata in multiple ways. Those tools are loosely “plugged” into the framework using an open-standard web services architecture. The Framework itself is also accessible via this open approach using SOAP methods. Therefore the Framework on the one hand acts as a “client” using the video processing “server” services internally and on the other hand it is also a “server” for 3rd party “client” applications that access the framework’s functionalities. This architecture was chosen to aim at a very loosely coupled approach.
There is also a need for some basic management of media files, database, metadata processing and peer-to-peer communication of different computer nodes. This is done by DVS’s SpycerAgent process that is used as the base of the framework.

5.2.1 Video Processing SOAP Server

The communication between the SEMEDIA Framework and the video processing services inside the framework is done via SOAP. The processing server is invoked through the following methods:
The main functioning is as follows:

1. Get clip’s path (including folder & name) and file type
2. Create a unique id (jobId)
3. Compare clips’s file type with GetSupportedTypes() values
4. Invoke GetStatus(), if return string is empty then start SOAP server
5. Create a jobXml for a specific video file or image sequence (using sequence start, sequence end, folder and filename) and send this XML file to the server by calling SubmitJob(jobXml)
6. Poll GetStatus(jobId) periodically. If it returns “processing” than SOAP Service is still working
7. If GetStatus(jobId) does not return “processing” any longer than processing is finished and we can call RemoveJob(jobId)

It is also possible to pause, stop, resume and shutdown the SOAP server using the method Control(action).

A more detailed description of the process flow is shown in Figure19, showing the methods of the Framework (green) that communicates with the database (red) and the video processing server (blue). An analyse job starts on top of the diagram with “IsValidExtractionJob()” and ends at the bottom by calling “RemoveJob()”.

Figure 19: SEMEDIA Framework video processing services communication
To instantiate a SubmitJob, we need a `sequenceStart` and a `sequenceEnd`. Both are mapped to a specific timecode format as it is used in the MPEG-7 standard. Finally we need the clip’s `path` and `filename` and a destination folder for the generated MPEG-7 file, as well as for the key frames and stripe images. All this information is then found as tags or attributes like in the following jobXml example.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Job xmlns="http://iis.joanneum.at/schema/mex/job"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation="http://iis.joanneum.at/schema/mex/job mexjob.xsd"
     peerID="clipster123">
  <Action>ContentAnalysis</Action>
  <Source>
    <ID type="UUID">record_009</ID>
    <Clip>
      <ImageSequence start="T00:00:00:200F25" end="T00:00:00:399F25">
        <Path type="absolute">X:\record_2106_A_009</Path>
        <File>record_2106_A_009_%07d.dpx</File>
      </ImageSequence>
    </Clip>
  </Source>
  <Mpeg7File>C:\Share\record_009\record_009.xml</Mpeg7File>
  <MetaEssenceLocation type="keyframe">
    <Path>C:\Share\record_009\debugkeyframes</Path>
  </MetaEssenceLocation>
  <MetaEssenceLocation type="stripeimage">
    <Path type="abstracted">C:\Share\record_009\debugstripe</Path>
  </MetaEssenceLocation>
</Job>
```

An application addressing this SOAP interface should generate this file automatically when invoking an analyse process.
5.2.2 Framework Access SOAP Server

To use the SEMEDIA Framework with other prototypes, it is necessary to add a communication port that could be accessed by the base systems of these prototypes. Therefore the SEMEDIA Framework also acts as a SOAP Server.

![Diagram of Prototype Approach with SpycerAgent and SOAP Access](image)

Figure 22: Prototype Approach with SpycerAgent and SOAP Access

A 3rd party application then acts as a SOAP client and can easily connect to the SEMEDIA Framework using the following methods:

1.) A simple (static) search functionality with predefined input fields that work in conjunction (AND-associated):

GetMedia()

**Input:**
- FileName, a string to search for a filename
- FileType, an int for any (0), mpg (1), avi (2), mov (3), flv (4), dpx (5), bmp (6), jpg (7), wav (8), aif (9) file type
- Duration, an int for any (0), < 1 min (1), > 1 min (2) duration
- Resolution, an int for any (0), low (1), high (2) resolution (low is < sd, high is >= sd)
- CameraMotion, an int for any (0), Pan (1), Zoom (2), Tilt (3)

**Output:**
- MediaArray of MediaItems (see appendix)

2.) A complex search functionality using a searchXml as input. The searchXml has to be built in the client application:

GetMediaByXml()

**Input:**
- SearchXml, an XML string built of search conditions

**Output:**
- MediaArray of MediaItems
3.) A simple (static) search functionality returning just a small set of metadata to reduce traffic:

**GetSmallMedia()**

**Input:**
- FileName, a string to search for a filename
- FileType, an int for any (0), mp4 (1), avi (2), mov (3), flv (4), dpx (5), bmp (6), jpg (7), wav (8), aif (9) file type
- Duration, an int for any (0), < 1 min (1), > 1 min (2) duration
- Resolution, an int for any (0), low (1), high (2) resolution (low is < sd, high is >= sd)
- CameraMotion, an int for any (0), Pan (1), Zoom (2), Tilt (3)

**Output:**
- SmallMediaArray of SmallMediaItems (see appendix)

4.) A complex search functionality returning just a small set of metadata to reduce traffic:

**GetSmallMediaByXml()**

**Input:**
- SearchXml, an XML string built of search conditions

**Output:**
- SmallMediaArray of SmallMediaItems

5.) A method expecting a media’s uuid as input and returning a MediaItem:

**GetSingleMediaByUuid()**

**Input:**
- Uuid, a string of a media’s uuid

**Output:**
- MediaItem a list of metadata values for a media file

6.) A method expecting some file related inputs and returning a MediaItem:

**GetSingleMedia()**

**Input:**
- Id a string of a media’s unique id

**Output:**
- MediaItem a list of metadata values for a media file

7.) A method expecting some file related inputs and returning the MPEG-7-XML of the file

**GetMpeg7()**

**Input:**
- Id a string of a media’s unique id

**Output:**
- GetMpeg7Result an path to an MPEG-7-XML containing the metadata of an essence file

8.) A method returning the paths of keyframes of an essence file:

**GetKeyframes()**

**Input:**
- Id a string of a media’s unique id

**Output:**
- KeyframePathArray an array of keyframe paths (strings)
9.) A method returning the paths of stripeimages of an essence file:

\textit{GetStripeimages()}

\textbf{Input:}
- \textit{Id} a string of a media’s unique id

\textbf{Output:}
- \textit{StripeimagePathArray} an array of stripeimage paths (strings)

10.) A method returning camera motion parameters of an essence file:

\textit{GetJrsCameraMotion()}

\textbf{Input:}
- \textit{Id} a string of a media’s unique id

\textbf{Output:}
- \textit{JrsCameraMotionArray} an array of camera motion segments, each containing:
  - start frame number, end frame number,
  - pan value, zoom value and tilt value

5.2.3 Framework Indexing vs. Base System Indexing

There are also two different options of using the SEMEDIA Framework, as described in more detail in deliverable 5.1. The first is to use the Framework for search queries. In that case the framework will exclusively maintain the metadata.

![Figure 23: Framework indexing](image)

It is also possible to index all clips in the base system’s own database by transferring either the MPEG-7 file or selected metadata items after the processing of a clip has been finished. Then searches could be performed on the base system alone, the framework would solely be used for the generation of the metadata.
Figure 23 shows the framework indexing approach. The difference to base system indexing (shown in figure 24) is that after processing of the media files all metadata is indexed in a database that is hold inside the Framework. Therefore the tools inside the framework not only generate metadata, a search on the media files is also possible by sending queries to the Framework.

Systems that have their own database for indexing the files for search can use the Framework solely for the creation of metadata. 3rd Party systems can do queries for selected metadata that is extracted from the MPEG-7 or they can access the MPEG-7 file as a whole to get the complete set of metadata generated by the Framework. The Framework holds a repository of MPEG-7 files whereas the indexing and search is done in the 3rd party application as seen in Figure 24.

5.2.4 Implementation of JRS’ Mpeg7UtilLib

If a 3rd party application is for example not interested in getting the whole MPEG-7 file and parsing it for camera motion information itself, the framework offers the “GetJrsCameraMotion” method.

This method triggers the Mpeg7UtilLib developed by JRS that is used inside the framework and returns a list of camera motion segments that belong to a clip. Those camera motion segments can then be used for visualisation in the 3rd party application.

The Mpeg7UtilLib is also used for reading the camera motion values directly after finishing the analyse process. Those values are then indexed in the database of the framework to use them as search parameters.

5.2.5 General Workflow of the SEMEDIA Framework

After installation and starting of the SEMEDIA Framework one can define one or more so-called “watch folders”. A watch folder is observed by the SpycerAgent process that
runs in the background and recognises adding, deletion and changing of files. Some metadata of those files is indexed in a database for faster search and retrieval.

It is also necessary to define a target folder for the MPEG-7 files, the stripe images and the key frames that are generated by the service. The folder chosen results in a mapping of the original clip’s file path to a so-called “proxy folder”. It is also possible to let the Framework generate a flash video file (flv) and a thumbnail (jpg) so that a SOAP client can access smaller representations of the original clip through network or web.

![Figure 25: Workflow of the SEMEDIA Framework prototype](image)

If an essence (e.g. a video file) is added to a watch folder, an analyse job is started. Adding more files results in a pending queue of analyse jobs started one after the other. The analyse job can be paused, resumed and stopped.

An unsupported file type can be converted into a format that the video processing service can handle. Compatible formats are image sequences (dpx, tiff), mpeg or avi files that use Directshow codecs.

Once the analyse process is finished the user can find an MPEG-7 file containing several metadata like “shot boundaries”, “camera motion” or “visual activity” next to the clip. In the “proxy folder” one can also find an MPEG-7 file, a folder for stripe images and key frames, an flv proxy and a jpg thumbnail.

The metadata and extracted representation are now ready for access through the Framework Access SOAP methods.

To give access via internet we first used an ssh-connection to get a connection to the SOAP server in Hanover. Later we added a Web server to simplify the access.
5.2.6 The Framework's search functionality

When looking at the SEMEDIA Framework's SOAP methods one can find the “GetMediaByXml()” and “GetSmallMediaByXml()” functions that make use of the framework’s search functionality. On the basis of a “searchXml” a flexible approach of creating search queries is given. An example search for a FileName that begins with “spain” or contains the word “barcelona” and has less than 50000 frames FrameCount looks like this:

```
<query type="search">
  <condition>
    <con meta="FileName" op="=" value="spain%%"></con>
    <con meta="FileName" op="=" value="%%barcelona%%"></con>
  </condition>
  <condition>
    <con meta="FrameCount" op="&lt;" value="50000"></con>
  </condition>
  ...  
</query>
```

Figure 26: SEMEDIA Framework example searchXml

All Metadata types that are searchable can be combined as conditions in the searchXml. The percent sign is used as wildcard to build “begins with…”, “ends with…” and “contains…” statements. Omitting the wildcards results in a “direct match”. The operators for “less than” and “greater then” are in XML-specific manner written as “&lt;” and “&gt;”. With this flexible searchXml approach it is possible to design the graphical user interface totally independently from the search logic that will then fit to every GUI demand.

6 Conclusion

During the lifetime of the project the SEMEDIA partners developed ten prototypes, each with a slightly different emphasis.

The six research prototype applications represent individual developments by separate partners that were started or enhanced during SEMEDIA’s activity with the main goal to create test environments for the tools. These could already be used in early phases of tool development, but also for some of the final evaluation tests.

The four integrated applications combine SEMEDIA technologies from different partners with real-world applications for specific use cases from the target domains broadcast, postproduction and Web. At current state some of these applications are already publicily deployed like the Video Search tool from Yahoo! or even in daily production use as the CMD asset management tool from Smoke&Mirrors.

The use of standards like MPEG-7 and a framework with a plug-in like architecture, connected via state of the art Web service technology, provides a simple connection to third-party products and allows the use of SEMEDIA tools in a wide range of search environments and systems.