Multimed Tools Appl
DOI 10.1007/s11042-011-0847-5

Search-based composition, streaming and playback of video archive content

**Dag Johansen · Pål Halvorsen ·
Håvard Johansen · Håkon Riiser ·
Cathal Gurrin · Bjørn Olstad ·
Carsten Griwodz · Åge Kvalnes ·
Joseph Hurley · Tomas Kupka**

Abstract Locating content in existing video archives is both a time and bandwidth consuming process since users might have to download and manually watch large portions of superfluous videos. In this paper, we present two novel prototypes using an Internet based video composition and streaming system with a keyword-based search interface that collects, converts, analyses, indexes, and ranks video content. At user requests, the system can automatically sequence out portions of single videos or aggregate content from multiple videos to produce a single, personalized video stream on-the-fly.

Keywords Video search engines · Personalized composition · Segmented adaptive HTTP streaming

D. Johansen · H. Johansen · Å. Kvalnes · J. Hurley
Department of Computer Science, University of Tromsø, Tromsø, Norway

D. Johansen
e-mail: dag@cs.uit.no

H. Johansen
e-mail: haavardj@cs.uit.no

Å. Kvalnes
e-mail: aage@cs.uit.no

J. Hurley
e-mail: joseph.hurley@cs.uit.no

P. Halvorsen (✉) · C. Griwodz · T. Kupka
Department of Informatics, University of Oslo, Oslo, Norway
e-mail: paalh@ifi.uio.no

C. Griwodz
e-mail: griff@ifi.uio.no

T. Kupka
e-mail: tomasku@ifi.uio.no

Published online: 30 July 2011

1 Introduction

Video has become a popular, first-class entity in the Internet media landscape. Consider, for instance, the emergent and rapid deployment and growth of public Internet video archives, which make a wide range of content available including prime-time newscasts, movies, and scholarly video lectures. The number of videos streamed from these services is in the order of tens of billions per month [16]. Consequently, Internet traffic has become dominated by video data. Leading industry movers conjecture that traffic on the mobile-phone networks will also soon be dominated by video content [10]. This unparalleled growth of data will clearly contribute to the emerging overload problem for Internet users, in particular since existing technologies hardly support information extraction from videos.

An interesting problem is how to efficiently automate the video search process across a large video archive. Novice Internet users are very familiar with text-based search interfaces, and we intended to avoid inventing a new interface that might confuse them. Simultaneously, we conjecture a major change to existing search solutions. First, we abandon the traditional textual-rich result page in response to a query, a page that would provide a list of links to relevant video content accompanied with anchor text. This does not scale properly since users must download large videos for manual parsing, a problem if a user is searching for just a single event or scene in a large video. Second, and more importantly, we do not just recommend links to contents provided by somebody else. Our main thesis is that the search engine should be actively involved in *producing* content by collecting the relevant video objects from the search results and aggregating the clips on-the-fly into a user-specific video summary. This is a major change in search engine functionality and impacts a broad range of issues, from intellectual property rights to scale and availability.

We have developed two different, but related large-scale search based video system prototypes to investigate this thesis properly. Both solves the intellectual property rights by being an Internet system for sign-in customers or an enterprise search system for internal use. The first system is Davvi[1] [25], an end-to-end prototype of

[1]Davvi is the indigenous Arctic Sami word for "north", but interpreted as "towards the coast" (the direction of water streams in the Arctic Norway). http://www.utexas.edu/courses/sami/dieda/anthro/concept-time.htm.

P. Halvorsen · C. Griwodz · T. Kupka
Simula Research Laboratory, Lysaker, Norway

H. Riiser
NetView Technology, Oslo, Norway
e-mail: haakon.riiser@netview.no

B. Olstad
Microsoft, Oslo, Norway
e-mail: bjornol@microsoft.com

C. Gurrin
Dublin City University, Dublin, Ireland
e-mail: cgurrin@computing.dcu.ie

an Internet based video system in the sports domain. It allows users to automatically search for and select specific soccer events from across large video archives and combine them into a single logically composed video for almost immediate playback. The novelty in Davvi is that it changes a traditional search service into a search, production, and streaming service. The second prototype system developed enhances a widely deployed commercial enterprise search engine with video streaming. The latest released enterprise search engine from Microsoft provides the foundation for our novel production enhancement. As such, we have extended this search engine with video composition properties never demonstrated before [20]. Subjective user assessments of both prototypes indicate that the enhanced functionality is preferred over the traditional way of performing Internet video search.

The rest of this paper is organized as follows. In the next section, we give examples of related work and further motivate the need for our system. In Section 3, we briefly outline the architecture of our proposed system integrating and enhancing several different components. Section 4 describes the soccer scenario where we also detail the different components used to build the system. In Section 5, we briefly present the video-enabled enterprise search engine mainly pointing at the differences from the soccer scenario. A user evaluation is presented in Section 6, and finally, we summarize and conclude in Section 7.

2 Related work

In the area of video streaming and multimedia search, there has been a huge amount of research. However, at least in our target soccer and slide presentation application scenarios, no existing system combines these components in a similar manner as presented in this paper. In this section, we give some examples of systems and work that relate to our system.

In the area of video streaming applications for soccer (and sports in general), there are several available streaming systems. Two examples system are VG live [45] and TV2 Sumo [44]. These systems often target live real-time streaming as an alternative to traditional television broadcasting and then later provide only very limited search functionality where a user can only search for games and a few main game events, like goals. The main drawback with these systems is the lacking ability to compose the search results in to a customized, on-the-fly generated personalized video. On the other hand, our architecture uses the search engine as a core system component.

Furthermore, in the slide presentation scenario, several approaches exist where slides and video presentations are synchronized. The Altus vSearch [4] system is perhaps most similar to our ideas with respect to the functionality of combining PowerPoint presentations, videos, and scrolling transcripts into an accessible and searchable video archive. Another example is FXPAL's TalkMiner [17] that return the decks of slides matching a query that each are synchronized with a video. However, both vSearch and TalkMiner are missing the ability to additionally present a customized video for a selected set of slides without manually navigating through each presentation.

Various higher-level techniques for automatic visual analysis of sports videos have been proposed and evaluated [39]. Although soccer is the perhaps most widely investigated topic [13, 26, 53], similar approaches exist for other sports like tennis [6], basketball [55], baseball [9], rugby [38], American football [28], and Formula 1 [46]. Such systems can be used in our metadata and annotation operations. However, their reported recall and accuracy when used alone is insufficient in the context of our target application areas where both high accuracy and low recall are required.

The ability to select key elements of a video for playback is similar to the well known problem of generating a video summary [31]. For instance, the framework proposed by Herranz and Martínez [22] analyzes a video and assigns a relevancy number to each group of pictures (GoP). The list of GoP numbers, sorted by their relative relevancy, is then embedded into the video. Variable length summaries can then be generated by choosing an appropriate prefixed subset of this list. The use of such an index structure implies that the video analytic phase needs only to be run once, which is critical for low-latency operation. However, the suggested technique cannot generate summaries composed of segments from multiple videos. A similar problem exists with the priority curve algorithm suggested by Albanese et al. [3]. The algorithm can efficiently compute video summaries by looking for peaks in the relevancy of adjacent video segments, but cannot operate on a larger corpus of videos. Summarization techniques that make the user interact with intermediate video data, like storyboards and video posters [33, 52], might be useful for understanding and navigating a large video search output. However, our goal is direct playback of the search output, which would mitigate the need for such tools.

There exist a lot of solutions for video delivery, and in general, most existing solutions could be used. However, systems providing adaptive segmented HTTP streaming have recently gained new interest with systems like Move Networks [32], Microsoft's Smooth Streaming [29], and Apple's HTTP Live streaming [36], and such approaches are currently under consideration for standardization as dynamic adaptive streaming over HTTP (DASH) [43]. Thus, we target a similar approach.

In summary, there exists many individual components that may be used in our scenario, and we do reuse and modify existing solutions. However, the novelty of our system is the full integration of search and customization with a corresponding video extraction, on-the-fly generation and synchronized video presentation. Next, we present our approach.

3 The Davvi architecture

Our thesis is that a traditional search engine should be able to be engaged more actively in the production of the concrete results presented to end-users. Today, a search engine crawls the web or similar proximities for content, produces an index based on obtained data, and provides a query based interface to its end-users. We denote this the search and recommendation component (❶) in our overall Davvi architecture. To facilitate production properties, we must next add two more components, the video archive and streaming (❷) and metadata extraction (❸) components. An overview of the architecture and how the different components interact in the implemented experimental prototype is given in Fig. 1. All the components within the Davvi-box are integrated. We do use some existing components like the Solr search engine [7], the Apache web-servers [5] and the FFmpeg coding tools [35]. Furthermore, the web-crawlers and the streaming solution are build upon known

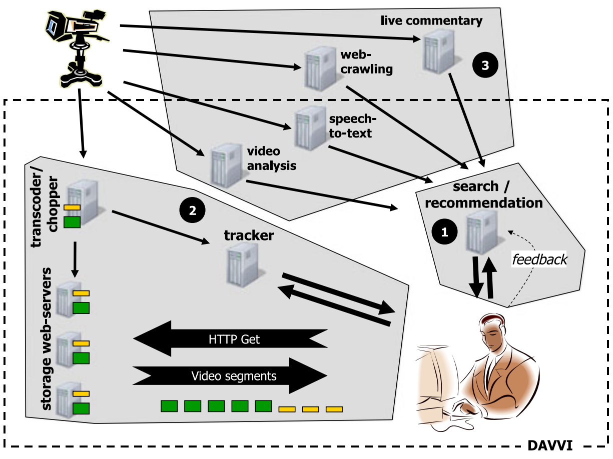


Fig. 1 Davvi architecture and main components

ideas, but they are implemented from scratch to have full control to be able to efficiently integrate the solutions.

First, we need the raw video content itself pre-fetched and pre-produced for production purposes. User-perceived search query latency is in the order of 200 ms for popular search engines, and we must strive for similar behavior. The trick search engines apply to provide the illusion of searching the entire Internet in this short time-frame is to crawl and pre-process an index that queries are resolved against. A similar approach is indeed needed for video production purposes, in particular since this voluminous media-type might add orders of magnitudes in resources required. Hence, we add a video component to the architecture that pre-fetches, pre-processes, archives, and streams video to end-users.

The video archive and streaming component (❷) needs to support several non-functional requirements. This includes how to scale the streaming servers incrementally, how to achieve load balancing, how to enable quality adaption, and how to support arbitrary combination of video clips from multiple sources. Using a multi-source streaming system with discrete media object units (segments) has already been successfully demonstrated and proven scalable [32]. Hence, we adopt a similar solution. As Fig. 1 shows, the captured videos are first coded in segments, each in multiple qualities, and then they are stored in the video archive consisting of plain web-servers. Such representation lends itself naturally to content distribution networks like Akamai [2] or L3 [27] for streaming purposes.

To disseminate the multi-quality video streams, we chose a torrent-like approach over HTTP for data delivery. Such an approach has recently gained new interest with systems like Move Networks [32], Microsoft's Smooth Streaming [29] and Apple's HTTP Live streaming [36]. The essence is that segments are downloaded using HTTP GET requests, and different segments might then be retrieved from different geographical locations. A tracker may be used to hold up-to-date information about the individual segment locations, something we also need to provide a general solution for more or less independent of the type of application built on top.

To analyze and to annotate the videos with searchable metadata (❸), we need tools for identifying the semantic meaning of the different video regions. Traditional search engines provide this poorly, so alternatives must be explored. One serious challenge for incorporating a general structure for this at the general architectural level, is that this task is very application specific. One solution does not fit them all. Thus, the metadata extraction component incorporates multiple alternatives. The net effect of this component is to produce (1) a short description that describes each event (and thus can be indexed by a search engine) and (2) a time interval defining the part of the video that relates directly to the event description.

As depicted in Fig. 1, metadata extraction can include fully automated video analysis and speech-to-text subcomponents. Additionally, semi-automated approaches can be included. There exist a lot of untapped metadata information on the Internet in several application domains including, for example, speech transcripts and live event commentaries [8, 45, 50]. A challenge then is to map this type of annotations to the timeline of the video. How this can be done will be further explored through an implementation of the Davvi architecture in real application domains, which is the topic of the two next sections.

To implement such a system, we integrate several existing components. Some examples include adaptive segmented HTTP streaming systems, web-servers, search engines, indexing systems and web-crawlers, to name a few. To follow good engineering practice, we do not reinvent every single aspect of a new architecture. Instead, we integrate and enhance existing solutions if possible, and then demonstrate novelty through the combination. The paradox is that the less novel components used in providing a novel holistic system, the more is the potential of wide adoption in practice. Enterprises with access to large video archives and investments in their own infrastructure should then be able to create similar practical systems with relatively minor additions to their existing infrastructure.

4 Soccer event search and summaries

We have designed and implemented an application prototype based on the Davvi architecture. This is in the soccer domain, and we call this application Davvi for short.

4.1 Davvi functionality

Searching and streaming video in the soccer domain is a multi-million dollar industry. Currently, numerous media companies are delivering real-time sports streaming via broadband Internet. This is primarily a supplement to television broadcasting schemes, and these companies also provide library footage of past events as an extra service. Live soccer streaming typically peaks a few hours during a weekend while the games are played; for the rest of the week, the streaming infrastructure is hardly used. Davvi aims to create new complementary activities for this silent mid-week period.

Although our application is new, Davvi strives for a familiar user experience. Keyword-based search was an obvious candidate, so Davvi provides a textual based interface for event querying as illustrated in Fig. 2. The result granularity is important, because a 90-min video of the whole game is not appropriate in this context. Searching for very specific information using the full content of digital videos would be a labor intensive, time-consuming process, since each result must be traversed manually following an initial search.

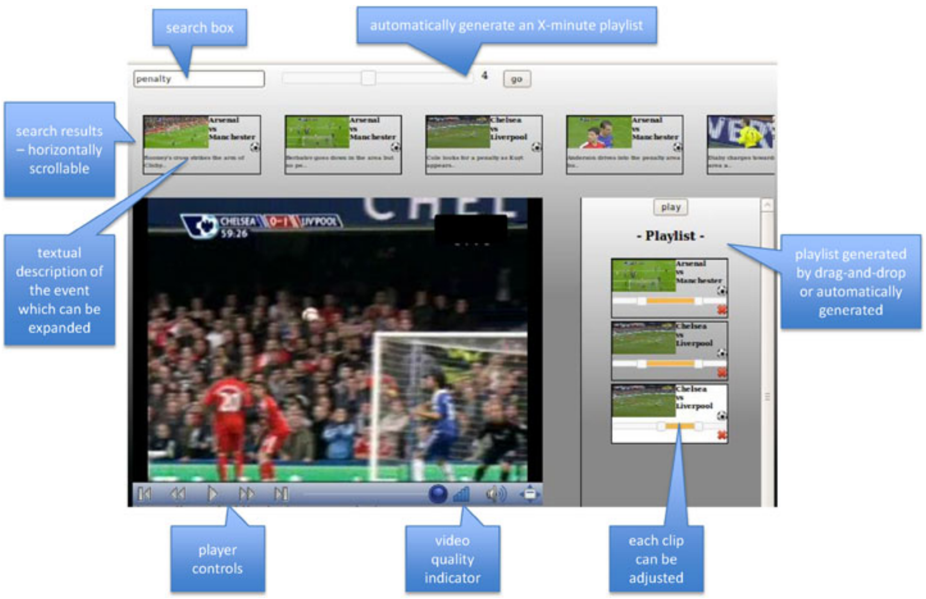

search box
automatically generate an X-minute playlist
penalty
go
search results – horizontally scrollable
textual description of the event which can be expanded
play
- Playlist -
playlist generated by drag-and-drop or automatically generated
player controls
video quality indicator
each clip can be adjusted

Fig. 2 One of Davvi's user interfaces

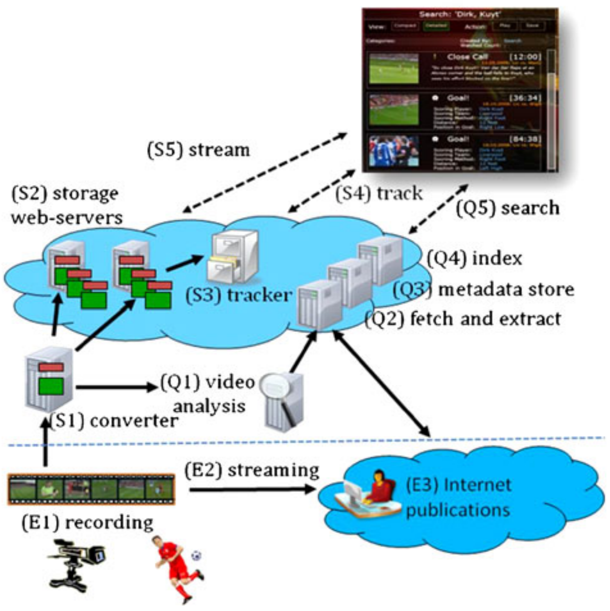
Traditional video shots are also too coarse grained, so by using video segments of short duration (2 s), Davvi provides finer granularity of the shot boundaries of an event, where each event is identified and annotated by web-sources (see Section 4.3). Thus, Davvi aims to identify an optimized video frame sequence for the given user query. An event is then potentially resolved to one or several consecutive sequences of fixed size video segments. If the relevant event happens in, for instance, 8 s of a 30-s shot, 22 s of the shot could be considered irrelevant.

Davvi offers two options for the user to choose how the final result of the search is presented. With the first option, the user can be directly involved in the query result refinement process by selecting explicitly from the result set what to watch from a list of video thumbnails. Thumbnails in Davvi are 180 $\times$ 132 pixel JPEG-pictures ranging from about 8–12 KB in size. When generated in a simple manner from the initial part of a video shot they have been shown to provide little value over a textual response [47], so our thumbnail is generated from the most relevant segment within a shot. Each search result then is represented by such a thumbnail and a corresponding textual description, and these results can be further manipulated (e.g., changing the time length of the event) before the selected playlist shots are streamed continuously.

With the second option, Davvi can be fully automated without any user involvement other than the initial query where the desired length of the composed video is also specified. In this case, the most accurate segments are immediately fetched and played back as a single composed video, thereby bypassing the user involvement in the composition process after the query has been submitted.

The inclusion of personalization and recommendation technologies can reduce user overhead and integrate an element of diversity into interactive multimedia systems. Davvi incorporates a recommendation element by gathering a user profile comprising user interests (explicitly gathered) and user attention data (implicitly

Fig. 3 Davvi operations in the soccer example



gathered listings of the video segments played for any given user query). The user interest element of the user profile stores details of the user's favorite teams and players, and this makes accessing related content easier. Although not the focus of our current Davvi work, the user attention data will be used to support later collaborative filtering across a large user-base to further improve the quality of recommendation.

Figure 3 describes the design of Davvi in order to provide the intended functionality in the soccer scenario. Below the dotted line are external components already used commercially for content capturing and live streaming. These are labeled (E1), (E2) and (E3). Davvi consists of two main parts: the video archive and streaming parts labeled (S1), (S2), (S3), (S4) and (S5); and the metadata extraction, search and reccomendation parts labeled (Q1), (Q2), (Q3), (Q4) and (Q5). In the following subsections, we will discuss design alternatives and detail the core software components needed for this type of video system.

4.2 Video archive and streaming

Our initial Davvi implementation runs over a large video archive including Norwegian Premier League soccer games. Such an archive would normally be managed by a single entity like our partner Schibsted, one of Europe's largest media-houses. The original sports videos are recorded initially (E1) in standard PAL SDTV (720 × 576 @ 25 fps interlaced) using a bitrate of 5 Mbits/s. This is a commercial standard used by Schibsted for content capturing. The video is then encoded in MPEG-2 and audio in MP2 (MPEG-1 Layer II) and provides the foundation for live streaming (E2) as offered today on a commercial basis.

4.2.1 Adaptive HTTP streaming

The current de facto video delivery solution over the Internet is adaptive streaming over HTTP [1, 29, 32, 36]. This is an approach that is currently under consideration for standardization as dynamic adaptive streaming over HTTP (DASH) [43]. Many alternatives exist, but for example, Quality-of-Service protocols are still not widely available, and UDP lacks congestion control. Hence, many current streaming applications are implemented using TCP, which has been shown to provide good streaming performance [48]. In these systems, the bitrate (and thus video quality) can be changed dynamically to match the fluctuating amount of available bandwidth, giving a large advantage over non-adaptive systems that are frequently interrupted due to buffer underruns or data loss.

Commercial video service providers typically use a combination of adaptive segmented HTTP-solutions (over TCP) and content distribution networks. Davvi currently uses a similar approach.[2] We deploy videos on its own web storage servers, but this can easily be replaced with a content distribution network. Davvi also uses a torrent-style HTTP streaming protocol (S5) with a tracker (S3) like in BitTorrent [11]. Our custom container is optimized for adaptive streaming and video composition over HTTP, and it reduces overhead with a factor of approximately 2–10 for small size files being distributed compared to popular commercial systems [37].

4.2.2 Segmentation

Video data can be delivered several ways over HTTP, and a first design choice concerned how to represent, store and stream videos. Davvi adapts a torrent-like approach splitting the video stream into many smaller independent, self-contained videos (segments) which each can be delivered over HTTP from different servers. Davvi can then benefit from a distributed infrastructure where segments from the same video can be stored at multiple repositories (S2) utilizing their parallel retrieval potential. Moreover, segments can be cached in the dissemination network for later reuse, similar to the way popular content distribution or streaming networks work.

The next challenge is to identify the ideal length of the video segments. Apple HTTP streaming uses fixed segment sizes of length 2–10 s, and Microsoft Smooth Streaming uses variable lengths between 2–4 s. For Davvi, we varied the size and experimented with lengths in this range to identify the optimal combination of granularity and quality. In this video search scenario, the lower range proved optimal, so Davvi uses two second segments. With shorter segment lengths, for instance one second segments, one can observe visible noise in the video playback due to flicker effects [34] and decreased compression efficiency [25]. With larger segments, we achieved negligible better compression, but since the search and quality adaption granularity is at segment boundaries, the two second length proved to be an efficient trade-off, i.e., a size also frequently used in commercial systems [29, 32, 36]. The bit rate in each segment varies (variable bit rate) depending on scene complexity.

[2]In principle, most existing solutions [1, 29, 32, 36] are applicable for our usage scenario. However, since these software packages have closed source code, limited support for arbitrary combination of clips, high header overheads, and slow start-up times, we have chosen to implement our own solution providing the same functionality as the existing systems while at the same time addressing their weaknesses.

When storing each segment as a separate file, file block allocation can also contribute to disk footprint expansion since each file is generally not an exact multiple of whole disk blocks (i.e., giving an average fragmentation of half the last block). In our concrete example, the worst case scenario would be 10,800 fragmented disk blocks. The Linux ext2/ext3 file system implementation configured with a typical 4 kB block size resulted in an average fragmentation overhead (with last disk block 50% utilized) ranging from 0.3% (750 kB segment) to 4% (50 kB segment). Given the low cost of disk storage, it is plausible to defend this overhead since it also enables the use of plain web-servers for storage and retrieval. Another approach is to remove the disk footprint expansion by trading off the number of files for server complexity. One could merge all the segments into one file and use HTTP range requests where each HTTP GET operation specifies the file offsets in the file that define each segment. The other option is to apply a solution similar to Smooth Streaming [29] where the web-server extracts the segment file from the large concatenated file. However, our design favors the simple solution enabling less complex segment retrieval operations.

4.2.3 Adaption: segment coding and encapsulation

Internet bandwidth and latency properties vary over time, from user to user, and even from device to device type for a single user. To support video adaption and bit rate scalability we could, for instance, use a scalable video codec like scalable MPEG (SPEG) [23], Multiple Description Coding (MDC) [18] and the Scalable Video Coding (SVC) extension to H.264 [41]. The videos could then be delivered using a scheduling scheme a la priority progress streaming [23] or priority-based media delivery [40] making a prefetch window which is first filled with the scalable codec's base layer. If there is more time (and bandwidth) before the window should be played out and the next window started to be filled, quality enhancing layers are downloaded. These solutions, however, suffer from lack of efficient, available players and large metadata overheads if various segments are combined into a single video playout. Additionally, since we are waiting to fill the window, the latencies also increase. In a multicast scenario, these overheads may pay off, but the Davvi default usage pattern is unicast. Consequently, the extra required bandwidth might damage the video quality.

In Davvi, we therefore code each segment using H.264 (Advanced Video Codec, AVC) with fixed length, closed group of pictures (GoPs) for video and MP3 for audio at several different rates for adaption. To additionally provide adaptiveness, Davvi encodes each individual two second segment in several quality levels as shown in Fig. 4, i.e., each segment is separately coded using H.264 AVC for each quality. For the soccer data, we have experimented with four quality levels (320 × 240 @

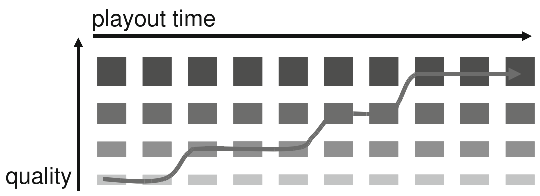
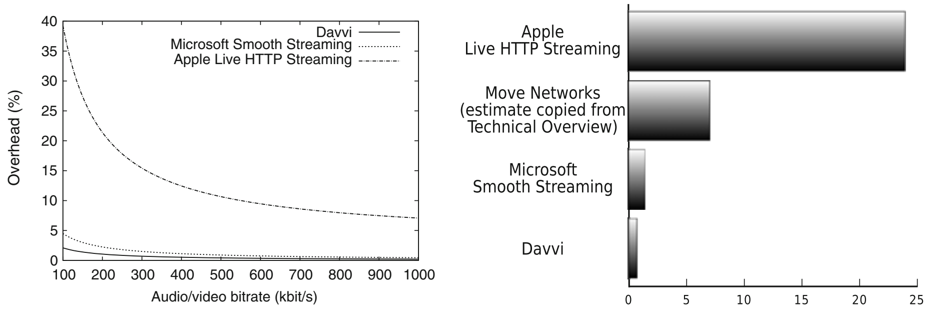


Fig. 4 Video quality and bit rate adaption



(a) Streaming rates of 100–1000 kbit/s (b) A 168 kbit/s stream zoom-out

Fig. 5 Relative container overhead as a function of the stream bitrate

0.6 Mbit/s, 512 × 384 @ 1.2 Mbit/s, 720 × 544 @ 2 Mbit/s, and 1,088 × 832 @ 3 Mbit/s). Then, the video player pulls video segments from the distributed servers with a quality matching the currently available resources, i.e., according to the segment download speed, the amount of buffered data and the size of the next segment, the quality of the next segment is determined. For instance, low quality video can be streamed to a cellular phone with a small display at 0.6 Mbit/s since higher quality in any case will be unnecessary. This saves bandwidth and device energy.

Furthermore, to simplify the multiplexing format and reduce startup latency and overhead of large containers, especially when transmitting low bitrate data, we use our specific container format [37] removing redundancy. The static data is sent once per session and frame indexes are not required, i.e., the only data that is potentially different and included in the container is payload type, time stamp, picture aspect ratio, frames per second and number of bytes until next header. Since each frame, both audio and video, is encapsulated with a header, the relative overhead is particularly high in low-rate streams. This is because each frame is smaller where the constant size header takes a larger share of the total amount of data transmitted. Figure 5 shows how the relative container overhead varies as function of the payload bitrate where we compare the Davvi format with state-of-the-art commercial systems.[3] We can observe that the Davvi container format reduces the overhead meaning that the amount of header information in each packet is reduced. Especially in low bandwidth scenarios like streaming to mobile devices over 3G network (Fig. 5b), this increases the bandwidth available for the video payload.

The Davvi player takes advantage of this multi-encoding together with the new container format to ensure almost immediate start-up viewing time by first streaming low quality encoded video for an initial viewing period. Typical startup

[3]Note that Move Networks' technology lacks specifications, and it could therefore not be properly analyzed. However, in their own technical overview [32], we found an estimation of 10% overhead including protocol headers over the bandwidth that is consumed for audio and video data. However, since we do not know if the overhead varies with the bitrate, it is only included in Fig. 5b as a best case.

Table 1 Startup and interaction latencies for various systems measured from Oslo to a remote storage server

System (storage location)	Latency (ms)		
	min	avg	max
Move Networks (demo system/home page [32], Level 3, California)	560	716	800
Move Networks (VG live [45], Level 3, California)	640	1,022	1,400
SmoothHD (demo system/home page [29], Akamai Oslo)	320	533	880
SmoothHD (Tromsø/Arctic Norway)	400	956	1,500
Davvi (Tromsø/Arctic Norway)	280	328	360
Davvi (Oslo)	320	328	360

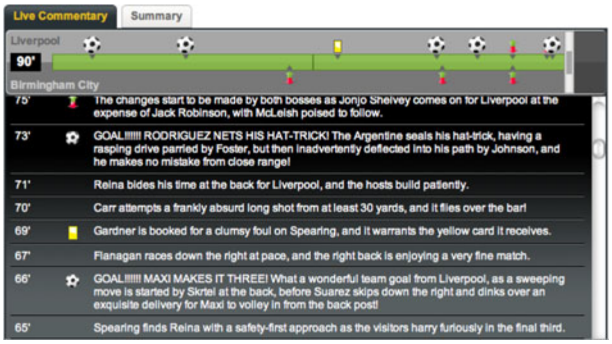
and interaction latencies, i.e., the time between the user hits a button and the first frame is displayed on the screen, are shown in Table 1 where we compare with commercial systems from Move and Microsoft. The table shows statistics over repeated experiments where the load in both the network and the nodes vary. Moreover, we have not taken into account load of and propagation delays to different servers, but the average round trip time to the different sites are less than the differences.

In parallel, high-quality data is fetched at a small offset further out in the video. When sufficient high-quality data has been buffered up in the player, a smooth transition to higher quality playback is performed. Having fixed segment boundaries every two seconds allows this flexibility in the middle of a logical scene where no glitch is observable when moving up or down quality levels.

4.3 Metadata extraction

A key property of Davvi is its ability to precisely identify events in large video archives to support textual video search. For this, we need to extract metadata related to the videos, and we have evaluated two different approaches for this. First, fully automated annotation tools (Q1) can extract low-level metadata from digital videos using complex visual analysis techniques [13, 39]. However, a significant semantic gap exists between the low-level visual features, like colors, textures, and shapes, that typically can be extracted by automatic analysis and the semantic meaning of the video content that users query for, like a sliding tackle by a specific player. Such automatic processing can rely on a number of visual and aural analysis outputs as source for a machine learning approach [39] to identify and label important events. Even though such a technique would seem to be ideal, there is still a trade-off between accuracy and recall. Hence, automated video annotation tools still have high potential for compelling research, but currently limited value in Davvi.

We solve this particular semantic extraction problem by leveraging off from human generated commentaries at external high-quality publication sources (E3) similarly to the approach proposed by Xu et al. [49]. One example is displayed in Fig. 6 where journalists from Yahoo Eurosport UK have annotated the most important events from the 2011 Liverpool—Birmingham City game [51] together with timing information. Fortunately, the sports domain contains numerous such news portals, social networking groups, official club pages, and the like that publish time-synchronized and relevant game data, often while the event happens. Here, the text is used for event annotation and the timing information is used to synchronize



75'	The changes start to be made by both bosses as Jonjo Shelvey comes on for Liverpool at the expense of Jack Robinson, with McLeish poised to follow.
73'	GOAL!!!!!! RODRIGUEZ NETS HIS HAT-TRICK! The Argentine seals his hat-trick, having a rasping drive parried by Foster, but then inadvertently deflected into his path by Johnson, and he makes no mistake from close range!
71'	Reina bides his time at the back for Liverpool, and the hosts build patiently.
70'	Carr attempts a frankly absurd long shot from at least 30 yards, and it flies over the bar!
69'	Gardner is booked for a clumsy foul on Spearing, and it warrants the yellow card it receives.
67'	Flanagan races down the right at pace, and the right back is enjoying a very fine match.
66'	GOAL!!!!!! MAXI MAKES IT THREE! What a wonderful team goal from Liverpool, as a sweeping move is started by Skrtel at the back, before Suarez skips down the right and dinks over an exquisite delivery for Maxi to volley in from the back post!
65'	Spearing finds Reina with a safety-first approach as the visitors harry furiously in the final third.

Fig. 6 Live text-commentaries from Yahoo Eurosport UK [51]

the text to the time in the video, i.e., by default, we define the event time interval as ±15 s of the timing information given in the live text commentaries, but this is also adjusted by the users as shown in Fig. 2. In summary, although humanly generated (often by professional sport journalists), this commentary data is available and is used to automatically annotate our video content. By leveraging it, we bring huge numbers of expert eyes to the task, and they are very likely to comment on just the events users will later want to retrieve.

The main source for Davvi has been the soccer web site VG Live [45] run by Schibsted, but this is gradually being complemented with local newspaper portals publishing similar commentary data. Using more sources gives us richer description of each moment and wider coverage. We consider more text is better since we will be able to answer to a broader set of queries. The published HTML data typically contains general information about the game, like the names of the playing teams and where the game is played. It also contains information about events that have occurred within the games, like descriptions of each scored goal or penalties. The event descriptions typically consist of coarse grained timing information and free-text commentaries.

The immense growth of this type of sports commentary portal is an international phenomenon. In particular, we have been impressed by portals like BBC Sport [8] and Yahoo Sports [50], with second and minute resolution annotations, respectively. A 90-min soccer video is voluminous in size, but the commentary accompanying a complete game is relatively small in size. A VG Live commentary web page is on average 124 kB for a game. We measured similar average sizes for the two international portals Yahoo and BBC which where 102 and 145 kB, respectively.

Davvi automates the task of fetching commentary data, extracting high-level events, storing these events internally, and building a search index. The Davvi architecture is rather design agnostic, in the sense that alternative design choices can be taken at each of these stages. For instance, we have used traditional web crawlers including Microsoft Search Server Express and Apache Nutch, to fetch external HTML data, but prefer our own fetching component based on the Scrapy [14] open-source web-scraping framework (Q2). This custom solution is clearly impacted by the fact that we currently are investigating an even closer coupling with a web publisher; a push structure where commentary updates immediately are submitted into Davvi.

In order to retrieve data, Davvi uses site-specific parsers and XPath queries to extract high-level events from the previously mentioned web pages. Such parsers amount to a few hundred lines of Python code. We did not have to consult technicians at our partner portal for developing these, so they are rather simple to develop. To evaluate this, we also wrote parsers for other external sources. The parser for BBC Sport amounts to 220 lines of Python code, and a similar parser for Yahoo Sports is 339 lines.

To give an indication of the data volume, we applied Davvi for indexing the 2008–2009 season of UK Premiere League soccer games using commentaries extracted from two different web sites. One source had a rather limited vocabulary, but provided timing information at the granularity of seconds. The other source provided timing information with minute resolution, but generally used a much richer language to describe each event. In total, these two sources provided us with 65,233 events, whereof 45,661 were considered unique. This averages to 120 events per game for the 380 games within that particular soccer season. The resulting index contained 9,515 unique terms. Querying this collection for "goal corner Liverpool" yields 2,580 matching events. This small study is indicative that ample metadata can be extracted from these types of web sites provided by expert commentators and journalists.

Davvi has an extensible architecture by design. One of the features of the metadata store is that we can continuously refine it to improve the quality of the video descriptions. We have developed automated analysis and feedback components to do just this. This functionality is enabled by the streaming backend and the flexibility of the metadata store and search index. We support many channels for improving and extending the metadata describing the video content. We have developed feedback components, query-time algorithms and analysis platforms. These components all have the purpose of enhancing the search capabilities and playback quality of the system and giving users enhanced access to the video data.

4.4 Search and recommendation

The next natural step could be to push these events into a traditional search engine, producing an inverted index that resolves keywords to a set of URIs. The problem with this solution is that a query would result in references to complete documents. Davvi is trying to help users find a specific piece of a large document (related to a short sequence in a large video), so Davvi needs to resolve keywords to a set of events within specific documents and their related time-offsets in the associated video.

To support resolution at this granularity, we have introduced a metadata store (Q3) as back-end to the Solr index (Q4). These two components are closely related in the sense that Solr indexes the content of the metadata store. A query is also resolved to content in this store, not the web documents the content originated from. This way, a query resolution is to an event within a game, which contains the offset into the video it relates to.

Translating the event descriptions into the correct position in the video stream automatically is less than trivial. Commentaries are usually published on the web with only a reference to the stadium clock. Some sources keep track of the real world clock timestamp of the moment the comment was published or inserted to their system. Neither of these clocks currently map directly to a video. However, determining the visible stadium clock value through a soccer video can be achieved automatically in certain conditions [54]. In the cases where the video quality or the clock overlay properties prevent this from succeeding, synchronization points need to be entered into Davvi. For continuous footage, one synchronization point per half of the soccer game is required. With these synchronization points in place, every event in a normal game can be mapped to the correct point in the video stream. Second precision at the commentary source results in a more precise mapping.

We have used both video description XML files and a relational database for the Davvi metadata store, and Microsoft SQL Server 2008 is used for the moment. Communication with the scraper is through .NET WCF web services and JSON (JavaScript Object Notation) serialization for packing the content.

Querying the index is through normal HTTP requests using Solr's query language (Q5). The important detail is that a successful query resolution references one or several events in this metadata store, not the original HTML documents covering the complete game. Each event is represented as a row in the database. The search output includes information that uniquely identifies the rows in the database that match a query. However, in most cases, the end user only needs to interact with the index since the information required to playback a matching video clip is stored as document attributes in the index and returned as part of the query response.

Figure 7 illustrates the flow of textual data in Davvi. We have used pure XML output from Solr, but we now use JSON as output format from Solr, partially since this is a human-readable format often used for serializing and transmitting structured data over a network connection. This is also flexible, as with most Davvi components and their interaction schemes. The net effect of this Solr output is that video segments can be immediately downloaded. The time-offset into the game is

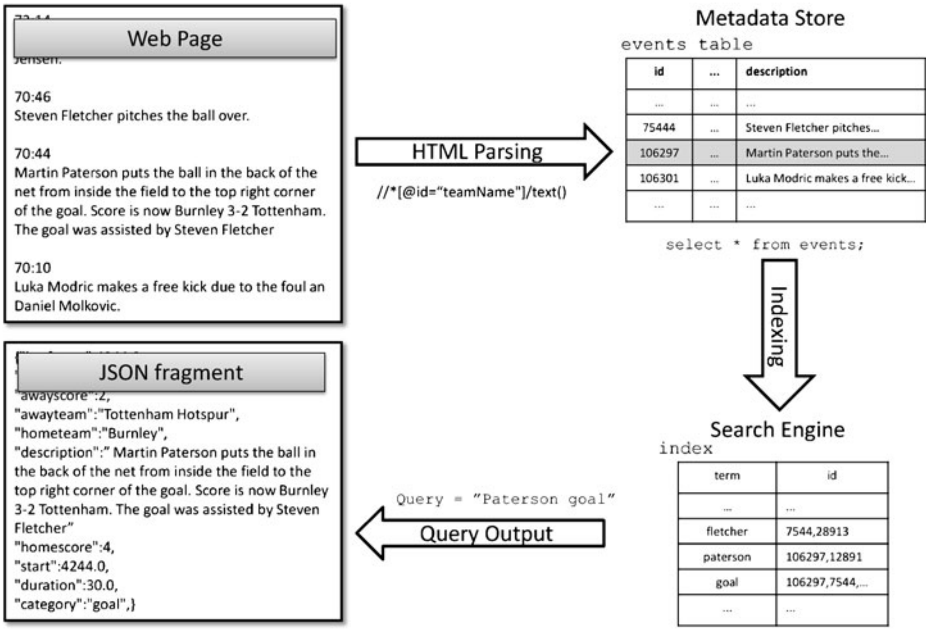

Web Page
70:46
Steven Fletcher pitches the ball over.
70:44
Martin Paterson puts the ball in the back of the net from inside the field to the top right corner of the goal. Score is now Burnley 3-2 Tottenham. The goal was assisted by Steven Fletcher
70:10
Luka Modric makes a free kick due to the foul an Daniel Molkovic.
HTML Parsing
//*[@id="teamName"]/text()
Metadata Store
events table
id
...
description
75444
Steven Fletcher pitches...
106297
Martin Paterson puts the...
106301
Luka Modric makes a free kick...
select * from events;
Indexing
Search Engine
index
term
id
fletcher
7544,28913
paterson
106297,12891
goal
106297,7544,...
Query = "Paterson goal"
Query Output
JSON fragment
"awayscore":2,
"awayteam":"Tottenham Hotspur",
"hometeam":"Burnley",
"description":" Martin Paterson puts the ball in the back of the net from inside the field to the top right corner of the goal. Score is now Burnley 3-2 Tottenham. The goal was assisted by Steven Fletcher"
"homescore":4,
"start":4244.0,
"duration":30.0,
"category":"goal",}

Fig. 7 Flow of data through Davvi

used when contacting (S4) the Davvi tracker (S3), and the tracker feedback is used to order the correct video segments to stream (S5).

When representing video content visually on screen, there are a number of alternative approaches that can be employed depending on the unit of retrieval in question. In most WWW video search systems such as YouTube or Google Video, the unit of retrieval is the entire video content with a single image (keyframe) and some metadata being employed to represent the entire video content visually on screen, as an entry in a ranked list. Depending on the type of video content and the length of the video, this may or may not be appropriate. In any case, this cannot provide any level of random access, or jumping into the content at an appropriate temporal position. A first step to provide a level of random access into a video would be to use a table of contents over the video. Visual storyboards are commonly used for this purpose, where one or more key-frames (usually JPEG images) are extracted from the video to represent the content and visual presentation of these keyframes produces the storyboard interface [21]. Typically, these are chosen with an even temporal distribution, or even distribution across the video shots that comprise the video content.

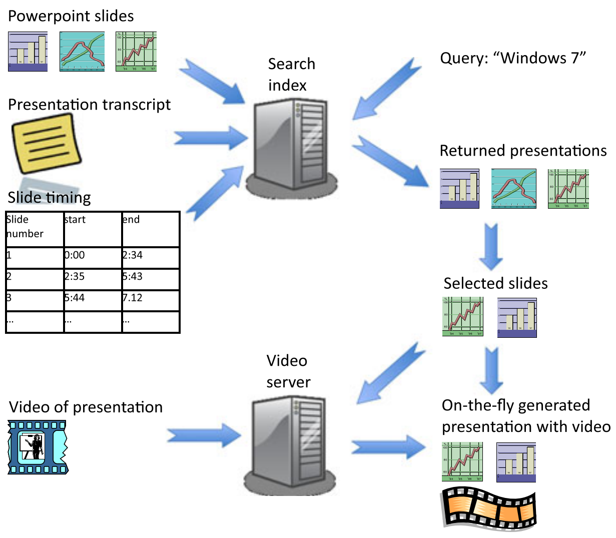
Clicking on any key-frame would typically begin video playback from that point, as would be the case with academic video search engines, such as Físchlár [42]. In this case, each keyframe is considered equally important. In order to indicate varying degrees of importance of keyframes in a storyboard, they may be sized differently, with larger keyframes being more important or with a higher degree of similarity to the query, as is the case in [12].

Davvi also employs storyboards, but where the key-frames represent selected logical units within the video content, such as goals, penalties and other significant events, in a manner similar to [15]. These key-frames map directly to the specific keyword query issued by the user. The benefit of such an approach is that Davvi can support different interfaces to the video content, such as a listing of the important events (e.g., goals or corners) in the content, or the presentation of a ranked list of events or scenes.

5 vESP

The main thesis of our work has been that search result processing should provide more than just references to third party sources. To evaluate this properly and prove that our architecture and conclusions are relevant for multiple application scenarios, we also branch into another application. Hence, we also implemented a prototype application outside the sports domain. The idea is to transparently integrate multimedia data with traditional textual-oriented query results in large enterprise search engines over large repositories of presentations.

We built an application prototype based on the Davvi architecture using internal corporate data from Microsoft. They have a large collection of multimedia data used for training, sales presentations, and marketing purposes. This includes video recordings with, for instance, related PowerPoint presentations and transcripts of the talk. The idea now is that the search engine should automatically discover and extract relevant scenes from a large repository of existing videos and produce new,



Slide number	start	end
1	0:00	2:34
2	2:35	5:43
3	5:44	7.12
...	...	...

Fig. 8 vESP operations: indexing, querying and data presentation

customized videos of events matching the user queries, possibly combining videos and slides from multiple presentations.

Our system has been integrated into the FAST enterprise search platform (ESP) [30] from Microsoft which is a scalable, high-end enterprise search engine commercially used world-wide. Today, the users already may browse documents returned from a query within the search result page, but to view the corresponding video, the user must manually start the video and search to the point where the particular slide is presented—for every relevant slide. The idea of our video-enabled enterprise search platform (vESP)[4] [19, 20] is thus to automate this process such that the user can generate personalized, customized presentations on the fly with the correspondingly generated video.

The typical flow of operations in vESP is depicted in Fig. 8. The first challenge is metadata extraction where it should be possible to have a fully automated approach. In this sceanro, the operation is much simpler since we already have a great deal of textual metadata. We used the transcript of talks, either published by the same publisher of the talk video or obtained using speech-to-text tools, and the associated slides. The resulting text and the slides themselves are indexed similar to normal documents in a traditional search engine. In our demo scenario, all the presentation data including the PowerPoint slides, the speech transcripts and slide timing information (slide change time) is fed into the search engine. The time granularity is in the initial prototype limited to the slide transition boundaries, but we are investigating how to use the corresponding timing information in the audio stream to make a fine-granular index where even smaller parts of the video belonging to a particular slide is returned. Furthermore, as the proposed functionality becomes available, it also creates an incentive for people to produce more metadata. In the scenario for slides,

[4] A video of the vESP demo is available at http://home.ifi.uio.no/paalh/DAVVI-vESP.mp4.

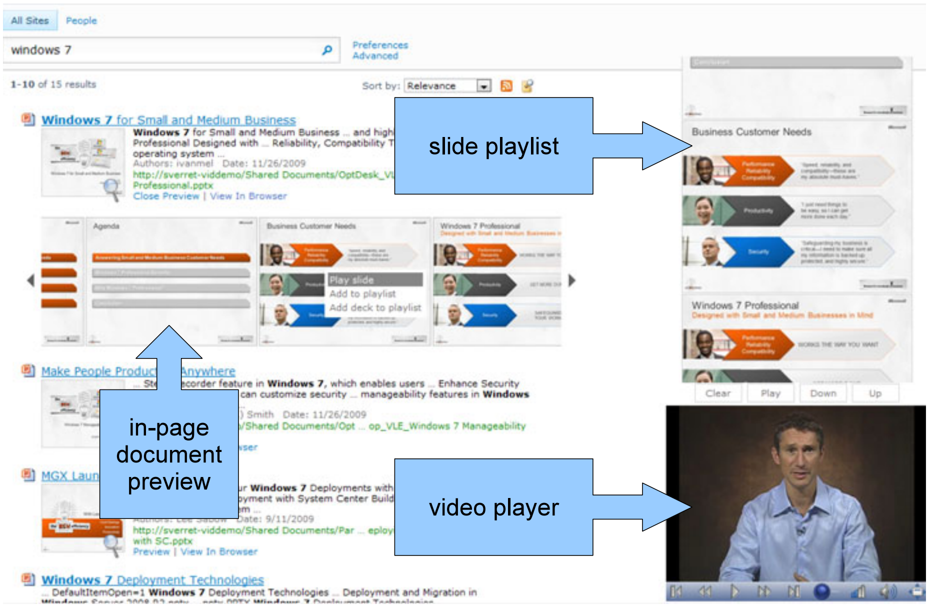


Fig. 9 vESP with document preview and integrated video functionality

we can for example use the already existing timing functionality in PowerPoint and have the slide transition times be recorded.

As in the soccer example, the video component is separate. The integration is performed in the user interface as depicted in Fig. 9. The video data is stored in a "video storage server" and streamed to the user using the adaptive segmented HTTP streaming approach as described in earlier sections, i.e., downloading segments based on the video time intervals delivered from the search results. This means that, using

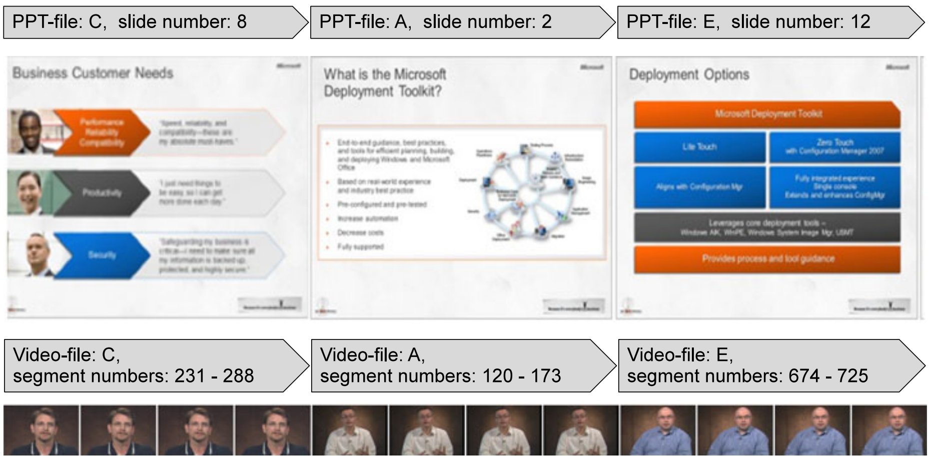

PPT-file: C, slide number: 8
PPT-file: A, slide number: 2
PPT-file: E, slide number: 12
Business Customer Needs
What is the Microsoft Deployment Toolkit?
Deployment Options
Video-file: C, segment numbers: 231 - 288
Video-file: A, segment numbers: 120 - 173
Video-file: E, segment numbers: 674 - 725

Fig. 10 Playlist playout

vESP, the user can based on a textual Google-like search, select individual slides or complete presentations in a new on-the-fly generated presentation and optionally have the related video displayed along with the slide presentation. Thus, the relevant parts of corresponding videos will subsequently be extracted and placed into a playlist as depicted in Fig. 10, and both the slides and the aligned videos are played as one continuous presentation.

6 User evaluation

We conducted a controlled evaluation with external users previously not exposed to our two example systems. We were interested in the first impression with video search added and the possibility to combine events into a personalized video summary. We denote this quantifiable value *functionality*. Since the first impression of a new feature might be too subjective, we also had the users carefully assessing and quantifying how well they considered the added functionality if being used over time. We denote this *value*, our most important measure.

The evaluation was done in varying order using the standard pair-comparison method [24]. They rated the user satisfaction on a standard Likert scale[5] with balanced seven-point keying (useless = 1; wow = 7).

6.1 Soccer scenario

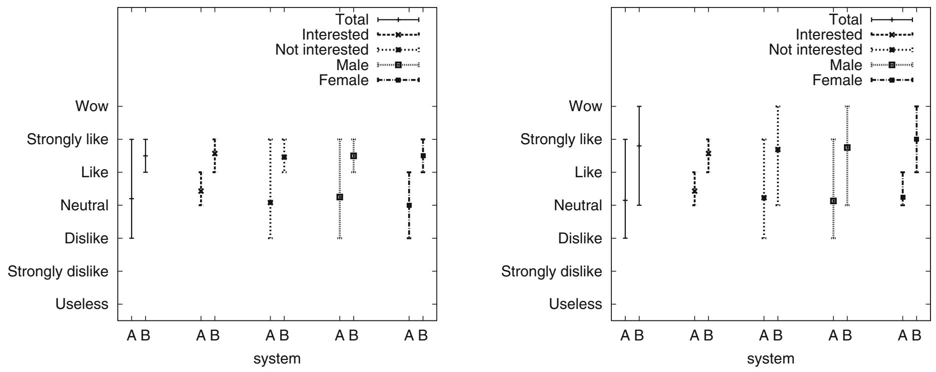
To evaluate Davvi in the soccer domain, we used 20 assessors (both male and female, ages 22–45, with and without interest for soccer) in a controlled experiment. They were exposed to both Davvi and a baseline system, the popular commercial portal VG Live operated by Schibsted. VG Live provides a real-time soccer streaming service, but also an archive of previously streamed games, manually produced highlights, a coarse-granularity search service, statistics and the like.

The results are presented in Fig. 11, and the experiments show no difference between order, gender, age, or scenario interest. Furthermore, the statistical significance was analyzed using a standard Wilcoxon signed-rank test.[6] The results are statistically significant since the probabilities that the results happened by chance are 1.335E−05 for value and 2.670E−05 for functionality.

The overall trend shown in Fig. 11 is that these users immediately appreciated the added functionality Davvi provides. Davvi on average scored 5.5, while VG Live scored more neutral 4.2. Even better results were obtained when asked about longer term value of this functionality. The average value of Davvi increased to 5.8, compared to an average of 4.15 for VG Live. The assessors typically liked the easy way of finding particular events and the way one could combine results on-the-fly into a playlist. They commented that this saves time compared to the benchmark system, and Davvi was the only system that was rated with the highest score ("wow"). This is more than an indication that these users would use Davvi for soccer event searching and summarizations on a more regular basis.

[5]http://en.wikipedia.org/wiki/Likert_scale

[6]http://en.wikipedia.org/wiki/Wilcoxon_signed-rank_test



(a) First impression of the functionality in the search result page

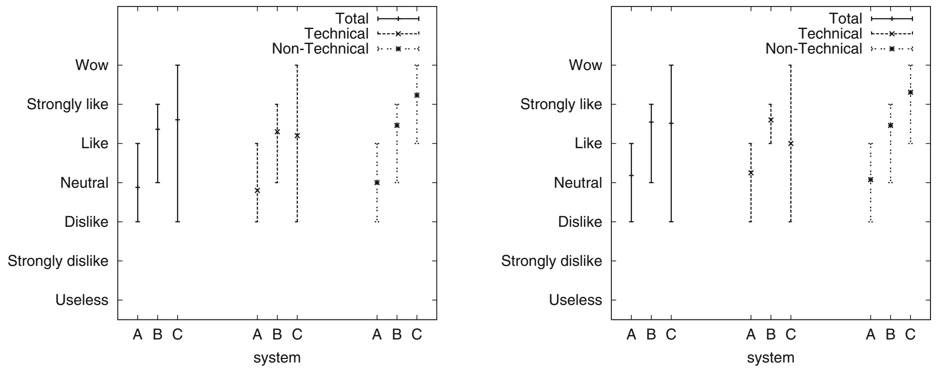
(b) System value,willingness to use the system

Fig. 11 User evaluation results with max/avg/min scores for soccer scenario ($A =$ VGLive, $B =$ Davvi)

6.2 Slide presentation scenario

To evaluate the slide presentation scenario described in Section 5, we collected a new group of people (not previously exposed to the soccer system) for a subjective evaluation of vESP. We had 33 assessors (both male and female Microsoft employees with and without technical background) that were familiar with and frequent users of search systems.

These assessors tried, in different order, all three versions of the system: (a) first approach returned a plain textual, Google-like interface requiring an external program like PowerPoint to open the returned documents; (b) the second approach had support for the in-page preview of slide thumbnails; and (c) the third approach further added support for presenting the corresponding video when clicking on a



(a) First impression of the functionality in search result page

(b) System value,willingness to use the system

Fig. 12 User evaluation results with max/avg/min scores in the presentation scenario ($A =$ plain, $B =$ document preview and $C =$ document preview + video)

particular slide and potentially on-the-fly building a personalized slide deck with the corresponding generated video.

The results from the subjective assessments of the systems are presented in Fig. 12. As in the previous assessment experiment in the soccer scenario, the results are statistically significant regardless of how the results are grouped (total, technical or non-technical, gender or order of the systems). Furthermore, there are no large differences between genders nor the order in which they tested the system. In general, looking at the overall (total) average, we observe that adding the in-page document preview increased the overall impression of the system, and the score is again slightly increased when adding the video features in vESP.

We observe similar trends for the value scores, but the willingness is approximately the same to use the document preview system with and without video. In this respect, the given scores heavily depend on the assessors' background.

Some of the people with technical background working in the area of system development were slightly more reluctant to adding the video features, and their scores varied more. Some liked it a lot, but some had comments that watching video would take a lot of extra time, i.e., assuming that the video had to be used instead of viewing it as an additional feature that could be used if the slide itself did not properly explain the topic. On the other hand, non-technical people like support and sales people gave very positive feedback using terms like "tremendous value" and "very impressive". Nevertheless, in total, the added video functionality was better ranked than the two other approaches, and the top score "wow" was only used when video was enabled. The respective average scores were 5.6 and 5.4 (out of 7) with and without video for the functionality and respectively 5.5 and 5.5 for the value.

7 Summary

A traditional search engine fetches documents, builds an index from their contents, and exposes a query interface to help users find the entire document they are looking for. The differences between Davvi and existing web indexing solutions are the extraction of document pieces (the output of Xpath queries), and the merging of those pieces from different sources to form new documents. Davvi uses that new interpretation to help video archive users instantly access specific moments in potentially very long videos.

We have shown that small, but important modifications to and use of existing Internet technologies result in a novel video search, composition, and streaming system. Davvi fetches documents from the web, chops them up, constructs a new set of documents from the pieces, and indexes those. This indexed data is correlated with the respective video data Davvi has converted, distributed, and made ready for smooth streaming upon request. The net effect is that users can tap and drill into immense amounts of remote video data otherwise not practically feasible. The Davvi soccer system is one implementation in a specific application domain, but we conjecture a much broader applicability. vESP is such an example.

We have deliberately followed good software engineering practice in the sense that we aimed for a novel application, but by leveraging off from related work and well-proven practices for the details. Davvi is the first system then that demonstrates smooth streaming of automatically composed videos in response to a traditional textual query. Similar systems can and should be built by others to efficiently leverage the huge archives of multimedia data out there. The rich potential of existing storage solutions, streaming protocols, and media players represents the keys to exploiting a vast collection of existing video.

Acknowledgements This work has been performed in the context of the *iAD* (Information Access Disruptions) Centre for Research-based Innovation, project number 174867—-funded by the Research Council of Norway. We are particularly grateful for contributions by Erik Aaberg, former Schibsted principal investigator on this project. Feedback from the Cornell professors Kenneth P. Birman, Johannes Gehrke and Fred B. Schneider clearly improved the quality of this paper.

References

1. Adobe: HTTP dynamic streaming on the Adobe Flash platform (2010) http://www.adobe.com/products/httpdynamicstreaming/pdfs/httpdynamicstreaming_wp_ue.pdf
2. Akamai Technologies, Inc. (2011) Akamai: the leader in web application acceleration and performance management, streaming media services, and content delivery. http://www.akamai.com/
3. Albanese M, Fayzullin M, Picariello A, Subrahmanian V (2006) The priority curve algorithm for video summarization. Inf Syst 31(7):679–695
4. Altus, Inc. (2010) Easy enterprise video tools from Altus. http://www.altuslearning.com/altus_vsearch.php
5. Ames G, Bannert A et al (2010) The Apache HTTP server project. Software version 2.2.17. The Apache Software Foundation. http://httpd.apache.org/
6. Anil RD, Kokaram A, Rea N, Denman H (2003) Joint audio visual retrieval for tennis broadcasts. In: Proceedings of the IEEE conference on acoustics, speech, and signal processing, pp 561–564
7. Au B, Cutting D, Gospodneti O et al (2010) Solr. Software version 1.4.1. The Apache Software Foundation. http://lucene.apache.org/solr/
8. BBC (2010) BBC SPORT, football, premier league, live scores. http://news.bbc.co.uk/sport2/hi/football/eng_prem/live_scores/
9. Chang P, Han M, Gong Y (2002) Extract highlights from baseball game video with hidden markov models. In: Proceedings of the IEEE ICIP 2002, pp 609–612
10. Cisco Systems, Inc. (2010) Visual networking index. http://www.cisco.com/en/US/netsol/ns827/networking_solutions_sub_solution.html
11. Cohen B (2008) The bittorrent protocol specification. http://www.bittorrent.org/beps/bep_0003.html
12. Cooke E, Ferguson P, Gaughan G, Gurrin C, Jones G, Le Borgne H, Lee H, Marlow S, Mc Donald K, McHugh M, Murphy N, O'Connor N, O'Hare N, Rothwell S, Smeaton A, Wilkins P (2004) TRECVID 2004 experiments in Dublin City University. In: TRECVID 2004—Text REtrieval conference TRECVID workshop
13. Ekin A, Tekalp AM (2003) Automatic soccer video analysis and summarization. IEEE Trans Image Process 12:796–807
14. Evans S, Hoffman P, Graña D, Olveyra M, García G et al (2010) Scrapy—an open source web scraping framework for Python. Software version 0.10.3, Insophia. http://scrapy.org/
15. Ferguson P, Gurrin C, Lee H, Sav S, Smeaton A, O'Connor N, Choi Y, Park H (2009) Enhancing the functionality of interactive TV with content-based multimedia analysis. In: Proceedings of the international workshop on content-based audio/video analysis for novel TV services
16. Flosi SL (2010) comScore releases April 2010 U.S. online video rankings. Press release. comScore, Inc
17. FXPAL (2011) TalkMiner. http://talkminer.com/

18. Goyal VK (2001) Multiple description coding: compression meets the network. IEEE Signal Process Mag 18(5):74–93
19. Halvorsen P, Johansen D, Olstad B, Kupka T, Tennøe S (2010) A video-enabled enterprise search platform. In: Proceedings of the international conference on data and knowledge engineering (ICDKE), pp 534–541
20. Halvorsen P, Johansen D, Olstad B, Kupka T, Tennøe S (2010) vESP: enriching enterprise document search results with aligned video summarization (demo). In: Proceedings of the ACM international multimedia conference (ACM MM), pp 1603–1604
21. Hauptmann A (2005) Lessons for the future from a decade of informedia video analysis research. In: Proceedings of the 4th international conference on image and video retrieval, LNCS, no. 3568. Springer, New York, pp 1–10
22. Herranz L, Martínez JM (2010) A framework for scalable summarization of video. IEEE Trans Circuits Syst Video Technol 20(9):1265–1270
23. Huang J, Krasic C, Walpole J, Feng W (2003) Adaptive live video streaming by priority drop. In: Proceedings of the IEEE international conference on advanced video and signal based surveillance, pp 342–347
24. ITU-T Study Group 12 (1997–2000) (1999) Subjective video quality assessment methods for multimedia applications. ITU-T Recommendation P.910, International Telecommunications Union
25. Johansen D, Johansen H, Aarflot T, Hurley J, Kvalnes Å, Gurrin C, Sav S, Olstad B, Aaberg E, Endestad T, Riiser H, Griwodz C, Halvorsen P (2009) DAVVI: a prototype for the next generation multimedia entertainment platform (demo). In: Proceedings of the ACM international multimedia conference (ACM MM), pp 989–990
26. Koichi OU, Miura K, Ide I, Sakai S, Tanaka H (2002) An object detection method for describing soccer games from video. In: Proceedings of the IEEE international conference on multimedia and expo (ICME), pp 45–48
27. Level 3 Communications, LLC (2011) Content delivery network (CDN). http://www.level3.com/Products-and-Services/Video/Content
28. Li B, Sezan MI (2001) Event detection and summarization in American football broadcast video. In: Society of Photo-Optical Instrumentation Engineers (SPIE) conference series, vol 4676, pp 202–213
29. Microsoft Corporation (2009) SmoothHD. http://www.smoothhd.com
30. Microsoft Corporation (2010) Enterprise search: FAST. http://www.microsoft.com/enterprisesearch/en/us/Fast.aspx
31. Money AG, Agius H (2008) Video summarisation: a conceptual framework and survey of the state of the art. J Vis Commun Image Represent 19(2):121–143
32. Move Networks, Inc. (2010) Move networks. http://www.movenetworks.com
33. Ngo CW, Ma YF, Zhang HJ (2005) Video summarization and scene detection by graph modeling. IEEE Trans Circuits Syst Video Technol 15(2):296–305
34. Ni P, Eichhorn A, Griwodz C, Halvorsen P (2009) Fine-grained scalable streaming from coarse-grained videos. In: Proceedings of the international workshop on network and operating system support for digital audio and video (NOSSDAV)
35. Niedermayer M et al (2011) FFmpeg. Software version 0.7-rc1. http://www.ffmpeg.org/
36. Pantos R, William May J (2010) HTTP live streaming. Internet-draft version 04. Apple, Inc. http://tools.ietf.org/html/draft-pantos-http-live-streaming-04
37. Riiser H, Halvorsen P, Griwodz C, Johansen D (2010) Low overhead container format for adaptive streaming. In: Proceedings of the multimedia systems conference (MMsys)
38. RTE Sport: Rugby News (2009) http://www.rte.ie/sport/rugby
39. Sadlier DA, O'Connor NE (2005) Event detection in field sports video using audio-visual features and a support vector machine. IEEE Trans Circuits Syst Video Technol 15(10):1225–1233
40. Schierl T, de la Fuente YS, Globisch R, Hellge C, Wiegand T (2010) Priority-based media delivery using SVC with RTP and HTTP streaming. Multimedia Tools and Applications (MTAP) 1–20. doi:10.1007/s11042-010-0572-5
41. Schwarz H, Marpe D, Wiegand T (2007) Overview of the scalable video coding extension of the H.264/AVC standard. IEEE Trans Circuits Syst Video Technol 17(9):1103–1129
42. Smeaton A, Lee H, Mc Donald K (2004) Experiences of creating four video library collections with the Físchlár system. Int J Digit Libr 4(1):42–44
43. Stockhammer T (2011) Dynamic adaptive streaming over HTTP—standards and design principles. In: Proceeding of the ACM MMSys, pp 133–144
44. TV2 AS (2010) TV2 Sumo, Sporten. http://webtv.tv2.no/webtv/sumo/?treeId=2
45. VG Nett (2010) VG Live. http://vglive.no

46. Vojkan MP, Petkovic M, Mihajlovic V, Jonker W, Djordjevic-kajan S (2002) Multi-modal extraction of highlights from tv formula 1 programs. In: Proceedings of the IEEE international conference on multimedia and expo (ICME), pp 817–820
47. Wactlar HD, Christel MG, Gong Y, Hauptmann AG (1999) Lessons learned from building a terabyte digital video library. Computer 32(2):66–73
48. Wang B, Kurose J, Shenoy P, Towsley D (2008) Multimedia streaming via tcp: an analytic performance study. ACM Transactions on Multimedia Computing, Communications, and Applications (TOMCCAP) 4:16:1–16:22
49. Xu C, Wang J, Wan K, Li Y, Duan L (2006) Live sports event detection based on broadcast video and web-casting text. In: Proceedings of the ACM international multimedia conference (ACM MM), pp 221–230
50. Yahoo! (2010) Eurosport: Premier League Football, Premiership Results & Fixtures, Yahoo! Eurosport UK. http://uk.eurosport.yahoo.com/football/premier-league
51. Yahoo! (2011) Eurosport: Liverpool v Birmingham City | Result | Premier League 2010–2011—Yahoo Eurosport UK. http://uk.eurosport.yahoo.com/football/premier-league/2010-2011/liverpool-birmingham-city-381459.html
52. Yeung M, Yeo BL (1997) Video visualization for compact presentation and fast browsing of pictorial content. IEEE Trans Circuits Syst Video Technol 7(5):771–785
53. Yow D, Yeo B-l, Yeung M, Liu B (1995) Analysis and presentation of soccer highlights from digital video. In: Proceedings of the Asian conference on computer vision, pp 499–503
54. Yu X, Li Y, Lee W (2008) Robust time recognition of video clock based on digit transition detection and digit-sequence recognition. In: Proceedings of the 19th international conference on pattern recognition
55. Zhou W, Vellaikal A, Kuo CCJ (2000) Rule-based video classification system for basketball video indexing. In: Proceedings of the ACM multimedia workshops, pp 213–216

Dag Johansen is a professor at University of Tromsø. His research interests include how to architect and build large-scale distributed systems. Contact him at dag@cs.uit.no

Pål Halvorsen is a professor at University of Oslo and a researcher at Simula Research Laboratory. His research focuses mainly at distributed multimedia systems. Contact him at paalh@ifi.uio.no

Håvard Johansen is a post doctor at the iAD Centre for Research-based Innovation in Tromsø, Norway. His research interests include pervasive computing, Byzantine fault-tolerant membership protocols, and peer-to-peer systems. Contact him at haavardj@cs.uit.no

Håkon Riiser is a PhD student at the University of Oslo; he is also the chief software architect at Netview Technology AS, where he is working on multimedia streaming. Contact him at haakon@netview.no

Cathal Gurrin is a lecturer at Dublin City University and an adjunct researcher at the University of Tromsø. His research interests focus on indexing and content-based retrieval of information in all media, and especially digital video content access from mobile devices. Contact him at cgurrin@computing.dcu.ie

Bjørn Olstad is a distinguished engineer at Microsoft and adjunct professor at the Norwegian University of Science and Technology (NTNU). His research interests include information access, search and image analysis. Contact him at bjornol@microsoft.com

Carsten Griwodz is a professor at the University of Oslo and a researcher at Simula Research Laboratory. His research focuses mainly at distributed multimedia systems. Contact him at griff@ifi.uio.no

Åge Kvalnes is an assistant professor at University of Tromsø. His research interests include datacenter operating systems. Contact him at aage@cs.uit.no

Joseph Hurley is a PhD student at the University of Tromsø. His research interests include search engine technologies and cloud computing. Contact him at joe@cs.uit.no

Tomas Kupka is a PhD student at the University of Oslo. His research interests include video streaming. Contact him at tomasku@ifi.uio.no