inEvent
Accessing Dynamic
Networked Multimedia Events

SEVENTH FRAMEWORK
PROGRAMME

inEvent

Accessing Dynamic Networked Multimedia Events

https://www.inevent-project.eu/

Funded under the 7th FP (Seventh Framework Programme)
FP7-ICT-2011-7
[Information and Communication Technologies]

D3.3: Report on storage and processing functionality

Due date: 01/05/2013
Submission date: 01/05/2013
Project start date: 01/11/2011
Duration: 36 months
WP Manager: Oleg Sternberg
Version: 1 **date**: 20/03/2013

Author(s): Oleg Sternberg (IBM)

Project funded by the European Commission in the 7th Framework Programme (2008-2010)		
Dissemination Level		
PU	Public	Yes
RE	Restricted to a group specified by the consortium (includes Commission Services)	No
CO	Confidential, only for members of the consortium (includes Commission Services)	No

inEvent
Accessing Dynamic
Networked Multimedia Events

D3.3: Report on storage and processing functionality

Abstract

Multimedia archives require a huge scalable storage space. In addition this space must be responsive enough in order to support high access rate on capturing and on streaming processes. Selecting a right solution is not easy, as fast local storage solutions lack scalability, while scalable network based solutions lack fast access rate capabilities. In inEvent, we aim to find a solution that will satisfy both fast access and scalability needs, considering all known available network solutions, such as HDFS (Hadoop File System), IBM GPFS and others. The goal is to provide a system that will allow users to retrieve media on demand, w/o a need to download media or to buffer it on rendering. We will allow for requirements satisfaction of both enterprise and geographically distributed internet-based solutions.

Processing of media is a complex task, while most systems are satisfied with processing of global metadata only. Attempts to make an advanced media processing, while using only one kind of analysis technology, usually bring unsatisfactory results, because each analysis technology is not perfect yet. The inEvent media archive target is to be able to initiate, manage and store results of plurality of analysis processes, coming from speech, image and other processing services. The provided system will try to synchronize between all kinds of media and annotations, build links between hyper-events and aims to find effective ways to store them as complex schemas. Taking into account that none of the analysis services provide perfect results, utilization of services combination will provide better and more confident semantic interpretation of the raw media. The archiving system aims to prepare stored media both for fast and efficient retrieval.

Contents

1 Introduction

This deliverable covers 3 main topics:

1. Storage structuring – the way media content and its processing results are stored in the file system.
2. Analysis processes management – this topic is mostly covered in D3.1 (Section 4 "Processing") and here we bring a short description of it.
3. Storage solution candidates:
    a. HDFS - The Hadoop framework transparently provides both reliability and data motion to applications. Hadoop implements a computational paradigm named MapReduce, where the application is divided into many small fragments of work, each of which may be executed or re-executed on any node in the cluster. In addition, it provides a distributed file system that stores data on the computer nodes, providing very high aggregate bandwidth across the cluster. Both Map Reduce and the distributed file system designed so that node failures are automatically handled by the framework. It enables applications to work with thousands of computation-independent computers and petabytes of data.

    b. GPFS - The General Parallel File System is a high-performance shared-disk clustered file system developed by IBM. It provides concurrent high-speed file access to applications executing on multiple nodes of clusters. It can be used with AIX 5L clusters, Linux clusters, on Microsoft Windows Server, or a heterogeneous cluster of AIX, Linux and Windows nodes. In addition to providing filesystem storage capabilities, GPFS provides tools for management and administration of the GPFS cluster and allows for shared access to file systems from remote GPFS clusters.

2 Storage structuring

The layout of the storage is hierarchal and described by the following figure:

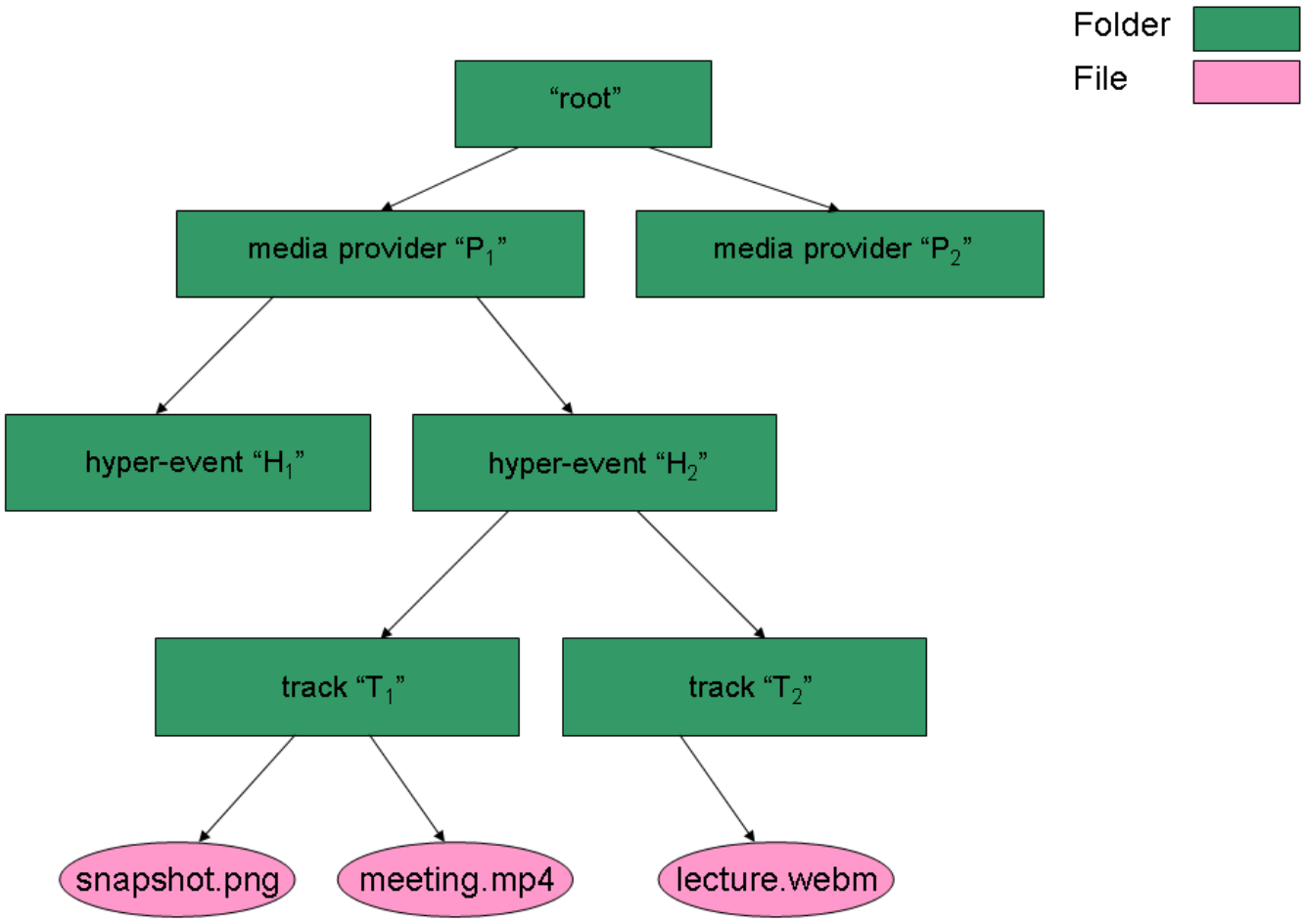

Figure 1

The access/retrieval of media content starts from the inEvent database (described in D3.1) where each hyper-event/track/track file entity has reference to specific folder/file on storage repository. The following figure shows this relation:

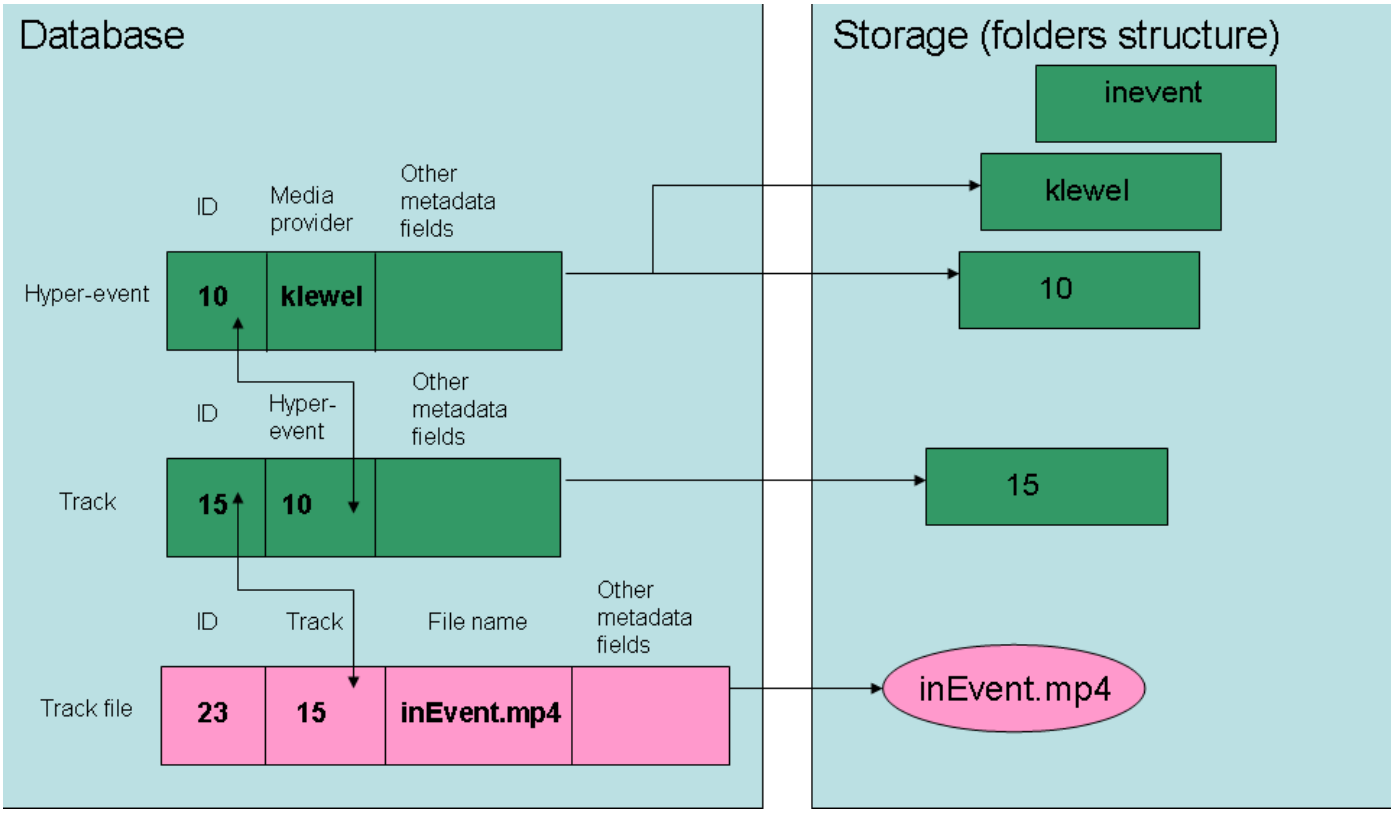

Figure 2

3 Analysis processes management

As mentioned in the introduction this topic is covered in D3.1 (Section 4 "Processing"). The following figure describes video analytics process management, but it will be applied for any kind of analytics process performed on that file:

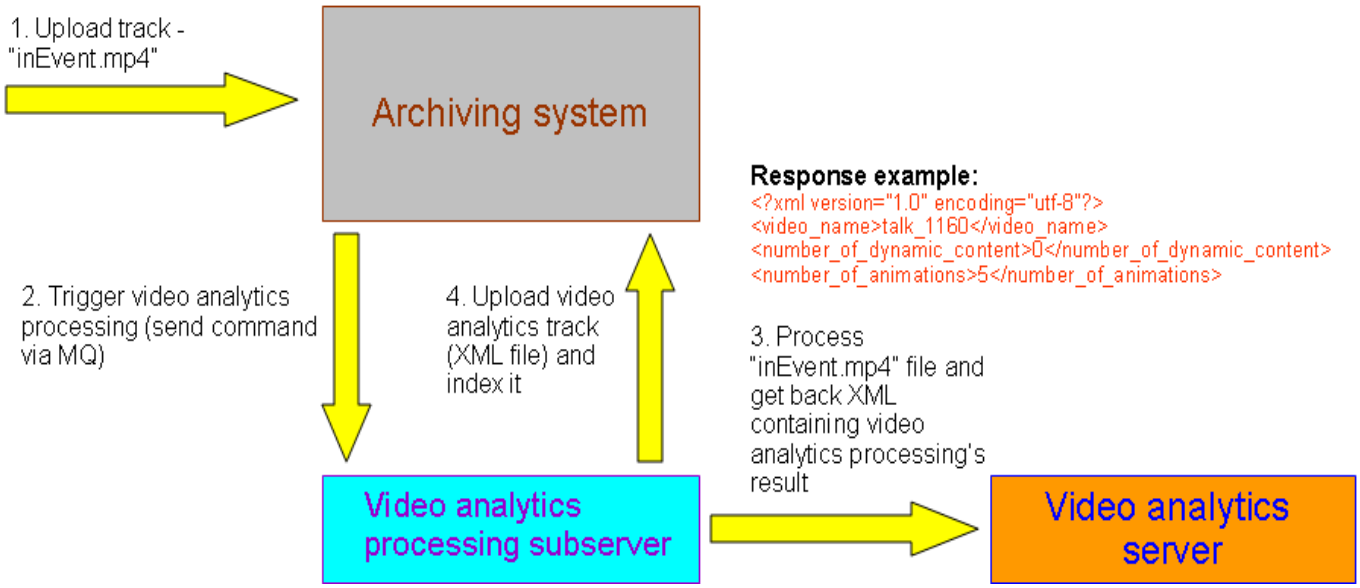

Figure 3

3.1 RTMF (Real-Time Media Framework)

The processing mechanism was designed to be pluggable: this means that the system builds chain of plug-ins (or filters), called plug-in (or filter) graph that executes the desired processing logic per mime type. Plug-ins per mime type and the way they construct graphs are defined in the database (via designated administration interface).
The main advantage of this approach is that it makes the processing mechanism highly extensible and configurable.

The following figure describes this mechanism:

Graph definition in database

Mime type	Filter
inevent/video	Video analytics
inevent/video	Transcript
inevent/speaker+events	Converge
inevent/transcript	Converge

Graph execution chain

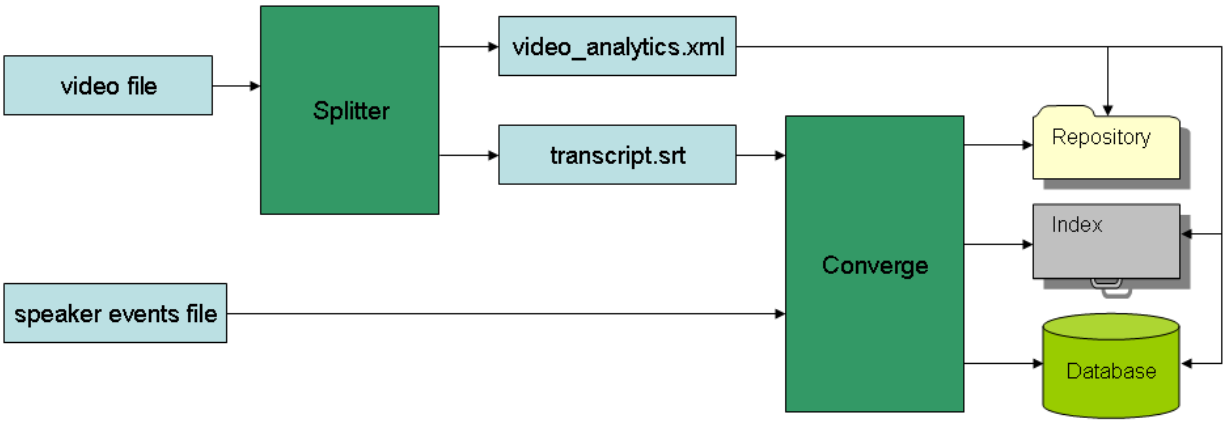

Figure 4

4 HDFS (Hadoop File System)

Hadoop provides a distributed filesystem and a framework for the analysis and transformation of very large data sets using the MapReduce paradigm. While the interface to HDFS is patterned after the Unix filesystem, faithfulness to standards was sacrificed in favor of improved performance for the applications at hand.

An important characteristic of Hadoop is the partitioning of data and computation across many (thousands) of hosts, and the execution of application computations in parallel close to their data. A Hadoop cluster scales computation capacity, storage capacity and I/O bandwidth by simply adding commodity servers. Hadoop clusters at Yahoo! span 40,000 servers, and store 40 petabytes of application data, with the largest cluster being 4000 servers. One hundred other organizations worldwide report using Hadoop.

HDFS stores filesystem metadata and application data separately. As in other distributed filesystems, like PVFS, Lustre2, and GFS, HDFS stores metadata on a dedicated server, called the NameNode. Application data are stored on other servers called DataNodes. All servers are fully connected and communicate with each other using TCP-based protocols. Unlike Lustre and PVFS, the DataNodes in HDFS do not rely on data protection mechanisms such as RAID to make the data durable. Instead, like GFS, the file content is replicated on multiple DataNodes for reliability. While ensuring data durability, this strategy has the added advantage that data transfer

bandwidth is multiplied, and there are more opportunities for locating computation near the needed data.

5 GPFS (General Parallel File System)

GPFS provides high performance by allowing data to be accessed over multiple computers at once. Most existing file systems are designed for a single server environment, and adding more file servers does not improve performance. GPFS provides higher input/output performance by "striping" blocks of data from individual files over multiple disks, and reading and writing these blocks in parallel. Other features provided by GPFS include high availability, support for heterogeneous clusters, disaster recovery, security, DMAPI, HSM and ILM.

A file that is written to the filesystem is broken up into blocks of a configured size, less than 1 megabyte each. These blocks are distributed across multiple filesystem nodes, so that a single file is fully distributed across the disk array. This results in high reading and writing speeds for a single file, as the combined bandwidth of the many physical drives is high. This makes the filesystem vulnerable to disk failures - any one disk failing would be enough to lose data. To prevent data loss, the filesystem nodes have RAID controllers - multiple copies of each block are written to the physical disks on the individual nodes. It is also possible to opt out of RAID-replicated blocks, and instead store two copies of each block on different filesystem nodes.

Other features of the file system include:

- Distributed metadata, including the directory tree. There is no single "directory controller" or "index server" in charge of the file system.
- Efficient indexing of directory entries for very large directories. Many file systems are limited to a small number of files in a single directory (often, 65536 or a similar small power of two). GPFS does not have such limits.
- Distributed locking. This allows for full Posix file system semantics, including locking for exclusive file access.
- Partition Aware. The failure of the network may partition the file system into two or more groups of nodes that can only see the nodes in their group. This can be detected through a heartbeat protocol, and when a partition occurs, the file system remains live for the largest partition formed. This offers a graceful degradation of the file system - some machines will remain working.
- File system maintenance can be performed online. Most of the file system maintenance chores (adding new disks, rebalancing data across disks) can be performed while the file system is live. This ensures the file system is available more often, so keeps the supercomputer cluster itself available for longer.

6 Summary

The following table summarizes different known features and compares them between HDFS and GPFS:

Feature	HDFS	GPFS
Breaking files into blocks and storing on different file system nodes	Yes – block size is 64MB or more to reduce storage requirements on the Namenode	Yes – blocks are small and this may fill up a file system's indices fast, so limit the file system's size.
Node crash recovery	Yes	High risk of data being (temporarily) lost
Data location	Not transparent – must be exposed so MapReduce programs can run near the data	Transparent
Posix standard	Partially supported	Fully supported
Metadata maintenance	Keeps this on the Primary and Secondary Namenodes, large servers which must store all index information in-RAM	Distributes its directory indices and other metadata across the file system

Table 1

The decision about which file system to use for storage of inEvent media and metadata will be made after taking into consideration the project's needs and performing benchmark tests that compare the two file systems presented above.

The tests are planed to be performed at the beginning of the next year. They will cover the following issues:

1. Legality - legal rules dictated by IBM will have major affect on these tests ,i.e. which version of each of the file system can be installed and tested
2. Performance - which file system gets better results in terms of storing/retrieving files
3. Failover recovery - which file system recovers faster and more reliably when one or more of its nodes crashed
4. Maintainability - how easily the files system can be maintained (administration, extending etc.)