NETWORK MANAGEMENT:

STATE-OF-THE-ART

SUBPART OF DELIVERABLE D5.1.1

by
Institut Télécom
CASSIDIAN

Due date of deliverable : t0+ **6**

Actual submission date: t0+ **8**

DOCUMENT HISTORY			
Version	Date	Comments	Author
00	March 02, 2011	ToC	Mihai Mitrea IT
01	May 19, 2011	Minor comments on document	Julien Francq CASSIDIAN
07	June 06, 2011	Integration	Mihai Mitrea IT
09	June 25, 2011	EOLANE contribution integration	Mihai Mitrea IT
10	August 10, 2011	Consolidation, English check, …	Mihai Mitrea IT

	Name and function	Date	Signature
Prepared by			
Reviewed by			
Approved by	Mihai Mitrea	August 10, 2011	
Authorized by			

CONTENTS

SUMMARY

The present document starts the series of deliverables in the SPY's WP5 by presenting the state-of-the-art background supporting the network management related technologies:

- ***data coding:*** *video, metadata and multimedia scene technologies are discussed and benchmarked;*
- ***data encapsulation:*** *the nowadays most intensively used solutions (belonging to the MPEG family) are described;*
- ***session layer protocol:*** *the two most intensively used protocols (RTSP and HTTP) are outlines and their matching to the SPY peculiarities is emphasized;*
- ***network based adaptation:*** *the means for ensuring multimedia content adaptation to the network constraint are hinted to;*
- ***data & stream integrity:*** *the basic principles in watermarking (be it robust, fragile or semi-fragile) are presented and their practical relevance is pointed to through literature examples; the tools for ensuring PMR (private mobile radio) networks integrity and authenticity are also presented and discussed;*
- ***remote control:*** *the way in which complex multi-camera systems can be remotely managed is pointed to.*

1. SCOPE

While complementing the user requirements deliverables, this document is meant to bridge the gap between the Final Project Proposal and the kick-off of the network management technical component specification and development. In this respect, it allows the reader to:

- identify the main technologies able to support the SPY use cases,
- pre-evaluated the related state-of-the-art achievements and the currents technical/methodological dead-locks,
- bring to light how novel synergies among existing technologies (*e.g.* watermarking – compression – hashing – multimedia scenes) can be established in order to properly serve the SPY peculiarities.

Note that this deliverable has an evolving character, *i.e.* it is expected to be reconsidered during/after the architectural specification phase.

2. ASSOCIATED DOCUMENTS

http://www.onvif.org/ - ONVIF website

3. WP5 – OVERVIEW & EXPECTED IMACT

The objective of the WP5 is to define the network management component for the SPY system. In this respect, the following structure has been adopted:

WP5100: Component design

Task 5.1.2 State of the art and feasibility analysis:

The objective of this task is to identify technology limitation and evaluate how to overcome these limitations to fulfil system requirements defined in WP3.

Task 5.1.2 Component architecture definition:

The objective of this task is to describe the component architecture and propose a software decomposition taking into account the SPY platform constraints.

Task 5.1.3 Interface description:

This task will define the Network management interface for SPY systems (mobile and control).

WP5200: Data coding techniques

Task 5.2.1 Adaptive video coding

This task considers the results of the video processing module in order to automatically define RoIs (regions of interest) in video. These regions are further involved in the MPEG-4 selective coding mechanism in order to match the compression of a region to its relevance in video-surveillance (e.g. lossless compression for licence plates and faces but a heavy compression for trees and other static area).

Additionally, “classical” tools for adaptive video coding will be dealt with: Dynamic adaptation of encoding parameters, Simulcast, Video Transcoding, Layered video coding (SVC), etc.

H.264 encoder/decoder in FPGA+DSP environments for mobile systems will be developed. This encoder/decoder will be support H.264 ITU-T Rec.H.264 /ISO/IEC 14496-15, Part 10 standards.

Task 5.2.2 Data Encapsulation

Define the encapsulation schemes able to unitary accommodate natural data (video/audio), video-surveillance relevant information (synthetic data, computed on the mobile side) and application-oriented metadata (GPS position, time stamps, authentication information, etc).

Analysis and development of error resilience techniques: for example as defined in H.264 (intra refresh, slice decomposition, data partitioning, etc.). Streaming problems identification, error rates control, error recovery and packet lost recovery among others.

WP5300: Transmission techniques

Task 5.3.1: Session layer protocols

Analysis and development of a session layer protocols for session establishment, negotiation and feedback to be used for :

- Source-based adaptation for which the source adapts the video transmission based on feedback from receivers and/or network.
- Receiver-based adaptation for which the receivers choose to subscribe to multicast session according to its capabilities or needs.

Task 5.3.2: Network-based adaptation

Analysis and development of techniques which can be used and deployed for network-based adaptation. A few examples are DiffServ (classification by application type or by scalable video coding layers), Unequal error protection, Multimedia Gateways, etc.

WP5400: Integrity techniques

"Integrity" of a digital object means that it has not been corrupted over time or in transit; so, we are sure to have in our hand the same set of bits that was produced when the object was created.

Consequently, an analysis and development of techniques which can be used to endure the data integrity will be carried out. One solution could be the use of watermarking.
The meaning and requirement of integrity in the context of surveillance and rescue framework shall be defined. This analysis implies threats and risks analysis. Due to the kind of information managed watermarking provides relevant technique for integrity. Then the survey of appropriate techniques such as water marking will be done. The outcome of that survey will be a selected technique.
Then one technique will be implemented and validated.

WP5500: Open Platform integration

In order to prove the interoperability of the proposed solutions, the network management component will be integrated in 2 different mobile platforms.

Task 5.5.1: Remote Control software

This task involves developing software for the Open platform so that it will be possible to change on-board camera/audio parameters remotely through a software interface. This task deals also with the development of the application manager on the mobile side.
A middleware will be designed that will enable remote control of the on-board camera parameters.
The application manager is a piece of software which highlights the events detected by the embedded algorithms integrated in the WP4 and notifies the Ground side of the SPY framework.

Task5.5.2: Components integration

This task will deal with the integration of the components developed in the previous tasks (WP5200, WP5300 and WP5400) to fit in the Smart Open platform and to propose the interface defined in WP3.

4. DATA CODING

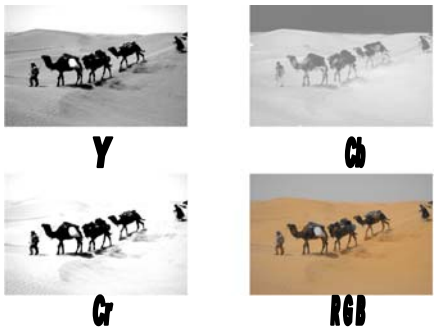
4.1 GENERALITIES

4.2 VIDEO CODING

Like all video codecs, the MPEG-4 AVC codec (coder/decoder), transforms the uncompressed data in a classic compression chain: prediction P, transformation T, quantization Q and arithmetic coding E [1-2]. A preparatory phase, preceding the video data compression chain consists in projecting the data in a YCrCb (respectively luma component, blue-difference and red-difference) color space that closely resembles the human perception of colors, Figure 1. YCrCb signal is created from the source RGB (Red, Green, Blue). The values of R, G and B are added together according to their relative weight to get the signal *Y*. The latter represents the luminance of the source. The signal *Cb* is obtained by subtracting the *Y* signal original blue; similarly signal *Cr* is obtained by subtracting the signal *Y* [3]:

$$Y = 0.299R + 0.587G + 0.114B$$
$$Cr = 0.492(B - Y)$$
$$Cb = 0.877(R - Y)$$

Subsequently a sub-sampling is applied to the *Cb* and *Cr*, as shown in Figure 1, for the *Y* component is more information than *Cb* and *Cr*.



4:4:4

4:2:2 4:2:0

Figure 1: Sub-sampling of color components.

4.2.1 Prediction

The prediction is meant to eliminate the spatial (intra prediction) and temporal (inter prediction) redundancy.

The image represented in each color component is divided into blocks of pixels 16×16 called macroblocks (Figure 2). For images where the number of columns/rows is not a multiple of 16, a padding (add columns/rows of pixels to obtain a multiple of 16). A particular procedure is applied to improve the resistance against errors and losses during the video reconstruction phase to gather macroblocs from the same or different images into slices. The intra prediction is made according to a prediction mode by copying the pixels in the row/column adjacent in a direction (vertical, horizontal, diagonal down/left, diagonal down/right). The prediction modes are illustrated in Figure 3.

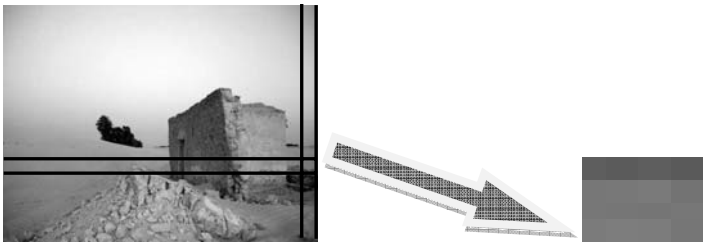


Figure 2: Macroblock illustrations.

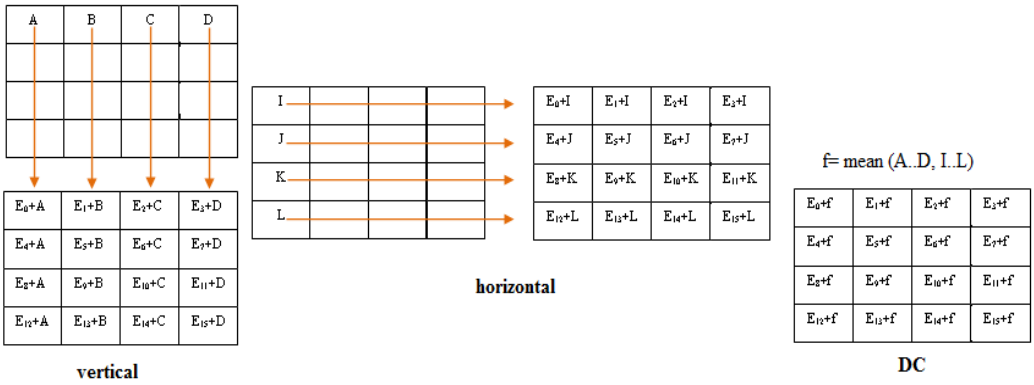


Figure 3: The three most intensively considered prediction modes.

For inter prediction, the blocks are predicted from previous or following frames, the spatial displacement of corresponding blocks of frames specified by a motion vector. This motion vector is estimated that for the luminance component and an integer value. MPEG-4 AVC codec uses an estimate for the quarter pixel motion compensation, enabling very precise description of the displacement of the moving regions. Block sizes for the prediction can be 16 × 16, 16 × 8, 8 × 16, 8 × 8, 8 × 4, 4 × 8 or 4 × 4, enabling very precise segmentation of areas to predict. Note that the reference frame for inter prediction cannot be found outside of a prediction module called GOP (Group Of Picture)[4].

A GOP consists of a number of images that can be 3 types, grouped according to a predetermined decoding order:

- **the I frames** correspond to a coded image independently; note that only one field I can be at the beginning of a GOP, as it serves as a starting point for coding images of two other types;

- **the P frames** are associated with an motion compensated image, predicted either from an I or from another P frame;

- **the B frames** refer to any image being double (forward and backward) motion compensated.

Regardless the type of prediction, the pixel values are subtracted from the corresponding predicted values; these differences are further transformed, quantified and binary encoded.

4.2.2 Transformation

Following the prediction, the transformation is applied with the view of representing the data as uncorrelated (separated into components with a minimum interdependence) and compacted (the energy is concentrated on a small number of values) information [3-4]. The MPEG-4 AVC codec uses a modified version of the classical DCT (Discrete Cosine Transform) in order to work with integer coefficients and eliminate errors caused by the fact that in a conventional DCT coefficients are irrational. The transformation matrix used by MPEG-4 AVC is calculated from the matrix H by taking the rounded values of the coefficients amplified by a factor α (experimentally set to 2.5):

$$X = H \cdot x$$

$$H = h(k,n)$$

$$h(k,n) = c_k \sqrt{\frac{2}{N}} \cos\left[(n+\frac{1}{2}) \frac{k\pi}{N} \right]$$

$$H' = round(\alpha \times H)$$

$$H' = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix}$$

This transformation also has the advantage of simplicity of implementation, as shown in Figure 4: its calculation is based on additions, subtractions and shifts (multiplications by 2).

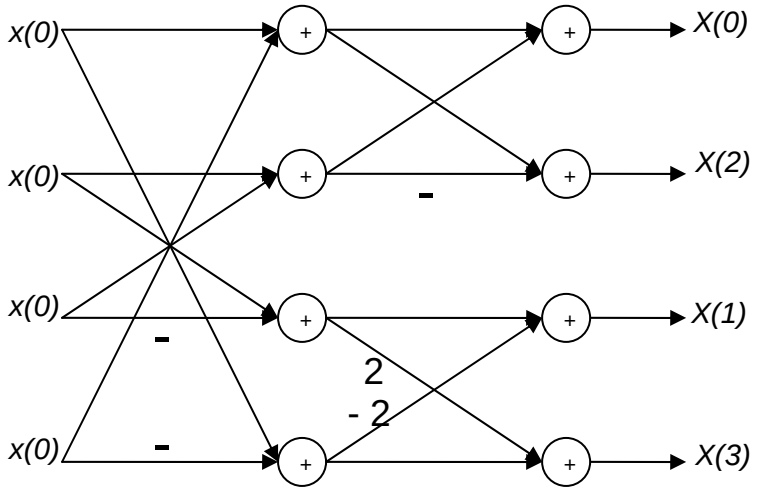


Figure 4: Fast implementation of the integer DCT in MPEG-4 AVC.

4.2.3 Quantizing

Quantizing is the phase where information is lost in the compression chain [3-4]. For a given quantization step (always an integer), the quantized value can be calculated as:

$$X_q(i,j) = \mathrm{sgn}\{X(i,j)\}\frac{X(i,j) + f(Q_s)}{Q_s},$$

where *i* and *j* are the indices of rows and columns and $f(Q_s)$ controls the quantization values near the origin.

For de-quantization, the reconstructed information $X_r(i,j)$ is calculated as

$$X_r(i,j) = Q_s \times X_q(i,j).$$

To circumvent the disadvantages of an entire division, the MPEG-4 AVC offers another form of quantification, this time involving a right shift:

$$X_q(i,j) = \mathrm{sgn}\{X(i,j)\}\left[|X(i,j)| \times A(Q) + f \times 2^L >> L\right]$$
$$X_r(i,j) = B(Q) \times X_q(i,j),$$

where f, $Q(A)$ and $Q(B)$ are association of the quantization parameter, L is the bit length parameter for the encoding process.

4.2.4 Entropy encoding

The final phase of MPEG-4 AVC is the entropy coding (lossless) which takes place in three stages. First, the quantized coefficients are scanned in a zigzag order, Figure 5, in order to gather the maximum of trailing zero and trailing one coefficient. Secondly, each quantized coefficient is RL (Run-Length) encoded so as to increase the compression rate. The construction of the bitstream is achieved according to two advanced methods of entropy coding [5-6-7]:

- CAVLC (Context-Adaptive Variable Length Coding) can be used for all encoding profiles.
- CABAC (Context-Adaptive Binary Arithmetic Coding) that can be used alternately with CAVLC only for main profile.

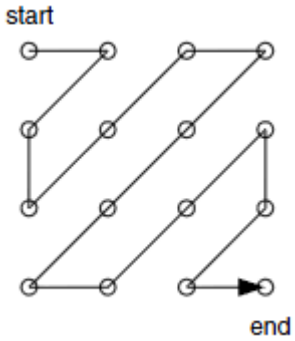


Figure 5: Zigzag scan for 4x4 blocks.

4.2.5 Reference

[1] ISO/IEC 14496-10 and ITU-T Rec. H.264, Advanced Video Coding.
[2] T.Wiegand, G. Sullivan, G. Bjontegaard and A. Luthra, “Overviewof the H.264 /AVCVideo Coding Standard,” *IEEE Transactions on Circuits and Systems for Video Technology*, 2003.
[3] A. Hallapuro, M. Karczewicz and H. Malvar, “Low Complexity Transform and Quantization – Part I: Basic Implementation,” JVT document JVT-B038, Geneva, February 2002.

[4] H.264 Reference Software Version JM6.1d, http://bs.hhi.de/~suehring/tml/, March 2003.

[5] S. W. Golomb, “Run-length encoding,” *IEEE Trans. on Inf. Theory*, IT-12, pp. 399–401, 1966.
[6] G. Bjontegaard and K. Lillevold, “Context-adaptive VLC coding of coefficients,” JVT document JVT-C028, Fairfax, May 2002.
[7] D. Marpe, H. Schwarz and T. Wiegand, “Context-Based Adaptive Binary Arithmetic Coding in the H.264 / AVC Video Compression Standard,” *IEEE Transactions on Circuits and Systems for Video Technology*, 2003.

4.3 METADATA CODING

4.3.1 Definition

The term *metadata* refers to information used to describe items and groups of items. It is data about data. It can be used to describe physical items as well as digital items (files, documents, images, datasets, etc.). A library catalogue, for example, is made up of metadata describing the books, journals and other items held by the library. The File Properties for a word processing document is a rudimentary (and imperfect) metadata record.
Item level metadata is used to describe a single object such as a photograph: who took the photograph, who is in it, the date it was taken, the place it was taken, the type of camera used to take the photograph, and so on.
Collection level metadata is used to describe an aggregation of objects such as the photo album (or CD-ROM or file folder) that contains a group of photographs: the size of the collection, who took the photographs (there may be more than one person), the time period over which the photographs were taken, and so on, Figure 6 [1].

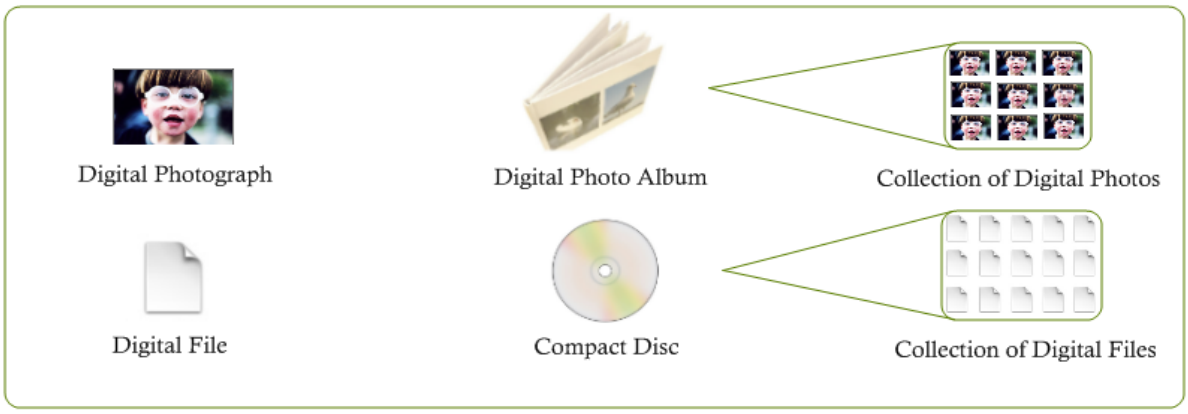


Figure 6: General overview of the metadata structure.

Some of these attributes, such as 'Title' may be the same as those used to describe an individual photograph. Metadata adds value to documents or images. For scientific data, metadata is even more important because it provides the context needed to make sense of what would otherwise be a collection of random numbers.

4.3.2 Types of metadata

The metadata elements used to describe either an item or a collection can serve different purposes. Some examples include:

- *Descriptive metadata* describes a resource for purposes such as discovery and identification. It can include elements such as title, abstract, author, and keywords
- *Structural metadata* indicates how compound objects are put together, for example, how pages are ordered to form chapters.
- *Administrative metadata* provides information to help manage a resource, such as when and how it was created, file type and other technical information, and who can access it. There are several subsets of administrative data; two that sometimes are listed as separate metadata types are:
    - *Rights management metadata*, which deals with intellectual property rights and,

- *Preservation metadata*, which contains information needed to archive and preserve a resource.

4.3.3 Metadata creation

Metadata can be created *by hand*, or it can be created *automatically* [2]. The camera can tell you the time and date, the type of camera, exposure times, file format, and so on, and can attach this metadata to the image file automatically. The camera cannot tell you who the photographer is, or what the subject of the photograph is. This information must be provided by a human and/or by some external software/hardware device. There is a significant cost associated with assigning metadata by hand and little cost associated with collecting it automatically.
Metadata can be embedded in a digital object or it can be stored separately. Metadata is often embedded in HTML documents and in the headers of image files. Storing metadata with the object it describes ensures the metadata will not be lost, obviates problems of linking between data and metadata, and helps ensuring that the metadata and object will be updated together. However, directly embeding metadata in some types of objects can be very difficult (*cf*. the watermarking chapter of this document). Storing metadata separately can also simplify the management of the metadata itself and facilitate search and retrieval. Therefore, current day solutions commonly consider metadata storage in an additional database system and the corresponding linking system to the described objects.

Metadata utilization is illustrated in Figures 7 and 8, [3]. Without metadata, the user must search each data holding individually, Figure 2 [3]. With metadata available, one search yields all, thus resulting in fast and accurate searches.

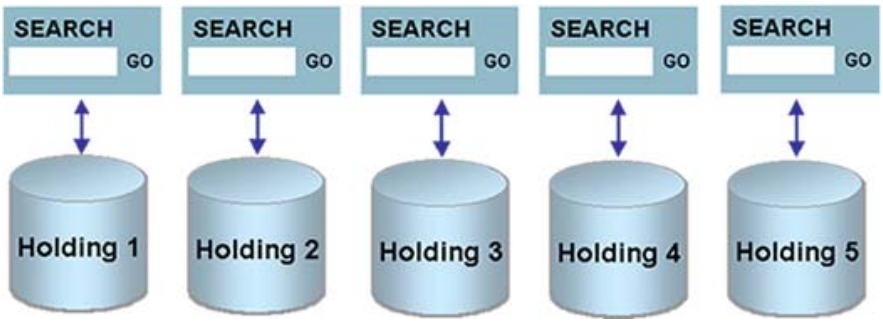


Figure 7: Searching through data holdings without existence of metadata.

Figure 8: Searching through data holdings with existence of the metadata.

4.3.4 Compatibility

It is important that any emerging video metadata format be compatible with the main metadata formats already in use (within existing media archives).
As practically all the main current day solutions considered the Dublin Core Metadata Element Set, their compatibility can be obtained by developing some appropriate parsers: the same information is generally available but its syntax is different, Table 1 [4], [5].

Table 1: Overview of the metadata used by existing media archives.

Bloqx element	Description	Dublin Core	Media RSS	SMIL	Internet Archive
identifier	The URL of the movie (video object).	Identifier	the "url" attribute of the "content" element	Resource	link
Title	The name of the movie.	title	title	Title	title
creator	The creator of the movie.	creator	tbd	Creator	creator
Date	Date that the movie was created (uploaded) DCMI Best Practice to use the ISO 8601 profile for date format: YYYY-MM-DD.	date	tbd	Date	date (contains year eg: "2004"), also "publicdate" in the correct format. Our "date" seems that it is more compatible with "publicdate."
format	The format of the video file: Quicktime, Windows Media, MPEG, etc. (Mime type)	format	"type" attribute of the "content" element	Format	format
Rights	A URL for a text-based explanation of the movie's licensing terms.	rights	tbd	rights (contains text description rather than a url)	licenseurl
subjetct	A set of keywords describing the topic of the movie.	Subject	tbd	Subject	subject
descreption	Provides a text-based description of the movie.	description	description	Description	description

4.3.5 Interoperability

Describing a resource with metadata allows it to be understood by both humans and machines in ways that promote interoperability.
Interoperability is the ability of multiple systems with different hardware and software platforms, data structures, and interfaces to exchange data with minimal loss of content and functionality. Using defined metadata schemes, shared transfer protocols, and crosswalks between schemes, resources across the network can be searched seamlessly.

4.3.6 MPEG and metadata

In October 1996, MPEG started a new work item to provide a solution to the questions described above. The new member of the MPEG family, named "Multimedia Content Description Interface" (in short MPEG-7) [6], provides standardized core technologies allowing the description of audiovisual

data content in multimedia environments. It extends the limited capabilities of proprietary solutions in identifying content that exist today, notably by including more data types. MPEG-7 is a standard for describing the multimedia content data that supports some degree of interpretation of the information meaning, which can be passed onto, or accessed by, a device or a computer code. MPEG-7 is not aimed at any one application in particular; rather, the elements that MPEG-7 standardizes support as broad a range of applications as possible.
MPEG-7 offers a comprehensive set of audiovisual Description Tools (the metadata elements and their structure and relationships, that are defined by the standard in the form of Descriptors and Description Schemes) to create descriptions (*i.e.* a set of instantiated Description Schemes and their corresponding Descriptors at the users will), which will form the basis for applications enabling the needed effective and efficient access (search, filtering and browsing) to multimedia content. This is a challenging task given the broad spectrum of requirements and targeted multimedia applications, and the broad number of audiovisual features of importance in such context.
Figure 9 below shows a highly abstract block diagram of a possible MPEG-7 processing chain, included here to explain the scope of the MPEG-7 standard. This chain includes feature extraction (analysis), the description itself, and the search engine (application). To fully exploit the possibilities of MPEG-7 descriptions, automatic extraction of features will be extremely useful. It is also clear that automatic extraction is not always possible, however. As was noted above, the higher the level of abstraction, the more difficult automatic extraction is, and interactive extraction tools will be of good use.
Note: Regardless their usefulness, neither automatic nor semi-automatic feature extraction algorithms are inside the scope of the standard; this is also the case for the search engines, filter agents, or any other program that can make use of the description.

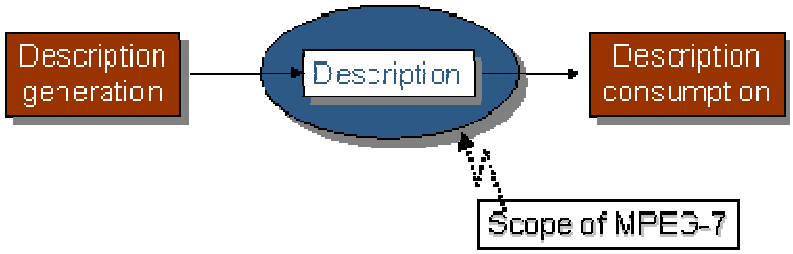


Figure 9: Scope of MPEG-7.

Figure 10 shows the relationship among the different MPEG-7 elements introduced above. The DDL allows the definition of the MPEG-7 description tools, both Descriptors and Description Schemes, providing the means for structuring the Ds into DSs. The DDL also allows the extension for specific applications of particular DSs. The description tools are instantiated as descriptions in textual format (XML) thanks to the DDL (based on XML Schema). Binary format of descriptions is obtained by means of the BiM defined in the Systems part.
Figure 11 explains a hypothetical MPEG-7 chain in practice.

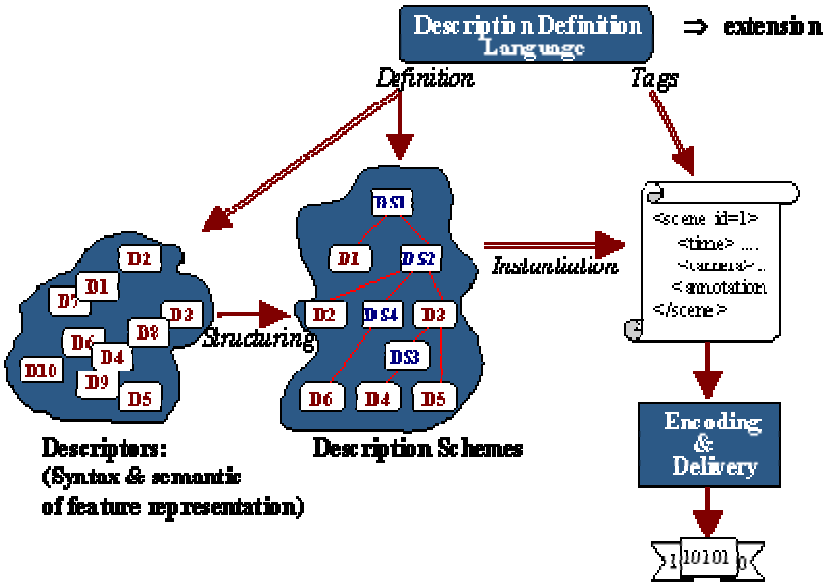


Figure 10: MPEG-7 main elements.

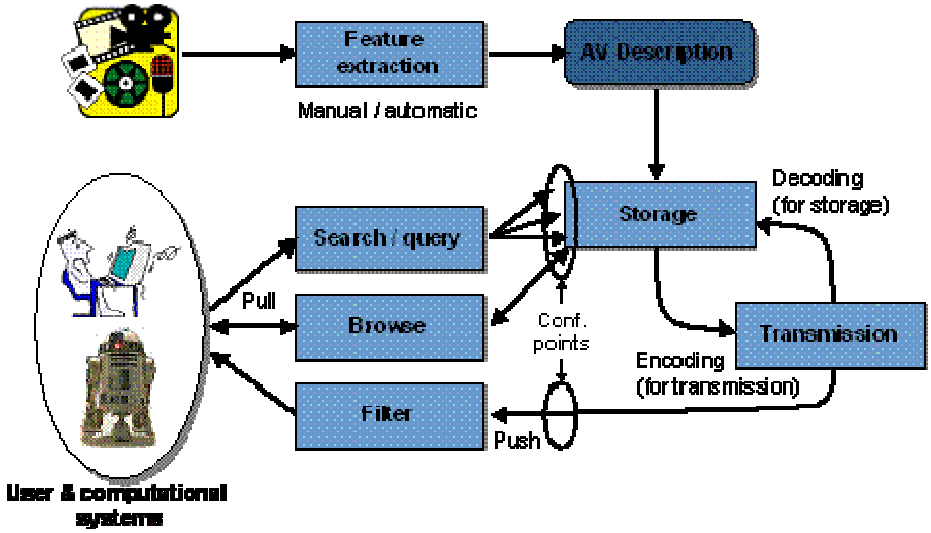


Figure 11: Abstract representation of possible applications using MPEG-7.

4.3.7 Metadata compression using BiM

XML has not been designed to deal ideally in a real-time, constrained and streamed environment like in the multimedia or mobile industry. As long as structured documents (HTML, for instance) were basically composed of only few embedded tags, the overhead induced by textual representation was not critical. MPEG-7 standardizes an XML language for audiovisual metadata. MPEG-7 uses XML to model this rich and structured data. To overcome the lack of efficiency of textual XML, MPEG-7 Systems defines a generic framework to facilitate the carriage and processing of MPEG-7 descriptions: BiM (Binary Format for MPEG-7). It enables the streaming and the compression of any XML documents. BiM coders and decoders can deal with any XML language. Technically, the schema definition (DTD or XML Schema) of the XML document is processed and used to generate a binary format. This binary format has two main properties. First, due to the schema knowledge,

structural redundancy (element name, attribute names, …) is removed from the document. Therefore the document structure is highly compressed (98% in average). Second, elements and attributes values are encoded according to some dedicated codecs. A library of basic datatype codecs is provided by the specification (IEEE 754, UTF_8, compact integers, VLC integers, lists of values ...). Other codecs can easily be plugged using the type-codec mapping mechanism defined in the standard.

4.3.8 References

[1] Australian National Data Service, ANDS guides, http://ands.org.au/guides/metadata-awareness.pdf
[2] A guide for libraries, published by NISO, “Understanding Metadata”, 2001.
[3] Online guide for Metadata, http://map.ns.ec.gc.ca/elearning/ec/english/metadata/common.html.
[4] L. Rein, T. Winningham, “Video metadata model”, http://microformats.org/wiki/video-metadata-model.
[5] Dublin Core Metadata Initiative, www.dublincore.org.
[6] MPEG 7 standard description, ISO/IEC JTC1/SC29/WG11 15938.

4.4 MULTIMEDIA SCENE REPRESENTATION TECHNOLOGIES

Moving Picture Experts Group (MPEG) introduced the MPEG 4 standard (ISO/IEC 14496 - Coding of audio-visual objects) as a collection of representation and compression methods, for dealing with video, audio, graphics and text coding formats. Under this framework, MPEG 4 defines a dedicated description language, called Binary Format for Scene (BiFS) [1] which is able to describe the heterogeneous content of the scene, to manage the scene object behavior (*e.g.* object spin) and to ensure the time/conditional updates (e.g. user input/interactivity).
The novelty of BiFS not only refers to the scene description but also to the scene compression. Traditionally, the heterogeneous visual content was represented by successive frames composing a single video to be eventually compressed by some known codecs (such as MPEG-2 [2] or MPEG-4 AVC [3]). BiFS follows a completely different approach: it allows each object to be encoded with its own optimal coding scheme (video is coded as video, text as text, and graphics as graphics).
The BiFS principles have also been optimized for thin clients and mobile networks purposes, thus resulting in a standard called Lightweight Application Scene Representation (LASeR) [4].
A comparison of existing technologies for heterogeneous content representation, carried out in terms of binary compression, dynamic updates and streaming capabilities is represented in Figure 6:

- Binary compression: Several solutions for the binary encoding of rich media content are already present on market, both on the inside and outside of MPEG worlds (Flash [5], Java [6], SMIL/SVG [7]). On the one hand, LASeR is the only technology specifically developed in order to serve mobile thin devices requiring at the same time strong compression and low complexity decoding. On the other hand, BiFS takes the lead over LASeR with the power of expression, the strong graphics features and the possibility of describing 3D scenes.
- Dynamic updates: While user interaction is nowadays supported by nearly all technologies in Figure 6, dynamic updates on the scene (generated at the server side) are only supported by BiFS and LASeR.
- Live streaming: BiFS and LASeR are the only binary compressed technologies that can be constantly transmitted and presented to the end user, at a rate determined by the content updating mechanism per se. Note that Flash does not support such a distribution mode: it requires for the complete content file to be downloaded to the end client and then displayed.

In the sequel, we shall focalise on the BiFS and LASeR technologies.

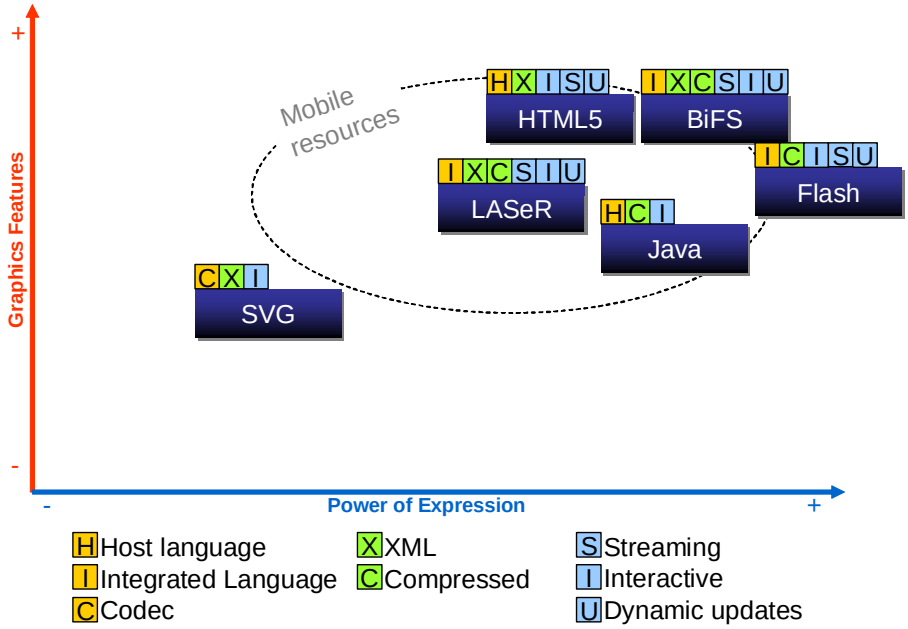


Figure 6: Concurrent solutions for heterogeneous content compression, updating and streaming. Power of expression: possibility of describing complex/heterogeneous scene. Graphics Features: describes the visual quality of the displayed content.

4.4.1 BiFS

BiFS (Binary Format for Scene) referred to as MPEG 4 Part 11, is an MPEG-4 scene description language designed for the optimization of interactive rich-media services (text, audio, video, 2D & 3D graphics) and this at the representation, delivery and rendering levels. Furthermore, in addition to its ancestors (*e.g.* VRML - Virtual Reality Modeling Language), BiFS provides distinguishing mechanisms such as scene updates, binary compression and data streaming.

4.4.1.1 Content representation

In order to support all of these features, BiFS exhibits an object-oriented and a stream-based design. All content is described by a scene-graph, providing a hierarchical representation of audio, video and graphical components. Consequently, each object becomes a BiFS node with abstract interfaces thus affording for the manipulation of its properties, independent of its media. Each node contains objects, either individual or grouped, to be displayed, transformations specifying the spatial coordinates of the objects, and a list of fields defining the particular behavior of the considered node. For example, a Box node has width, height and depth fields specifying the size of the box. MPEG-4 has around 100 nodes with 20 basic field types: boolean, integer, floating point, two- and three-dimensional vectors, time, normal vectors, rotations, colors, URLs, strings, images, and other more arcane data types such as scripts.

4.4.1.2 BiFS compression

The scene-graph structure allows high level description of the graphical content and makes the compression independent of the source media (video, images, audio, etc…): the coding is only performed on the scene-graph description, the underlying media encoding remaining unchanged.

4.4.1.3 User interaction

BiFS provides a well-defined framework for handling the user interaction. In general MPEG-4 covers two types of interaction: client-side and server-side.

- *Client-side interaction* It deals with content manipulation on the end user terminal, where only local scene updates are available: the player captures the user events and uploads the scene corresponding to the present actions on the scene, without contacting the server.
- *Server-side interaction* It supposes that the content manipulation on the end user terminal is sent to the scene source, by using the uplink channel. There are two possible solutions for serving the server-side interaction:
    - *using ECMA script (JavaScript language) [8*]: an object-oriented scripting language used to enable programmatic access to objects, used by the MPEG for enforcing scene capabilities. In order to complete a server side interaction, an XMLHttpRequest[6] object is used to send user interaction information to the server. This solution is commune to BiFS and LASeR.
    - *using "ServerCommand"*: the functionality of the "ServerCommand" signals the server of the occurrence of a event directly from the scene as a result of user interaction. While the ServerCommand enables event routing to the server, the ServerCommandRequest structure specifies the syntax for the messages communicated to the server over a back channel.

4.4.2 LASeR

Properly referred to as MPEG 4 Part 20, MPEG 4 LASeR (Lightweight Application Scene Representation) is designed for representing and delivering rich-media services to re-source-constrained devices such as mobile phones. It addresses the requirements of the end-to-end rich media publication chain: ease of content creation, optimized rich media data delivery and enhanced rendering on all devices. Inspired by the best concepts of state-of-the-art solutions (e.g. W3C/SVG, Macromedia Flash, ISO/IEC MPEG/BIFS), LASeR tunes and optimizes each feature required by Rich Media services to effectively fulfill the need of an efficient open standard.
LASeR uses an SVG scene tree at its core. It imports composition primitives from the different W3C specifications (all of SVG Tiny 1.1 [7], some of SVG 1.1 [9] and SMIL 2 [10]) and uses the SVG rendering model to present the scene tree. Among all its composition primitives, LASeR specifies hyper linking capabilities, audio and video media embedding, vector graphics representations, animation and interactivity features.

4.4.2.1 Content representation

Graphic Animations, Audio, Video and Text are packaged and streamed altogether. Contrary to existing technologies on mobile that are mostly aggregation of various components, not necessarily well integrated together (e.g. XHTML + SMIL + SVG + CSS + Ecmascript + …), LASeR grounds its design from what made the success of Macromedia Flash on the Web: a single, well defined and deterministic component that integrates all the media. This integration ensures both the richness and quality of the end user experience.

4.4.2.2 LASeR compression

LASeR has been designed to deliver Rich Media Service starting from 10 Kb/s. The key technology used here is vector graphics compression and dynamic updates of the scene. This feature enable to drastically limit the waiting time of the endusers as opposed to a standard Web-like approach where

the complete page is re-sent even though only small changes had been made. Needed for low bitrate networks such as GPRS, this functionality is also useful on higher bit rate network where Rich Media services can be sent at low rate, therefore preserving bandwidth to improve audio and video quality.

4.4.2.3 User interaction

The interaction philosophy is abridged by *Full screen and interactivity with all streams*. With the use of vector graphic technology, content can easily be made to fit the screen size. This feature enables to provide an optimal content display although the screen resolution is highly varying. In addition, virtually all pixels can be used as elements of the user interface. This allows the design of rich and user-friendly interfaces, similar to what people used to have when interacting with their devices. The interaction mechanism is similar as BiFS covering two types of interaction: client-side and server-side (by using ECMA Script).

4.4.3 Reference:

[1] MPEG-4 BiFS, ISO/IEC JTC1/SC29/WG11 14496-11.
[2] MPEG-4 BiFS, ISO/IEC JTC1/SC29/WG11 14496-11.
[3] MPEG-2 standard specification, ISO/IEC 13818.
[4] MPEG-4 AVC standard specification, ISO/IEC JTC1/SC29/WG11 14496-10.
[5] MPEG-4 LASeR standard specification, ISO/IEC°JTC1/SC29/WG11 14496-20.
[6] Adobe, http://www.adobe.com/
[7]The Java Language Specification, http://java.sun.com/docs/books/jls/third_edition/html/j3TOC.html
[8] Mobile SVG Profiles: SVG Tiny and SVG Basic, http://www.w3.org/TR/SVGMobile/
[9] Bruno, Eric J. “Ajax: Asynchronous JavaScript and XML,” *Dr. Dobb's Journal*, v. 31, n. 2, February. 2006, p. 32-35
[10] W3C, Scalable Vector Graphics Specification, http://www.w3.org/TR/2002/WD-SVG12-20021115.
[11] Synchronized Multimedia Integration Language (SMIL 2.0), http://www.w3.org/TR/2005/REC-SMIL2-20050107/

5. DATA ENCAPSULATION

5.1 GENERALITIES

Starting from different natural data (image, graphics, video, audio) encoded in elementary streams, and from metadata, the data encapsulation provides a meta-file format whose specification describes how these elements coexist in a file to be stored or in a stream to be on-the-fly consumed.
The meta-file format is called the container Format. The Container format parts may have various names: "chunks" as for PNG, "atoms" in QuickTime/MP4, "packets" in MPEG-TS (from the communications term) and "segments" in JPEG. Some containers are devoted to audio data: WAV (RIFF file format, widely used on Windows platform), and other containers are exclusive to still images: TIFF (Tagged Image File Format) still images and associated metadata.
Flexible containers can hold many types of audio and video, as well as other media. The most popular multi-media containers are:

- 3GP (used by many mobile phones; based on the ISO base media file format)
- Flash Video (FLV, F4V) (container for video and audio from Adobe Systems)
- MP4 (standard audio and video container for the MPEG-4 multimedia portfolio, based on the ISO base media file format defined in MPEG-4 Part 12 and JPEG 2000 Part 12) which in turn was based on the QuickTime file format.

ISO base media file format defines a general structure for time-based multimedia files such as video and audio. It is used as the basis for other media file formats
ISO base media file format contains:

- timing
- structure
- media information for timed sequences of media data

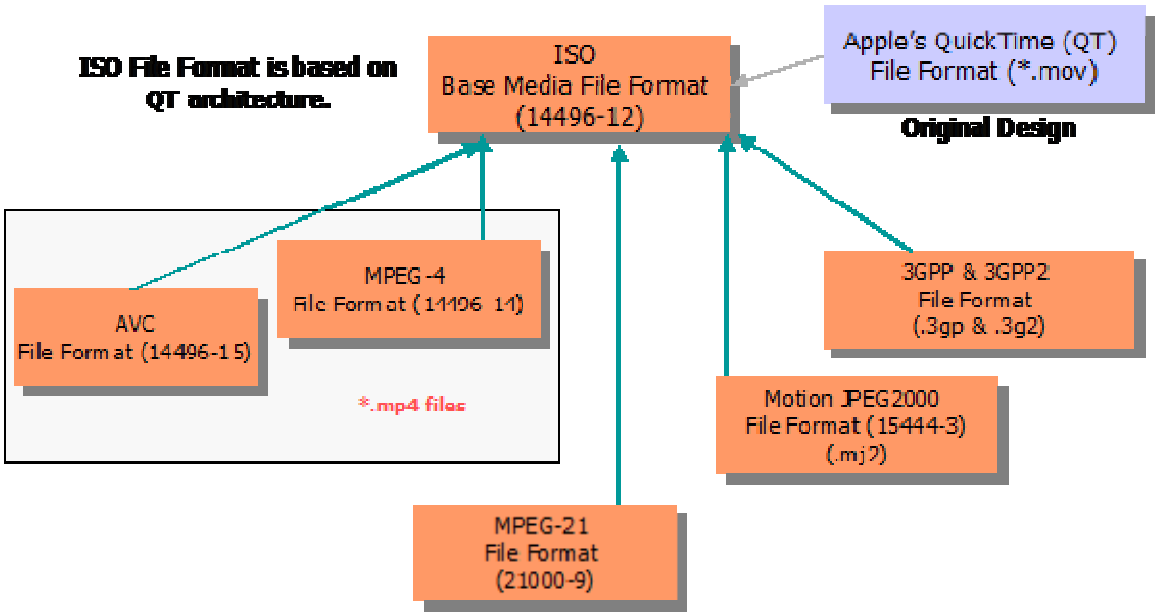


Figure 7: Relationship between the ISO, MP4, AVC, MPEG-21 File Formats.

5.1.1 Structure

Physical

The family of the storage file formats is based in the concept of box-structured files. A box-structured file consists of a series of boxes (sometimes called atoms), which have a size and a type. The types are 32-bit values and usually chosen to be four printable characters, for ease of inspection and editing. Box structured files are used in a number of applications, and it is possible to form 'multi-purpose' files which contain the boxes required by more than one specification. Examples include not only the ISO Base File Format family described here, but also the JPEG 2000 file format family, which for the most part is a still-image file format. There is provision for using extension boxes with a Universal Unique Identifier type (UUID) [4], and specification text is provided on how to convert all box types into UUID's.
All box-structured files start with a file-type box (possibly after a box-structured signature) that defines the best use of the file, and the specifications to which the file complies.
The physical structure of the file separates the data needed for logical, time, and structural de-composition, from the media data samples themselves. This structural information is concentrated in a *movie box*, possibly extended in time by *movie fragment* boxes. The movie box documents the logical and timing relationships of the samples, and also contains pointers to where they are located. Those pointers may be into the same file or another one, referenced by a URL.
Un-timed data may be contained in a metadata box, at the file level, or attached to the movie box or one of the streams of timed data, called tracks, within the movie.

Logical

Each media stream is contained in a track specialized for that media type (audio, video, etc.), and is further parameterized by a sample entry. The sample entry contains the 'name' of the exact media type (i.e., the type of the decoder needed to decode the stream) and any parameterization of that decoder needed. The name also takes the form of a four-character code. There are defined sample entry formats not only for MPEG-4 media, but also for the media types used by other organizations using this file format family. They are registered at the MP4 registration authority [3].
Tracks (or sub tracks) may be identified as alternatives to each other, and there is support for declarations to identify what aspect of the track can be used to determine which alternative to present, in the form of track selection data.

Time

Each track is a sequence of timed samples; each sample has a decoding time, and may also have a composition (display) time offset. Edit lists may be used to over-ride the implicit direct mapping of the media timeline, into the timeline of the overall movie.
Sometimes the samples within a track have different characteristics or need to be specially identified. One of the most common and important characteristic is the synchronization point (often a video I-frame). These points are identified by a special table in each track. More generally, the nature of dependencies between track samples can also be documented. Finally, there is a concept of named, "parameterized sample groups". Each sample in a track may be associated with a single group description of a given group type, and there may be many group types.

5.1.2 Application

Meta data support

Support for meta-data takes two forms.

First, timed meta-data may be stored in an appropriate track, synchronized as desired with the media data it is describing.
Secondly, there is general support for non-timed collections of metadata items attached to the movie or to an individual track. The actual data of these items may be in the metadata box, elsewhere in the same file, in another file, or constructed from other items. In addition, these resources may be named, stored in extents, and may be protected. These metadata containers are used in the support for file-delivery streaming, to store both the 'files' that are to be streamed, and also support information such as reservoirs of pre-calculated Forward Error-Correcting (FEC) codes.
The generalized meta-data structures may also be used at the file level, above or parallel with or in the absence of the movie box. In this case, the meta-data box is the primary entry into the presentation. This structure is used by other bodies in order to wrap together other integration specifications (*e.g.* SMIL [9]) with the media integrated.

Streaming support

There is support for both streams prepared for transmission by streaming servers, and also for streams recorded into files, in the form of hint tracks, server and reception.
When media is delivered over a streaming protocol it often must be transformed from the way it is represented in the file. The most obvious example of this is the way media is transmitted over the Real Time Protocol (RTP) [6]. In the file, for example, each frame of video is stored contiguously as a file-format sample. In RTP, packetization rules specific to the codec used, must be obeyed to place these frames in RTP packets.
A streaming server may calculate such packetization at run-time if it wishes. Hint tracks contain general instructions for streaming servers as to how to form packet streams, from media tracks, for a specific protocol. Because the form of these instructions is media-independent, servers do not have to be revised when new codecs are introduced. In addition, the encoding and editing software can be unaware of streaming servers. Once editing is finished on a file, then a piece of software called a hinter may be used that adds hint tracks to the file, before placing it on a streaming server.
There are defined formats for server and reception hint tracks for RTP and MPEG-2 Transport, and there is a server hint track format for streaming file delivery (FD) hint tracks, that can be used to support protocols such as FLUTE [10].

Protection

Protected streams (such as encrypted streams, or those controlled by a Digital Rights Management (DRM) system) are also supported by the file format (*e.g.* streams encrypted for use in DRM). There is a general structure for protected streams, which documents the underlying format, and also documents the protection system applied and any parameters it needs.
This same protection support can be applied to the items in the metadata.

5.2 BASIC MPEG-4 ENCAPSULATION

5.2.1 Definition

The MP4 file format is designed to contain the media information of an MPEG-4 presentation in a flexible, extensible format which facilitates interchange, management, editing, and presentation of the media. This presentation may be 'local' to the system containing the presentation, or may be via a network or other stream delivery mechanism (a TransMux). The file format is designed to be independent of any particular delivery protocol while enabling efficient support for delivery in general. The design is based on the QuickTime® format from Apple Computer Inc.

5.2.2 Structure

The MP4 file format is composed of object-oriented structures called ‘atoms’. A unique tag and a length identify each atom. Most atoms describe a hierarchy of metadata giving information such as index points, durations, and pointers to the media data. This collection of atoms is contained in an atom called the ‘movie atom’. The media data itself is located elsewhere; it can be in the MP4 file, contained in one or more ‘mdat’ or media data atoms, or located outside the MP4 file and referenced via URL’s.

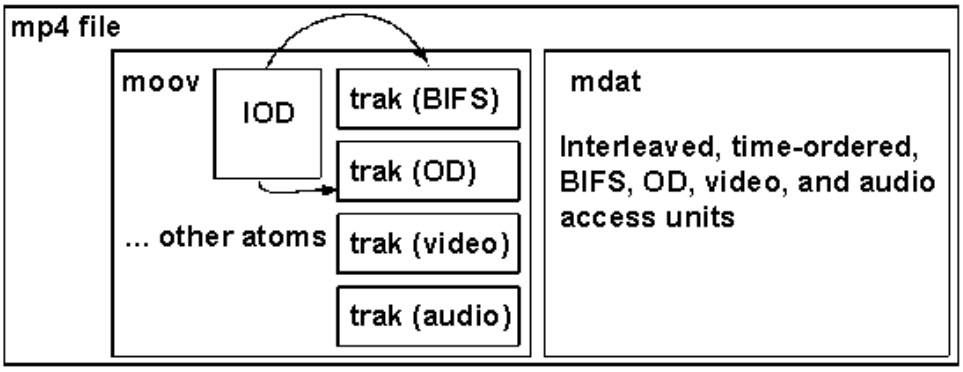


Figure 8: Example of a simple interchange file, containing three streams.

The MP4 file format is composed of object-oriented structures called ‘atoms’. A unique tag and a length identify each atom. Most atoms describe a hierarchy of metadata giving information such as index points, durations, and pointers to the media data. This collection of atoms is contained in an atom called the ‘movie atom’. The media data itself is located elsewhere; it can be in the MP4 file, contained in one or more ‘mdat’ or media data atoms, or located outside the MP4 file and referenced via URL’s.

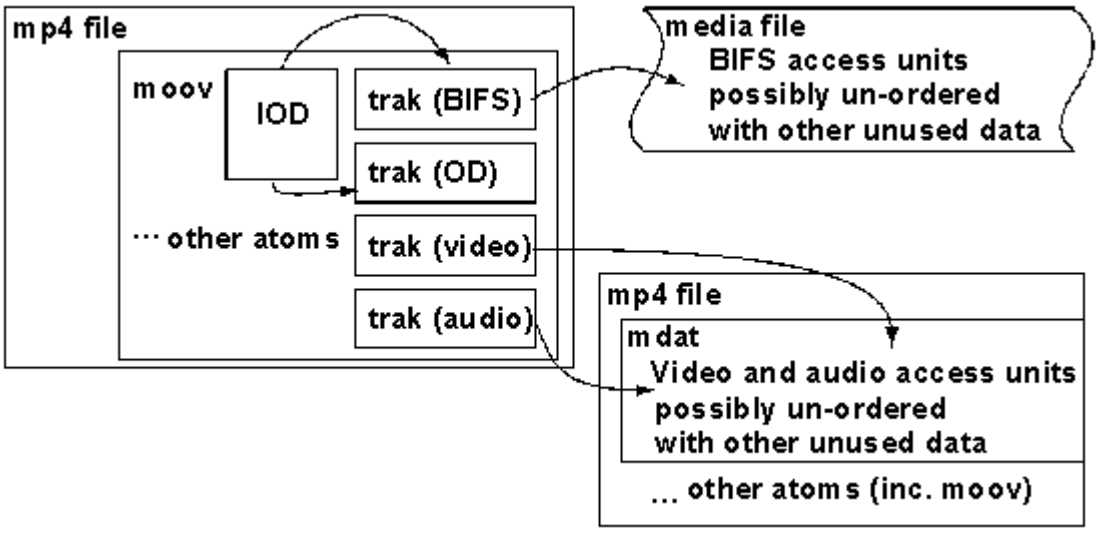


Figure 9: Complex file with external media data.

The metadata in the file, combined with the flexible storage of media data, allows the MP4 format to support streaming, editing, local playback, and interchange of content, thereby satisfying the requirements for the MPEG4 Intermedia format.

5.2.3 Examples

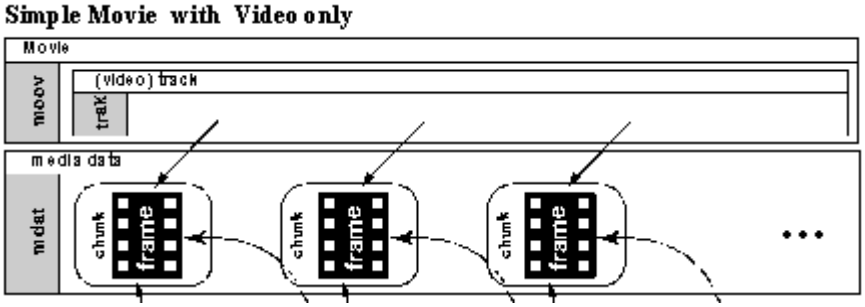
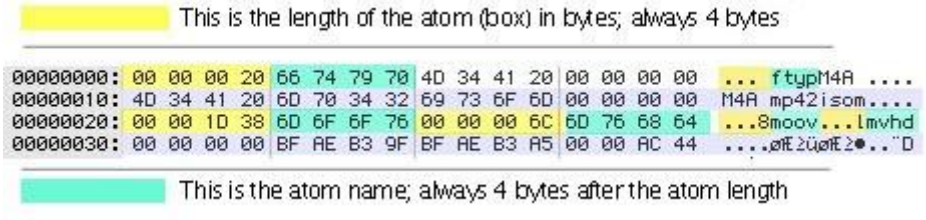


Figure 10: sample showing simple movie file with video track only.

For physical structure, an atom has a format:



The ftyp atom is ALWAYS first, and has a certain type of format - it tells what type of file it is & the basic versioning of the atom structures.
In the above example the mov atom has a length of 0x00001D38 or 7480 bytes. Immediately following the moov name however is a new atom. This is the mvhd atom, and its length is 0x0000006C or 108 bytes.
The minimum length of an atom then would be 8 bytes.
The 'Atom Is A Parent Or Holds Data' rule is made to be broken. Often the atom under moov.trak.mdia.minf.stbl.stsd is a parent and contains data. Apple's drm implementation breaks this rule further. The other standard atom that breaks this rule is moov.udta.meta for historical reasons. Still, the MPEG-4 container is relatively easy to understand and highly flexible.
The most important part of a Quicktime file is the mdat atom which actually holds the raw information for the file is stored. This top level atom takes up the bulk of a MPEG-4 file. However, the moov atom comprises a number of different atoms and hierarchies, and provides for basic functionality - like specifying the dimensions of a video file, or the duration of a song.
uuid atoms are user-defined atoms, and are similar to normal atoms, but their name is 8 bytes (4 bytes holding uuid and the name of the uuid atom). Sony PSP mp4 files notably use uuid atoms. AtomicParsley supports setting & reading its own uuid atoms to carry supplemental metadata.

Known iTunes Metadata Atoms
Metadata to be used with iTunes comes in the moov.udta.meta.ilst hierarchy. The atoms directly under the ilst atom have specific names, but they do not carry the data directly. The children of these named atoms (the data atom) carry the actual information. The 4 letter code of the parent is listed below, while the atom flags after the data atom are listed in the Class column. It is the class of the data atom that broadly determines whether text or numbers or binary data is contained.

4char code	Name	Class/Flag		Appearance
©alb	Album	1	text	iTunes 4.0
©art	Artist	1	text	iTunes 4.0
aART	Album Artist	1	text	??
©cmt	Comment	1	text	iTunes 4.0
©day	Year	1	text	iTunes 4.0
©nam	Title	1	text	iTunes 4.0
©gen | gnre	Genre	1 | 0 1	text | uint8	iTunes 4.0
trkn	Track number	0	uint8	iTunes 4.0
disk	Disk number	0	uint8	iTunes 4.0
©wrt	Composer	1	text	iTunes 4.0
©too	Encoder	1	text	iTunes 4.0
tmpo	BPM	21	uint8	iTunes 4.0
cprt	Copyright	1	text	? iTunes 4.0
cpil	Compilation	21	uint8	iTunes 4.0
covr	Artwork	13 | 14 2	jpeg | png	iTunes 4.0
rtng	Rating/Advisory	21	uint8	iTunes 4.0
©grp	Grouping	1	text	iTunes 4.2
stik	?? (stik)	21	uint8	??
pcst	Podcast	21	uint8	iTunes 4.9
catg	Category	1	text	iTunes 4.9
keyw	Keyword	1	text	iTunes 4.9
purl	Podcast URL	21 | 0 4	uint8	iTunes 4.9
egid	Episode Global Unique ID	21 | 0 4	uint8	iTunes 4.9
desc	Description	1	text	iTunes 5.0
©lyr	Lyrics	1 3	text	iTunes 5.0
tvnn	TV Network Name	1	text	iTunes 6.0
tvsh	TV Show Name	1	text	iTunes 6.0
tven	TV Episode Number	1	text	iTunes 6.0
tvsn	TV Season	21	uint8	iTunes 6.0
tves	TV Episode	21	uint8	iTunes 6.0
purd	Purchase Date	1	text	iTunes 6.0.2
pgap	Gapless Playback	21	uin8	iTunes 7.0

5.2.4 Reference:

[1] ISO/IEC 14496-12, ISO Base Media File Format; technically identical to ISO/IEC 15444-12
[2] ISO/IEC 14496-14, MP4 File Format
[3] ISO/IEC 14496-15, Advanced Video Coding (AVC) file format
[4] ISO/IEC 14496-10, Advanced Video Coding
[5] ISO/IEC 21000-9, MPEG-21 File Format
[6] ISO/IEC 15444-1, JPEG 2000 Image Coding System
[7] The MP4 Registration Authority, http://www.mp4ra.org/
[8] ISO/IEC 9834-8:2004 Information Technology, "Procedures for the operation of OSI Registration of Universally Unique Identifiers (UUIDs) and their use as ASN.1 Object Identifier components" ITU-T Rec. X.667, 2004
[9] SMIL: Synchronized Multimedia Integration Language; World-Wide Web Consortium (W3C) http://www.w3.org/TR/SMIL2/
[10] RTP: A Transport Protocol for Real-Time Applications; IETF RFC 3550, http://www.ietf.org/rfc/rfc3550.txt

6. SESSION LAYER PROTOCOL

This section describes the session layer protocols. In today's video systems, the most common standardised session layer protocols are the Real-Time Streaming Protocol (RTSP) and HyperText Transfer Protocol (HTTP). They are used in most video streaming solutions to provide video session set-up and control between a video source and a video client.

6.1 REAL-TIME STREAMING PROTOCOL (RTSP)

Real-Time Streaming Protocol (RTSP) is standardised in IETF RFC 2326 and currently being revised (at the time of writing in April 2011) by the IETF draft draft-ietf-mmusic-rfc2326bis-27 - Real Time Streaming Protocol 2.0.

RTSP is often considered as a « remote control » protocol because it enables the following features from a multimedia client to a multimedia server:

- SETUP: to establish the transport method
- PLAY: to request the start of the video transmission
- PAUSE: to stop the video transmission temporarily
- TEARDOWN: to stop the video transmission

RTSP relies on the TCP or UDP protocols for transport and it often used with the Session Description Protocol or XML for the session description. It is a text-based protocol an similar to HTTP.
An RTSP-based multimedia session message exchange is shown in Figure 11.

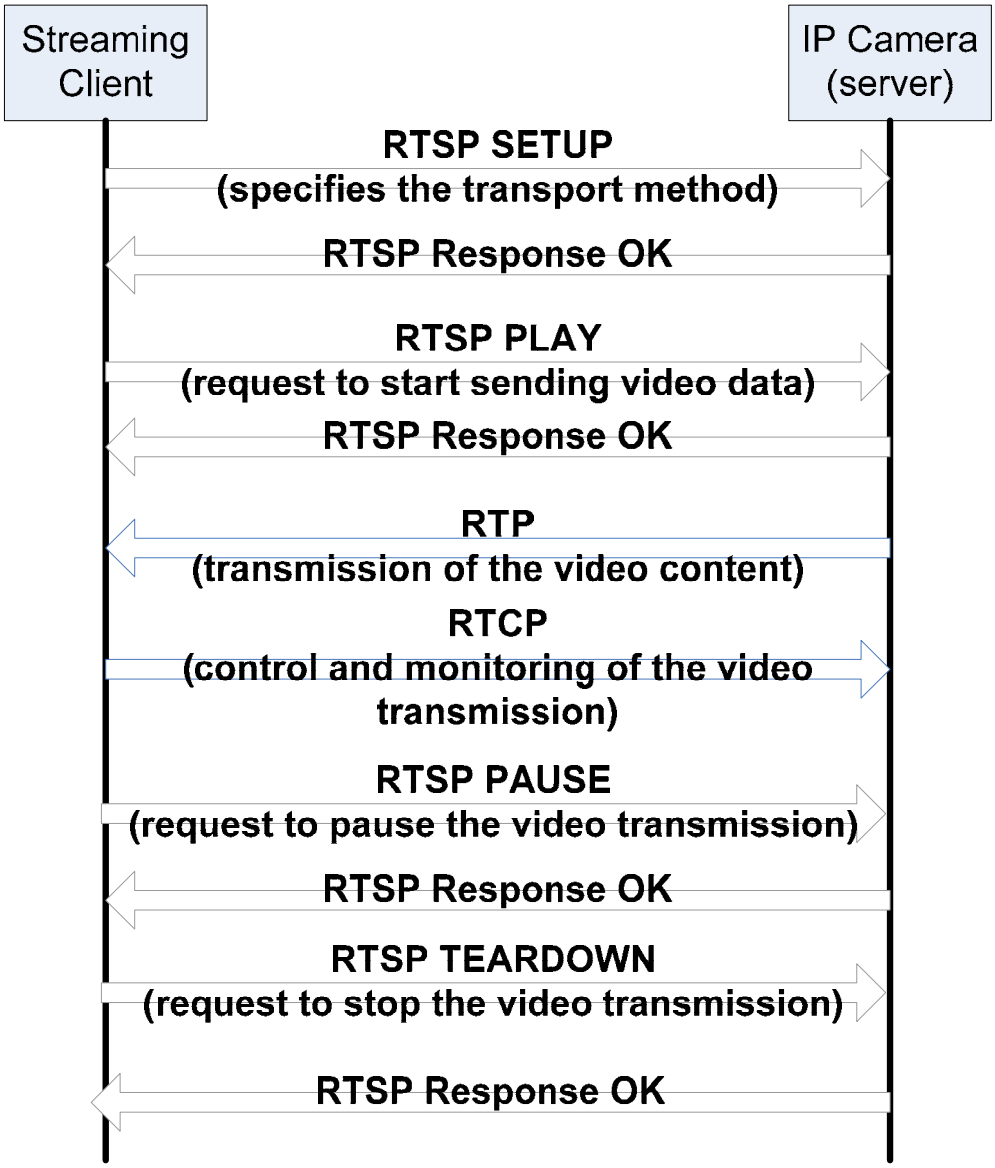


Figure 11: Multimedia session based on RTSP.

Advantages

Limitations

Relevance to SPY

6.2 HYPERTEXT TRANSFER PROTOCOL (HTTP)

HTTP is standardised by the IETF in RFC1945 (HTTP/1.0) and RFC2616 (HTTP/1.1).

In today's video systems and IP cameras, the HyperText Transfer Protocol (HTTP) is used for the following purposes:

- configuration of the system elements (for example IP camera codec parameters, IP address, etc.)
- Live view of video with direct access to IP cameras using a standard web browser.

7. NETWORK BASED ADAPTATION

Several network-based adaptation techniques exists:

- Feedback-based mechanisms to be used with layered video coding or dynamic encoder parameter modification
- Joint video channel coding (JVCC)

7.1 FEEDBACK-BASED ADAPTATION

When sending video stream to the control centre (CC), a field team (FT) must respect the following constraints:

- Use best possible video quality,
- Not overload wireless network uplink channel.

For that purpose, before sending video data on the network, a bandwidth negotiation between the FT and the CC must take place. Moreover, two phenomena linked to FT mobility make that network conditions vary with time:

- When a FT moves near or away from the network base station he/she is connected to, the bandwidth should also increase or decrease,
- The available bandwidth for a base station is function of the number of FTs connected to this BS and the bandwidth consumed by each FT.

A simple protocol can be envisioned to negotiate bandwidth allocated to FTs:

1. FT asks the CC available bandwidth,
2. CC considers the actual network load (number of video uploads from the same BS…),
3. CC sends the FT available bandwidth,
4. FT uses video encoding parameters in accordance with available bandwidth.

Moreover, on a continuous basis the following operations are executed:

- CC continuously monitors network load,
  - When network load evolves, CC may send new available bandwidth to uploading FTs,
- FT continuously monitors wireless signal quality,
  - When wireless signal quality evolves, FT may need to change video encoding parameters to respect actual available bandwidth.

This protocol is illustrated in Figure 12.

Figure 12: Video adaptation mechanism.

The Real-Time Streaming Protocol (RTSP) is appropriate for implementing such a mechanism.

7.2 JOINT VIDEO CHANNEL CODING (JVCC)

Separation of source coding and channel coding is a major principle in traditional video communication systems: video source is first encoded to a bit stream which is then channel encoded and modulated, as illustrated in Figure 13. Source-channel separation leads to modular system design that allows independent optimization of source and channel coders. It also allows interoperability by providing a common digital interface. While simple to implement (no modification of video encoder), this mechanism does not allow to exploit information about video signal when protecting bit stream against errors. For example, the video components and their bit representation are not equally important and we may decide to give a higher protection level to more important video information (e.g. I pictures); this is possible only if information about video stream is kept and passed to the channel coding and modulation module. Besides separation theorem assumes known channel capacity, which is rarely the case in wireless channel.

The joint source-channel coding is the approached proposed by JVCC. The latter consists in regrouping video encoding and channel coding and modulation steps in a single step, as illustrated in Figure 13.

Information about video stream is thus available during the whole process and can be exploited at any moment. With this approach, it is possible to implement Unequal Error Protection (UEP) mechanism to give different protection levels to video information of different importance.

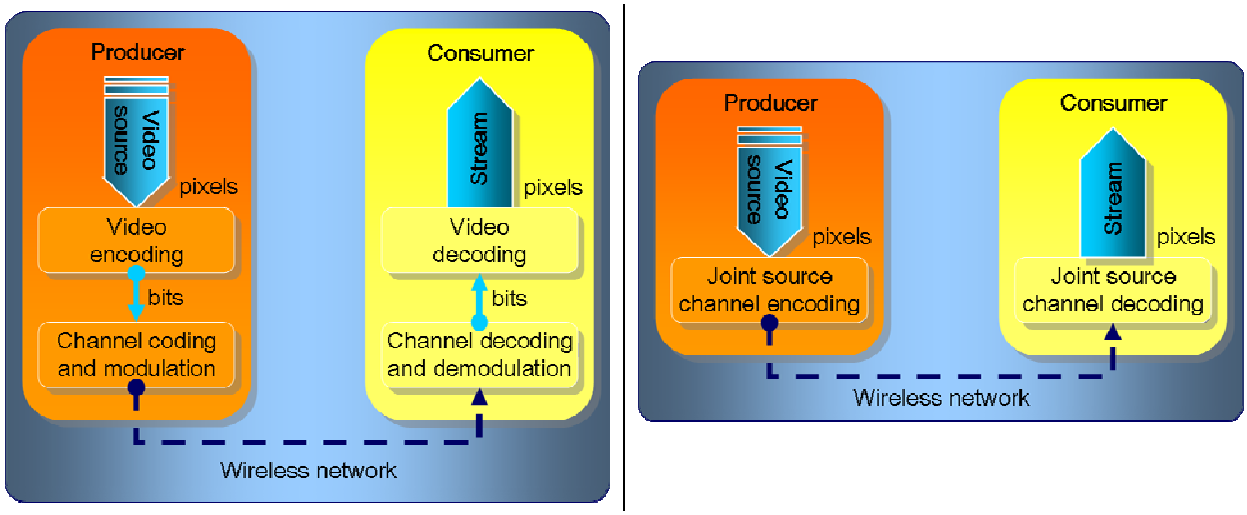


Figure 13: Classical video encoding vs joint video channel coding.

The JVCC is composed of three elements, as detailed in Figure 14:

- Video processing and representation to prioritize the video components according to their importance,
- Unequal error protection to encode the most significant bits of the important components better than the least significant bits of less important components,
- Combination of modulation and UEP to generate the proper constellation in the channel size space.

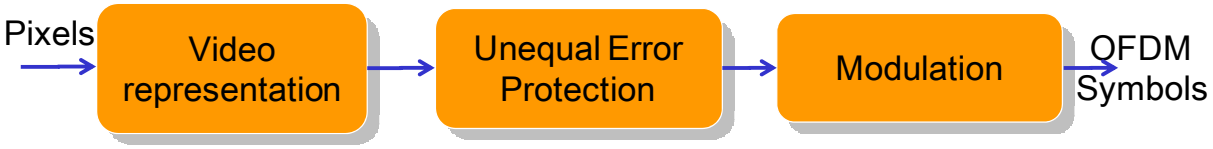


Figure 14: JVCC elements.

8. DATA & STREAM INTEGRITY

Ensuring that digital objects (videos) have not been corrupted over time or in transit on SPY network is one of the most crucial and challenging security issue of SPY network management. In brief, we must implement video (data and stream) integrity techniques.

The most used method for doing this is digital watermarking. Subsection 8.1. will detail the state-of-the-art of this technical domain. Subsection 8.2. will give some alternative technologies, and subsection 8.3. will list bibliographical references.

8.1 WATERMARKING TECHNOLOGIES

The amount of high quality digital video data is ready available on the Internet so that users can conveniently be able to enjoy watching on-line video, transmit and exchange video files. Digital video is also useful in many other applications: surveillance video systems and broadcasting are good examples. However, at the same time a number of security problems have been introduced, since digital video sequences are very susceptible to manipulations and alterations using widely available editing software. This way video content is not reliable anymore. For example, a video shot from a surveillance camera cannot be used as a piece of evidence in a courtroom because it is not considered trustworthy enough. Therefore, authentication techniques are consequently needed in order to ensure the authenticity and integrity of video content. Till date, there have been various such techniques [1], of which digital watermarking is one of the most popular. Digital watermarking is a technique that embeds a secret, unnoticeable signal (called watermark) into the original multimedia object, like audio, image and video. The watermark can be detected or extracted later to claim the authenticity of the media content.

From the information theory point of view, the watermarking process can be considered as a communication system with side-information at the encoder as shown in figure 15. Using a secure key *k,* the watermark message *m* is embedded into the host signal *x*. The watermarked signal *s* is then transmitted over the channel which can introduce a signal *n* resulting from an attack. The decoder receives the signal *r* and, using the same key as *k* which using during the embedding process, extracts the watermark message *m'*

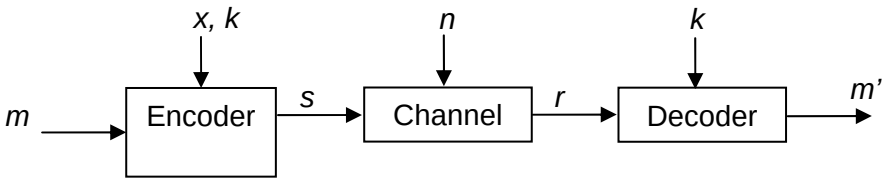


Figure 55: Watermarking synopsis diagram.

There is no universal requirement to be satisfied by all watermarking applications. Nevertheless, some general directions can be given for most of the applications. In order to be effective, the watermark should be perceptually invisible (transparent) and robust against the attacks; moreover, it should allow the insertion of a sufficient amount of information (*e.g.* a serial number identifying a user, a time stamp, etc.):

- **Transparency**: watermarking should be imperceptible and invisible to a human observer and the embedded watermark should not affect the quality of the host data. In this sense, two concepts are considered: fidelity and quality. A watermarking technique is faithful if no difference can been seen in the host media and the marked media. However, a watermarking has a good quality if artifacts are not disturbing for a human observer.
- **Unremovable**: They do not get removed when Works are displayed or converted to other file formats.
- **Data-payload**: the amount of information which can be reliably embedded and extracted from the host data.
- **False-positive rate**: the frequency with which we should expect watermarks to be falsely detected in unwatermarked content.
- **Robustness**: the watermark should be successfully recovered even if the watermarking method is public, and must survive normal and malicious processing of content.
- **Fragility**: the watermark cannot be detected after slightest modifications.
- **Security**: only an authorized party can insert/detect the presence of the watermark. Watermarks must resist hostile attacks.
- **Oblivious:** the watermark should be extracted without using the original host.

Moreover, watermarks offer the advantage over other techniques that they undergo the same transformations as videos in which they are embedded.

In order for the watermarking techniques to be easily integrated into practical applications, additional requirements are imposed:

- **Low complexity**: watermarking method should not be computationally complex, *i.e*. they should not require sophisticated operations, like decoding/re-encoding, spectral representations, *etc.*; one way of achieving such a desideratum is to consider compressed domain watermarking techniques, *i.e.* techniques inserting the mark directly into the compressed stream.
- **Constant bit-rate**: watermarking method should not increase the size of the compressed data and the bit-rate, at least for constant bit-rate applications where the transmission channel bandwidth has to be obeyed.

The required degree of each constraint presented above depends on the watermarking application. It must be noted that if a watermarking system is improved so that it has better performance in one property (such as robustness, security, transparency), this can often be traded for better performance in other properties.

Benchmarking is a reasonable means of comparing watermarking systems. However, no benchmark is likely to be relevant to all applications. The following section presents various applications in which watermarking methods are investigated.

Watermarking applications are illustrated in Table 1. For a more thorough investigation we can refer to [2-3].

Table 1: Applications and purposes.

Applications	Purpose
Copyright protection	Proof of ownership
Video authentication	Detect that the original content has been alternated or not
Video enrichment	Enhance video content and make it more interactive

Copyright protection: For the protection of intellectual property, the video data owner can embed a watermark representing copyright information in his data. Thus, watermarks provides owner identification (by embedding the identity of videos' copyright holder), proof of ownership (by providing evidence in ownership dispute), transaction tracking (by identifying people who obtain content legally but illegally redistribute it).This type of application requires a maximum strength of robustness.

Video authentication: With the ease of visual data modification in the digital domain, unauthorized alterations could be made without any perceptible trace. Consequently, the video recorded by a video surveillance system has no value as evidence in court unless the integrity can be verified. This type of application requires robustness against mundane video processing operations but fragility against malicious modification attacks. Watermarks embed signature information in content that can be later checked to verify it has not been tampered with.

Video enrichment: Watermarking can be useful for video enrichment applications. At any time viewing, the user can click on an element of the video to extract information that has already been embedded about that item. This type of application usually requires the insertion of a large amount of information compared to the rest of the conventional video applications.
Watermarking techniques can be divided into different categories according to various criterions [4]. The general classification of the currently available watermarks is shown in Figure 16.

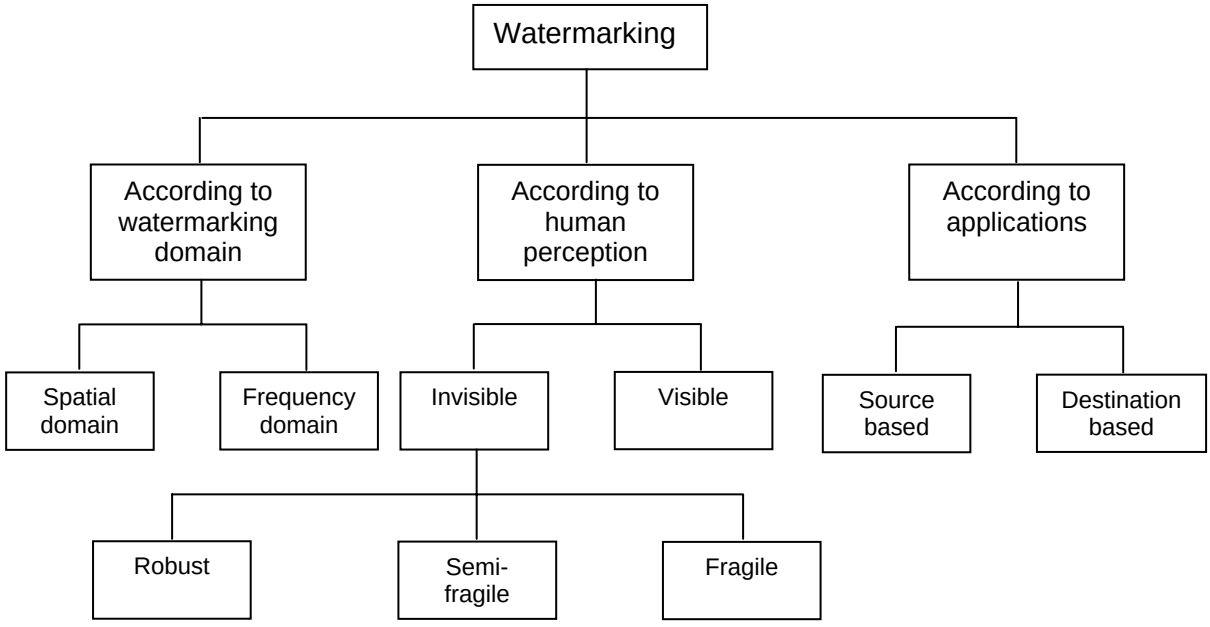


Figure 16: General classification of watermarking techniques.

According to the domain in which video watermarking is performed, watermarking processing methods can be classified into two categories: spatial domain and frequency domain. In the spatial domain, directly applying minor changes to the values of the pixels in a minor way is mainly used. This technique makes the embedded information hardly noticeable to the human eye. For example, pseudo-random watermarking works by a simple addition of a small amplitude pseudo-noise signal to the original media data. In the frequency domain, the object first goes through a certain transformation (as the DCT or the DWT, for instance), then the watermarking is embedded in the transform coefficients which are finally inverse transformed to obtain the watermarked data. The frequency domain methods are more robust than the spatial domain techniques.

According to the human visual system, watermarking techniques can be divided into two different types: visible and invisible. Transparent watermarking techniques are further sub-divided into three classes: robust, semi-fragile and fragile, according to their robustness degree.

From the application point of view, digital watermarking could be source based or destination based. Source based watermarking can be used to authenticate whether a received media data has been manipulated and the destination based watermarking can trace the source of illegal copies.

In the following section, we investigate the state of the art of watermarking techniques based on their robustness (robust, semi-fragile and fragile).

8.1.1 Robust watermarking

Robust watermarking techniques should survive to any change or alteration of the protected data. They are primarily used in applications such as copyright protection, which require the algorithm to be as robust as possible so that severe modifications and degradations cannot remove the watermark.

Many robust watermarking schemes have been proposed, consisting in either spatial domain or transform domain watermarks. The main issue addressed for these schemes is the robustness of watermarks against various intentional or unintentional alterations.
To achieve the desired robustness, many watermarking techniques are based on spread-spectrum techniques [5].

Zang [6] present a spread-spectrum based robust technique. The watermark is spread into random sequence to be embedded in mid-frequency coefficients of transformed blocs. Good robustness against Gaussian filtering, contrast modification and noise addition is ensured.

F. Hartung *and* B. Girod [7] propose a watermarking method which can be applied directly in the compressed stream by modifying only Intra images. The watermark is inserted into TCD coefficients of blocks whose size is less than original blocks, to not increase the flow of the video.

The most recent approach is based on Quantized Index modulation (QIM), proposed by B. Chen and W. Wornell. The QIM superiorities include better robustness, large data payload and less complexity. But the traditional *QIM* has worse transparency because of the invariable modulation step. Therefore, many adaptive watermarking schemes have been proposed [8].

Spread transform dither modulation (ST-DM) is a particular form of QIM. The watermark is not directly embedded into the original signal but into its projection onto a randomly generated normed vector.

A. Golikeri, P. Nasiopoulos and Z. J. Wang [9] propose an ST-DM watermarking method. Although improving the performances of the traditional ST-DM, this method features a quite small data payload and has no robustness against the geometric attacks.

8.1.2 Fragile watermarking

Fragile watermarking techniques is designed to reflect even slightest manipulation or modification of the media data, since the embedded watermark can easily become altered or destroyed after common attacks, such as compression, cropping and spatial filtering.

The purpose of fragile watermarks is to check the integrity and authenticity of digital contents. Fragile digital watermarking is of great importance in courtroom defense, reliable e-business, medical image database, *etc*. When the content of multimedia is suspected, the extraction of fragile watermark can be used to detect and localize tampers, even present the category of tampering.

Lu *et al*. [10] present a simple watermarking scheme for image authentication, fragile against the quantization attack; the principle consists in breaking the position relationship among the watermark and watermarked image. They employed two secret keys to protect the watermark against possible attacks. Besides, their method also embed binary watermark into the cover image by using random permutations and XOR operations. Meanwhile, they claimed that their scheme is not only secure and fast but also can detect and localize the modification position.

C. C. Wang and Y. C. Hsu [11] present a new fragile watermarking algorithm for verifying the integrity of the H.264 stream. The proposed watermarking scheme is based on the residual macro-block which is the subtraction between current macro-block and predicted macro-block. The proposed scheme uses the MD5 hash function for watermarking purposes to enhance the strength of keeping the integrity of the watermarked multimedia content.
The simulation results confirmed that the proposed watermarking scheme is capable of establishing the integrity of the bit-streams of the H.264 codec. The method detects GOP or frame removal. Experiment results reveal the ability of watermarking scheme to identify an unauthorized recompression of the original video content.

8.1.3 Semi-fragile watermarking

A semi-fragile watermark combines the properties of fragile and robust watermarks offering a level of tolerance to some “acceptable” alterations. Like a robust watermark, a semi-fragile watermark is capable of tolerating some degree of change to the watermarked data, such as the addition of quantization noise from compression. On the contrarily, like a fragile watermark, the semi-fragile watermark is capable of localizing regions of the image that have been tampered and distinguishing them from regions that are still authentic. The primary application of semi-fragile watermarking techniques is to differentiate between content-changing alterations and content-preserving alterations.

Semi-fragile watermarking techniques can be classified into two classes according to their purposes: watermarking techniques for privacy protection and watermarking techniques for data authentication and integrity.

Privacy protection:
F. Dufaux and T. Ebrahimi [12] address the problem of privacy protection in video surveillance. They introduce two efficient approaches to conceal Regions Of Interest (ROIs) based on transform-domain or code-stream-domain scrambling. In the first technique, the sign of selected transform coefficients

is pseudo randomly flipped during encoding. In the second method, some bits of the code-stream are pseudo-randomly inverted. Simulation results show that the proposed scrambling techniques are successful at concealing privacy-sensitive information while leaving the scene comprehensible.
P. Meuel *et al*. [13] propose a method to protect faces in video surveillance scenes. Their method deletes any visible information of faces in a video and uses a data-hiding technique to embed information in the video that allows further reconstruction of the faces if needed.

Authentication and integrity verification:
In their paper [14], S. Chen and H. Leung propose an authentication scheme based on chaotic semi-fragile watermarking. The timing information of video frames is modulated into the parameters of a chaotic system. The system output, which is a noise-like signal, is used as a watermark and embedded into the block-based discrete cosine transform domain. The embedded information is demodulated by a maximum likelihood estimator. The GOP index, ig, and the frame index, ir, within a Group Of Pictures (GOP) are used as the timing information. A mismatch between the demodulated and the observed timing information indicates the existence of temporal tampering. Meanwhile, inspecting the individual watermark components is able to reveal spatial tampering. Experimental results show the effectiveness of their chaotic scheme, the embedded timing information is robust to common spatial processing, such as JPEG compression, median filtering and contrast enhancement.
S. Thiemert *et al*. [15] present a semi-fragile watermarking scheme for authenticating intra-coded frames in compressed digital videos. The scheme provides the detection of content-changing manipulations while being moderately robust against content-preserving manipulations. They describe a watermarking method based on invariant features referred to as interest points. The features are extracted using the Moravec-Operator. Out of the interest points they generate a binary feature mask, which will be embedded robustly as watermark into the video.

For each I frame, a binary feature mask is generated. Using a robust watermarking scheme they embed the binary feature mask into the underlying middle and high frequency DCT coefficients of one of the adjacent I-frames. For instance *I*-frame *n* may contain the binary feature mask of *I*-frame *n+1*. Results show the robustness to content- preserving manipulations and fragility to content-changing manipulations. The scheme detects adding or deleting objects.

8.2 ALTERNATIVE TECHNOLOGIES

There are other alternative technologies for data protection. In the following section, we focus on the steganography technology.

In order to understand the differences between Digital Watermarking and Steganography, we use the precise definitions given in [1]:

- We define Steganography as the practice of undetectably altering a video recording to embed a secret message.
- We define Digital Watermarking as the practice of imperceptibly altering a video recording to embed a message about that video.

Steganography is derived from the Greek words for covered writing and essentially means “to hide in plain sight”. As defined by Cachin [16], steganography is the art and science of communicating in such a way that the presence of a message cannot be detected. Differences between steganography and watermarking are both subtle and essential.

The main goal of steganography is to hide message *m* in a host data *d*, to obtain new data *d'*, practically indistinguishable from *d*, by attackers, in such a way that the presence of *m* cannot be detected in *d'*. However, the main goal of watermarking is to hide message *m* in a host data *d*, to obtain new data *d'*, practically indistinguishable from *d*, by attackers, in such a way that the message *m* cannot be replaced or removed in *d*.
Steganography provides a means of secret communication which cannot be removed without significantly altering the data in which it is embedded. The embedded data will be confidential unless an attacker can find a way to detect it.
There are different applications for steganography. It can be used to hide a message intended for later retrieval by a specific individual or group. In this case the aim is to prevent the message being detected by any other party. The other major area of steganography is copyright marking, where the message to be inserted is used to assert copyright over a document.

8.3 DATA & STREAM INTEGRITY ON PMR NETWORKS

In this subsection, we will detail data and stream integrity mechanisms used on PMR networks.

8.3.1 The Context of PMR Systems

Within the context of PMR systems, the verification of PMR systems, the verification of the integrity and the authentication of the origin of a signal consist in verifying that the signal has not intentionally corrupted by a malicious third party. The aim is, for each mobile terminal, to verify that the radio signal received originates from a base station of the system, and not from a pirate base station, and *vice versa*, for each base station to verify that a radio signal received originates from a mobile terminal of the system, and not from a pirate mobile terminal. Stated otherwise, this check makes it possible to detect attacks against the system which consist in sending a message having the characteristics (synchronization, protocol format, coding, etc.) of a radio message of the system, but while nevertheless being a false message or a message falsified by an adversary who has intercepted an authentic message [17].

8.3.2 Non-Malicious *vs.* Malicious Threats to Data Integrity

The techniques required to provide data integrity on noisy channels differ substantially from those required on channels subject to manipulation by adversaries. *Checksums* or *Cyclic Redundancy Codes (CRCs)* provide protection against accidental or non-malicious errors on channels, which are subject to transmission errors. The protection is non-cryptographic, in the sense that neither secret keys nor secured channels are used. Checksums generalize the idea of a parity bit by appending a (small) constant amount of message-specific redundancy. Both the data and the checksum are transmitted to a receiver, at which point the same redundancy computation is carried out on the received data and compared to the received checksum. Checksums can be used either for error detection or in association with higher-level error-recovery strategies (*e.g.*, protocols involving acknowledgements and retransmission upon failure). Trivial examples include an arithmetic checksum (compute the running 32-bit sum of all 32-bit data words, discarding high-order carries), and a simple XOR (XOR all 32-bit words in a data string). *Error-correcting codes* go one step further than error-detecting codes, offering the capability to actually correct a limited number of errors without retransmission; this is sometimes called *forward error correction* [17].
CRCs have been adopted by numerous PMR systems (for example TETRAPOL, TETRA, *etc.*) to protect the transmission of radio frames against unintentional errors due to poor radio conditions [17].

While of use for detection of random errors, *k*-bit checksums are not of cryptographic use, because typically a data string checksumming to any target value can be easily created. One method is to

simply replace/modify the transmitted message x with/into a message x', then calculate the code CRC(x') with the perfectly well known CRC, and finally code and transmit the information ($x' \| CRC(x')$) in a frame without the receiver or receivers detecting the least anomaly. This example shows that it is mandatory to implement integrity checking techniques that don't allow attackers constructing false messages which remain valid as regards the receivers. This latter must be able to detect intentional errors introduced by a malicious third party [17].

In contrast to checksums, data integrity mechanisms based on (cryptographic) hash functions are specifically designed to preclude undetectable intentional modification. The hash-values resulting are sometimes called *Integrity Check Values (ICV)*, or *Cryptographic Check Values* in the case of keyed hash functions. Semantically, it should not be possible for an adversary to take advantage of the willingness of users to associate a given hash output with a single, specific input, despite the fact that each such output typically corresponds to a large set of inputs. Hash functions should exhibit no predictable relationships or correlations between inputs and outputs, as these may allow adversaries to orchestrate unintended associations [17].

8.3.3 Data Integrity using MAC Alone

Data integrity is ensured thanks to a Message Authentication Code (MAC). MAC is a class of hash functions which allows data integrity checking, data origin authentication and identification. Hash functions play a fundamental role in cryptography: they take an arbitrary finite length bitstring to fixed length digest. They should have desirable properties such as preimage and collision resistance.
MAC algorithms may be viewed as hash functions which take two functionally distinct inputs, a message and a secret key, and produce a fixed-size (say n-bit) output, with the design intent that it be infeasible in practice to produce the same output without knowledge of the key [18].

Message Authentication Codes (MACs) as discussed earlier are designed specifically for applications where data integrity (but not necessarily privacy) is required. The originator of a message *x* computes a MAC $h_k(x)$ over the message using a secret MAC key *k* shared with the intended recipient, and sends both (effectively $x \| h_k(x)$). The recipient determines by some means (*e.g.*, a plaintext identifier field) the claimed source identity, separates the received MAC from the received data, independently computes a MAC over this data using the shared MAC key, and compares the computed MAC to the received MAC. The recipient interprets the agreement of these values to mean the data is authentic and has integrity – that is, it originated from the other party which knows the shared key, and has not been altered in transit. This corresponds to Figure 17 [18].

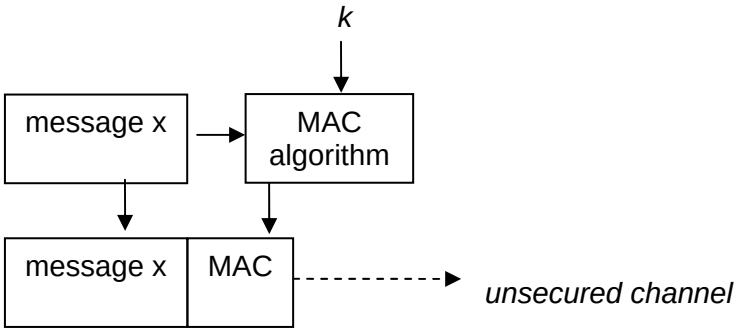


Figure 17: Method for Providing Data Integrity using MAC [18]

MACs can be based on block ciphers. The most commonly used MAC algorithm based on a block cipher *E* makes use of Cipher-Block-Chaining (CBC). Usually, AES is used as the block cipher *E* (for confidentiality reasons, *E* used in PMR systems is not given), n = 128 in what follows, and the MAC key can be a 128, 192 or 256-bit AES key.

Algorithm CBC-MAC [18]

INPUT: message x; specification of block cipher *E*; secret MAC key k for *E*.
OUTPUT: n-bit MAC on x (n is the blocklength of *E*).

1. *Padding and blocking.* Pad x if necessary. Divide the padded text into n-bit blocks denoted x1, …, xt.
2. *CBC processing.* Letting Ek denote encryption using *E* with key k, compute the block Ht as follows: $H_1 \leftarrow E_k(x_1)$; $H_i \leftarrow E_k(H_{i-1} \oplus x_i)$, $2 \le i \le t$.
3. *Optional process to increase strength of MAC.* Using a second secret key $k' \neq k$, optionally compute: $H'_t \leftarrow E^{-1}_{k'}(H_t)$, $H_t \leftarrow E_k(H'_t)$.
4. *Completion.* The MAC is the n-bit block H_t.

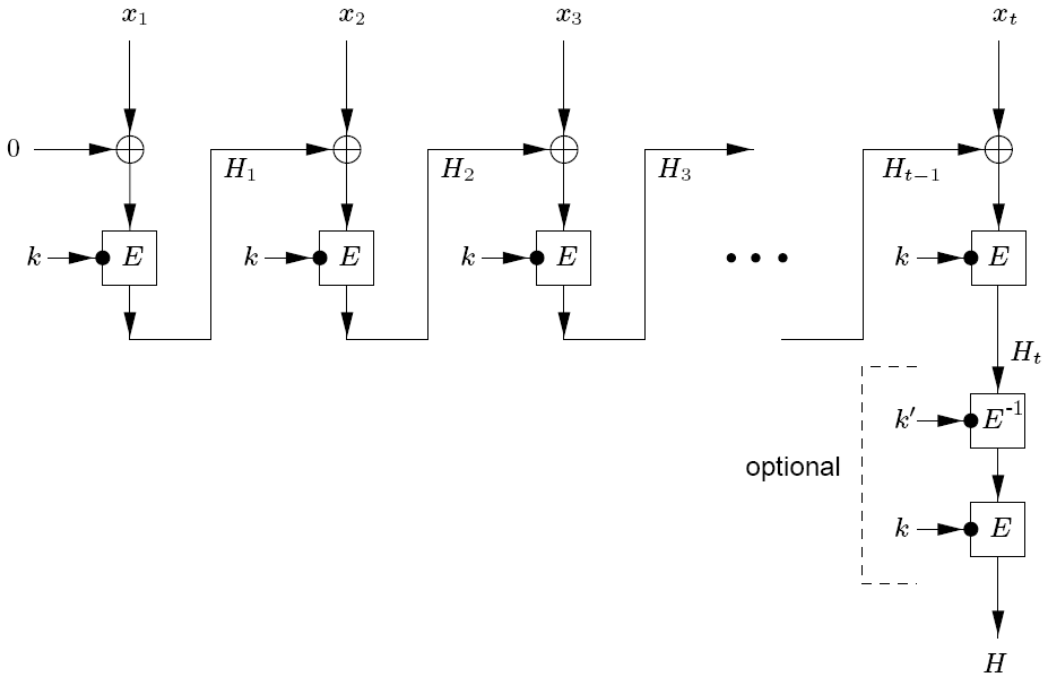


Figure 18: CBC-based MAC Algorithm [18]

8.3.4 Data Integrity Combined with Encryption

In the context of PMR systems, the mechanism depicted in Figure 17 is not fully satisfactory since it only ensures integrity. In PMR systems, **both integrity and confidentiality** are required. In this case, the following data integrity technique employing a MAC algorithm $h_{k'}$ may be used. The originator of a message x computes a hash value $H = h_{k'}(x)$ over the message, appends it to the data, and encrypts the augmented message using a symmetric encryption algorithm *E* with shared key *k*, producing ciphertext $C' = E_k(x \| h_{k'}(x))$ [18] (for confidentiality reasons, E_k used in PMR systems is not given).
This is transmitted to a recipient, who determines (*e.g.*, by a plaintext identifier field) which key to use for decryption, and separates the recovered data x' from the recovered hash H'. The recipient then independently computes the hash $h_{k'}(x')$ of the received data x', and compares this to the recovered hash H'. If these agree, the recovered data is accepted as both being authentic and having integrity. This corresponds to Figure 19 [18].
The intention is that the encryption protects the appended hash, and that it will be infeasible for an attacker without the encryption key *k* to alter the message without disrupting the correspondence between the decrypted plaintext and the recovered MAC. Moreover, the use of a MAC here offers the advantage that should the encryption algorithm be defeated, the MAC still provides integrity. A drawback is the requirement of managing both an encryption key and a MAC key. Care must be exercised to ensure that dependencies between the MAC and encryption algorithms do not lead to security weaknesses, and as a general recommendation these algorithms should be independent [18].

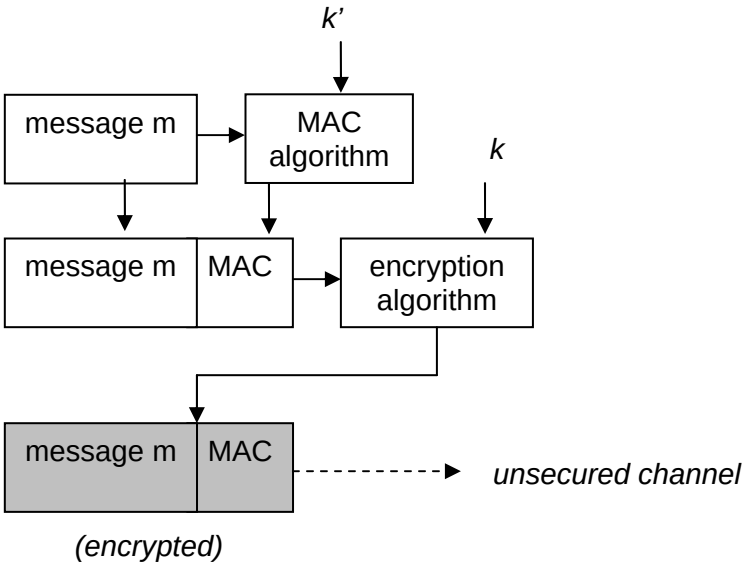


Figure 19: Method for Providing both Data Integrity and Confidentiality [18]

8.3.5 Cautionary Note about MAC Processing

The way that the MAC result is processed brings security issues.
According to a property of the functions used in a MAC context, a modification of a bit in the message x brings about, on average, the modification of one bit out of two in the result $h_{k'}(x')$. This property must be put in conjunction with the fact that the MAC will be sent concatenated (usually, we send

only MSBs or LSBs of the MAC). If not, the quantity of sent information is greater than the original frame size, and then the bandwidth will drastically decrease.
In brief, the advantage of this first implementation mode is to allow the use of any function h, with a digest cut to the desired size by truncating the result of this function if necessary. On the other hand, it is possible to have unintentional error detection properties different from those obtained with a linear CRC, for certain types of errors. Specifically, although the detection of errors is the same for an error probability that is uniform over the whole set of messages transmitted, it will be less favourable in the case of a non-uniform probability.
This is why a second mode of calculating the digest, illustrated by the flowchart of Figure 20 (and entirely described in [17]), provides the use of a specific “sealing function”.
This function is adapted to guarantee the detection of unintentional errors in the same way as a CRC. A mathematical function is proposed which comprises the combination, on the one hand, of a pseudo-random generating function GPA and, on the other hand, of a non-linear code CNL. The function GPA generates, from a secret key K and from a determined initialization variable, an encryption string of any length, for example of at most 2^{64} distinct values. The CNL code must have a Hamming distance equal to or greater than that of a CRC customarily used in the contemplated type of applications. For equal sizes it is known that there exists a non-linear code which satisfies this property.
With a mathematical function of this type, the detection of intentional errors results from the GPA function, and that of unintentional errors results from the non-linear code CNL. The performance is optimized by choosing a non-linear code CNL having properties to guarantee good Hashing.
Based on a message x to be sealed with a secret key K, an example of such a function comprises the following calculations.
In a first step, a variable X is calculated with the aid of the GPA function applied to the key K and to a first initialization variable VI1, in such a way that: X = GPA(VI1, K).
Then, in a second step, an information item Y(M) is calculated with the aid of a linear matrix A_X constructed from the variable X, and applied to a message M, in such a way that: $Y(M)=A_X(M)$.
In a third step, which may be performed in parallel with or before previous steps, the calculation of a variable Z is carried out with the aid of the GPA function applied to the key K and to a second initialization variable VI2, in such a way that: Z=GPA(VI2,K).
Finally, in a last step, which necessarily takes place after the two other ones, the seal S(M) is calculated with the aid of a linear matrix A_z constructed from the variable Z, and applied to the information item Y(M), in such a way that: $S(M) = A_z(CNL(Y(M)))$.
As will be immediately apparent to the person skilled in the art, there exists a plurality of functions GPA, of non-linear codes CNL and of linear matrices A satisfying the soughtafter aims (for confidentiality reasons, GPA, CNL and A used in PMR systems are not given).

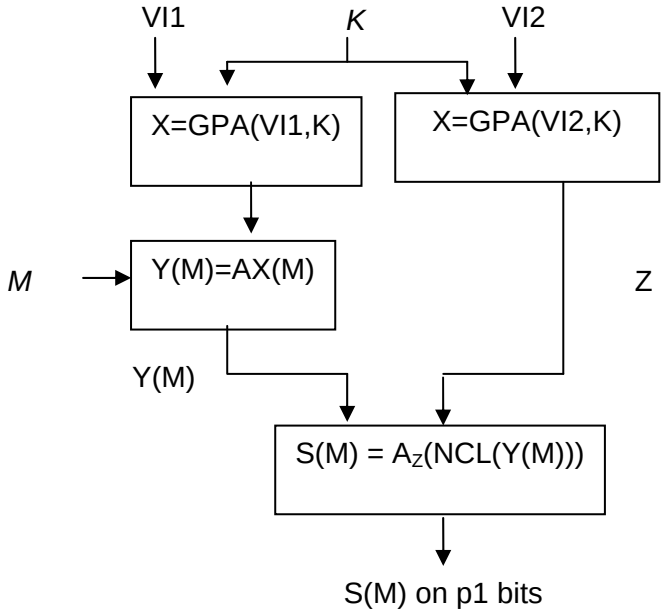


Figure 20: Another Procedure Providing better Security for Sealing [17]

8.4 REFERENCES

[1] A. Piva, and M. Barni, "Managing Copyright in Open Networks," *IEEE Internet Computing*, MAY-June 2002.
[2] G. Doërr and J.-L. Dugelay, "A guide tour of video watermarking," *Signal Processing: Image Commun*, vol. 18, no. 4,pp. 263-282, Apr. 2003.
[3] F. Bartolini, A. Tefas, M. Barni, and I. Pitas, "Image authentication techniques for surveillance applications," Proc. IEEE, vol. 89, no. 10, pp. 1403-1418, Oct. 2001.
[4] Sin-Joo Lee, and Sung-Hwan Jung, "A survey of watermarking techniques applied to multimedia," *IEEE International Symposium on Industrial Electronics,* Vol. 1, pp. 272-277, Korea, June 2001.
[5] I. Cox, J. Kilian, T. Leighton, and T. Shamoon, "Secure spread spectrum watermarking for multimedia," *IEEE Transactions on Image Processing*, vol. 6, no. 12, pp. 1673-1687, December 1997.
[6] J. Zhang, A. T, and S. Ho, "Efficient robust watermarking of compressed 2-D grayscale patterns," *IEEE Workshop on Multimedia Signal Processing*. pp. 1-4, October 2005.
[7] F. Hartung, and B. Girod, "Watermarking of uncompressed and compressed video," *Signal Processing*, vol. 66, pp. 283_301, May 1998.
[8] Q. Li, I. J. Cox, "Using Perceptual Models to Improve Fidelity and Provide Resistance to Valumetric Scaling for Quantization Index Modulation Watermarking," *IEEE Trans. on Information Forensics and Security*, vol. 2, pp. 127- 139, 2007.
[9] A. Golikeri. P. Nasiopoulos, and Z. J. Wang, " Robust digital video watermarking scheme for H.264 advanced video coding Standard," *Jornal of Electronic Imaging*, vol. 16 (4), pp. 8-12, Oct-Dec 2007.
[10] H. T. Lu, R. M. Shen and F. L. Chung, "Fragilewatermarking scheme for image authentication," *Electronics Letters*, vol. 39, no. 12, pp. 898-900, 2003.
[11] C. C. Wang and Y. C. Hsu, "Fragile watermarking scheme for H.264 video authentication," *Optical Engineering*, vol. 49, no. 2, Feb. 2010.
[12] F. Dufaux and T. Ebrahimi, "Scrambling for Privacy Protection in Video Surveillance Systems," *IEEE Trans On Circuits and Systems For Video Technology*, vol. 18, no. 8, pp. 1168–1174, Oug. 2008.
[13] P. Meuel, M. Chaumont and W. Puech, " Datat Hiding In H.264 Video for Lossless Reconstruction of Region of Interest," *EURASIP*, pp. 2301-2305 ,2007

[14] S. Chen and H. Leung, “Chaotic Watermarking for Video Authentication in Surveillance Applications,” *IEEE Trans On Circuits and Systems For Video Technology*, vol. 18, no. 5, pp. 704–709, May. 2008.
[15] S. Thiemert, H. Sahbi and M. Steinebach, “Using entropy for image and video authentication watermarks,” SPIE-IS&T, vol. 6072, no. 18, 2006.
[16] C. Cachin, “An Information-Theoretic Model for Steganography,” *Proceedings of 2nd Workshop on Information Hiding*, MIT Laboratory for Computer Science, May 1998
Intentional Transmission Errors”. US Patent number 7,774,677 B2. 2010.
[18] A. Menezes, P. van Oorschot and S. Vanstone. “Handbook of Applied Cryptography”, chapter 9 “Hash Functions and Data Integrity”. CRC Press. ISBN 0-8493-8523-7. 1996.

9. REMOTE CONTROL

FOV Field of view
NVC Network Video Client
NVT Network Video Transmitter
PTZ Pan Tilt Zoom

9.1 BENEFITS OF THE REMOTE CONTROL

An important aspect of a video surveillance system is managing video for live viewing, recording, playback and storage of video flows, and configuration of the video devices. If the system consists of only one or a few cameras, viewing and some basic video recording can be managed via the built-in web interface of the network cameras and video encoders. When the system consists of more than a few cameras, using a network video management system is necessary. This aspect is more linked to the ground infrastructure and the associated devices, like for the network management.
On-board devices, the network devices as well as the video devices, are not too many, particularly when the vehicle is a patrol car. But a network management as well as a network video management may be useful according to the size of the car fleet.
The remaining of this section will focus on the benefits of the network video management, related to the sensors, that is to say the IP cameras, without dwelling on the management of the network devices, neither discussing on the video streaming.

9.2 CAMERA CONFIGURATION

The management of the IP camera has two main areas: the configuration of the camera and its use. The configuration is generally done once at the set-up of the video network or the installation of the camera in the network. Despites the basic network settings such as IP address, the configuration takes into account parameters related to the advanced network settings such as DNS, NTP, servers enabling (FTP, HTTP…). Network features are also set once, such as SMTP to send a mail on a defined trigger, QoS for setting the priority level of video, audio, the streaming characteristics…
Another type of parameters set once are the video parameters to define settings of the image such as the resolution, the frame rate, the color, brightness, sharpness, and so on… Coding parameters are also set at this moment. They include the compression rate, the bit rate, the transmission mode (CBR, VBR)…

9.3 CAMERA CONTROL

The camera control can allow the modification of the current configuration, but it should not occur very often. Most of the actions performed on the camera control are related to the PTZ and the management of the movement (pan, tilt) and the zoom of the camera.
An important topic of the PTZ management is the definition of the coordinate space. The ONVIF forum is working on the standardization of communication between network video devices and the interoperability between network video products regardless of manufacturer, within video surveillance and other physical security areas. Following considerations are issued from ONVIF results.
Spaces are used to specify absolute, relative and continuous movements. Whereas absolute movements require an absolute position, relative movements are specified by a position change, and

continuous movements require the specification of a velocity (relative movement over time). For these three cases, different coordinate systems, also called spaces, are used describing the desired movement.

9.3.1 Absolute Position Spaces

The Absolute Position Spaces are used when the NVC wants to move the camera to the requested position. The absolute movement from current position A to an arbitrarily chosen position B doesn't have to follow a specific path. Instead, the PTZ NVT may choose the shortest path in order to reach the target destination.

Figure shows a camera with pan and tilt mechanics and the corresponding spherical coordinate system. The space description assumes that the dome is mounted on the ceiling. The definition of a Pan movement is the rotation of the camera module around the pan axis. Thereby, the tilt axis is also rotated in the same direction in the plane orthogonal to the pan axis, so that it is still orthogonal to the camera lens axis. Tilt movement is the rotation of the camera module around the tilt axis. With the tilt axis the camera direction can be changed from looking downward to looking at the horizon. Some devices may support a camera which can look above the horizon.

The angles describing the rotation around pan and tilt axis are referred to as pan and tilt angles, where pan is represented by the X coordinate of the Position vector and tilt is represented by the Y coordinate of the Position vector. Both angles are specified in degrees. The initial position of this coordinate system is when the direction of the camera lens is parallel to the ceiling. The pan and tilt angles in this initial direction are zero (0,0).

When starting from the initial direction and increasing the pan angle, objects that have previously been in the centre of the image will move towards the left of the image. When starting from the initial direction and increasing the tilt angle, objects which have been previously in the middle of the image move towards the bottom of the image.

The maximum range for pan and tilt angles are between -180 and +180 degrees. The NVT can restrict the tilt range arbitrarily. The tilt angle of a camera can change its direction in the space of a hemisphere like a dome camera is typically bounded from 0 to -90 degrees. If a device cannot pan the full range, it may limit the pan range to an appropriate interval.

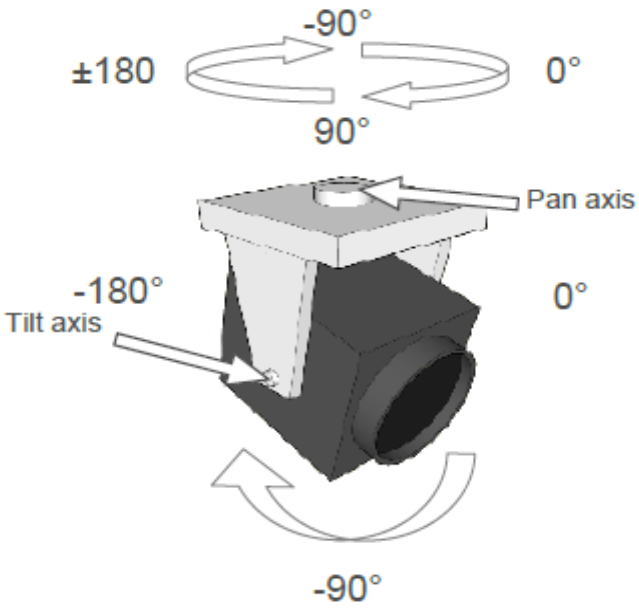


Figure 21: Spherical Pan / Tilt Position Space in Degrees for a camera mounted on the ceiling.

The Digital Pan / Tilt Position Space is suitable for Digital PTZ cameras, where the pan and tilt coordinates represent the centre point of a window positioned on a sensor , also known as absolute Digital PTZ.
The pan movement is a horizontal movement in the X direction on the sensor plane and the tilt movement is a vertical movement in the Y direction on the sensor plane. The coordinate system originates from the lower left of the sensor. Figure 6 displays an example of a window located at the left upper most coordinate (0.1,0.9) with a window size of (0.2*plane width, 0.2* plane height).

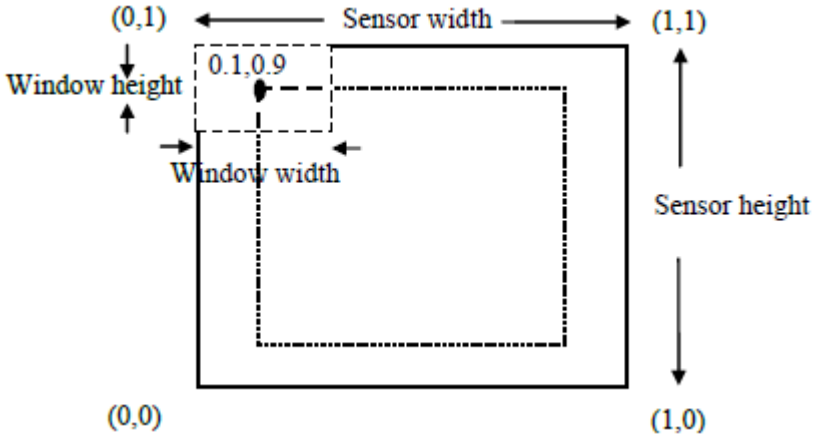


Figure 6: Digital Pan / Tilt Position Space.

9.3.2 Relative Translation Spaces

The Relative Pan / Tilt Translation Spaces is suitable when a NVC wants to move the camera in a certain direction without knowing the camera‘s current Pan / Tilt position.
A Relative Pan / Tilt Translation can be derived from a corresponding (digital / spherical) Absolute Pan / Tilt Position Space by taking the difference of two absolute Pan / Tilt positions. However, there are also relative Pan / Tilt translations where no corresponding absolute Pan / Tilt space can be defined.
The Spherical Pan / Tilt Translation Space In Degrees derives from the Absolute Spherical Pan / Tilt Position Space In Degrees. Instead of an absolute Position space where the reference position is fixed, the relative spherical space specifies the reference position as the cameras current position at all times. Thereby, the Pan / Tilt Translation is expressed as the coordinate difference from the current position to the target position. If a NVC wants to pan the camera by 5 degrees, it can use this relative spherical space and set the X coordinate of the direction to 5 and the Y coordinate to 0.
The Relative Pan / Tilt Translation Space In FOV is introduced to simplify the navigation with dome cameras in graphical user interfaces. When the user wants to centre the camera on a certain position in the current camera view, the user requests a movement with respect to the current FOV. Due to the mechanics of a dome, the image content may rotate (see Figure). Figure 7 shows a rectangle representing the image content. The relative Pan / Tilt Translation Space in FOV has its origin in the centre of the image. The upper right corner corresponds to the normalized coordinate (1,1).

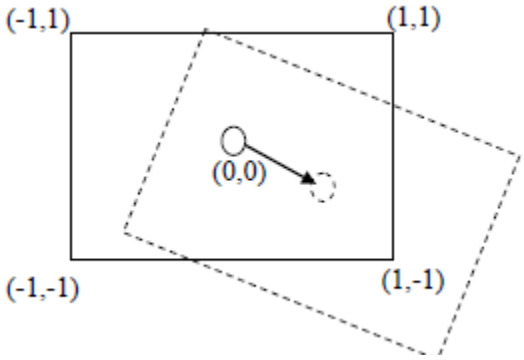


Figure 7: Relative Pan / Tilt Translation Space In FOV for a mechanical dome.

The Digital Pan / Tilt Translation Space is derived from the Absolute Digital Pan / Tilt Position Space (see section 3.1.2). Instead of an absolute position space where the reference position is fixed, the relative space specifies the reference position as the cameras current position at all times. Thereby, the pan/tilt translation is expressed as the coordinate difference from the current position to the target position. If a NVC wants to move the window area of the Video Source Configuration by a tenth of the sensor width horizontally, it can use this relative spherical space and set the X coordinate of the direction to 0.1 and the Y coordinate to 0.
The following figure shows the space description of this Digital Pan / Tilt Translation coordinate system. The outer box represents the image sensor, the dotted inner box the cropped area, and the arrow demonstrates a translation request of the cropped area with a pan/tilt vector of (0.1,-0.2).

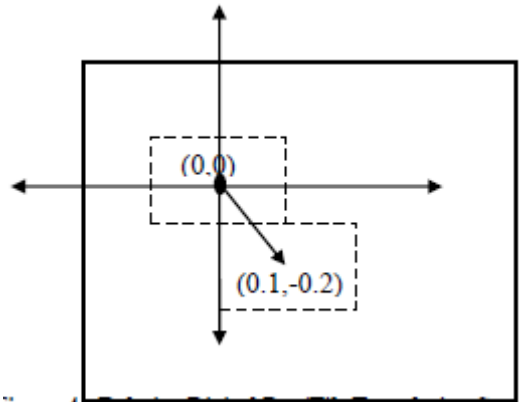


Figure 8: Relative Digital Pan / Tilt Translation Space.

The maximum translation in positive direction for pan and tilt in this coordinate system would be represented by (1,1) the same way as the maximum negative translation would map to the coordinate (-1,-1). For example, a translation of a cropped area on a megapixel sensor from its lower left corner to its upper right corner corresponds to a relative movement using the translation vector (1,1) assuming that the cropped area has zero size.

9.3.3 Continuous Velocity Spaces

The Continuous Velocity Spaces are used when the NVC wants to move the dome continuously in a certain direction with a defined speed.

If a camera supports e-flip[1] and the tilt translation is passing nadir[2] position (including the room for a hysteresis), the camera should rotate the image at nadir (only for cameras supporting absolute pan/tilt positioning). When a rotation occurs, the camera will continue the current movements according to the directions given when the command was issued. If the command is interrupted with a new request (after the flip), that request will be handled according to the new (flipped) direction and coordinates. A camera that doesn't support e-flip or has it disabled will not rotate the image and directions during a tilt movement passing nadir.

9.3.4 Speed Spaces

The Speed Spaces are introduced to specify the speed when moving to an absolute or relative position. Thereby, the NVC specifies the combined speed for the two direction parameters.
If Relative Translation Space and Continuous Velocity Space are already defined, the corresponding Speed Space is derived as follows: Requesting a continuous movement with a velocity V for T seconds, is identical (up to acceleration and positional inaccuracies) to requesting a relative movement with Relative Position R and Speed S, where R equals V times T and S equals the length of vector V. Therefore, Speed values are always positive.

9.4 NETWORK REQUIREMENTS

Information between network video management and video device is generally exchanged with protocol such as SNMP. The remote control generates a few commands, without a huge volume of data, as well as the status returned by the video network. The bandwidth required for these transfers is therefore insignificant over the bandwidth required by the streaming of the video flows.

[1] E-flip is the term used to describe the behaviour when a PTZ Dome rotates the image and control directions when passing nadir direction during a tilt movement. This functionality is useful when controlling domes using human joystick control, where a client can track an object that passes nadir and doesn't have to bother about inverted controls.

[2] Nadir is defined as the direction below a dome camera that is mounted in the ceiling and looking downwards.

10. CONCLUSIONS

The present document starts the series of deliverables in the SPY's WP5 by presenting the state-of-the-art background supporting the network management related technologies:

- **data coding**: video, metadata and multimedia scene technologies are discussed and benchmarked;
- **data encapsulation**: the nowadays most intensively used solutions (belonging to the MPEG family) are described;
- **session layer protocol**: the two most intensively used protocols (RTSP and HTTP) are outlines and their matching to the SPY peculiarities is emphasized;
- **network based adaptation**: the means for ensuring multimedia content adaptation to the network constraint are hinted to;
- **data & stream integrity**: the basic principles in watermarking (be it robust, fragile or semi-fragile) are presented and their practical relevance is pointed to through literature examples; the tools for ensuring PMR (private mobile radio) networks integrity and authenticity are also presented and discussed;
- **remote control**: the way in which complex multi-camera systems can be remotely managed is pointed to.

As it can be seen all these issues are currently object to intensive and extensive activities raised not only by the industrial and academic communities, but by the standardization bodies as well. Hence, during the project life, some new emerged technologies (like the HTML5, for instance) might be taken aboard. Should such a situation arise, the present deliverable wpould be likely to be updated accordingly.