



**SEVENTH FRAMEWORK PROGRAMME**  
**ICT-1-1.5**  
**Networked Media**

*Specific Targeted Research Project*

**My-e-Director 2012**  
(FP7-215248)

**My-e-Director 2012 - Real-Time Context-Aware and  
Personalized Media Streaming Environments for  
Large Scale Broadcasting Applications**

**[D3.1 Contextual-Database Creation Tool]**

Due date of deliverable: [15-03-2009]

Actual submission date: [13-03-2009]

Start date of project: 04-02-2008

Duration: 36 months

### Summary of the document

<b>Code:</b>	<b>D3.1 Contextual-Database Creation Tool</b>
<b>Last modification:</b>	06/03/2009
<b>State:</b>	Final
<b>Participant Partner(s):</b>	ATOS, FBK
<b>Editors &amp; Author (alphabetically):</b>	Editor: Elena Garrido(ATOS)  Authors: Elena Garrido (ATOS) Paul Chippendale (FBK)
<b>Fragment:</b>	<b>No</b>
<b>Audience:</b>	<input checked="" type="checkbox"/> public <input type="checkbox"/> restricted <input type="checkbox"/> internal
<b>Abstract:</b>	<i>This document is deliverable D3.1 "Contextual-Database Creation Tool".</i>
<b>Keywords:</b>	
<b>References:</b>	Refer to the corresponding section at the end of the deliverable

## Document Control Page

<b>Version number</b>	V04
<b>Date</b>	06/03/2009
<b>Modified by</b>	Elena Garrido
<b>Comments</b>	Final document
<b>Status</b>	<input type="checkbox"/> draft <input type="checkbox"/> WP leader accepted <input type="checkbox"/> Technical coordinator accepted <input checked="" type="checkbox"/> Project coordinator accepted
<b>Action requested</b>	<input type="checkbox"/> to be revised by partners involved in the preparation of the deliverable <input type="checkbox"/> for approval of the WP leader <input type="checkbox"/> for approval of the technical coordinator <input type="checkbox"/> for approval of the project coordinator Deadline for action:

## Change history

Version number	Date	Changed by	Changes made
01	03/02/2009	E. Garrido	1 <sup>st</sup> version
02	05/02/2009	P. Chippendale	2 <sup>nd</sup> version, section 4 has been added
03	16/02/2009	E. Garrido	Intoduction
04	06/03/2009	E. Garrido	Updates of section 5 with decisions taken in a telco

## Table of Contents

1	Executive Summary .....	6
1.1	Scope .....	6
1.2	Audience .....	6
1.3	Summary .....	6
1.4	Structure .....	6
2	Introduction .....	7
3	Sport Information System .....	8
3.1	Services of the SIS .....	8
3.2	Services Description .....	9
3.2.1	SISService .....	9
3.2.2	SISSubscriptionService .....	10
3.2.3	Real-Time message format .....	10
3.3	SIS Data Diagram .....	13
3.4	Examples .....	15
3.4.1	Example of usage of the web-services .....	15
3.4.2	Examples of real-time data .....	16
4	Visual processing output .....	18
5	Future .....	20

## **1 Executive Summary**

### **1.1 Scope**

My-e-Director will use metadata coming from the Olympic Games written in a proprietary format. The metadata will span a diverse multitude of sources, and will contain information such as: the event being viewed, results, medals, records, weather conditions, athletes in the event, with their numbers, names and their expected lane designation, team colours and visual appearance models. Also metadata will be generated from in the image processing module. This information will be used all over the My-e-Director platform.

### **1.2 Audience**

This deliverable is public. It contains interfaces exposed by metadata components and metadata format used within the My-e-Director project.

### **1.3 Summary**

It describes intermediary software that will process all messages coming from commercially available service called IDF (offered to broadcasters of the Olympic Games family) and store relevant information into a structured database that can be accessed using web services.

This metadata will be enriched by the outputs of the visual processing tasks.

### **1.4 Structure**

The rest of the deliverable is structured in the following way:

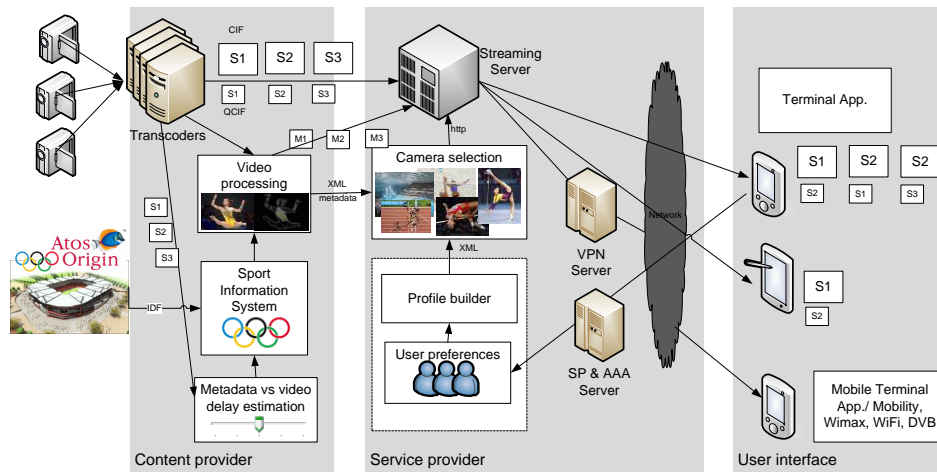
Section 2 is an introduction to the goals of this document – the description of the interfaces of the components in My-e-Director that will generate metadata.

Section 3 details of the Sports Information System.

Section 4 depicts the output that for the moment is generated in the image processing algorithms.

## 2 Introduction

The following figure depicts the My-e-Director system architecture and where the metadata components, Sport Information System and Video Processing, are placed.



**Figure 1: My-e-Director System Architecture**

Metadata is used all over the My-e-Director system. This information will help other components to constrain the domain and therefore facilitate the multimedia entities automatic detection. It will be also used to create the user profiles and to select the best camera for the user.

During the Olympic Games, results data distributed to the official website, broadcasters and other internet customers using the Internet Data Feed (IDF)[IDF]. IDF contains all the results and real-time data from both INFO<sup>1</sup> and the Commentator Information System<sup>2</sup> (CIS). The Sport Information System will act as a wrapper hiding the complexity of this system and providing a 'clean output' from this.

The output from SIS will be enriched by the outputs from the visual processing tasks in WP3.

The available interfaces from SIS and the tools created in WP3 for the image processing with the other components will be based in open standard.

These standards are SOA and NewsML for the SIS and MPG7 for a future version from the image processing output.

SportML optimizes the sharing of sports statistics and information such as schedules, results, standings, team statistics and individual statistics, but not information of actions that happen during an event. NewsML is an XML standard designed to provide a media-independent, structural framework for multimedia news, and therefore real-time actions. NewsML has been chosen to send the real-time information from SIS.

MPEG-7 will be used as standard for the image processing output due to the fact it is a multimedia content description standard. It uses XML to store metadata, and can be attached to timecode in order to tag particular events.

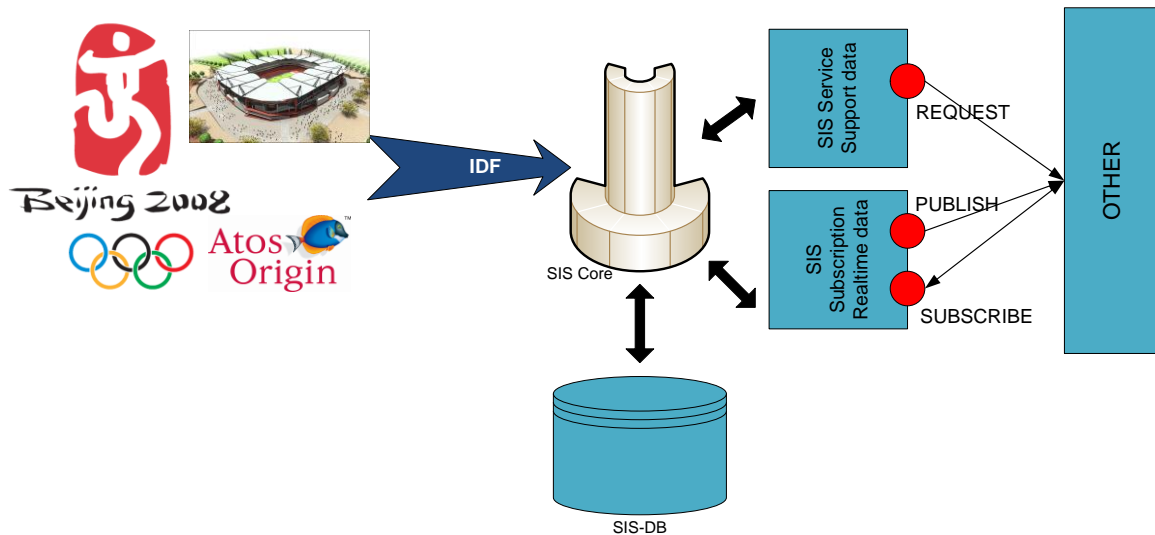
<sup>1</sup> INFO is an Intranet that is available to accredited media and the Olympic Family of athletes and IOC officials.

<sup>2</sup> Commentator Information System (CIS) is a Java-based application that displays results on touch-screen PCs at the venue broadcast sites in a fraction of a second, so they can be instantaneously dispatched across the globe. It provides event results to broadcasters before they hear the roar of the crowd. 'Color' for the commentary is provided by sport-specific screens.

### 3 Sport Information System

The Sport Information System is the intermediary software that will process all messages coming from Olympic Information systems supplied to the broadcasters. It has mainly two purposes:

1. **Providing Support data**, i.e. it provides ‘well-known’ concepts and facts before events start. Examples of “support information” will include data related with Athletes, Officials, Horses, and Teams in addition to (preliminary) event structures. Generally, this data is needed to use real time data.
2. **Providing Real Time Data**, i.e. it provides information which is generated throughout the course of events, like a match result, detailed stats about the participation of an athlete, actions that happen in a match, etc.



**Figure 2: Architectural overview on the ATOS SIS**

As seen in figure 3 the interface between the SIS and the OTHER My-e-Director components uses two modes of operation:

1. **Pull Mode:** The OTHER component from My-e-Director actively requests information from the SIS. This information is primarily support information, which is used to provide preparation data for the staging of the event.
2. **Push Mode:** Whenever data about events has been retrieved from the SIS, this data is assigned a unique ID. Using this ID, the OTHER component is able to subscribe for notification about changes in this event. The OTHER component can react on changes in the event structure (i.e. changes in the support information) and can receive real time information for events (e.g. results) using the subscription mechanism.

#### 3.1 Services of the SIS

Two different services are available for the OTHER components: SIService and SISSubscriptionService.

The methods exposed by these two services are shown in the following two figures:

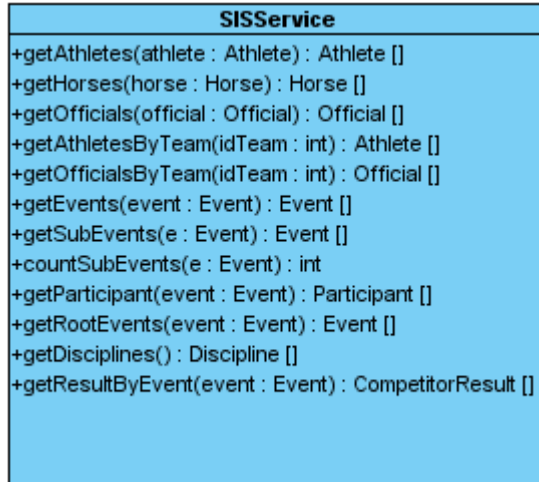


Figure 4: SIS Service Interfaces

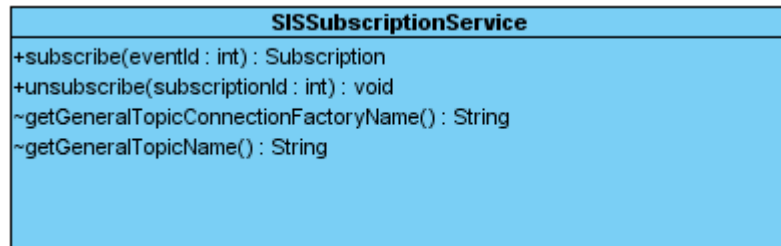


Figure 5: SIS Subscription Service Interfaces

- **SISService:** The SISService describes the service Endpoint at the SIS site. It provides methods to retrieve Athletes, Officials, Events and Teams, based on different filters.
- **SISSubscriptionService:** The SISSubscriptionService deals with subscriptions. Subscriptions are event based. To subscribe to an event, the caller has to call subscribe with the ID of the event it is interested in. If the event has any sub-events, all information about these sub-events are handled using this subscription, too (i.e. all information about the sub-events are sent to the same subscriber). The Subscription class contains information of the topic to use JMS to receive the Real-Time information.

## 3.2 Services Description

### 3.2.1 SISService

In general, the methods that will need a parameter in order to filter the returned results will use an object of the same type as the returned. The object passed as a parameter will be known as 'base', and the result set will be found by applying the intersection operation between the 'base' object and the set of objects of the same type.

In summary the set of objects returned will be the result of fulfil the empty attributes in the object that will be passed as a parameter.

- **Official[] getOfficials(Official official):** Return a list of Officials, using 'official' as base.
- **Athlete[] getAthletes(Athlete athlete):** Return a list of Athletes, using 'athlete' as base.
- **Horse[] getHorses(Horse horse):** Return a list of Horses, using 'horse' as base.
- **Team[] getTeamsByEvent(Event event):** Return a list of Teams, which will participate in the

event supplied as parameter.

- **Athlete[] getAthletesByTeam(int teamID):** Return a list of Athletes, which are part of the team
- **Event[] getEvents(Event event):** Return a list of Events, using 'event' as base.
- **Participant[] getParticipant(Event event):** Return a list of Athletes, which will participate in the event supplied as parameter.
- **Horse[] getHorsesByEvent(Event event):** Return a list of Horses, which will participate in the event supplied as parameter.
- **Discipline[] getDisciplines():** Returns a list of disciplines.
- **int countSubEvents(Event event):** Returns the number of sub-events which are direct children in the event structure of the given event.
- **Event[] getSubEvents(Event event):** Returns a list of sub-events which are direct children in the event structure of the given event.
- **Event[] getRootEvents():** Returns a list of events in the SIS which do not have a parent (i.e. which represent a tournament).
- **CompetitorResult[] getResultByEvent (Event event):** Returns a list results for the given event.

### 3.2.2 SISSubscriptionService

Methods in this service allow subscribing and unsubscribing to event notifications, and provide methods to retrieve JMS-specific information (like the name of the ConnectionFactory). Additionally, it provides the name of the general update topic which provides update information for support data.

- **String getGeneralTopicConnectionFactoryName ():** Returns the JNDI name of the Connection-Factory used to create connections (JMS-specific).
- **String getGeneralTopicName():** Returns the JNDI name of the topic which is used for general support updates.
- **Subscription subscribe(int eventId):** Used to subscribe to the event specified using "eventId". This method returns a subscription object, which contains the information needed to create a JMS connection to receive updates.
- **void unsubscribe(int eventId):** Removes the subscription for this event.

### 3.2.3 Real-Time message format

Real-Time messages are sent using JMS to JMS-Topics, which are created and subscribed to using the SISSubscriptionService. For each Subscription, a JMS-Topic is available. The JMS messages sent on these topics have a text (xml) payload, using the NITF NewsML standard as container. We use the inlineXML option in the contentSet element of the NewsML standard to transmit the SIS data. Additionally, meta-Information about the data (mainly creation and sent date) is included in the payload.

The payload of the inline XML element may either be:

1. An **eventUpdate (type: sis:EventUpdateList)** indicating real-time updates of support information. Content of this element is a list of Event-IDs, indicating the events with changed support information. The subscriber is responsible to request the support data from the SISService, and update its data according to the changes.
2. A **participantUpdate (type: sis:ParticipantUpdateList)** indicating changes for athletes and officials. Content of the list is a number of Participant IDs with changed information. As with changes in events, the subscriber is responsible for requesting the support information from the SISService and updating its data,
3. A **realtimeUpdate (type: sis:RealTimeUpdateType)**. This element is used for real-time up-

dates containing actual incidents during competitions. The payload of this element is firstly the event during which the incident happened (as EventID), and secondly a payload information depending on the sports of the competition (element: eventDetail, type: sis:EventDetailType). Possible content types (i.e. sub-types of EventDetailType) for this element are described in the following section.

Subtypes of the xml schema type sis:EventDetailType are valid data types for real-time updates. Real-time information for different disciplines is supplied by the system,. A different schema has to be created for each discipline due to the huge differences in the kind of sports information. Not all disciplines can be processes by SIS, therefore, only some will be described here. The described disciplines are: athletics, beach volleyball, and swimming.

### 3.2.3.1 Athletics

Athletics is the most complicated sport disciplines from the Olympic Games. It comprises a high diversity of sport events. Two schemas have been defined: one to send the athletic information and other to additional information that might be helpful.

To send information related with the athletics, messages of type **sis:AthleticDetailType** is used for **Athletic** real-time updates. Prior the start of an event statistics and information about the athlete like the seasonal best, personal best marks and bib is send. During the event the reaction time is send. Depending of the kind of event, other information can be send with the result information like the wind speed, for example in long jump.

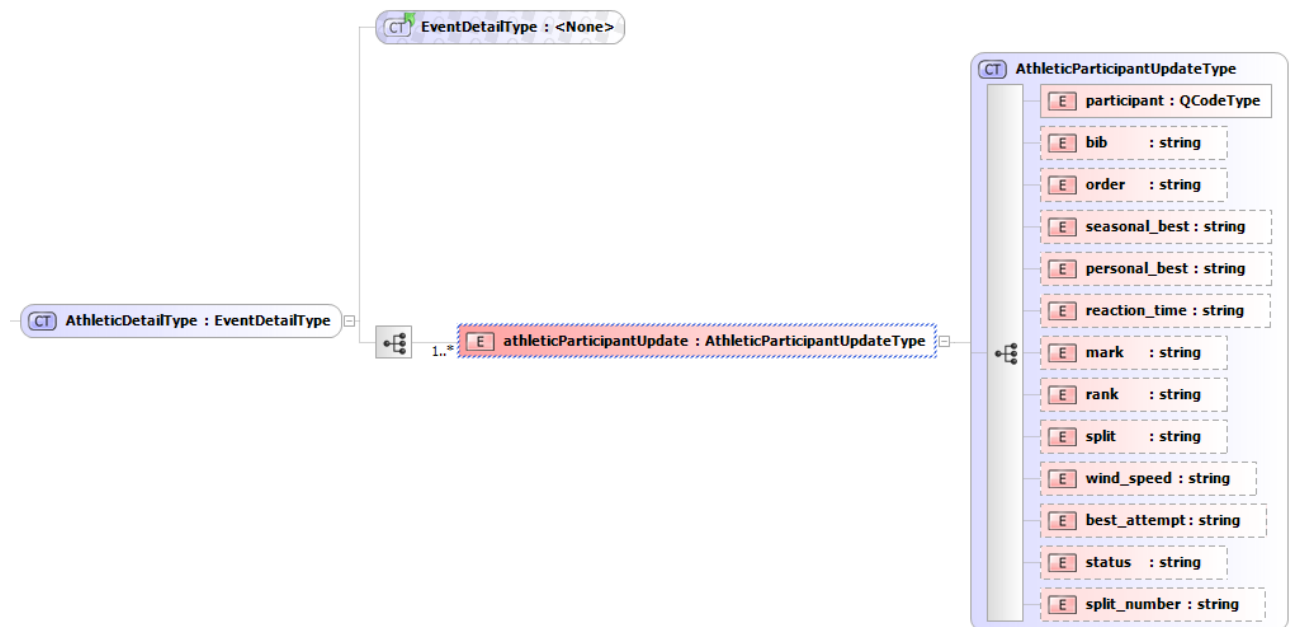


Figure 6: Graphical representation from the athletic schema

In messages of type **sis:AthleticInformationDetailType** some extra information is send about the event. This extra information is for example the number of splits and the distance of these splits. Also the information from the current participant (the athlete that is going to jump, for example is send).

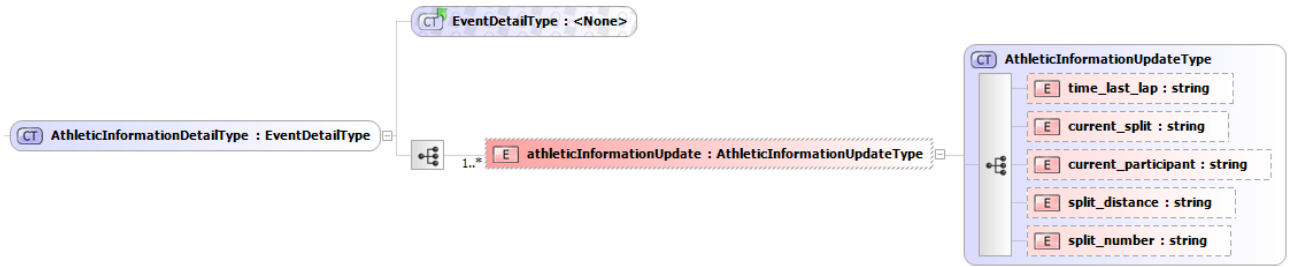


Figure 7: Graphical representation from the athletic schema

### 3.2.3.2 Beach Volleyball

Messages of type **sis:BeachVolleyBallDetailType** is used for **Beach Volleyball** real-time updates. Content of the messages is a number of actions, which describe the incident using an **sis:ActionUpdateType**. An action contains the information about the current rally (set, rally, order in the rally), the current score in the match (overall), and a list of actions which happened during the current set. Additional to the action, a flag indicating the validity of the information is contained in the message.

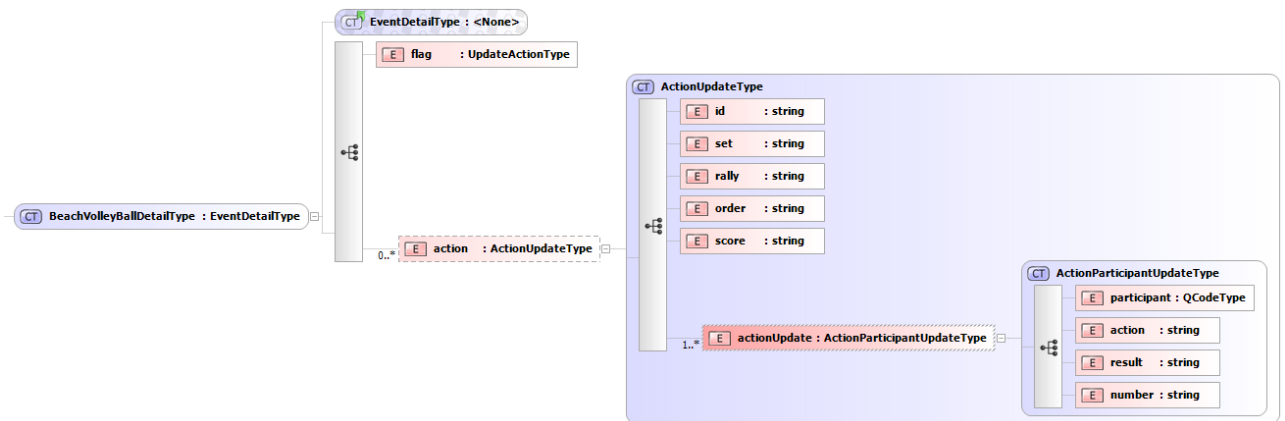


Figure 8: Graphical representation from the beach volleyball schema

### 3.2.3.3 Swimming

Messages are defined using the **sis:SwimmingDetailType**. Messages of this type are sent whenever a swimmer reaches a turning point (intermediate). The content of such a message include the result for the current athlete, as well as all results for athlete who have reached the turning point prior to the current athlete. Each athlete is encoded using a **sis:SwimmingParticipantDetailType**, which includes the leg he is on, the rank, time, lane and team (if applicable).

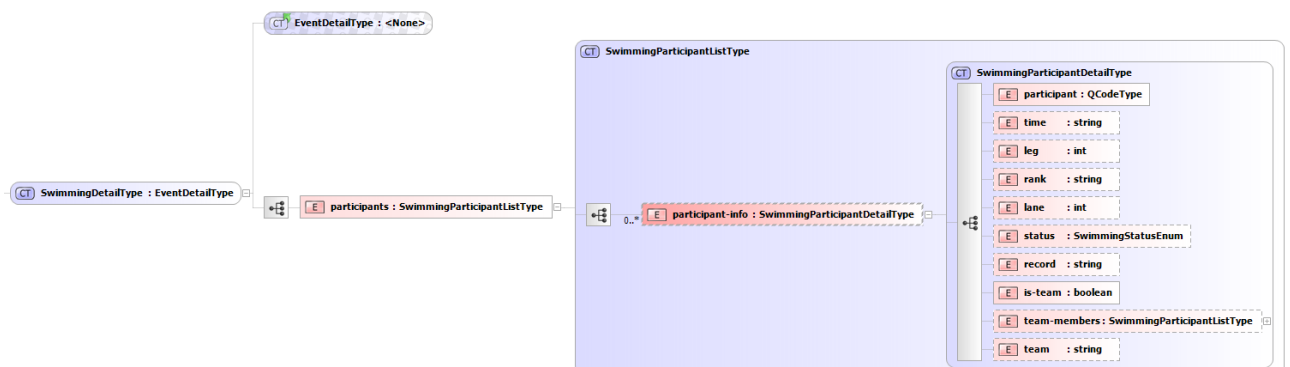


Figure 9: Graphical representation from the swimming schema

### 3.3 SIS Data Diagram

The following figure shows the data model for support data as used by the SIS.

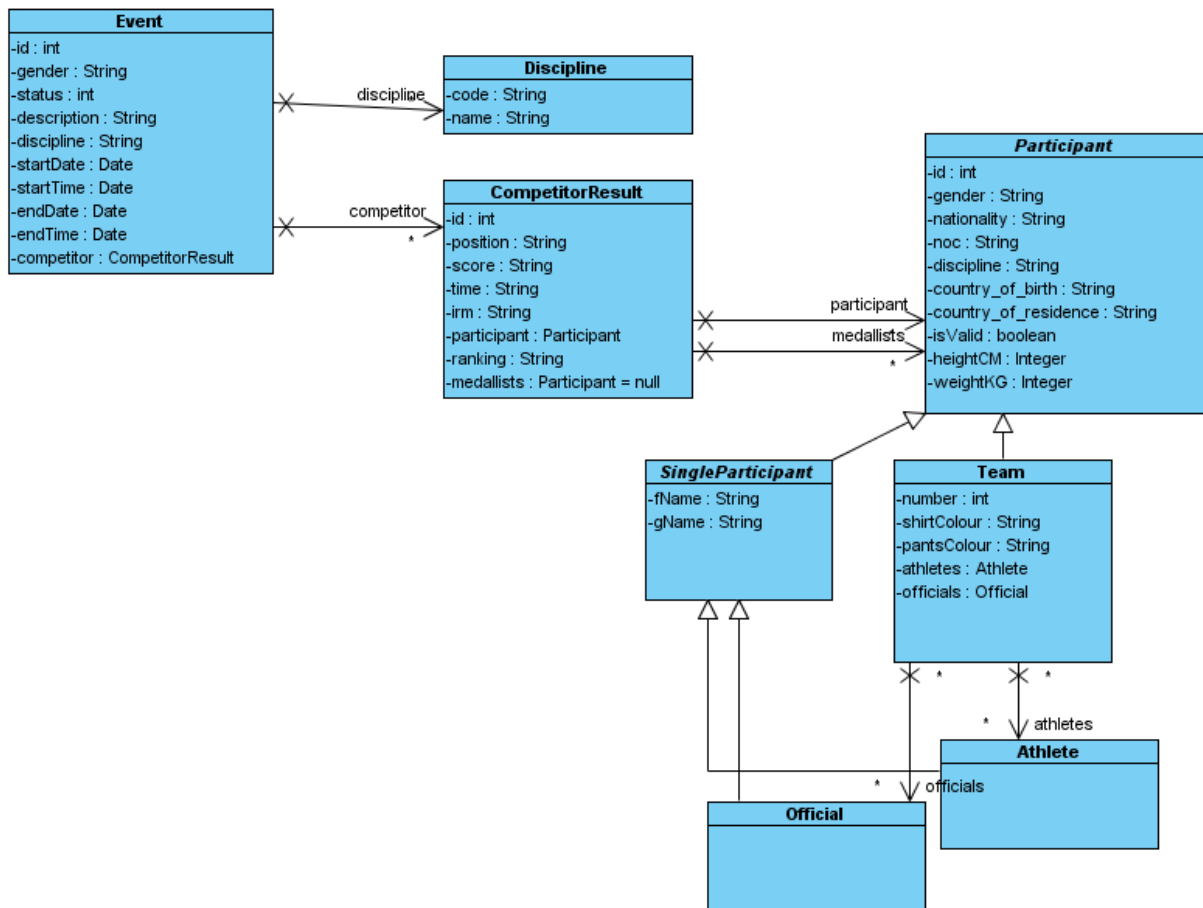


Figure 10: SIS Data Diagram

An event describes a sport competition unit within the SIS. Events within the SIS build a hierarchical structure where the root refers to the whole tournament and the leaves refer to the actual events.

- *id* as unique identifier
- *description* human readable free text describing the event
- *gender* explained later
- *start/end dates* and *times* from the event specified in local time
- *status* explained later
- list of *competitors*, athletes, officials or teams that a going to compete in the event.

The *status* of an event is defined by an integer code. Table 1 shows the different status codes as defined by the SIS:

Code	Name
1	Planned
2	Scheduled
4	In Progress
6	Results Unofficial
7	Results Official

9	Delayed
11	Cancelled
12	Protested
13	Postponed
14	Scheduled – New

**Table 1: SIS event status codes**

A *Discipline* is defined by a unique code and a human readable name. The SIS uses a predefined list of Disciplines shown in Table 2:

Code	Name	Code	Name
AR	Archery	HB	Handball
AT	Athletics	HO	Hockey
BB	Baseball	JU	Judo
BD	Badminton	MP	Modern Pentathlon
BK	Basketball	RO	Rowing
BV	Beach Volleyball	SA	Sailing
BX	Boxing	SH	Shooting
CF	Canoe Flatwater	SO	Softball
CM	Cycling - Mountain Bike	SW	Swimming
CR	Cycling Road	SY	Synchronized Swimming
CS	Canoe Slalom	TE	Tennis
CT	Cycling Track	TK	Taekwondo
DV	Diving	TR	Triathlon
EQ	Equestrian	TT	Table Tennis
FB	Football	VO	Volleyball
FE	Fencing	WL	Weightlifting
GA	Gymnastic Artistic	WP	Waterpolo
GR	Gymnastic Rhythmic	WR	Wrestling
GT	Gymnastic Trampoline		

**Table 2: SIS disciplines types**

The *gender* type defined three possible values shown in Table 3:

Code	Name
M	Male
F	Female
X	Mixed

**Table 3: SIS gender types**

*CompetitorResult* are the competitors that will participate in a specific event. When the result from the competitor is received it is also stored with this element. The *competitorResult* data types are:

- unique *id*
- *position* that has the athlete within the event when it start (lane number, order)
- *participant* from the Olympic Games. One participant can compete in several events.
- *Rank* within the event result
- *Time* when the event result is indicated with the time (for example 100m run)
- *Score* when the event result is indicated in points (for example archery)
- *IRM* invalid rank mark line DNF (did not finish), DSQ (disqualified)

- *Medallist* only applies to teams, the list of participants from a team that will get the medal

*Participant* has more information about the people that participate during the Olympic games. These participants can be athletes, officials or teams. The participant has

- Unique *id*
- *Gender*, see Table 3
- *NOC* National Olympic Committee, country the athlete / team is competing for.
- *Discipline* see Table 2
- *Nationality* from the athlete or official, the one from the passport
- *Country of birth and residence* only applies to officials and athletes
- *heightCM*, *weightKG* only informed for an athlete
- *isValid* indicate if his accreditation has been denied or not.

In case of athletes and officials (*SingleParticipant*) it is possible to retrieve the family and given names.

*Teams* are described in an own data type. Team members are referred by a list of *officials* and *athletes* using the official and athlete data type respectively. In addition colour of the shirt and the pants are defined as attributes of the team data type.

## 3.4 Examples

### 3.4.1 Example of usage of the web-services

Invocation requesting all athletic root events

```
<?xml version="1.0" ?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns3:getRootEvents xmlns:ns2="http://sis.atosorigin.es/types" xmlns:ns3="http://sis.atosorigin.es/ws">
      <arg0>
        <ns2:id>-1</ns2:id>
        <ns2:discipline>AT</ns2:discipline>
        <ns2:parentId>0</ns2:parentId>
      </arg0>
    </ns3:getRootEvents>
  </S:Body>
</S:Envelope>
```

Result (extraction from the result)

```
<?xml version="1.0" ?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns3:getRootEventsResponse xmlns:ns2="http://sis.atosorigin.es/types" xmlns:ns3="http://sis.atosorigin.es/ws">
      <return>
        <ns2:id>19385</ns2:id>
        <ns2:gender>M</ns2:gender>
        <ns2:description>Men's 100m</ns2:description>
        <ns2:discipline>AT</ns2:discipline>
        <ns2:parentId>0</ns2:parentId>
      </return>
      <return>
        <ns2:id>19425</ns2:id>
        <ns2:gender>M</ns2:gender>
        <ns2:description>Men's 200m</ns2:description>
        <ns2:discipline>AT</ns2:discipline>
        <ns2:parentId>0</ns2:parentId>
      </return>
      <return>
        <ns2:id>19465</ns2:id>
        <ns2:gender>M</ns2:gender>
        <ns2:description>Men's 400m</ns2:description>
        <ns2:discipline>AT</ns2:discipline>
        <ns2:parentId>0</ns2:parentId>
      </return>
      <return>
        <ns2:id>19503</ns2:id>
        <ns2:gender>M</ns2:gender>
        <ns2:description>Men's 800m</ns2:description>
        <ns2:discipline>AT</ns2:discipline>
        <ns2:parentId>0</ns2:parentId>
      </return>
    </ns3:getRootEventsResponse>
  </S:Body>
</S:Envelope>
```

```

<return>
  <ns2:id>19540</ns2:id>
  <ns2:gender>M</ns2:gender>
  <ns2:description>Men's 110m Hurdles</ns2:description>
  <ns2:discipline>AT</ns2:discipline>
  <ns2:parentId>0</ns2:parentId>
</return>
...
</ns3:getRootEventsResponse>
</S:Body>
</S:Envelope>

```

### Retrieving a results from a specific event

```

<?xml version="1.0" ?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns3:getResultByEvent xmlns:ns2="http://sis.atosorigin.es/types" xmlns:ns3="http://sis.atosorigin.es/ws">
      <arg0>
        <ns2:id>19385</ns2:id>
        <ns2:gender>M</ns2:gender>
        <ns2:description>Men's 100m</ns2:description>
        <ns2:discipline>AT</ns2:discipline>
        <ns2:parentId>0</ns2:parentId>
      </arg0>
    </ns3:getResultByEvent>
  </S:Body>
</S:Envelope>

```

### Result from the web-service invocation

```

<?xml version="1.0" ?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns3:getResultByEventResponse xmlns:ns2="http://sis.atosorigin.es/types" xmlns:ns3="http://sis.atosorigin.es/ws">
      <return>
        <ns2:id>137496</ns2:id>
        <ns2:participant xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ns2:athlete">
          <ns2:id>46166</ns2:id>
          <ns2:gender>M</ns2:gender>
          <ns2:nationality>JAM</ns2:nationality>
          <ns2:noc_organization>JAM</ns2:noc_organization>
          <ns2:discipline>AT</ns2:discipline>
          <ns2:country_of_birth>JAM</ns2:country_of_birth>
          <ns2:country_of_residence>JAM</ns2:country_of_residence>
          <ns2:valid>true</ns2:valid>
          <ns2:height_cm>196</ns2:height_cm>
          <ns2:weight_kg>86</ns2:weight_kg>
          <ns2:fname>BOLT</ns2:fname>
          <ns2:gname>Usain</ns2:gname>
        </ns2:participant>
        <ns2:ranking>1</ns2:ranking>
        <ns2:time>00:00:09.690</ns2:time>
      </return>
      <return>
        <ns2:id>137497</ns2:id>
        <ns2:participant xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ns2:athlete">
          <ns2:id>46224</ns2:id>
          <ns2:gender>M</ns2:gender>
          <ns2:nationality>TRI</ns2:nationality>
          <ns2:noc_organization>TRI</ns2:noc_organization>
          <ns2:discipline>AT</ns2:discipline>
          <ns2:valid>true</ns2:valid>
          <ns2:fname>THOMPSON</ns2:fname>
          <ns2:gname>Richard</ns2:gname>
        </ns2:participant>
        <ns2:ranking>2</ns2:ranking>
        <ns2:time>00:00:09.890</ns2:time>
      </return>
      <return>
        <ns2:id>137498</ns2:id>
        <ns2:participant xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ns2:athlete">
          <ns2:id>46382</ns2:id>
          <ns2:gender>M</ns2:gender>
          <ns2:nationality>USA</ns2:nationality>
          <ns2:noc_organization>USA</ns2:noc_organization>
          <ns2:discipline>AT</ns2:discipline>
          <ns2:valid>true</ns2:valid>
          <ns2:height_cm>180</ns2:height_cm>
          <ns2:weight_kg>89</ns2:weight_kg>
          <ns2:fname>DIX</ns2:fname>
          <ns2:gname>Walter</ns2:gname>
        </ns2:participant>
        <ns2:ranking>3</ns2:ranking>
        <ns2:time>00:00:09.910</ns2:time>
      </return>
      ...
    </ns3:getResultByEventResponse>
  </S:Body>
</S:Envelope>

```

### 3.4.2 Examples of real-time data

The real time data is send in a push mode. With the first message example everything will be shown, and the rest only the payload from inlineXML will be exposed here as example.

Start list with personal best and seasonal best from the athletes. Bib is also included. The `<ns3:participant>` corresponds to

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<newsItem xmlns:ns2="http://myedirector.org/event/2008-02-19/update" xmlns="http://iptc.org/std/nar/2006-10-01/"
xmlns:ns3="http://myedirector.org/event/2008-04-29/sports/update">
<catalogRef href="http://www.atosorigin.com/live/swimming_cv.xml"/>
<itemMeta>
<itemClass qcode="http://www.atosorigin.com/eventupdate"/><provider qcode="http://www.atosorigin.com"/>
<versionCreated>2009-01-27T12:43:01.687+01:00</versionCreated>
</itemMeta>
<contentMeta>
<contentCreated>Sun Jul 13 12:00:01 CEST 2008</contentCreated>
<genre qcode="http://cv.iptc.org/newsCodes/theme/15000000"/>
</contentMeta>
<contentSet>
<inlineXML>
<ns2:realtimeUpdate>
<ns2:event>817</ns2:event>
<ns2:eventDetail xsi:type="ns3:AthleticDetailType" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<ns3:athleticParticipantUpdate>
<ns3:participant>802</ns3:participant>
<ns3:bib>BRADLEY</ns3:bib>
<ns3:order>1</ns3:order>
<ns3:seasonal_best>27:21.99</ns3:seasonal_best>
<ns3:personal_best>26:32.00</ns3:personal_best>
</ns3:athleticParticipantUpdate>
<ns3:athleticParticipantUpdate>
<ns3:participant>803</ns3:participant>
<ns3:bib>GRANGER</ns3:bib>
<ns3:order>2</ns3:order>
<ns3:seasonal_best>28:50.79</ns3:seasonal_best>
<ns3:personal_best>26:55.10</ns3:personal_best>
</ns3:athleticParticipantUpdate>
<ns3:athleticParticipantUpdate>
<ns3:participant>804</ns3:participant>
<ns3:bib>MCGORUM</ns3:bib>
<ns3:order>3</ns3:order>
<ns3:seasonal_best>29:46.81</ns3:seasonal_best>
<ns3:personal_best>26:54.8</ns3:personal_best>
</ns3:athleticParticipantUpdate>
<ns3:athleticParticipantUpdate>
<ns3:participant>805</ns3:participant>
<ns3:bib>O'RAWHE HOBBS</ns3:bib>
<ns3:order>4</ns3:order>
<ns3:personal_best>26:22.51</ns3:personal_best>
</ns3:athleticParticipantUpdate>
...
</ns2:eventDetail>
</ns2:realtimeUpdate>
</inlineXML>
</contentSet>
</newsItem>
```

Message that informs the time that leader took to make the last lap (400m)

```
<inlineXML>
<ns2:realtimeUpdate>
<ns2:event>817</ns2:event>
<ns2:eventDetail xsi:type="ns3:AthleticInformationDetailType" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<ns3:athleticInformationUpdate>
<ns3:time_last_lap>1:40.0</ns3:time_last_lap>
</ns3:athleticInformationUpdate>
</ns2:eventDetail>
</ns2:realtimeUpdate>
</inlineXML>
```

Message when the leader passes a split distance (splits are at 1000, 2000, 3000 and 4000m)

```
<inlineXML>
<ns2:realtimeUpdate>
<ns2:event>817</ns2:event>
<ns2:eventDetail xsi:type="ns3:AthleticDetailType" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<ns3:athleticParticipantUpdate>
<ns3:participant>813</ns3:participant>
<ns3:mark>4:11.95</ns3:mark>
<ns3:split>1</ns3:split>
</ns3:athleticParticipantUpdate>
</ns2:eventDetail>
</ns2:realtimeUpdate>
</inlineXML>
```

Message send with results

```
<inlineXML>
<ns2:realtimeUpdate>
<ns2:event>817</ns2:event>
<ns2:eventDetail xsi:type="ns3:AthleticDetailType" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<ns3:athleticParticipantUpdate>
<ns3:participant>813</ns3:participant>
<ns3:order>12</ns3:order>
<ns3:mark>21:06.37</ns3:mark>
<ns3:rank>1</ns3:rank>
</ns3:athleticParticipantUpdate>
<ns3:athleticParticipantUpdate>
<ns3:participant>807</ns3:participant>
<ns3:order>6</ns3:order>
<ns3:mark>21:30.75</ns3:mark>
```

```
<ns3:rank>2</ns3:rank>
</ns3:athleticParticipantUpdate>
<ns3:athleticParticipantUpdate>
  <ns3:participant>814</ns3:participant>
  <ns3:order>13</ns3:order>
  <ns3:mark>22:21.40</ns3:mark>
  <ns3:rank>3</ns3:rank>
</ns3:athleticParticipantUpdate>
<ns3:athleticParticipantUpdate>
  <ns3:participant>805</ns3:participant>
  <ns3:order>4</ns3:order>
</ns3:athleticParticipantUpdate>
<ns3:athleticParticipantUpdate>
  <ns3:participant>803</ns3:participant>
  <ns3:order>2</ns3:order>
</ns3:athleticParticipantUpdate>
<ns3:athleticParticipantUpdate>
  <ns3:participant>804</ns3:participant>
  <ns3:order>3</ns3:order>
</ns3:athleticParticipantUpdate>
<ns3:athleticParticipantUpdate>
  <ns3:participant>810</ns3:participant>
  <ns3:order>9</ns3:order>
  <ns3:status>DQ</ns3:status>
</ns3:athleticParticipantUpdate>
</ns2:eventDetail>
</ns2:realtimeUpdate>
</inlineXML>
```

#### 4 Visual processing output

The vision processing technologies being realized within the other WP3 tasks require a metadata input to assist their operation and produce their own XML meta-data output based on their findings, which is subsequently passed on to other WPs (primarily WP4).

As the content of the video-streams presented to the vision algorithms varies greatly, an understanding of context can greatly improve the efficiency and accuracy of automatic content detection. For example if we know that we are 'probably' viewing a 100m race, and we also know which athletes 'should' be in the race, then we can restrict our person ID search space to a list of maybe eight people and our physical search space to a limited portion of the running-track. This knowledge comes from the meta-data described above and is initially parsed into a database, utilized by all of the image processing algorithms. Relevant/useful meta-data fields such as race type (e.g. 100m Men's final), details of which athletes are competing (e.g. names, shirt IDs, lanes, etc) and their basic attributes (e.g. heights, nationalities, etc.) are extracted. A graphical representation of the database can be seen in the diagram below; note that the two fields filled by external meta-data are: Event and Athlete.

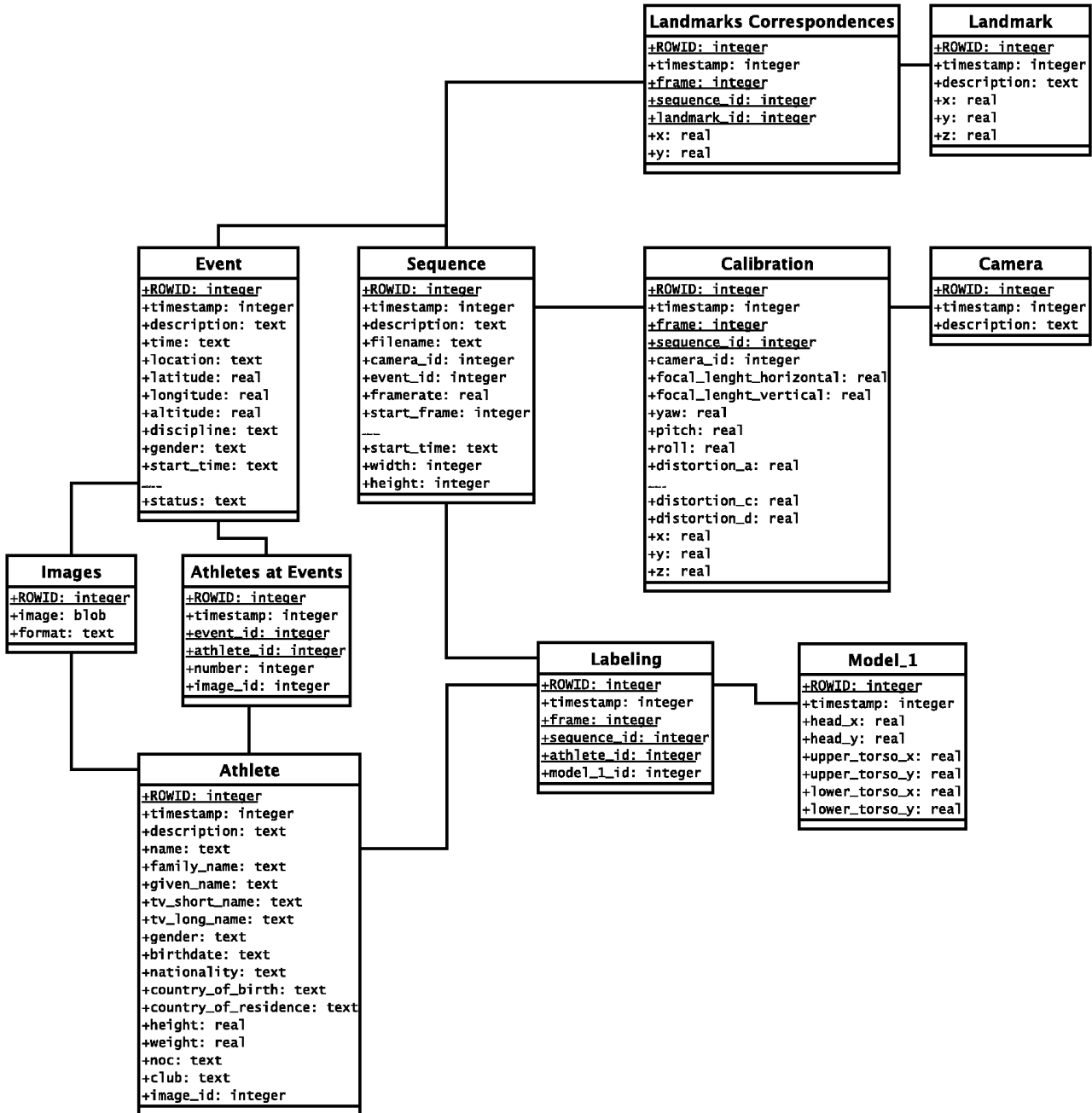


Figure 11: Visual representation of the meta-data handled by the image processing components

A more detailed description/example of the XML structure of the data relating to Event and Athlete statistics, can be seen below:

```

<?xml version="1.0" encoding="utf-8"?>
<sequence>
  <end_frame>7445</end_frame>
  <event_id>1</event_id>
  <filename>d1 e19 mens 100 final 2008_07_12 17h51m13s24 A cam01.mov</filename>
  <framerate>25</framerate>
  <start_frame>7130</start_frame>

```

```
<event>
  <description>Birmingham mens 100m final</description>
  <discipline>mens 100m</discipline>
  <gender>M</gender>
  <location>Birmingham</location>
  <start_time>2008-07-12T17:51:13.24+01:00</start_time>
  <time>2008-07-12T17:51:13.24+01:00</time>
</event>
```

```
<athlete id="1">
  <athlete_id>1</athlete_id>
  <club>Victoria P</club>
  <family_name>Fifton</family_name>
  <gender>M</gender>
  <given_name>Rikki</given_name>
  <image_id>1</image_id>
  <name>Rikki Fifton</name>
  <nationality>Uk</nationality>
  <number>1</number>
</athlete>
```

.....

The remaining fields of the database are subsequently filled by the WP3 algorithms themselves on a frame-by-frame basis based on estimated parameters such as person position, heads found, identification data such as text, etc. A short section of a typical output from a body tracker can be seen below:

```
<frame n="7130" time="2008-07-12T17:55:58.440+01:00">
  <body_tracker>
    <athlete id="1">
      <head_x>0.720</head_x>
      <head_y>0.684</head_y>
      <upper_torso_x>0.714</upper_torso_x>
      <upper_torso_y>0.675</upper_torso_y>
      <lower_torso_x>0.706</lower_torso_x>
      <lower_torso_y>0.703</lower_torso_y>
    </athlete>
  </body_tracker>
</frame>
```

The frame number in this case relates to frame 7130 of the pre-recorded video stream (d1 e19 mens 100 final 2008\_07\_12 17h51m13s24 A cam01.mov ) and has been manually labelled. Each frame also has an absolute timestamp associated (2008-07-12T17:55:58.440+01:00 in this case). As can be seen above the three points (head\_x, head\_y), (upper\_torso\_x, upper\_torso\_y) and (lower\_torso\_x, lower\_torso\_y) in 2D space have been recorded. These points represent the centroid of the head, the mid-shoulder position and the mid-hip position. The values are given as percentages across and down the image using the top left corner as a (0,0) reference.

## 5 Future

In the future we're studying the possibility to convert all XML messages to MPG7 standard. We discuss with WP5 of the information generated in the image processing will be send to the user terminal application. If it is used in the terminal application MPG7 is a usable standard option, otherwise, due to speed performance, the above output for image processing will be maintained.

Also weather information might be included in the real-time information. IDF sends general information about the weather per Venue, this information gleans the temperature (Celsius, Fahrenheit), humidity (%), wind speed (m/s), wind direction (North, North-East, etc) and precipitation (mm).

## References

[IDS] [http://www.atosorigin.com/en-us/olympic\\_games/services\\_solutions/information\\_diffusion\\_systems/](http://www.atosorigin.com/en-us/olympic_games/services_solutions/information_diffusion_systems/)