SEVENTH FRAMEWORK PROGRAMME

Collaborative project
Small or medium-scale focused research project
FP7-SEC-2011-1
Grant Agreement no. 285533

TACTICS

Tactical Approach to Counter Terrorists in Cities

Deliverable details	
Deliverable number	D6.3
Title	Fusion Unit
Author(s)	TNO UPV Morpho ISCA
Due date	30/11/2014
Delivered date	28/12/2014
Dissemination level	PU
Contact person EC	PO

Cooperative Partners	
1.	UPV
2.	TNO
3.	Morpho
4.	ISCA

Table of Contents

Executive Summary

The purposes of this report D6.3 are:

- To present a way information fusion can be used in a system as presented in the TACTICS project for handling terrorist threats in an urban environment, increasing the usefulness of information.
- To present a system design of selected examples of fusion, to be implemented for validation with the Threat Management Tool.

In the TACTICS project, key aspects of a generic approach have been selected to be demonstrated in a relevant –but simulated- environment. Report D3.2 focused on that particular approach, as example of a system according to the more generic approach described in report D3.1. This deliverable D6.3 describes both a generic use of fusion in relation to D3.1 and a concrete system according to D3.2.

The FP7 research project TACTICS is concerned with improving terrorist threat mitigation in an urban environment. In report D3.1 (TACTICS Consortium, 2013) a conceptual solution description is given, presenting an integral yet generic approach to managing a terrorist threat in an urban environment. For the approach in the TACTICS project key aspects of the generic approach have been selected to be demonstrated in an operational environment. Report D3.2 (TACTICS Consortium, 2013) focused on that particular approach, providing a design of such a system.

The TACTICS system amongst others has sensors as sources of information, often requiring further human assistance to obtain information that is directly relevant for managing the threat. Information fusion can automatically process and combine information from different sources, to provide information that is more directly relevant. Information fusion is the topic of this report, both regarding the general concept as provided in D3.1, as well as regarding a system design of selected sub-systems to be used in the system architecture described in D3.2.

In this report, information fusion for counter terrorism in general is discussed as well as a description provided of a specific Fusion Unit that interfaces with information sources and the Threat Management Tool as part of the TACTICS system.

The purpose of the fusion engine is to supply accurate and timely information, which leads to situational awareness for the threat manager. This fusion is done through observation capabilities built with resources such as databases and sensors. These capabilities and resources together create the fusion process. The fusion process accomplishes this by two distinct means:

1) data is transformed into information on a higher abstraction level: from sensor output to a signal, from a signal to an attribute of an object, from an object to a situation and from a situation to an appreciation of that situation;
2) multiple data streams are combined to improve the quality of the data with respect to the function of the respective capabilities.

In this report fusion is discussed, as it could be provided by Fusion Units communicating with a Threat Management Tool in a TACTICS system. Such Fusion Units in a general sense would be similar to other Information Sources, but instead of providing sensor data, provide information at a higher level. This would in most cases be object level information, describing an observed or detected entity, or situation information, relating entities to each other or the environment. A common object model for such information would therefor preferably be based on describing an entity as well. Defining a detailed data model would still require all users of the model to understand the meaning of all included information fields and values, and would only make sense when this is done for a larger number of systems. In the TACTICS project, two fusion systems are selected to illustrate how such systems could be used with a TACTICS system. These systems are a Face Recognition system and a Behaviour Analysis System.

The information provided by the selected fusion systems could also be provided by an operator looking at the camera feeds. The advantage of these automated systems, is that an operator can only look at a limited

number of cameras and recognize faces or appearance from a (limited) list of suspects. Behaviour recognition may be easier and more accurate by a person, but not for many systems at the same time and for a prolonged time. The information provided by the system is also not the final information used in decision making, as still an operator will observe the provided information and determine its value. For example, after an alert that a person is recognized, or a behaviour is detected for a suspect person, an operator could easily validate this information with the provided data, and decide whether this information is used in threat management.

Due to the Snowden revelations, the impression could emerge that TACTICS is overtaken by reality: “intelligence services already know everything”. First, the intelligence process is not the topic for TACTICS. In European countries there is typically a strict information barrier between secret services, and police forces. Second, so far, the revelations showed a government that also forced companies to give up information, or even spied on companies’ data flows. TACTICS is different. First, because it relies on preparation and cooperation with the owners of resources. This implies that companies put their relations with their customers also at stake. Second, because these resources are not constantly tapped, but only when there is a specific need for a specific threat. Third, TACTICS goes into depth with several technologies for physical surveillance (face recognition, behaviour recognition and behaviour profiling), which are not part of the SIGINT mission of the NSA.

ICT systems that are based on this project could have a wider scope than that of this project (countering terrorist threats in an urban environment), as it was linked to an EU research topic. As a consequence, police would have to use different systems depending on the motivation and location of the threat. This has the disadvantage that it may be necessary to switch system when a threat becomes more clear. It is therefore recommended to have police forces work with similar fusion systems also outside of terrorist threats. It follows that questions of proportionality should therefore be addressed at the level of specific capabilities in relation to specific types of threat, not at the level of a TACTICS system in general.

1 Introduction

The FP7 research project TACTICS is concerned with improving terrorist threat mitigation in an urban environment. The project goals are:

1. to make security forces capable of responding quicker, without being biased in decision making and to be more precise in the kind of information they request and the orders they send out by providing expert knowledge at the fingertips of the professionals of the security services at the time of an actual threat in urban environments;
2. to improve preparedness of security forces by decomposing threats into observable terrorist behaviours specific for urban environments;
3. to improve the capabilities at security forces' disposal by improving their management, efficiency and their cooperation in urban environments;
4. to facilitate a cross-European approach by offering a 3-levelled strategy on the tactical, operational and strategic level.

The challenges involved in mitigating a terrorist threat in an urban environment are described in deliverable D2.1 (TACTICS Consortium, 2013). In report D3.1 (TACTICS Consortium, 2013) (addressing goals 1-3) a conceptual solution description is given, presenting an integral yet generic approach to managing a terrorist threat in an urban environment.

For the approach in the TACTICS project key aspects of the generic approach have been selected to be demonstrated in an operational environment (TRL 6 [1]). Report D3.2 (TACTICS Consortium, 2013) focused on that particular approach, providing a design of such a system.

The "TACTICS system"[2] thus has a double meaning:

(1) a hypothetical class of operational systems which in some way or form resemble the TACTICS approach to mitigating terrorist threats in an urban environment. (Focus of D3.1)
(2) the concrete result of the TACTICS project; (Focus of D3.2)

The TACTICS system amongst others has sensors as sources of information, often requiring further human assistance to obtain information that is directly relevant for managing the threat. Information fusion can automatically process and combine information from different sources, to provide information that is more directly relevant. Information fusion is the topic of this report, both regarding the general concept as provided in D3.1, as well as regarding a system design of selected sub-systems to be used in the system architecture described in D3.2.

In section 2, information fusion for counter terrorism in general is discussed. In section 3 the specific Fusion Unit is described as a system that contains different fusion processing parts as sub-systems, and itself interfaces with information sources and the Threat Management Tool. The system design of these systems is described in section 4. The external interfaces to the Threat Management Tool and the interface of the Threat Management Tool are described in Annex A. In section 5 the use of a Fusion Unit as described in the earlier sections is discussed.

---

[1] Technology maturity is the degree to which a technology has been proven in a realistic operational setting (Sauser, Verma, Ramirez-Marquez, & Gove, 2006).

[2] A system is a construct or collection of different elements that together produce results not obtainable by the elements alone. The elements, or parts, can include people, hardware, software, facilities, policies, and documents; that is, all things required to produce systems-level results. The results include system level qualities, properties, characteristics, functions, behaviour and performance. The value added by the system as a whole, beyond that contributed independently by the parts, is primarily created by the relationship among the parts; that is, how they are interconnected (The Open Group Architecture Framework, 2011).

2 Information Fusion for Counter Terrorism

The purpose of the fusion engine is to supply the threat management tool with accurate and timely information, which leads to situational awareness for the threat manager. This fusion is done through observation capabilities built with resources such as databases and sensors. These capabilities and resources together create the fusion process. The fusion process accomplishes this by two distinct means:

1) data is transformed into information on a higher abstraction level: from sensor output to a signal, from a signal to an attribute of an object, from an object to a situation and from a situation to an appreciation of that situation;
2) multiple data streams are combined to improve the quality of the data with respect to the function of the respective capabilities.

These two means are discussed in more detail in the next sections.

2.1 Abstraction levels of information

In this context, different abstraction levels of information are defined, for example by the Joined Directors of Laboratories (JDL) (Steinberg, Bowman, & White, 1999), or in (Kester, 2010):

- Sensor level: data produced by a sensor (e.g., a camera image, radar signal) and related meta information (e.g., location, view direction)
- Object level: combining information about objects from different sources, for example person tracking, adding actions to observed persons, linking information by identity (not necessarily location). For example, if a person sees that a certain van (with number plate given) is located somewhere, this information should be stored, so if this van is seen at another time (and its number plate read by a person), the information of its previous location is known, and can be presented to the threat manager (possibly filtered by relevance).
- Situation level: Relations between objects can be recorded as well (i.e., an object was near a church, or is at an event), so that relevant relations can be shown.
- Impact level: object properties (van, identity) and relations (parked near church) that fit threat indications, increase priority to show the information.

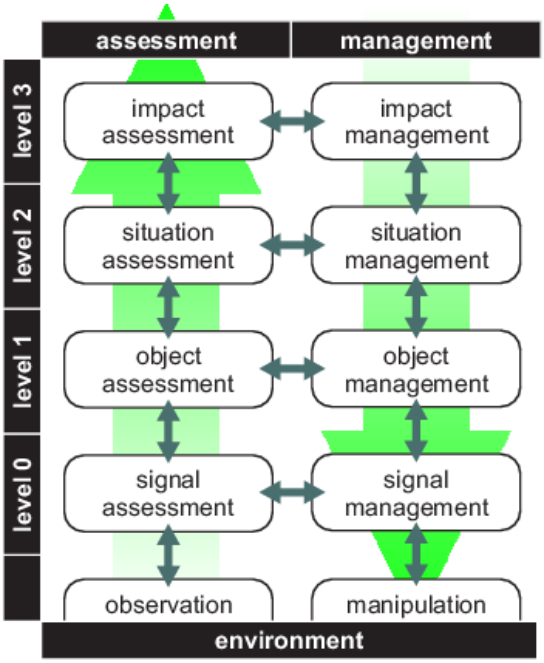

Figure 2-1 Information assessment and management on multiple abstraction levels.

Sensors are hardware information sources that produce sensor level information. Processing this information towards higher information levels can be automated, using computers applying algorithms. However, the higher the level, the more complicated such automation becomes. Often, humans perform parts, most or

even the complete information processing chain. For example, a camera with automated detection and number plate detection may provide object information (including a form of identity) as information, where a human links the location to a building, and estimates impact based on identity and location. In other cases, a human acts both as sensor and object and situation information source, for example an officer in the street, reporting the blacklisted van is seen near a religious building. Cases where higher level information processing is automated are rare. However, if parts of the chain are automated, processing by humans can be improved, for example by filtering relevant information (having an operator look at a few instead of 200 screens), or making situation assessment easier by indicating object information in a geographical view (i.e., all cars on a map). Such automated processing may require combining information about an object from different sources (for example by associating based on location, or number plate, or face description), and linking to geographical locations, and is combined in a Fusion Unit.

2.2 Merging of information

In a strict sense, information fusion is just the merging of information from disparate sources with differing conceptual, contextual and typographical representations. There are many definitions (University of Bonn – Institute of Computer Science - Communication and Networked Systems, 2012) and some related concepts:

- Data fusion is the merging of data representing the same real world object.
- Sensor fusion is the combining of sensory data or data derived from sensory data from disparate sources such that the resulting information is in some sense better than would be possible when these sources were used individually.

Because of the nature of TACTICS with resources potentially coming from a lot of different other organisations, the fusion engine limits fusion mainly to object level, i.e., getting better descriptions of an object (i.e. person, location, vehicle, etc.), by identity, location in time, or activity. Some sensor level fusion may be done, but this requires accurate metadata (a calibration for example) to get 3d world coordinates, thereby allowing for wide baseline or even non-overlapping tracking of people. This in turn also again enables data fusion, by linking information regarding these people from one location, to another.

A special case is that of the human as a sensor (3.5). A human professional can autonomously decide that something or someone is a threat, thereby seemingly bypassing the fusion engine. Such information should still be merged into the shared situational awareness, but then only on the threat level. A threat can obviously consist of multiple hostile elements together.

2.3 Surveillance design patterns

In (Rest, Roelofs, & Nunen, 2014) and (Rest, et al., 2014) a set of *surveillance patterns* were introduced: general reusable solutions to commonly occurring surveillance challenges. These are widely used in intelligence and surveillance systems. They have in common that they take data as input (situational awareness), and generate alarms as output (threat assessment). Five surveillance patterns have been identified: threshold alarm, profiling, concentric circles of protection, bag of words and scenario view. They can be employed by humans and machines alike. A fusion engine for counter terrorism should be able to apply each of these patterns to create capabilities from resources.

2.4 State of the art in fusion systems

These surveillance patterns are very generic, and widely applicable. So they will already be in use in some form or shape with many end users. This can be in the form of a physical security information management (PSIM) system for object security, as part of a “smart city” ICT system, or as part of an existing counter terrorism police ICT system.

A PSIM system is typically used in object security for critical infrastructure and other high risk objects where the workflow is rather strict. The output of subsystems such as access control, smoke detectors and intrusion detection systems is gathered in a PSIM system. The PSIM system uses programmable business rules in order to execute prescribed actions, or to present the operator with clear instructions depending on the input from the subsystems. For example, if access control registers that an access control badge has been submitted to three different doors where the owner is not authorized, then his access to his PC system is also revoked until he shows up in person at a reception desk. PSIM systems do not dynamically link to other

resources depending on the nature and development of a threat though, whereas the fusion engine of a TACTICS engine can be adapted instantly to the actual demands of the threat manager.

In the city of The Hague, a lot of international institutes are located such as the International Crime Court, Europol and others. Individually, these institutes already have a high threat level. Their physical proximity creates additional risk, but therefore also opportunities for increased security. The presence of a specific threat in their observation area could also lead to a dynamic support of the local police with the institutes' (security) resources and capabilities. This leads to the concept of a "smart city".

A smart city is a less clear concept. It is typically used to refer to a city that uses "smart ICT" in multiple aspects: healthcare, mobility, government and others. As such, it should be seen as an approach to tackle multiple big city problems, including climate change, ageing populations and changing retail and entertainment demands. Urban security can be one of those challenges, but typically not at the high risk level of (counter-)terrorism. On the other hand, it is a valid question whether "a TACTICS system" should be interpreted as a separate system-of-systems, or more like "one of the actors in a smart city". This is the topic of the next section.

Due to the Snowden revelations, the impression could emerge that TACTICS is overtaken by reality: "intelligence services already know everything". First, the intelligence process is not the topic for TACTICS. In European countries there is typically a strict information barrier between secret services, and police forces. Second, so far, the revelations showed a government that also forced companies to give up information, or even spied on companies' data flows. TACTICS is different. First, because it relies on preparation and cooperation with the owners of resources. This implies that companies put their relations with their customers also at stake. Second, because these resources are not constantly tapped, but only when there is a specific need for a specific threat. Third, TACTICS goes into depth with several technologies for physical surveillance (face recognition, behaviour recognition and behaviour profiling), which are not part of the SIGINT mission of the NSA.

The Dutch reality is the project "Sight". The goal of this project is to investigate how the situational awareness of a police officer can be improved in the case of a crisis (not just terrorism) on the basis of dynamically connecting to public and private resources, using integration standards and analysis and visualization techniques. In terms of TACTICS, Sight is taking a combination of the fusion engine and some parts of the threat manager to a next technology readiness level (D3.1 TACTICS Conceptual Solution Description, 2013).

2.5 Procurement of a TACTICS fusion system

The fact that an EU research topic that led to this project had a certain scope (countering terrorist threats in an urban environment), does not imply that ICT systems that are based on this project should have the same scope. That would have several consequences:

- Police would have to use different systems depending on the motivation of the threat (terrorism versus other motivations);
- Police would have to use different systems depending on the location of the terrorist threat (urban environment versus other environments);

These consequences have some obvious disadvantages. First, it becomes unclear or –on the other hand- too abrupt to decide when to switch from one system to another. For example, at the moment that the second plane flew into the World Trade Centre, it was all of sudden clear that it was a terrorist attack. Up to that moment it could still be an accident. That means that in the middle of the crisis, police forces would have to switch to another system. That should be avoided as much as possible. Another disadvantage would be that training and building experience with the system would be difficult, including finding and fixing teething troubles.

So, it is recommended to have police forces work with similar fusion systems also outside of terrorist threats. It follows that questions of proportionality should therefore be addressed at the level of specific capabilities in relation to specific types of threat, not at the level of a TACTICS system in general.

3 TACTICS Fusion Unit

3.1 Introduction

In this section, the Fusion Unit is described in relation to the TACTICs system, as a system that contains different fusion processing parts as sub-systems, and itself interfaces with information sources and the Threat Management Tool. It links directly to the System Architecture as described in D3.2 (TACTICS Consortium, 2013). The fusion data system is defined in relation to the other parts in section 3.2. As examples of fusion, two systems are chosen. In section 3.3 these are described at a functional level. An object data model is discussed in 3.4. A fusion process can be fully automated, but can also incorporate help by a human operator. For example, an operator could assist an automated process in following a person over different cameras, providing the corrected recognition as output to the Threat Manager. The whole process could even be done by a trained person, who acts as human sensor, combines information at object and situation level, and provides high level information to the Threat Manager. An example of this is provided in section 3.5.

In addition to the chosen fusion and information sources, more complex combinations, using different information sources are possible as well. In section 3.6 such scenarios are described, explaining the potential of fusion, using the chosen systems as basis.

3.2 Fusion System Definition

In D3.2, the system architecture of a TACTICS system is presented. It is based on three functionalities:

- The Threat Decomposition Tool allows obtaining indicators of a threat from a database, based on known indicators, and assists in communicating this information back to Threat Management.
- Capability Management (CM) improves the knowledge on available capabilities at security forces' disposal. The Capability Management Tool helps the Capability Manager to keep track of capabilities, and match them to required information needs of Threat Management.
- The Threat Management process makes security forces capable of responding quicker, without being biased in decision making and to be more precise in the kind of information they request and the orders they send out by providing expert knowledge at the fingertips of the professionals of the security services at the time of an actual threat in urban environments. The Threat Management Tool assists in this process, by providing obtained information in a more useful way, and enabling clearer interaction with Threat Decomposition and Capability management.

Figure 3-1 shows these three tools, with three users (the managers) that operate the tools and are the main connection to the outside of the system. In addition, there are the Information Sources, which are not part of the system, but are used to obtain new information. Information sources can be sensors, but also officers in the field, or databases of information. Information sources can have different capabilities. A simple sensor may only be able to provide a single measurement (e.g., weight), more complex sensors or persons can have a wide range of capabilities. Capability management keeps track of these capabilities, including information about location and availability. In deliverable D5.2 (TACTICS Consortium, 2014), it is described how capabilities can be described, and matched to requests from Threat Management, using a similar description as used in Threat Decomposition, using Morphological Analysis (Zwicky, 1969).

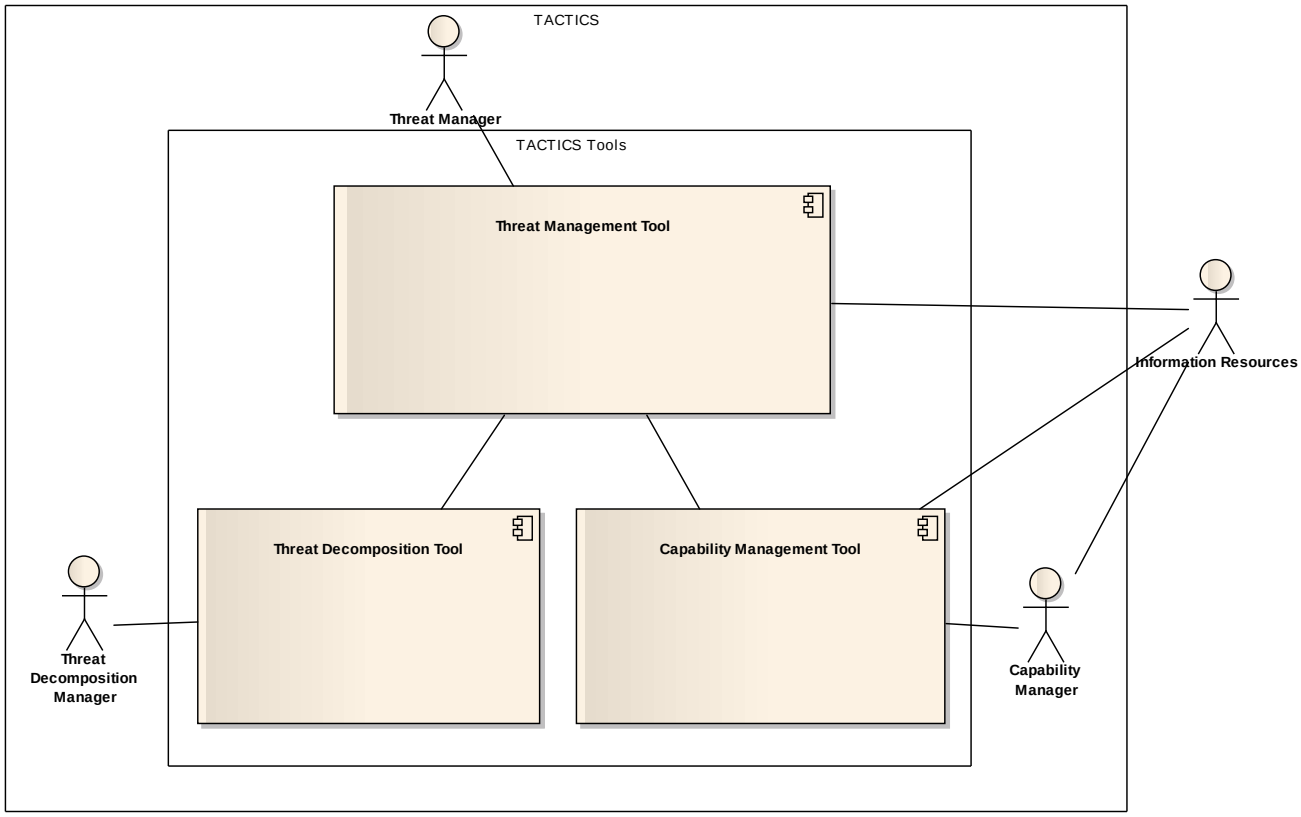

Figure 3-1 System view with main actors and three sub-components

The flow of information would then be an iterative process, where Threat Management queries Threat Decomposition based on current information for new hypotheses, then queries Capability Management for capabilities that could provide this information. Based on provided information about for example availability (e.g., how long would it take to obtain information), the Threat Manager would chose a capability and get access to the information source, which would then report to Threat Management. The Tools in TACTICS facilitate this process, by formalising some of the descriptive language, and providing (pre-formatted) communication.

Information Fusion is the process of combing information of one or more sources and transform this into new information, that is more relevant. A Fusion Unit provides this processing, and is an Information Source in itself. It has a capability that depends on that of the Information Sources it uses, for example in geographical extent of its observation area, or availability. As such, it takes the place of an Information Source as shown in Figure 3-1, resulting in Figure 3-2.The Information Sources used by a Fusion Unit can be capabilities by itself as well.

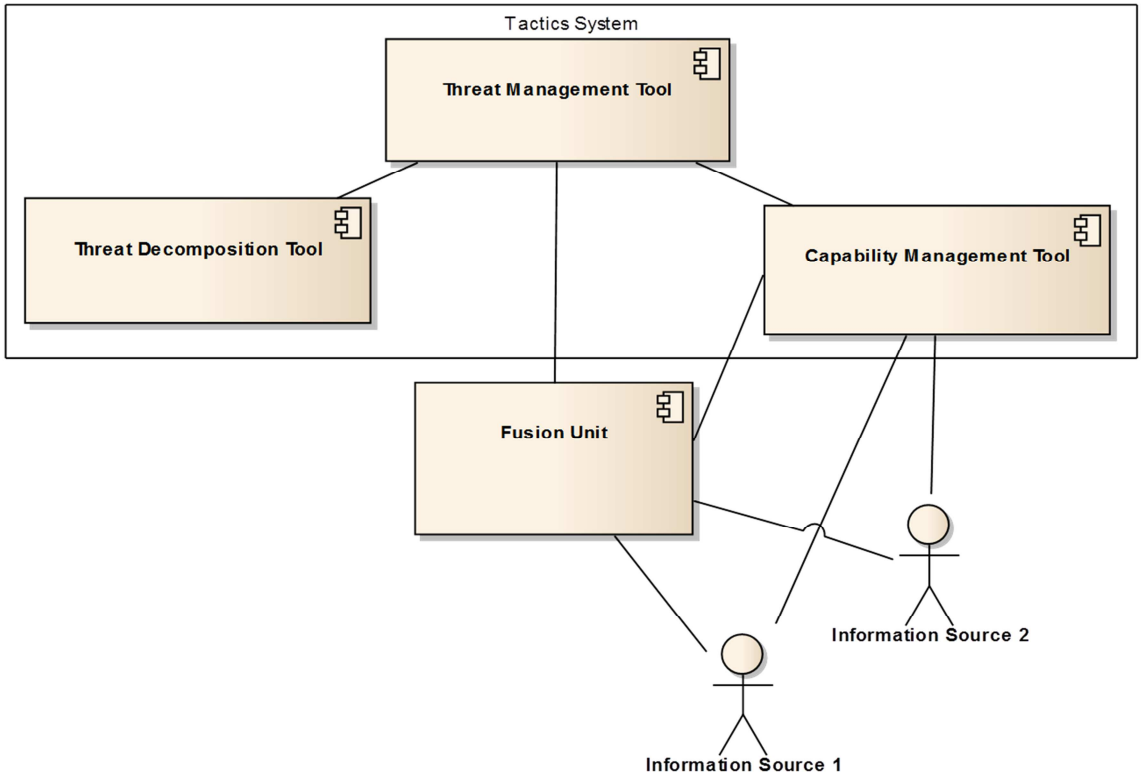

Figure 3-2 System view with the Fusion Unit as new Information Source

3.3 Selected Fusion Systems

As examples of Fusion Systems, different forms of fusion are considered to be demonstrated with the Threat Management Tool. All use camera systems as their Information Sources, and provide higher level information on persons. In general these are:

- Face recognition: matching persons in a camera to persons in a database
- Single camera tracking: tracking persons in a single camera
- Multi-camera tracking: linking single camera tracks over multiple cameras, allowing people to be followed over larger distances
- Activity Recognition: Analysing persons in a camera, detecting behaviours

Each of these forms of fusion could be a separate capability, as shown in Figure 3-3. In this case, three of them are linked to a single camera, providing information related to the coverage of that camera. The multi-camera tracking includes single camera tracking, but subsequently links tracks in one camera to earlier tracks in other cameras, using logical reasoning (can a person have moved this way) and some similarity (can this be the same person). This latter functionality could be used to follow persons that were marked suspect. Following persons over a longer time may become less accurate, as more persons may look alike and track breaks may occur. For that reason, combination with other capabilities could improve the information. For example, face recognition could be used to link single camera tracks, as shown in Figure 3-4.

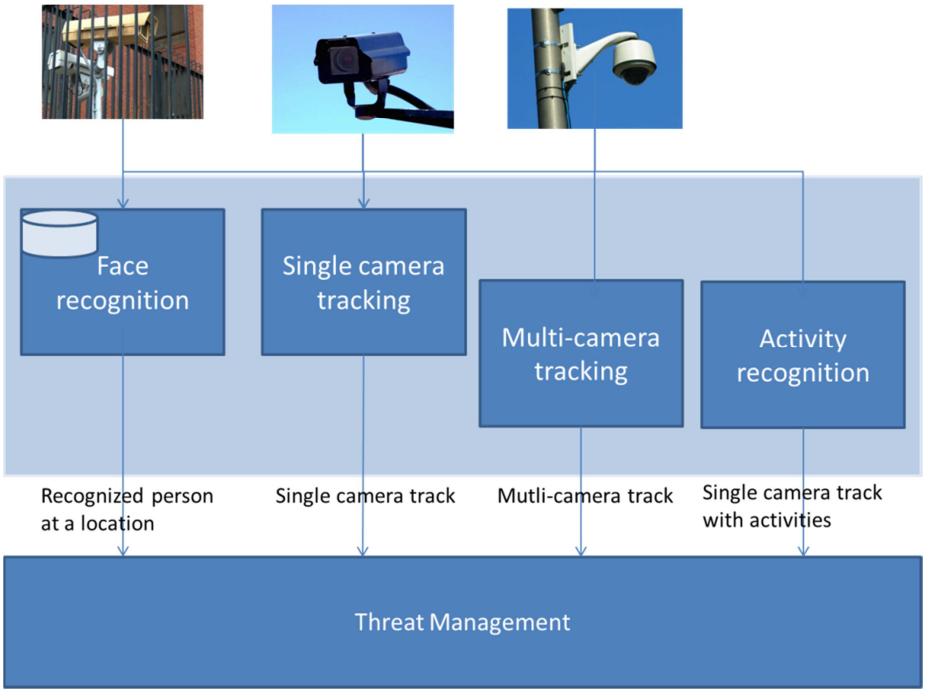

Figure 3-3 Different forms of fusion, as separate systems

In some cases, it may not be clear who of several persons is related to a threat. For example, an explosives sniffer could give an alarm, when several people are within range of the sensor. Following all persons would provide difficult information to process by an operator, and, with possible inaccuracies in tracking, could only provide weak evidence. The same may be the case for Behaviour Analysis. Short term behaviours, such as running or loitering would not necessarily make a person suspect. However, combining this with the weak evidence of multi-camera tracking, might link this behaviour to possibly suspect persons, limiting the number of occurring cases to a manageable number. In Figure 3-4 and Figure 3-5 this combination is indicated, for a combination with and without Face Recognition respectively. The latter is selected as an example of use of fusion in the TACTICS system, with the following two Fusion Systems:

- ***Face recognition*** Based on a video flow, this sub-system provides two different functionalities:
  - Blacklisting: Recognize a person considered as a suspect (in this case, the photo of this suspect needs to be previously registered in the sub-system)
  - Whitelisting: Indicate if a person is not on a list of people that are expected at a location
  - Loitering detection: Indicate if a person is repeatedly seen at the location
- ***Behaviour Analysis System***: From local observations of persons in a camera, the system can label a number of activities, such as loitering, meeting. Requested behaviours can be detected, but only sent as alarm for persons similar to persons present in an indicated area in an image of a given camera.

These systems will be further described in the System Design presented in section 4.

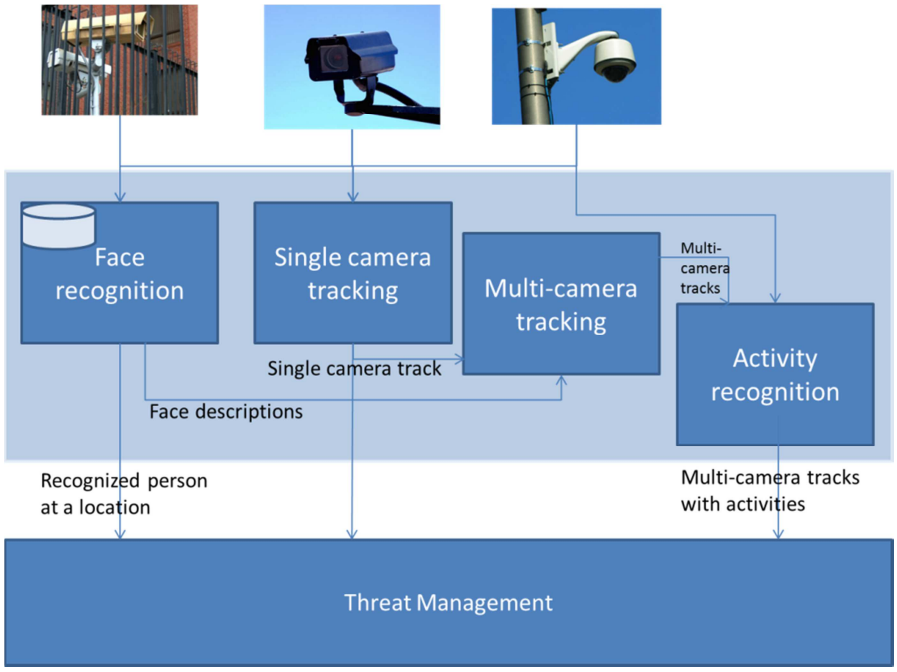

Figure 3-4 Different forms of fusion, combining face recognition with multi-camera tracking and behaviour analysis.

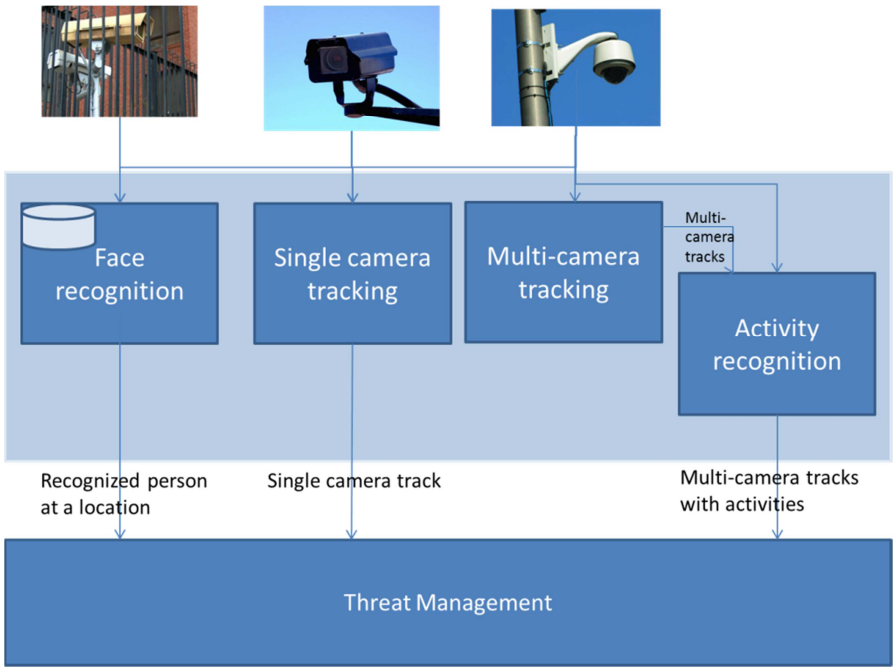

Figure 3-5 Different forms of fusion, combining tracking and behaviour analysis

3.4 Object Data Model

For the Threat Management Tool to be able to interact with many different fusion systems, it has to understand all different forms of information these systems provide. To make this easier, it is advantageous to have a data model that is the same for several different systems.

Information by most fusion systems will be related at object or situation level. I.e., it will specify information about an object (person, car) or their relation to other objects (geographical objects, location, groups they belong to). It therefor makes sense to have the Object Data Model be at object level and contain (part of) the following information:

- Identity information: for example, any of:
  - an ID, unique to the Fusion Unit
  - labels linking to known identities (e.g., faces in a blacklist)
  - A classification label (i.e., person, car, suitcase, etc.)
  - Other identity information, such as a license plate number
- Location information: where the object is now, was before, and how it is moving (speed, maximum velocity)
- Descriptive information:
  - Related to appearance, such as for example colour, size, shape
  - Related to behaviour, such as current activity
- Relation information:
  - Current geographical relation: is the object near an object, e.g., a building, or is a person meeting another (specified) person
  - Relations in time: is the object related to an earlier observed object.

For the current selected Fusion Units, several variables in the object information are fixed. For example, all objects the systems report on are persons. The Face Recognition system provides identity information linked to the blacklist provided by the TMT, which therefor knows the meaning of the information. For the Behaviour Analysis, a set of tracks is marked suspect by defining a time and location in a camera, and a behaviour to be detected is defined. In both systems, the location of the object is mostly defined by the location of the camera. Therefore, the data provided by the systems is relatively simple. Internally, the TMT could link information from different sources (i.e., an identity from Face Recognition and behaviour from Behaviour Analysis) for which an object model could be of use, but with current level of implementation this is not needed.

3.5 Search Detect React: The Human as a Sensor

Search Detect React® (SDR) is a proactive security method and philosophy that provides the capabilities to detect imminent threats to public order in mass crowd environments, and the tools to effectively prevent harmful events (Cohen, SDR From Stop and Search, to Search and Stop, 2013). SDR reinvents prevention by introducing an advanced proactive approach that uses heightened awareness and human behavior as leading components. The SDR standard (Cohen, The SDR Standard, 2013) bridges gaps between upper echelon expectations and on-the-ground capabilities (see section 2.2), enhances the value of technologies by providing heightened awareness capabilities to operators, and makes use of latent resources. The tools provided by SDR enhance capabilities to prevent illicit activities ranging from minor crime to major attacks. SDR gives both a broader and more nuanced aspect to situational awareness because it is based on a comprehensive understanding of the local definition of normality of a specific environment. When someone has a comprehensive understanding of what is normal in a specific environment from a behavior perspective, abnormal (deviant) behavior stands out (Cohen & Pliner, Searching for Abnormalities Instead of Suspects, 2012).

SDR training is possible in many different levels for any capacity, meaning that an officer working traffic control and an officer tending to a potentially violent demonstration can each contribute to the greater situational awareness with their heightened awareness capabilities to identify key indicators. To be able to effectively contribute each trainee is trained with SDR heightened awareness capabilities and patented auto-critique tools that allow them to detect abnormal behavior with regard to their specific environment while negating ethnic profiling and other biases from their policing practices.

To effectively contribute, SDR-trained officers would furthermore need the freedom to follow their professional opinions. In some frameworks law enforcement officers are hesitant to take proactive steps to prevent potential incidents because they are only authoritzed to react to crimes in progress or those that have already been committed. SDR is a method focused on prevention, and so trained officers need the freedom to take legitimate actions (calmly speaking to a citizen, for example)—in cases where auto-critique measures dictate—to observe their behavior and make an informed decision.

SDR training includes several auto-crititque mechanisms that ensure that officers' decisions are knowledge-based decisions based on behaviors and not on suspicions, biases or personal judgements. Furthermore, since SDR-trained personnel search for behavioural abnormalities instead of suspect profiling, achievement is not just measured by the number of stops or arrests, but rather by behavioural abnormalities noticed, therefore maintaining importance on the effectiveness and efficiency of the method in practice and not on competing statistics.

3.6 Use of the Fusion Unit

In deliverable D2.3 (TACTICS Consortium, 2013), several scenarios are defined in which the use of a TACTICS system would be relevant. In such scenarios, the selected Fusion Units would be used instead of operators looking at for example CCTV cameras, trying to observe deviant situations, such as unknown persons at restricted locations, known terrorists, or possibly suspect persons acting in a certain way. To fit the Fusion Units in such a scenario, information that might be in the mind of the operators, needs to be specified: who is known or unknown, when is someone possibly suspect and what behaviour is deviant. This information would be input through the TMT and therefore not be part of the Fusion Unit itself as defined in the next chapter. Therefore, here examples are provided of how such systems could be used. The first three explain how the implemented systems could be used. Further variations include possible combination of systems, but are not implemented.

Face Recognition with Blacklisting

After receiving intelligence information about a certain threat to a possible event, and after consulting with Threat Decomposition, focus may be on a specific terrorist organisation. A list of members of that organisation may be available, and for some of those a passport photograph may exist. When querying Capability Management for possibilities to be alerted of the presence of members of the organisation at the location of the event, the list of capabilities may include officers that could walk around, operators that might look at CCTV cameras, but also a Face Recognition System located at an access route to the event. After selecting this capability, the TMT would send images of members of the organisation to form a black list in the Face Recognition System. When a person on the list would be recognized with some certainty, the Threat Manager would be alerted via the Threat Management Tool, and verify whether this indeed is the person on the list. If this is not directly clear, via Capability Management another capability might be selected to verify, for example, a security guard near the entrance could be sent to check the passport of the person.

Face Recognition with Whitelisting

After receiving intelligence information about a certain threat to a possible event, and after consulting with Threat Decomposition, it might be concluded because the event is not public, or large areas of the venue have limited access, to look for persons that are not supposed to be at certain locations. Capability Management may indicate that many areas are already controlled by access through personal keys, but that of other areas some are covered by cameras with Face Recognition capability. Via he Threat Management Tool photographs of personnel that might be expected to be present in these areas will be uploaded to the Face Recognition System. If a person is detected in the covered areas that is not found on the white list, an alarm is sent to the TMT, alerting the Threat Manager of possible unauthorized access. Via the Capability Management a capability may then be selected to further investigate, for example, having an operator use a pan-tilt-zoom camera to have a closer look at the person and what he is doing, or have a security officer go to the location and have a look. As an extension of this functionality, the alarms may be limited to persons that are observed at the location several times, and are not on the white list, indicating people loitering in the area.

Behaviour Analysis

After receiving intelligence information about a threat at an event, and after consulting with Threat Decomposition, it might be concluded that since the event is open to public, and no specifics are known about who might perform the threat, more general capabilities will be used to detect suspect persons. One of such capabilities might be an explosives sniffer, or other CBRN sensor. Due to the number of people moving around, use of such a sensor may not point to a single suspect, but mark all persons around that location at the time of detection as possible threats. A capability to accurately follow all these people around the area may not be available, as many people are moving around, and mistakes in tracking them are likely. Another possible way of detecting persons involved in the threat could be by looking at (short term) behaviour, such as a person loitering in an area where this is less normal. On its own, this behaviour may lead to too many false alarms, as people may loiter for different reasons. As a combined capability however, detecting people loitering that are similar to one of the persons present at the earlier detection, may lead to finding someone

involved in an attack, without too many false alarms. Using this Behaviour Analysis System, the persons present at the sniffer at time of detection have to be indicated in the TMT, by selecting them in an image of a camera that is part of this system and observes the area covered by the sniffer. In addition, a behaviour that the system can detect and is expected to be deviant, such as loitering is indicated. The system will then send an alarm to the TMT when the behaviour is detected for a person similar to the indicated persons, with information indicating the area and time in the camera that observed the behaviour. An operator could than follow this person further using the cameras, or security personnel could be send to check on the suspect person.

SDR: The Human as a Sensor

A situation in which TACTICS is employed would, by definition, be one of a known terrorist threat and/or an impending or already executed terrorist attack. In such a situation SDR-trained personnel—human sensors—can detect individuals who are playing some part in the execution of the attack, by detecting searching behaviors that in SDR terminology are called the Urban Hunter™ energy. Someone involved in an impending or executed attack is an Urban Hunter—they are on the lookout for their signal, transportation, contact person, event, whatever it is that indicates to them their next move. The Urban Hunter energy is characterized by physical mental and physiological indicators of a person's demeanor that reveal their search for, wait for, and/or persuit of something. SDR provides trained individuals with the capabilities to detect the urban hunter energy, and fusion with additional situational information allows them to determine if an urban hunter is a threat. As such, a police officer may observe Urban Hunter behaviors such as loitering in an inlikely spot, a tense demeanor/muscle tension, and a fixed gaze in a specific direction. These observations fused together with the officer's comprehension of normal behavior in that environment and information and/or intelligence received from the TDM, may lead that officer to determine that the observed individual is a potential threat.

Variation combining Face Recognition and Behaviour Analysis

A variation on the above, using a combination of the selected systems, could use Face Recognition to assist linking persons based on similarity. As in the previous example, some trigger would indicate persons as possible suspects. If imagery of the persons would be of enough quality, their faces could be stored in a database. If the Behaviour Analysis would then detect the requested behaviour, the detected person could be checked with the stored faces and an alarm only sent if there is a match.

Variation combining person similarity with other event detection

Without behaviour analysis, using similarity of persons detected in different occasions may already be useful. For example, if CBRN sensors in a crowded event give alarms at different locations and/or at different times, this would result in two groups of suspect persons. Using other similarity, from face recognition or from the multi-camera tracking with recognition, might allow for a quick indication of the person present at both detections. If such processing would be activated at a first alarm, it would alert the Threat Manager at a subsequent alarm with already an indication of the probable person, allowing for a quicker action.

4 Fusion Unit system design

4.1 Introduction

In this chapter, the implemented Fusion Systems are described in more detail. In section 4.2 a fusion unit as a closed system is defined, with sensor input and alarm output, and interfacing for control. In sections 4.3 and 4.4 a functional description is given of the two selected fusion systems. In section 4.5 requirements are discussed, as defined in work package 6.4 [D6.4]. Interfacing of the fusion unit is described in general in section 0, with detailed interfacing descriptions in Annex A.

4.2 System definition

In section 3.2, fusion systems are defined in relation to the tools of a TACTICS system, and basic information sources. In this chapter, the two selected fusion systems will be described in more detail, regarded as systems on their own, taking the place of the generic Fusion Unit in Figure 4-1.

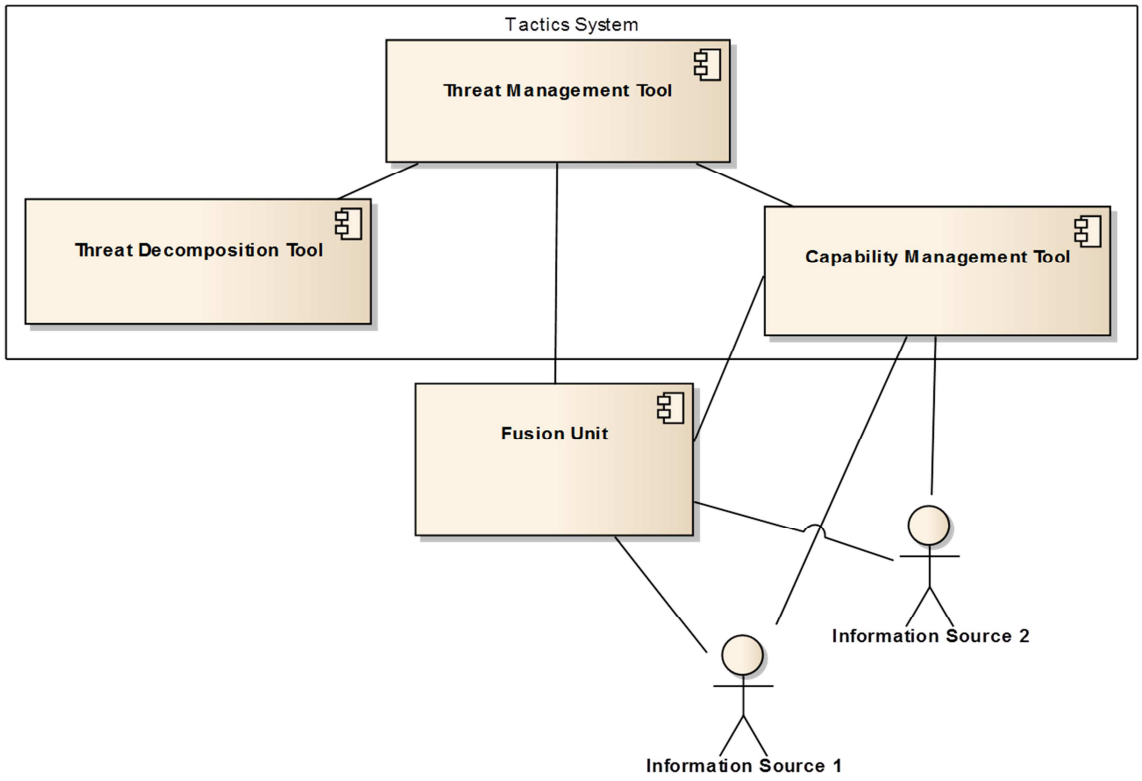

Figure 4-1 System view with the Fusion Unit as a system connected to the TACTICS system tools and information sources

For both the selected fusion systems, Face Recognition and Behaviour Analysis System, input of the system is one or more camera systems as Information Sources. The systems have made a description of themselves available to Capability Management. The Threat Management Tool communicates to the Fusion Systems information needed to define the requested functionality, such as faces to be recognised, indication of suspect persons, and requested suspect behaviours. The systems in turn communicate alarms to the TMT if the requested event occurs. The functionality of the Face Recognition system, including what information needs to be provided to the system, and what alarms will be sent, is described in section 4.3. Similarly, the Behaviour Analysis System is described in 4.4. Although their functionality is different, the use of the two systems is very similar: information on who is suspect is sent to the system, and alarms are obtained if such a person is probably observed, with information linking to the camera that made the observation. Interfacing of the systems with the TMT is similar for both as well, as described in section 0, although with current implementation small differences exist and have to be handled by the TMT. A more detailed common object data model (see section 3.4) could facilitate an even more uniform interface, but it would still require

knowledge of the meaning of information. For example, there is a difference in meaning between using an image to specify a suspect person, or an area in an image. Defining a common specification for the current systems would therefore not give much advantage, as most of the information would have one or the other system as only example.

Information about the systems will be provided to the CMT, indicating functionality for detecting entities, but especially coverage of the system (related to the coverage of the cameras). As a validation, the TMT could request from the CMT a list of systems that can detect (higher level information about) entities, in a certain area, and obtain a link to the specific system, after which the system would be used as explained in the next two sections.

4.3 Fusion subsystem Face Recognition

In the case of a terrorist attack, it is important to be able to catch the perpetrator as soon as possible. When the analysis of the cameras of the crime scene helps identifying the perpetrator, the possibility of indicating the current location of this suspect really is an asset.

Detecting abnormal behaviour (such as someone who is passing several times in front of a given camera when he is not supposed to), can also help preventing a terrorist attack. However, it is also important not to create too many false alarms and being able to manage staff who has to pass in front of a given camera several times.

Consequently, 3 different “face recognition related” functionalities have been implemented in TACTICS:

- Identifying a suspect on a given camera. This functionality is called “blacklist management” (this list contains faces of suspects who may have generated an attack)
- Detecting a suspect who passes several times in front of the same camera. This functionality is called “loitering management”
- Not raising any alarm when the detect person is authorized to do so. This functionality is called “whitelist management” (staff list)

If all those functionalities are activated, the sub-system works as follows:

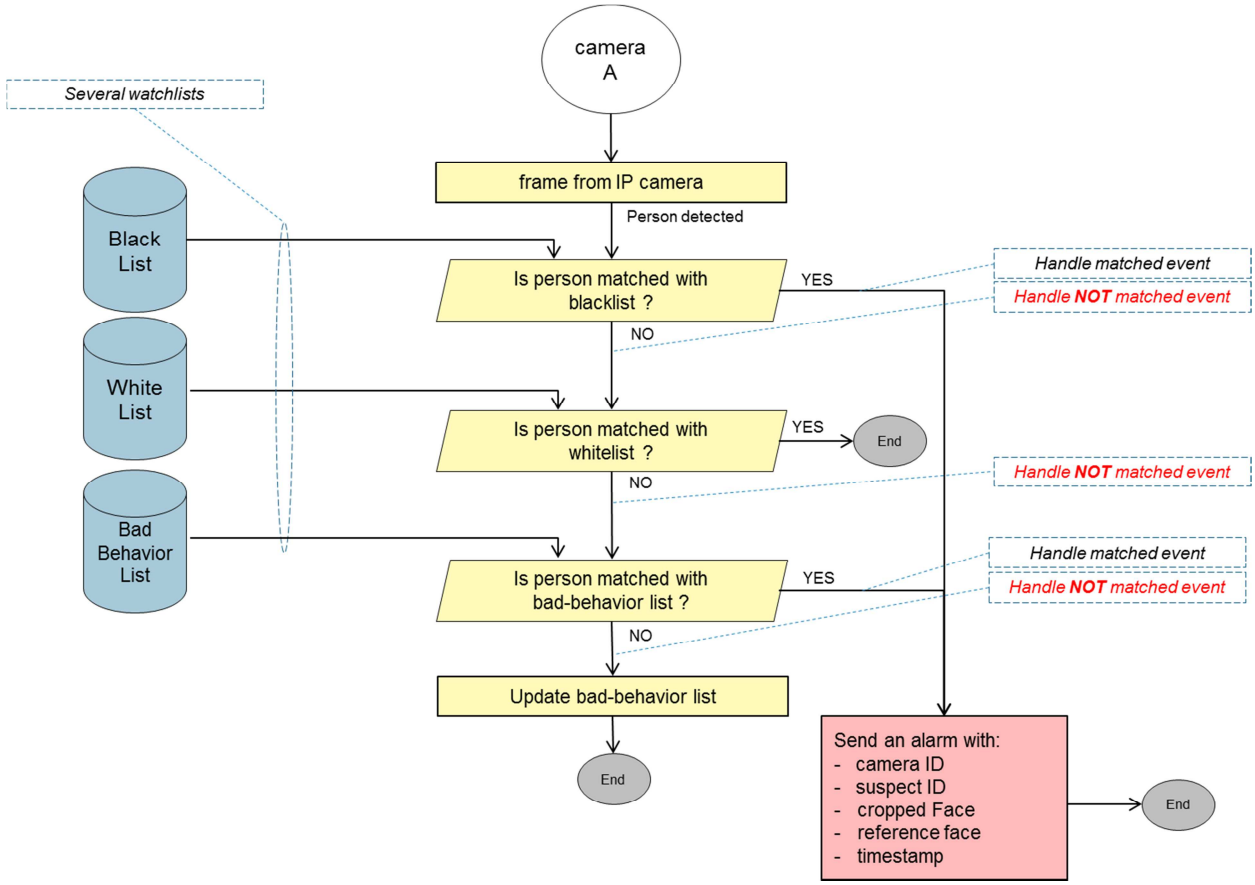

Figure 4-2: Face recognition workflow

The selected camera sends its frames to the sub-system. Each frame is interpreted and faces of detected people are retrieved. Those faces are then compared to a list of other faces.

First of all, they are compared to the blacklist. If the person looks like one of the suspects, an alarm is raised and sent to the TMT. If not, the person is matched against the whitelist. If the result indicates the person is one of the authorized people that can go through the camera several times, no alarm is raised and the system will interpret the next detected person or frame. If not, the sub-system verifies the detected person has not been seen passing several times in front of the camera. If it is the case, this person is considered as a suspect and an alarm is raised.

Those functionalities can be activated or not, depending on the TMT manager decision.

Technically, the face recognition sub-system needs several data as inputs. For each selected camera, the following elements are required:

- Blacklist (list containing the pictures of the suspect the system wants to identify)
- Whitelist (list containing the faces of the authorized people -staff members-)
- Mode selection (if the 3 functionalities are activated or not on the selected camera)
- Camera characteristics (width, height, frame rate)
- Video flow

As a result, the possible outputs would be:

- Alarm 1: “Person identified as a suspect”
- Image of the detected person on the selected camera
- Reference image from which the sub-system has been compared to
- Alarm 2: “Person identified several times on the same camera”
- Image of the detected person on the selected camera
- Reference image from which the sub-system has been compared to

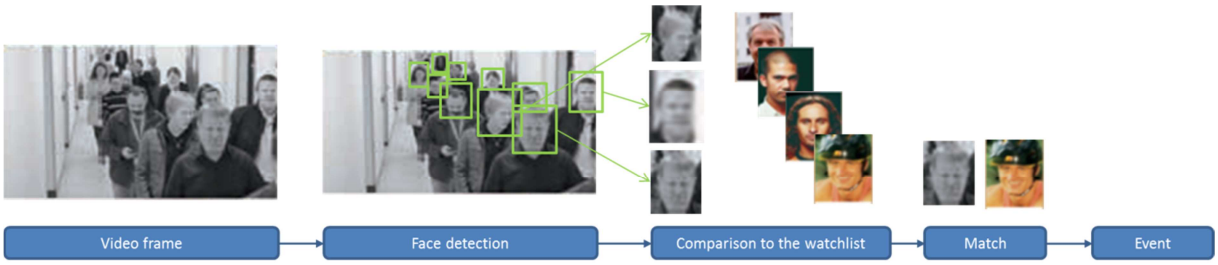

Figure 4-3: Example of a blacklist management case

4.4 Behaviour Analysis System

The Behaviour Analysis System combines two forms of processing: tracking persons over multiple cameras, and detecting behaviour. Where each of these may not provide enough support to declare someone suspect, or provide a high probability of false alarms, the combination of the two can be used to obtain relevant information.

Multi-camera tracking

In areas with multiple cameras with non-overlapping field-of-view (i.e., security cameras in a stadium, or CCTV in a town square), it is beneficial if tracks of persons in one camera can be linked to tracks in others, by recognizing the person, in order to show a longer history (where has this person been), or to follow a person (where has this person gone). At TNO a system is being developed that performs tracking of persons in cameras, and then links tracks in different cameras based on colour and kinematics (Bouma, et al., 2013). This is an operator assisted system, i.e., an operator who is looking for a person (indicated by a track), is shown possible other tracks of this person, ordered by likelihood. A real-time version is currently running in a shopping mall. As a more automated system, without assistance of an operator to provide tracks, the cross-track similarity can be used to link currently observed tracks to persons that were previously marked as suspicious, for example if they were present at a certain location at a given time.

Of tracks of a person in one camera, over time a description is obtained, which will be used to compute a likeliness to other person tracks. Selecting one (or more) tracks in one camera, it can indicate likely similarity candidates at other times (past or current). In an offline use, this can help an operator to more efficiently find a person. The current system will be used in a real-time mode, labelling current tracks as likely matches to an initial reference set. This set can be defined by indicating an area (in view of at least one camera) at a given time, marking all persons at that location as reference. A trigger to do this could for example be an uncertain explosives detection by a sniffer.

Behaviour detection

Labelling of behaviour may give a good indication of a suspect person, either by detecting specific behaviours that are known to be suspect, or by finding behaviour that is not found in non-suspect persons. Given a track of a person, activities can be determined from describing motion key points in the camera image sequences, and other information derived from the tracks, such as motion of tracks relative to each other, changes in shape of tracks, etc. (Bouma, et al., 2013). The processing used for the TACTICS Fusion Unit are track based, and was shown to detect more complex behaviours, such as pick-pocketing (Bouma, et al., 2014). For the current implementation, a limited number of less complex behaviours is implemented, such as ‘loitering’ (moving around in a limited area for a longer time). Similar to the multi-camera tracking, this processing uses single-camera tracks to determine behaviour, and in itself would provide a list of tracks that at a time show the requested behaviour.

Combination of multi-camera tracking and behaviour detection

The recognition of tracks that are similar to reference tracks, or show a requested behaviour each are not accurate, or not specific enough to result in an acceptable number of (relevant) alarms. It makes therefor sense to combine the two processing steps into one fusion system, as shown in Figure 4-4. It includes detection and tracking of persons in a single camera, providing single camera tracks to both the recognition and behaviour analysis processing. For each single camera track, an appearance is determined, and behaviour is described. In a multi-camera step, similarities of any track to another can be determined. This takes in account kinematics, eliminating similarity if a person could not have moved from one observed area to another in the time that passed. Combinations can also be limited to similarity with the reference tracks, i.e., those that were present in the area in the indicated camera at the reference time. By thresholding similarity and confidence in the detected behaviour, tracks are filtered and only tracks remain for which an alarm is sent to the Threat Management Tool. This alarm includes a link to the current observation, defined by the timestamp, camera id and region in the image where the person is shown, and a reference to the request by the TMT.

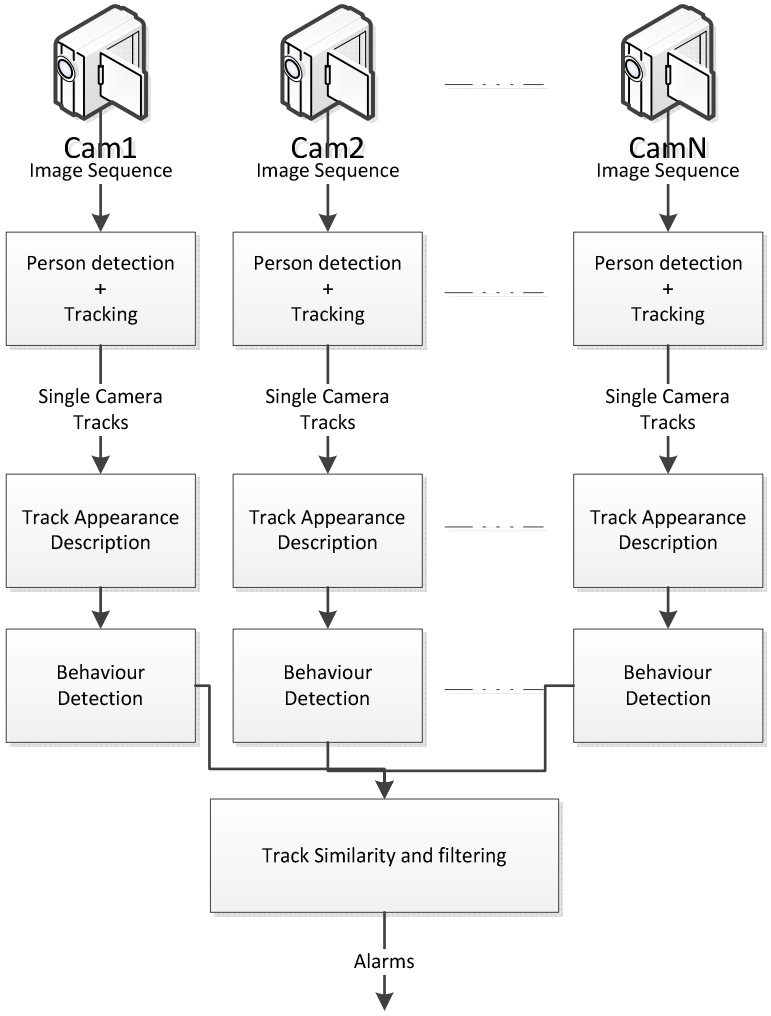

Figure 4-4 Behaviour Analysis System

The processing is controlled by the Threat Management Tool, which sends requests to send a message if a certain action occurs, with high similarity to one of the reference tracks defined by the specified image region of interest of a camera (by ID) and time.

4.5 Requirements

In document D6.4 (TACTICS Consortium, 2014), requirements are defined for the Threat Management Tool related to the Fusion Unit. For most of these requirements, this also puts requirements on the fusion unit. In this section, the requirements are discussed as related to the Fusion Unit. These requirements are repeated in Table 4.1, with the ID as it is listed in D6.4 in the first column. In the second column, ‘M’ indicates it must

be implemented, 'S'(should be implemented) is a slightly lower requirement. The third column gives the description of the requirement, with comments for clarification between brackets, if one was provided. In the 4th column, a description is given of how this requirement could be met. The last column indicates whether this was done in the current example implementations.

For requirements *FU_TMT_REQ#001* and *FU_TMT_REQ#006*, there is a link to the Capability Management Tool. The description of the capability of a Fusion Unit, as it is provided to the CMT for the Fusion Unit, should include a description of the resulting information, but also on the sensors it uses as input, such as what area is covered by the sensor (and thus the Fusion Unit). The CMT should also know and indicate possible ethical and juridical restrictions on the use of the data.

For requirement *FU_TMT_REQ#005*, with an alarm, information about the confidence or accuracy could be passed. Internally, such information probably is present as a numerical value (e.g., between 0% and 100%). For a user, such information may not have a clear meaning. For the TMT as recipient of the data, without knowledge of how this value came to be, it will also be difficult to convert it to a meaningful value, e.g., for one system 50% may indicate the alarm is still likely, where from another system this may indicate it is likely not true. Therefore, labels such as 'Very probably not', 'likely not', 'likely', 'very likely', 'certainly' are more useful than numeric values, but they should be provided by the Fusion Unit to the TMT. Since such translation is not currently available for the chosen units, no indication is given with the data, and a 'very likely' indication should be assumed.

Table 4.1 Fusion Unit requirements

ID		Description (comment)	How to handle	Implemented
FU_TMT_REQ#001	**S**	Description Data from sensors will be able to be shown directly at the TMT in a specific screen.	Make description (e.g., location) of sensors available via CMT	Fixed definition in CMT
FU_TMT_REQ#002	**M**	Fusion Unit Module will receive inputs directly from sensors and perform different fusion processes offering the results of these processes to the TMT as enhanced information.	Sensor data is direct input of a Fusion Unit, and for example sensors are not controlled separately by the TMT, which sees the fusion output.	Cameras are accessed by the fusion units, and alarms are send to TMT
FU_TMT_REQ#003	**S**	Sensor fusion module will be configured through TMT.	Information defining the requested fusion result are send by TMT to the fusion units.	E.g., Blacklist images, camera area with suspect persons and suspect behaviour are send by TMT
FU_TMT_REQ#004	**M**	Sensor Fusion approach must be re-traceable (This means that on request the user gets all information required to assess whether the fusion makes sense)	Information about used sensors and software (e.g., version, manufacturer) should be retrievable	Manufacturer and version is assumed implicit with CMT (not specifically in provided data). With alarms, information to retrieve image of the person is sent.
FU_TMT_REQ#006	**S**	Ethical and juridical restrictions must be shown to the end user.	Such restrictions would be listed with the capability as provided by the CMT	No specific implementation
FU_TMT_REQ#005	**M**	Description Information on the fusion quality should be provided	Some confidence on the output should be provided, e.g., a label with 'low', 'high' could be included with alarm data.	Currently not implemented (assume 'very likely').

4.6 Interfaces

From Figure 4-1, three interfaces can be determined:

- From Fusion Unit to Capability Management
- Between Threat Management and the Fusion Unit
- Between information sources and the Fusion Unit

In this section, these are discussed in general, and to what extent they are implemented for the selected Fusion Unit systems. A detailed description of interfaces defining the interfaces between the Threat Management Tool and the Fusion Units is given in Annex A.

Interface from Fusion Unit to Capability Management

The capabilities of the Fusion Unit have to be known to Capability Management, so they can be matched to requests for capabilities from Threat Management. This description includes on what entities information is provided (e.g., person, car), and what functionality is provided (e.g., detection, identification). Other relevant information is where this functionality is available. This would generally be the area covered by the information sources that are input to the Fusion Unit. This information would be available in the Fusion Unit, but since interfacing between the Fusion Unit and the Capability Management Tool is not foreseen in this project (where a Fusion Unit was almost defined as being part of the Threat Management Tool), the information will be added offline.

Interface between Threat Management and the Fusion Unit

This is the most important interface and has communication in both directions. Communication is initiated by the Threat Management Tool, providing information to the Fusion Unit describing the requested information, e.g., faces to recognize or behaviours to detect. The Fusion Units will acknowledge that they received and understand this information. In case the Fusion Unit would not be able to provide the requested information (e.g., because it is not available or due to access rights) it would send a rejection. The Fusion Unit would also provide information on how it would receive the information, for example providing a URL where the Fusion Unit is requested to send alarms. Either when a Fusion Unit has received specific information fully defining the request, or using a separate control, processing would start, and information will be sent, corresponding to the request. Processing would stop when either the request is removed by the TMT, or via a control message. Finally, the Fusion Unit could be released and made available again.

For the selected Fusion Units, information about available information sources (the cameras) can be obtained. Information needed to define the requested information can be set, i.e., faces for black or white lists for Face Recognition, and regions in an image of a camera indicating suspect persons for Behaviour Analysis. Processing is started either by setting a mode and value to enable processing for a camera in the Face Recognition system, or defining a behaviour. Processing is stopped when this information is removed.

Between information sources and the Fusion Unit

The connection of the Fusion Unit to the information sources is handled by the Fusion Units. If processing is only for selected information sources, this information has to be passed by the Threat Management Tool to the Fusion Unit. Available information sources are either provided by Capability Management, or can be requested from the Fusion Unit, as is the case with the implemented Face Recognition system. In a real system, where information sources may be useable for different capabilities (e.g., an operator might want to control a camera as well), there would also be communication between the information sources and Capability Management, to regulate access to the information source. In the implemented systems, it is assumed the Fusion Units have control over the information sources and such communication is not needed.

5 Discussion and Conclusions

In this report fusion is discussed, as it could be provided by Fusion Units communicating with a Threat Management Tool in a TACTICS system. Such Fusion Units in a general sense would be similar to other Information Sources, but instead of providing sensor data, provide information at a higher abstraction level. This would in most cases be object level information, describing an observed or detected entity, or situation information, relating entities to each other or the environment. A common object model for such information would therefor preferably be based on describing an entity as well. Defining a detailed data model would still require all users of the model to understand the meaning of all included information fields and values, and would only make sense when this is done for a larger number of systems. In the TACTICS project, two fusion systems are selected to illustrate how such systems could be used with a TACTICS system. These systems are a Face Recognition system and a Behaviour Analysis System. The communication of these systems to the Threat Management Tool is implemented in a similar way, with small differences for which the Threat Management Tool needs to adjust. As a communication protocol, a REST interface over HTTP is used. This allows for a flexible communication, passing information in standard forms such as JSON. The HTTP protocol also handles communication and error messaging in a structured way, where more dedicated connections over TCP/IP would for example require more effort to keep a connection alive. Securing communication, and adding access right handling can be done using the secured version of HTTP (HTTPS) and user authentication using a password.

The information provided by the selected fusion systems could also be provided by an operator looking at the camera feeds. The advantage of these automated systems, is that an operator can look at a limited number of cameras and recognize faces or appearance from a (limited) list of suspects. Behaviour recognition may be easier and more accurate by a person, but not for many systems at the same time and for a prolonged time. The information provided by the system is also not the final information used in decision making, as still an operator will observe the provided information and determine its value. For example, after an alert that a person is recognized, or a behaviour is detected for a suspect person, an operator could easily validate this information with the provided data, and decide whether this information is used in threat management.

ICT systems that are based on this project could have a wider scope than that of this project (countering terrorist threats in an urban environment), as it was linked to an EU research topic. As a consequence, police would have to use different systems depending on the motivation and location of the threat. This has the disadvantage that it may be necessary to switch system when a threat becomes more clear. It is therefore recommended to have police forces work with similar fusion systems also outside of terrorist threats. It follows that questions of proportionality should therefore be addressed at the level of specific capabilities in relation to specific types of threat, not at the level of a TACTICS system in general.

6 References

Bouma, H., Baan, J., Burghouts, G., Eendebak, P., van Huis, J., Dijk, J., & van Rest, J. (2014). Automatic detection of suspicious behavior of pickpockets with track-based features in a shopping mall. *Proc. SPIE, 9253*.

Bouma, H., Baan, J., Landsmeer, S., Kruszynski, C., Van Antwerpen, G., & Dijk, J. (2013). Real-time tracking and fast retrieval of persons in multiple surveillance cameras of a shopping mall. *Proc. SPIE, vol. 8756*.

Bouma, H., Burghouts, G., de Penning, L., Hanckmann, P., ten Hove, R., Korzec, S., . . . Schutte, K. (2013). Recognition and localization of relevant human behvaiour in videos. *Proc. SPIE, vol. 8711*.

Cohen, R. (2013, 11). *SDR From Stop and Search, to Search and Stop*. Retrieved from http://rancohensdr.wordpress.com/2014/11/13/sdr-from-stop-and-search-to-search-and-stop/

Cohen, R. (2013). *The SDR Standard*. Retrieved from SDR: http://sdr.eu.com/vision.php#TheSDRStandrd-tab

Cohen, R., & Pliner, J. (2012). Searching for Abnormalities Instead of Suspects. *Future Security Conference.* Berlin.

Fielding, R. (2000). *Architectural Style and the design of network-based software architectures.* University of California Irvine: PhD Dissertation.

Kester, L. (2010). Method for designing networking adaptive interactive hybrid systems. *Interactive Collaborative Information Systems, vol. SCI 281*, 401 – 421.

Rest, J. v., Grootjen, F., Grootjen, M., Wijn, R., Aarts, O., Roelofs, M., . . . Kraaij, W. (2014). Requirements for multimedia metadata schemes in surveillance applications for security. *Multimedia Tools and Applications 70.1*, 573-598.

Rest, J. v., Roelofs, M., & Nunen, A. v. (2014). *Deviant behaviour - Socially accepted observation of behaviour for security - Summary.* TNO.

Sauser, B., Verma, D., Ramirez-Marquez, J., & Gove, R. (2006). From TRL to SRL: The concept of systems readiness levels. *Proceedings of the Conference on Systems Engineering Research*.

Steinberg, A. N., Bowman, C. L., & White, F. (1999). Revisions to the JDL data fusion model. *International Society for Optics and Photonics, AeroSense'99*.

Steinberg, N. (1999). Revisions to the JDL data fusion process model. *In Proceedings of the 1999 National Symposium on Sensor Data Fusion.*

TACTICS Consortium. (2013). *D2.1 Urban Factors Overview.*

TACTICS Consortium. (2013). *D2.2 Requirements.*

TACTICS Consortium. (2013). *D2.3 Scenario's.*

TACTICS Consortium. (2013). *D3.1 TACTICS Conceptual Solution Description.*

TACTICS Consortium. (2013). *D3.2 System Architecture.*

TACTICS Consortium. (2014). *D5.2 Matching Capabilities to Needs.*

TACTICS Consortium. (2014). *D6.4 Information Management tool functional requirements.*

The Open Group Architecture Framework. (2011). *TOGAF 9.1, Introduction, Section 3: Definitions*. Retrieved 2014, from http://pubs.opengroup.org/architecture/togaf9-doc/arch/

University of Bonn – Institute of Computer Science - Communication and Networked Systems. (2012). *Definitions of Sensor Data Fusion in the Literature*. Retrieved 2014, from http://net.cs.uni-bonn.de/wg/sensor-data-and-information-fusion/what-is-it/sdf-definitions/

Zwicky, F. (1969). *Discovery, invention, research through the morphological approach.* Toronto: The Macmillian Company.

Annex A External interfaces

In this section, the interfacing of the fusion modules is described, as they are used by the TMT as well as the use of the modules by the TMT. A general description of interfaces is given in section A.1. In sections A.2 and A.3 the interfaces for the Morpho Face Recognition modules are described, and in sections A.4 and A.5 the interfaces to the TNO Behaviour Analysis System.

A.1 General interfacing

A general, open, scalable and modular communications paradigm must be established between TMT and data fusion processing units, paying attention to the possibility to be scalable considering the addition of other systems in the near future. With the TMT as data collector and the fusion production trigger and the TNO and Morpho systems elaborated information generators on demand of the TMT, and being the communication a request-response mechanism between TMT and fusion subsystems, one natural approximation can be the usage of Service Oriented Architecture (SOA) approaches.

SOA is a communications paradigm that decouples producers from consumers abstracting them from the underlying communication means. Communication is oriented to services which provide information at a high level. There is a step forward in simplicity and abstraction called RESTful services (Representational State Transfer) (Fielding, 2000).

The REST approach simplifies a Service Oriented Architecture by the following :

- A stateless client/server protocol
- A well-defined set of operations, atomic and unambiguous
- A url-based universal syntax to uniquely identify resources
- The usage of hypermedia, both for information and the exchange and maintainment of state.

There is also a wide availability of APIs, tools and resources to implement REST services on any kind of S.O (for example HTTP protocol) and programming language/environments. Therefore, for the communication between TACTCIS data fusion systems a REST approach will be used which is described in the next sections.

A.2 Face Recognition Module interfaces

A.2.1 Morpho API

The MORPHO API is based on a HTTP REST interface following the CRUD (Create, Read, Update, and Delete) rules. JSON can be used for message formatting. This interface allows interacting with 3 resources:

- Reference
- Watchlist
- Camera

A Reference resource is a person face picture.

A Watchlist resource is composed of a list of references. Two types of watchlists can be defined for TACTICS:

- Blacklist
- List of authorized people

All detection from a given blacklist will trigger an alarm message.

The list of authorized people is only required in case of an enabled loitering mode. Only loitering persons who are not in the authorized list will trigger an alarm message.

A Camera resource contains all information required to access the video stream (url, login, password, …) but it is also linked to watchlists. The same watchlist can be used by several cameras.

A.2.2 API flow

The process for launching a face recognition activity on a specific camera can be presented as follows:

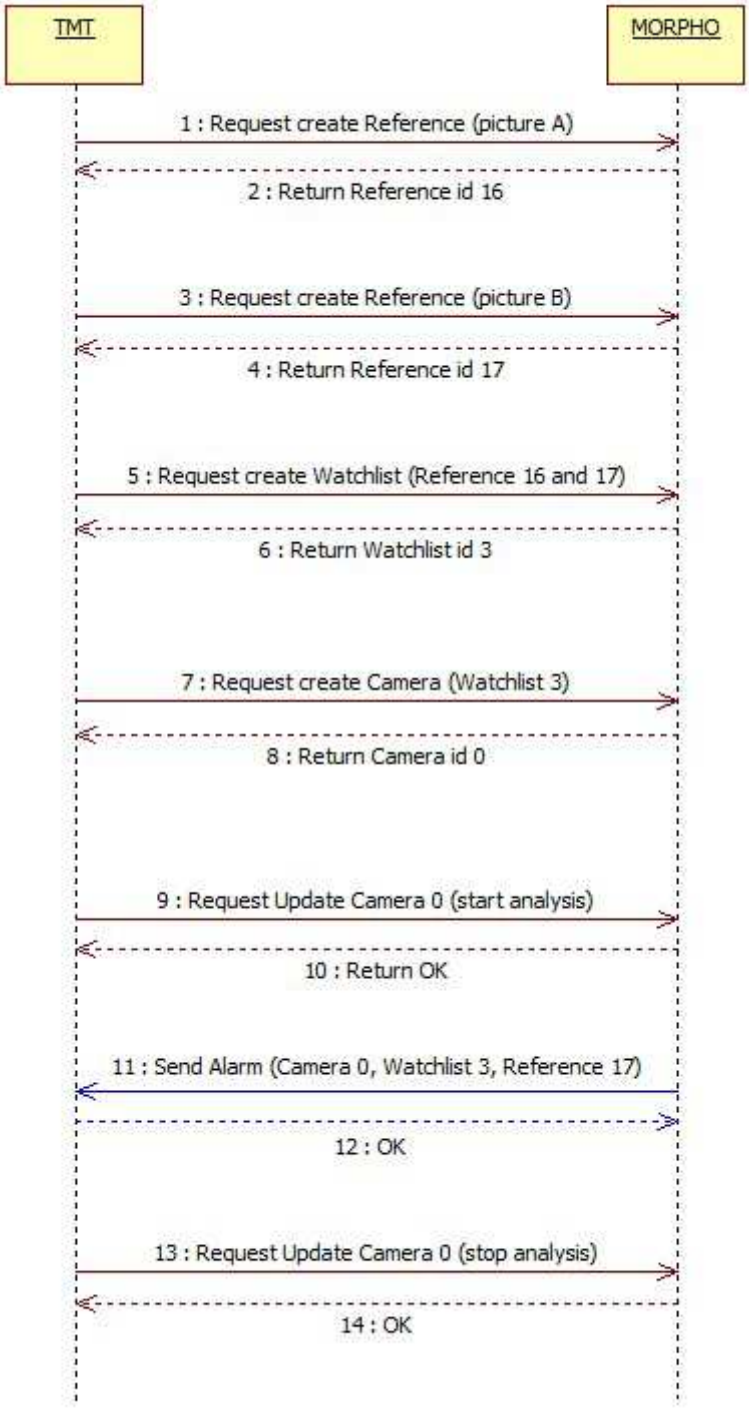

Figure 6-1: Face recognition process example

Here, 2 picture references (steps 1-4) have been inserted into a given watchlist (steps 5-6), which has been associated to a given camera (steps 7-8). The face recognition analysis starts (steps 9-10). An alarm is generated when one of the people in the blacklist is recognized or when an unauthorized person is loitering in front of the camera (steps 11-12). Finally, the analysis is stopped (steps 13-14).

A.2.3 Reference

A reference resource represents the face picture of a person. The table below lists the required data for describing a reference.

Field	Type	Read / Write	Example
id	int	Read	16
picture	string	Read & write	<base64 encoded picture string>

Table 6.1: Data required for defining a reference resource.

A.2.3.1 Create

This is an example of a HTTP POST request to create a new Reference.

request	POST morpho/references
	{
	“picture”: “<base64 encoded picture string>”
	}

OK response	HTTP 201 Created
	{
	“id”: 16,
	“picture”: “<base64 encoded picture string>”
	}

Error response	HTTP 400 Bad Request
	{
	“message”: “bad picture format.”
	}

A.2.3.2 Read

An already created Resource can be accessed by sending a HTTP GET request. The resource's id should be part of the URL.

request	GET morpho/references/16 (no body)

OK response	HTTP 200 OK
	{
	"id": 16,
	"picture": "<base64 encoded picture string>"
	}

Error response	HTTP 404 Not found
	{
	"message": "reference (id=16) not found"
	}

A.2.3.3 Delete

A reference can be deleted by requesting a HTTP DELETE on the full URL resource.

request	DELETE morpho/references/16 (no body)
OK response	HTTP 200 OK (no body)

Error response	HTTP 404 Not found
	{
	"message": "reference (id=16) not found"
	}

A.2.4 Watchlist

A Watchlist is composed by several references.

The table below lists the required data for describing a watchlist.

Field	Type	Read/Write	Example
id	int	Read	3
references	int[]	Read & write	[16, 17,18]
type	string	Read & write	“blacklist” or “authorizedPeopleList”

Table 6.2: Data required for defining a watchlist resource

A.2.4.1 Create

To create a new watchlist, a simple HTTP POST request should be sent with all the fields.

request	POST morpho/watchlists
	{
	“references”: [16, 17, 18],
	“type”: “blacklist”
	}

OK response	HTTP 201 Created
	{
	“id”: 3,
	“references”: [16, 17, 18],
	“type”: “blacklist”
	}

Error response	HTTP 400 Bad Request
	{
	“message”: “unknown type.”
	}

A.2.4.2 Read

A watchlist can be retrieved by sending a HTTP GET request.

request	GET morpho/watchlists/3 (no body)

OK response	HTTP 200 OK
	{
	“id”: 3,
	“references”: [16, 17, 18],
	“type”: “blacklist”
	}

Error response	HTTP 404 Not found
	{
	“message”: “watchlist (id=3) not found”
	}

A.2.4.3 Update

References can be added on an existing watchlist by sending a HTTP PUT request.

request	PUT morpho/watchlists/3
	{
	“id”: 3,
	“references”: [16, 17, 18, 19, 20],
	“type”: “blacklist”
	}

OK response	HTTP 200 OK
	{
	“id”: 3,
	“references”: [16, 17, 18, 19, 20],
	“type”: “blacklist”
	}

Error response	HTTP 404 Not found
	{
	“message”: “watchlist (id=3) not found”
	}

A.2.4.4 Delete

Watchlist can be deleted by sending a HTTP DELETE request.

request	DELETE morpho/watchlists/3 (no body)
OK response	HTTP 200 OK (no body)

Error response	HTTP 404 Not found
	{
	“message”: “watchlist (id=3) not found”
	}

A.2.5 Camera

The camera can have one or several watchlists. Camera access information values cannot be changed after the resource creation.

- Blacklist mode can be enabled by setting true on the field “is_blacklistmode_enabled”.
- Loitering mode can be enabled by setting true on the field “is_loiteringmode_enabled”.
- The video analysis can be started by setting true on the field “is_started”.

The table below lists the required data for describing a camera.

Field	Type	Read/Write	Example
id	int	Read	0
url	string	Read	“http://192.168.0.1:8080/axis-cgi/mjpg/video.cgi”
login	string	Read	“tactics”
password	string	Read	““t@ct!c$”
resolution	Int[2]	Read	[800, 600]
watchlists	Int[]	Read & write	[3, 4]
is_started	boolean	Read & write	false or true
is_blacklistmode_enabled	boolean	Read & write	false or true
is_loiteringmode_enabled	boolean	Read & write	false or true

Table 6.3: Data required for defining a camera resource

A.2.5.1 Create

A HTTP POST request is sent to create a new Camera resource.

request	POST morpho/cameras
	{
	“url”: “http://192.168.0.1:8080/axis-cgi/mjpg/video.cgi”,
	“login”: “tactics”,
	“password”: “t@ct!c$”,
	“resolution”: [800, 600],
	“watchlists”: [3, 4],
	“is_started”: false,
	“is_blacklistmode_enabled”: true,
	“is_loiteringmode_enabled”: false
	}

OK response	HTTP 201 Created
	{
	“id”: 0,
	“url”: “http://192.168.0.1:8080/axis-cgi/mjpg/video.cgi”,
	“login”: “tactics”,
	“password”: “t@ct!c$”,
	“resolution”: [800, 600],
	“watchlists”: [3, 4],
	“is_started”: false,
	“is_blacklistmode_enabled”: true,
	“is_loiteringmode_enabled”: false
	}

Error response	HTTP 400 Bad Request
	{
	“message”: “Camera authentification failed”
	}

A.2.5.2 Read

A HTTP GET request is sent to retrieve the information of a camera.

request	GET morpho/cameras/0 (no body)

OK response	HTTP 200 OK
	{
	“id”: 0,
	“url”: “http://192.168.0.1:8080/axis-cgi/mjpg/video.cgi”,
	“login”: “tactics”,
	“password”: “t@ct!c$”,
	“resolution”: [800, 600],
	“watchlists”: [3, 4],
	“is_started”: false,
	“is_blacklistmode_enabled”: true,
	“is_loiteringmode_enabled”: false
	}

Error response	HTTP 404 Not found
	{
	“message”: “camera (id=0) not found”
	}

A.2.5.3 Update

A HTTP PUT request can be used to:

- Add/remove watchlists from the camera
- Enable/disable blacklist and loitering mode
- Start/stop the camera analysis

request	PUT morpho/cameras/0
	{
	“id”: 0,
	“url”: “http://192.168.0.1:8080/axis-cgi/mjpg/video.cgi”,
	“login”: “tactics”,
	“password”: “t@ct!c$”,
	“resolution”: [800, 600],
	“watchlists”: [3, 4],
	“is_started”: true,
	“is_blacklistmode_enabled”: true,
	“is_loiteringmode_enabled”: false
	}

OK response	HTTP 200 OK
	{
	“id”: 0,
	“url”: “http://192.168.0.1:8080/axis-cgi/mjpg/video.cgi”,
	“login”: “tactics”,
	“password”: “t@ct!c$”,
	“resolution”: [800, 600],
	“watchlists”: [3, 4],
	“is_started”: true,
	“is_blacklistmode_enabled”: true,
	“is_loiteringmode_enabled”: false
	}

Error response	HTTP 404 Not found
	{
	“message”: “camera (id=0) not found”
	}

A.2.5.4 Delete

Camera settings can be deleted by sending a HTTP DELETE request.

request	DELETE morpho/cameras/0 (no body)
OK response	HTTP 200 OK (no body)

Error response	HTTP 404 Not found
	{
	“message”: “camera (id=0) not found”
	}

A.2.6 Alarm

Whenever an alarm is sent by the system, the following parameters are given.

Field	Type	Read/Write	Example
timestamp	String (UTC format)	Read	'*YYYMMDD***T***HHMMSS.MS*'
camera_id	int	Read	2
watchlist_id	int	Read	4
reference_id	int	Read	16
type	string	Read	"blacklist" / "loitering"
camera_picture	string	Read	<base64 encoded picture string>
reference_picture	string	Read	<base64 encoded picture string>

Table 6.4: Data composing an alarm resource

An example is given below.

request example	POST tmt/alarms/faceRecognition
	{
	"timestamp": "20140621T112400.000",
	"camera_id": 0,
	"watchlist_id": 3,
	"reference_id": 17,
	"type": "blacklist",
	"camera picture": "<base64 encoded picture string>",
	"reference_picture": "<base64 encoded picture string>"
	}

OK response example	HTTP 201 Created (no body)

A.3 TMT interfaces with face recognition implementation

The TMT system uses the Morpho API to connect to the Morpho face recognition system. The system exposes several HMI interfaces to the user in order to access to the face recognition system results as well as its management.

In the face recognition system there are the following main entities:

- Reference
- Watchlist
- Camera
- Alarm

A reference is a suspect element (mainly a face) image to be tracked by the face recognition system. It is mainly composed, in the Morpho API, by an id and an image. A watchlist is a set of references to be compared to the faces detected in some camera flows. There are several kinds of analysis, for instance recognizing someone from a blacklisted set of references. A camera entity represents a real world camera to which the face recognition system can apply its analysis linked to a watchlist (or more). Also it has extra practical data such as its url, access credentials, etc. An alarm is an asynchronous event, generated by the face recognition system when it detects a face.

Basically, the user accesses face recognition system by doing the following actions, grouped by the entities they are related to:

- Reference Actions
  - Create a reference
  - Update a reference
  - View existing references
  - View a particular reference
- Watch list actions
  - Create watchlist
  - Edit watchlist
  - View existing watchlists
  - View a particular watchlist
- Camera actions
  - Create a camera
  - Edit a camera
  - View existing cameras
  - View a particular camera
- Alarm actions
  - Setup an alarm
  - Receive an alarm

A.3.1 Reference actions

In this section the reference actions and their implementation in the TMT system are stated. Following we can see the UML use case diagram representing the actors involved and the TMT reference actions:

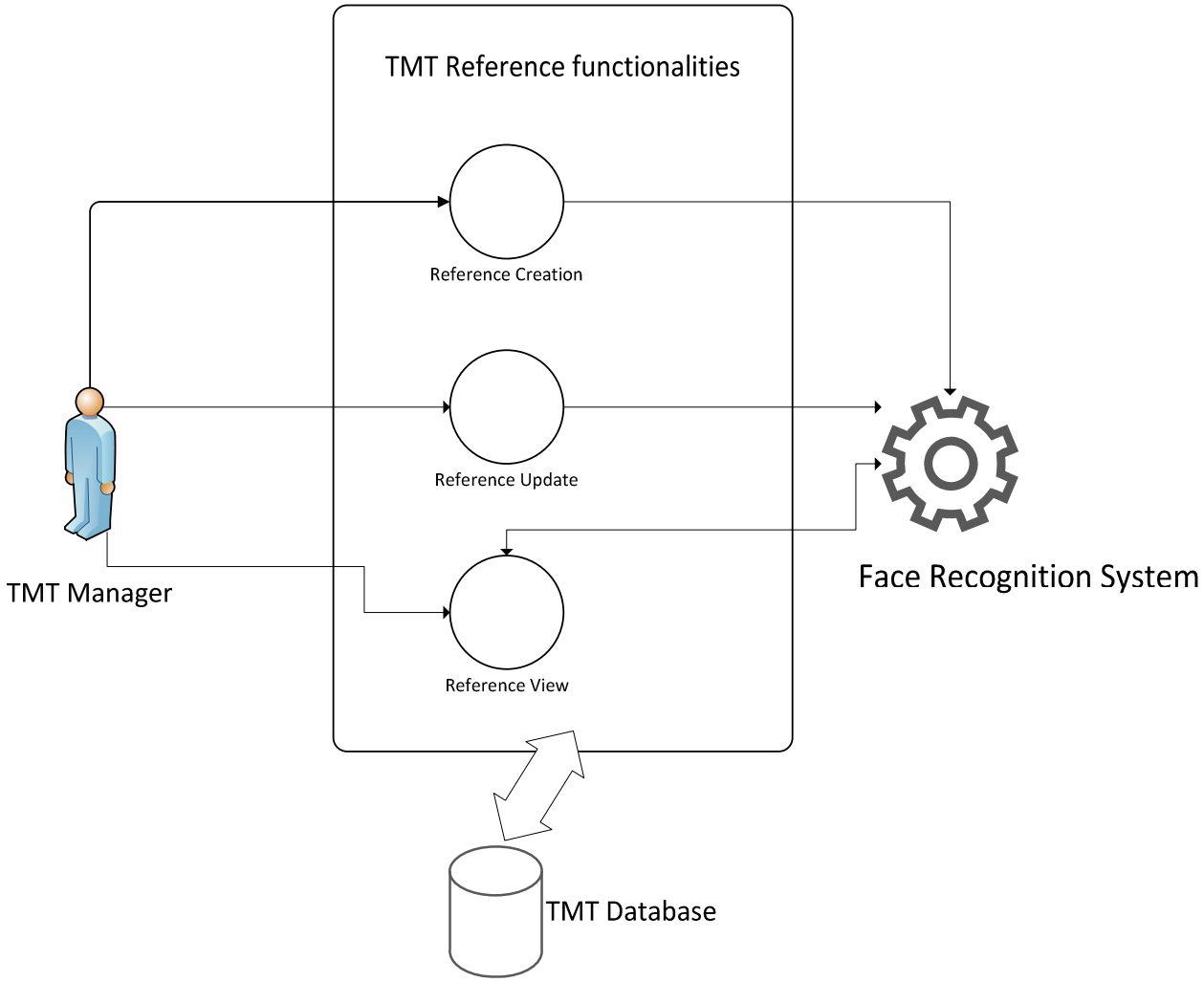

Figure 6-2: TMT reference functionalities use case diagram

In this case, the TMT manager can create, update or visualize existing references. There is a database at the TMT where persistent data resides, continually synchronized with the face recognition system, enabling limited offline working.

A.3.1.1 Reference creation

The first action that the TMT manager can do is to create a reference. To do so, TMT internally communicates with the face recognition system (in here after FRS) using REST interfaces, particularly the POST of the references interface with the corresponding JSON data. For instance, TMT does:

```
POST http://FRSserver:8080/references
```

With JSON parameters

```
{
  "picture" : "FAKE_BASE64_JPG_STRING_CREATED_BY_CLIENT_SAMPLE_JOHN"
}
```

Then, if things work properly TMT receives a HTTP 200 OK message or an error if there is any problem.

From the point of view of the TMT manager, the TMT interface provides several screens to manage this creation process. Being at the main page of the TMT system, the manager clicks on the data fusion management icon highlighted in the following figure:

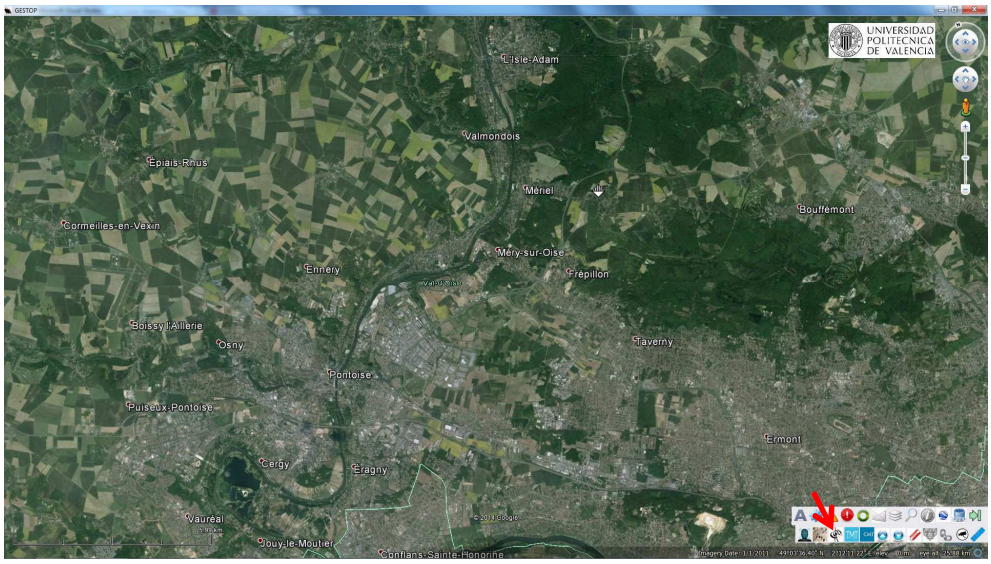

Figure 6-3: TMT data fusion management icon

Once clicked, the system will lead the TMT manager to the following screen:

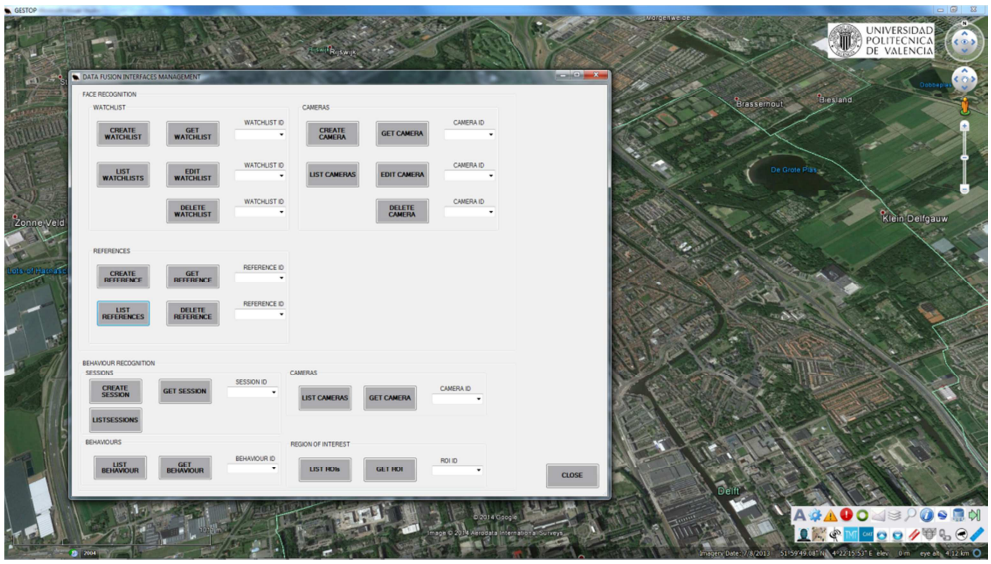

Figure 6-4: TMT data fusion management screen

In the previous screen we can see the main data fusion management functionalities, divided into two main groups: face recognition and behaviour recognition, the former from Morpho and the latter from TNO.

In order to create a new reference, the user clicks on the ‘CREATE REFERENCE’ button that leads to the following screen:

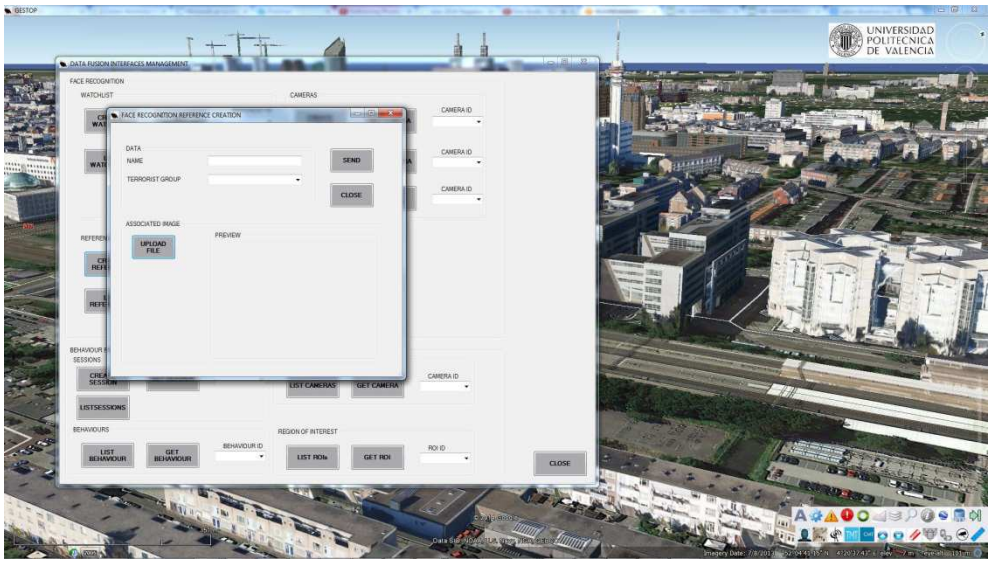

Figure 6-5: TMT Reference creation

Then, the user can insert data related to the reference such as name and terrorist group and upload the corresponding image, as can be seen in the following figure:

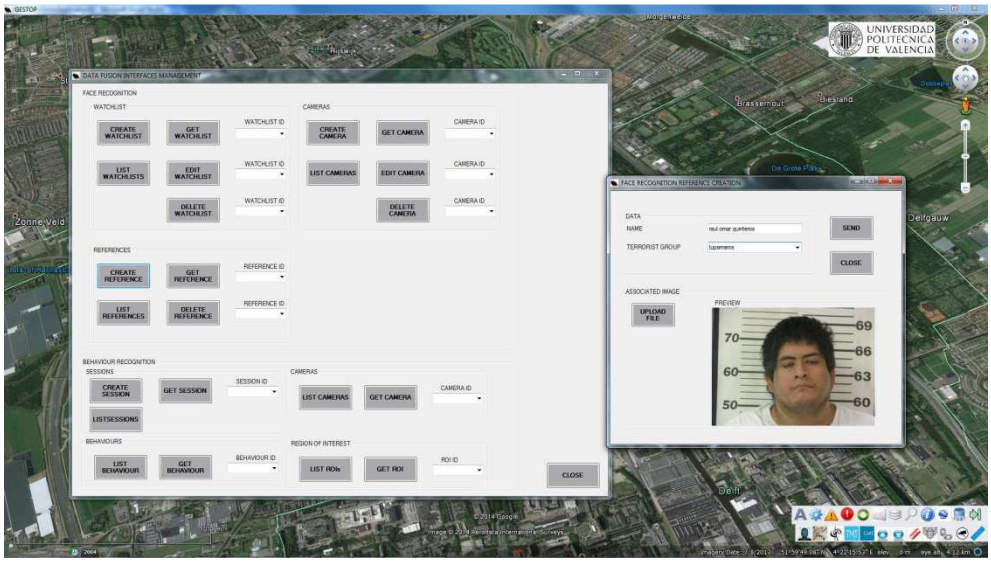

Figure 6-6: TMT Reference file upload

Once the user pushes the button ‘SEND’, the previously stated HTTP POST is sent to the FRS and the reference is created.

A.3.1.2 Reference update

References cannot be updated with the FRS API but can be deleted which is a sort of update. This is achieved by using the following HTTP request:

```
DELETE http://FRSserver:8080/references/0
```

Where the 0 corresponds to the id of the reference to be removed. If everything goes OK a HTTP 200 OK is received or else a HTTP 404 NOT FOUND.

Being at the main management screen, the user can only update ‘negatively’ the reference by deleting it. To do so, it must select which reference to delete by selecting it at the combo box and clicking the ‘DELETE REFERENCE’ button, as can be seen in the following figure:

Figure 6-7: TMT Reference update

A.3.1.3 Reference View

At any time, the TMT manager can consult existing references. In the TMT display there is an option to visualize all the references or view a particular reference.

To visualize all the references the user must click the button ‘LIST REFERENCES’. The TMT will issue a GET request to FRS like the following:

```
GET http://FRSserver:8080/references
```

And will receive a reply with JSON data as:

```
[ {
  "id" : 0,
  "picture" : "FAKE_BASE64_JPG_STRING_CREATED_BY_CLIENT_SAMPLE_JOHN"
}, {
  "id" : 1,
  "picture" : "FAKE_BASE64_JPG_STRING_CREATED_BY_CLIENT_SAMPLE_EMMA"
} ]
```

Which will be shown on TMT screen.

To visualize a particular reference we have to select the corresponding number in the combo box and then click on the button ‘GET REFERENCE’. Then, TMT will request FRS with the following HTTP GET:

```
GET http://FRSserver:8080/references/#selected_number
```

and will receive, if no error, a HTTP 200 OK message with the corresponding information in JSON format. As an example:

```
{
  "id" : 0,
  "picture" : "FAKE_BASE64_JPG_STRING_CREATED_BY_CLIENT_SAMPLE_JOHN"
}
```

In the TMT system, a screen will pop up with the following information:

Figure 6-8: TMT Reference visualization

A.3.2 Watch list actions

In this section watch list actions and their implementation in the TMT system are stated.

Following we can see the UML use case diagram representing the actors involved and the TMT watch list actions:

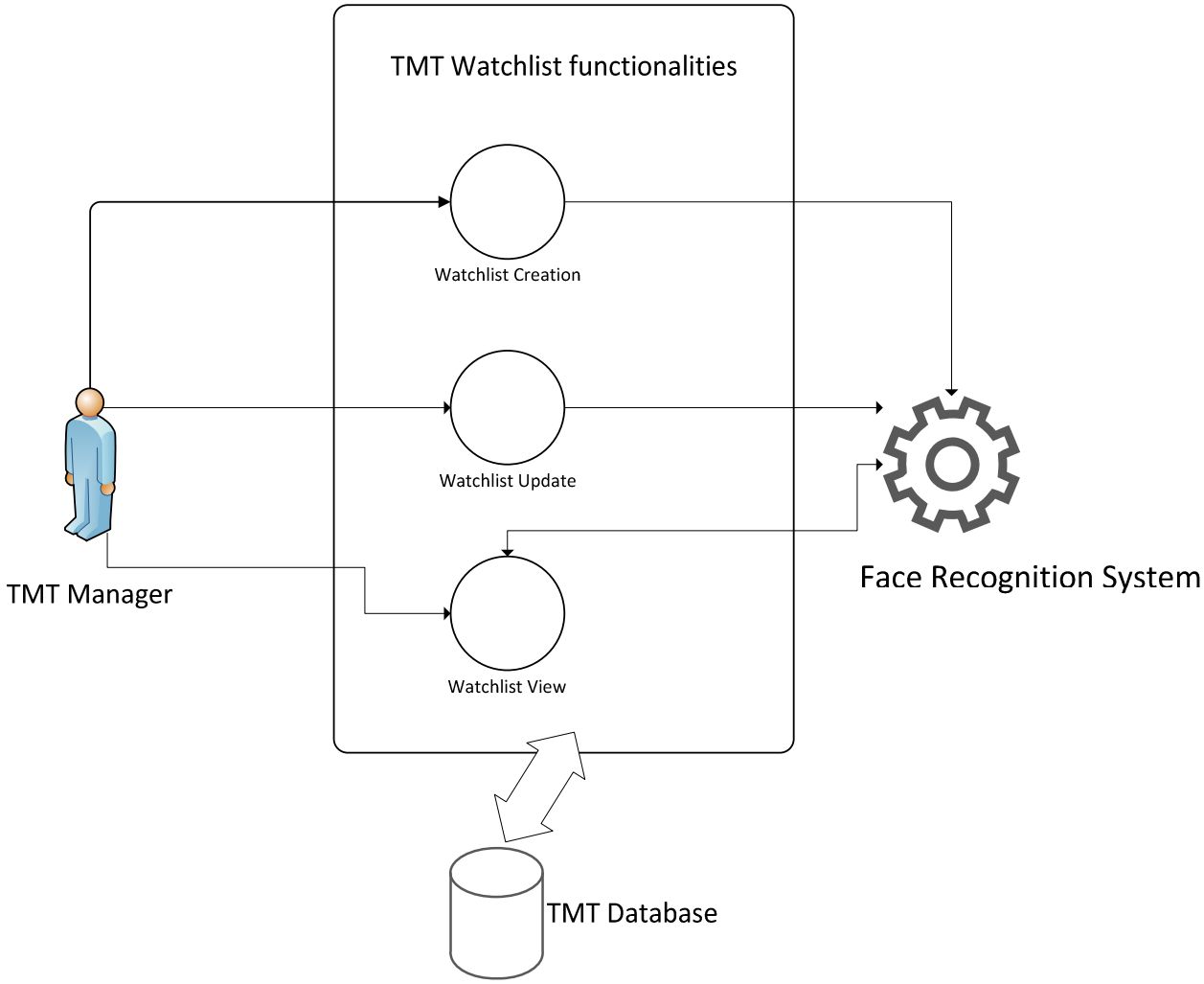

Figure 6-9: TMT watchlist functionalities use case diagram

In this case, the TMT manager can create, update or visualize existing watch lists. There is a database at TMT where persistent data resides, continually synchronized with the face recognition system, enabling limited offline working.

A.3.2.1 Watch list creation

The first action that the TMT manager can do is to create a watch list. To do so, starting at the data fusion management screen, TMT manager clicks on the button 'CREATE WATCHLIST'. Then, a screen pops-up requesting the type of the watchlist and the references it is linked to. References can be added at any time. In the following figure can be seen the screen:

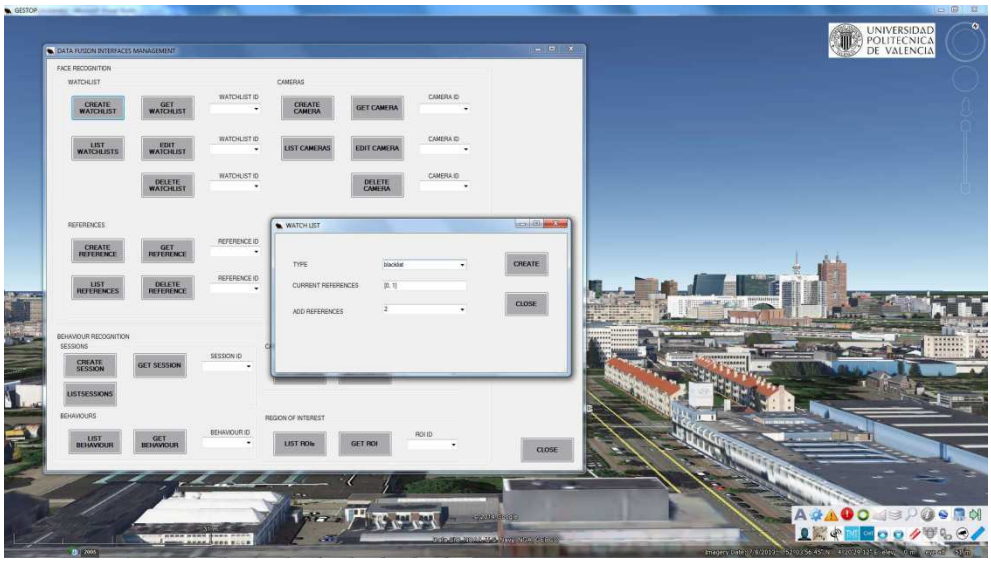

Figure 6-10: TMT watch list creation

Once the user clicks on the button ‘CREATE’, TMT communicates with FRS using REST interfaces, particularly the POST of the references interface with the corresponding JSON data. For instance, TMT does:

POST http://FRSserver:8080/watchlists

With JSON parameters

```
{
  "type" : "blacklist",
  "references" : [ 0, 1 ]
}
```

A.3.2.2 Watch list Update

At any time, the TMT manager can modify parameters of the watch list, basically the type and the references it is linked to. To do so, TMT manager must select the watchlist it is interested in in the corresponding combo box and then click on the button ‘EDIT WATCHLIST’. A screen pops-up, as can be seen in the following figure, allowing for the modification of the kind of watch list and the references.

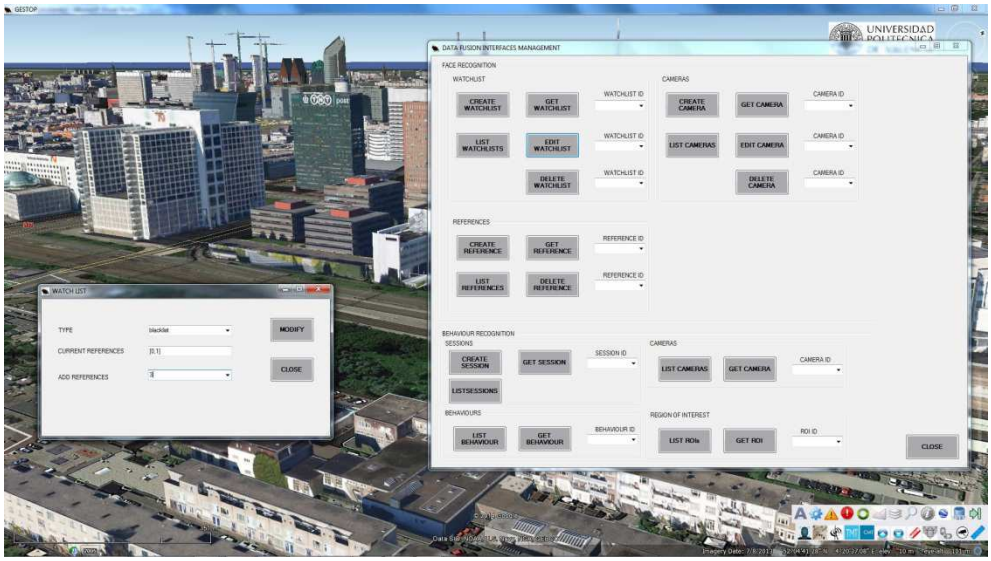

Figure 6-11: TMT watch list edit

Once the user clicks on the modify button, a HTTP PUT request is sent to the server following the API. The request is similar to the following:

```
PUT http://FRSserver:8080/watchlists/id_of_watchlist
```

With the following JSON parameters:

```
{
  "type" : "blacklist",
  "references" : [ 1 ]
}
```

A.3.2.3 Watch list View

At any time, the TMT manager can consult existing watch lists. In the TMT display there is an option to visualize all the watch lists or view a particular watch list.

To visualize all the watchlists, the user must click on the button ‘LIST WATCHLISTS’. The TMT will issue a GET request to FRS like the following:

```
GET http://FRSserver:8080/watchlists
```

And will receive a reply with JSON data as:

```
{
  "type" : "blacklist",
  "references" : [ 1 ]
}
```

Which will be shown on the TMT screen.

To visualize a particular watch list the user has to select the corresponding number in the combo box and then click on the button ‘GET WATCHLIST’. Then, TMT will request FRS with the following HTTP GET:

```
GET http://FRSserver:8080/watchlists/0
```

If there is no error, FRS will reply with the information associated with the particular watchlist and it will be shown at the TMT system.

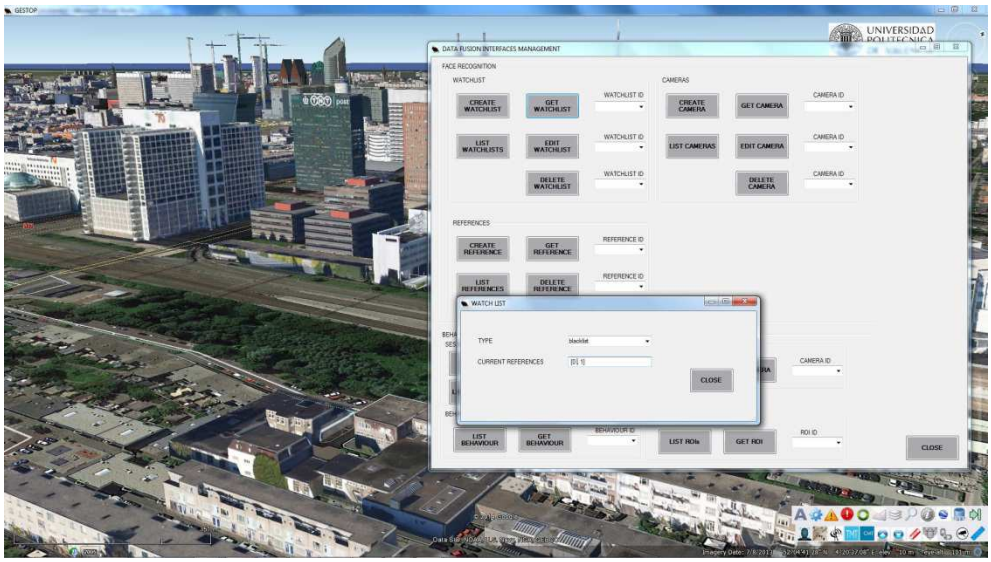

Figure 6-12: TMT Watch list view

A.3.3 Camera actions

In this section camera actions and their implementation in the TMT system are stated.

Following we can see the UML use case diagram representing the actors involved and the TMT camera actions:

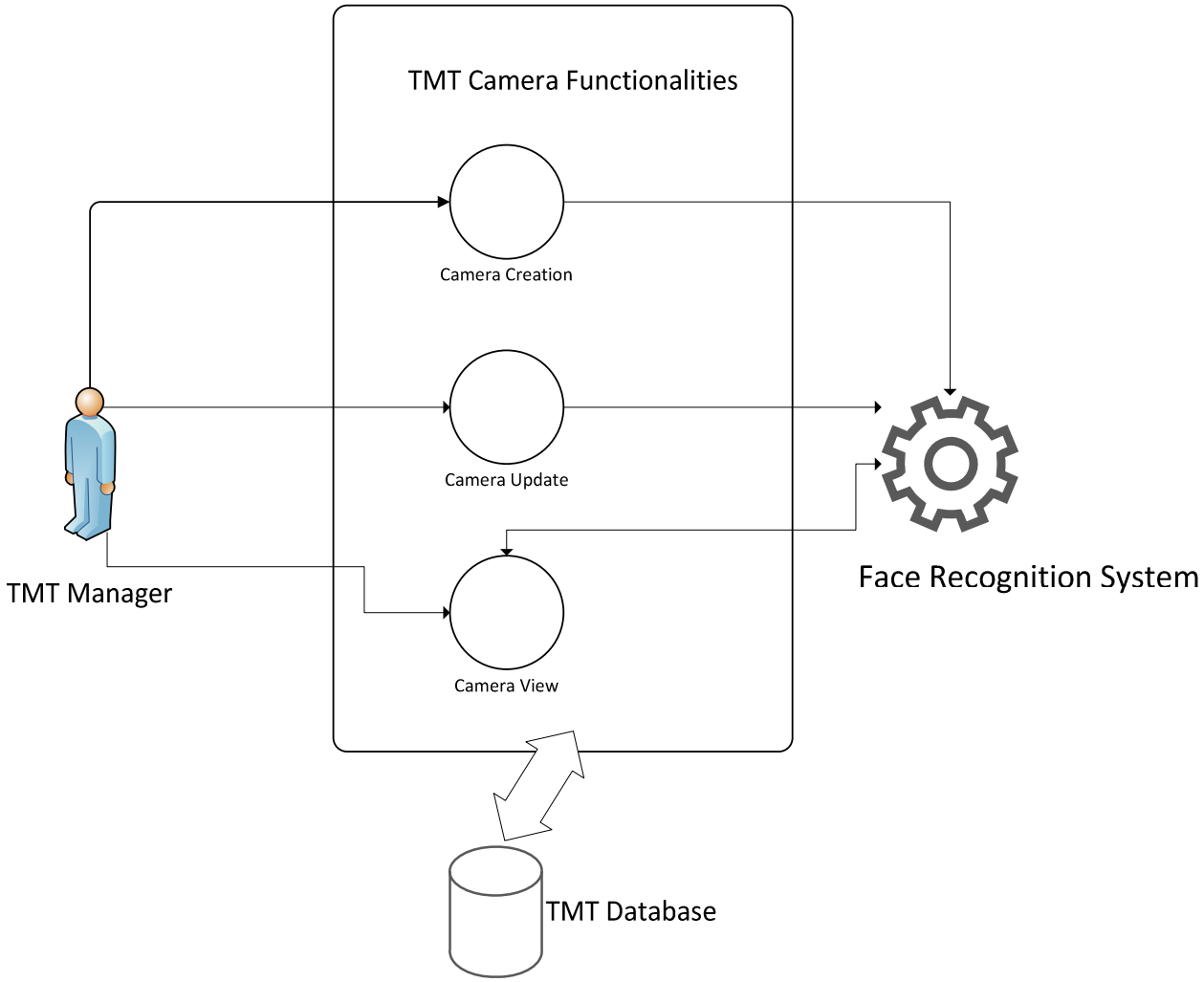

Figure 6-13: TMT Camera functionalities use case diagram

In this case, the TMT manager can create, update or visualize existing cameras. There is a database at TMT where persistent data resides, continually synchronized with the face recognition system, enabling limited offline working.

A.3.3.1 Camera creation

The first action that the TMT manager can do is to create a camera. To do so, starting at the data fusion management screen, TMT manager clicks on the button 'CREATE CAMERA'. Then, a screen pops up requesting the different parameters a camera can have. In particular, we can specify:

- Name
- URL
- Credentials (login and password)
- Resolution (X and Y)
- Watch list
- Location (longitude and latitude)

In the following figure can be seen the camera creation screen:

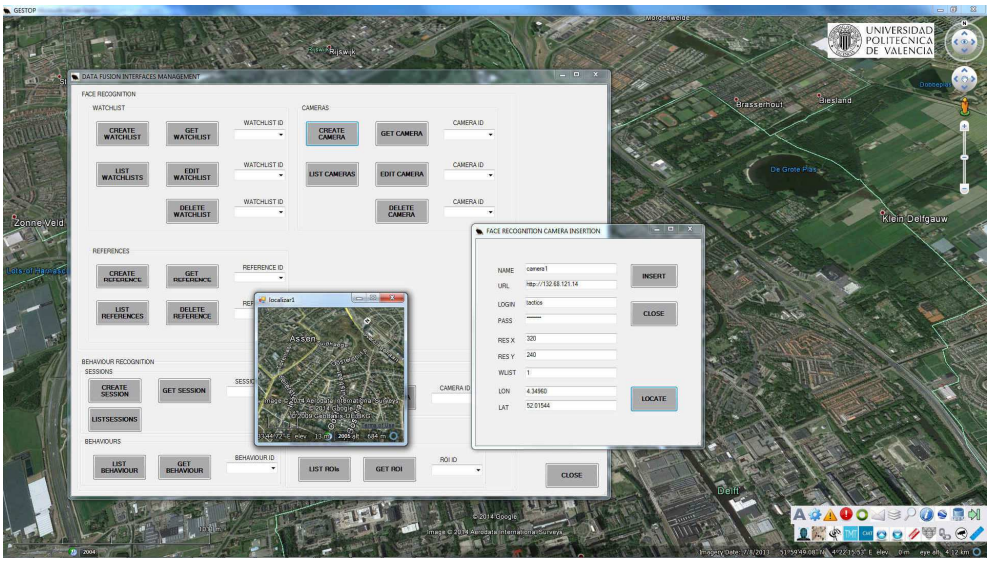

Figure 6-14: TMT Camera creation

Camera location is information not relevant to the FRS but to the TMT. That is why it is not exchanged among the two systems but needed at the TMT to georeference and show properly the item location. The user can insert it manually or by means of clicking its location in a world map. Once the user has inserted relevant data, an HTTP POST is sent to the server:

```
POST http://FRSserver:8080/cameras
```

With the relevant information in JSON format:

```
{
  "url" : "http://127.0.0.1:80/camera",
  "login" : "user",
  "password" : "pa$$w0rd",
  "resolution" : [ 800, 600 ],
  "watchlists" : [ 0 ],
  "is_started" : false,
  "is_blacklistmode_enabled" : false,
  "is_loiteringmode_enabled" : false
}
```

A.3.3.2 Camera Update

At any time, the TMT manager can modify parameters of the camera. In particular, static parameters as URL, credentials, resolution, location and watchlists, and dynamic parameters such as its state (‘is_started’ is used to control its operation and start it). To do so, TMT manager must select the camera of interest in the corresponding combo box and then click on the button ‘EDIT CAMERA’. A screen pops-up, as can be seen in the following figure, allowing for parameters modification:

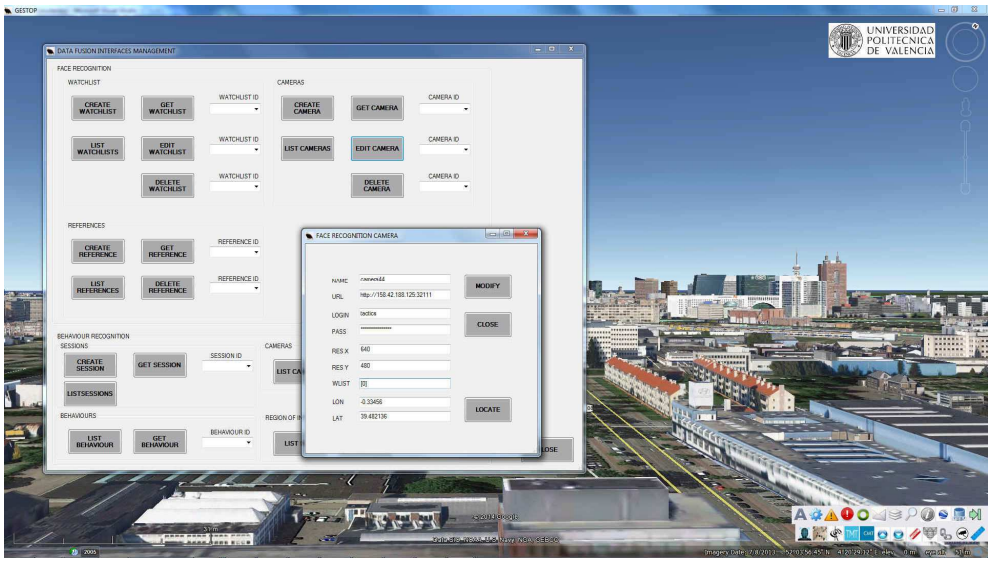

Figure 6-15: TMT Camera update

Once the form is filled up, an HTTP PUT is sent to the FRS server:

```
PUT http://FRSserver:8080/cameras/0
```

With the parameters in JSON format:

```
{
  "url" : "http://127.0.0.1:80/camera",
  "login" : "user",
  "password" : "pa$$w0rd",
  "resolution" : [ 800, 600 ],
  "watchlists" : [ 0 ],
  "is_started" : true,
  "is_blacklistmode_enabled" : true,
  "is_loiteringmode_enabled" : false
}
```

A.3.3.3 Camera View

At any time, the TMT manager can consult existing cameras. In the TMT display there is an option to visualize all the cameras or just view a particular camera.

To visualize all the cameras the user must click in the button ‘LIST CAMERAS’. The TMT will issue a GET request to FRS like the following:

```
GET http://FRSserver:8080/cameras
```

And will receive a reply with JSON data as:

```
[ {
  "id" : 0,
  "url" : "http://127.0.0.1:80/camera",
  "login" : "user",
  "password" : "pa$$w0rd",
  "resolution" : [ 800, 600 ],
  "watchlists" : [ 0 ],
```

```
  "is_started" : false,
  "is_blacklistmode_enabled" : false,
  "is_loiteringmode_enabled" : false
} ]
```


Which will be shown on TMT screen.

To visualize a particular camera the user has to select the corresponding number in the combo box and then click on the button ‘GET CAMERA’. Then, TMT will request FRS with the following HTTP GET:

```
GET http://FRSserver:8080/cameras/cam_number
```

If there is no error, FRS will reply with the information associated with the particular watchlist, a JSON like the following:


```
{
  "id" : 0,
  "url" : "http://127.0.0.1:80/camera",
  "login" : "user",
  "password" : "pa$$w0rd",
  "resolution" : [ 800, 600 ],
  "watchlists" : [ 0 ],
  "is_started" : false,
  "is_blacklistmode_enabled" : false,
  "is_loiteringmode_enabled" : false
}
```


And it will be show at the TMT system.

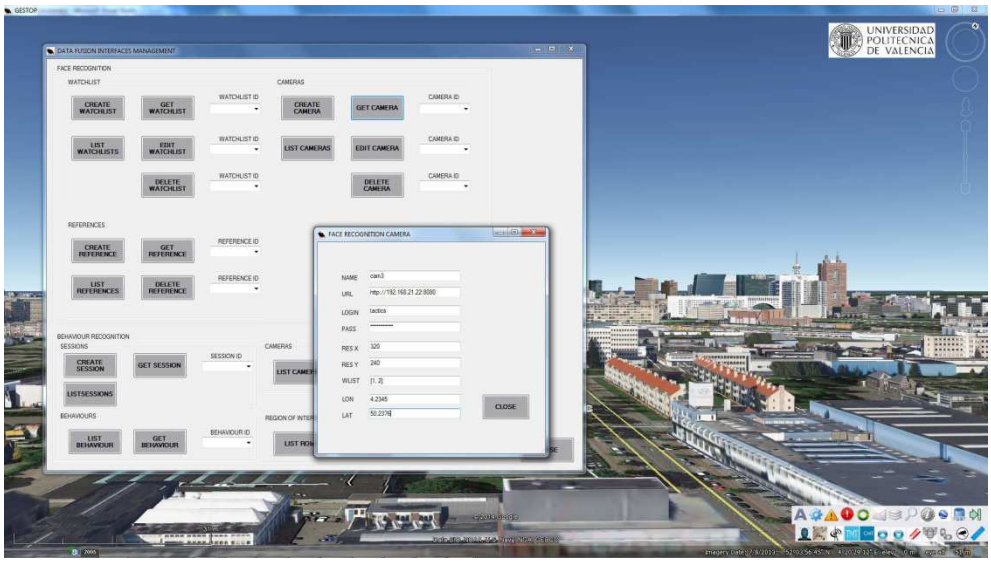

Figure 6-16: TMT camera parameters

A.3.4 Alarm actions

In this section alarm actions and their implementation in the TMT system are stated.

Following we can see the UML use case diagram representing the actors involved and the TMT alarm actions:

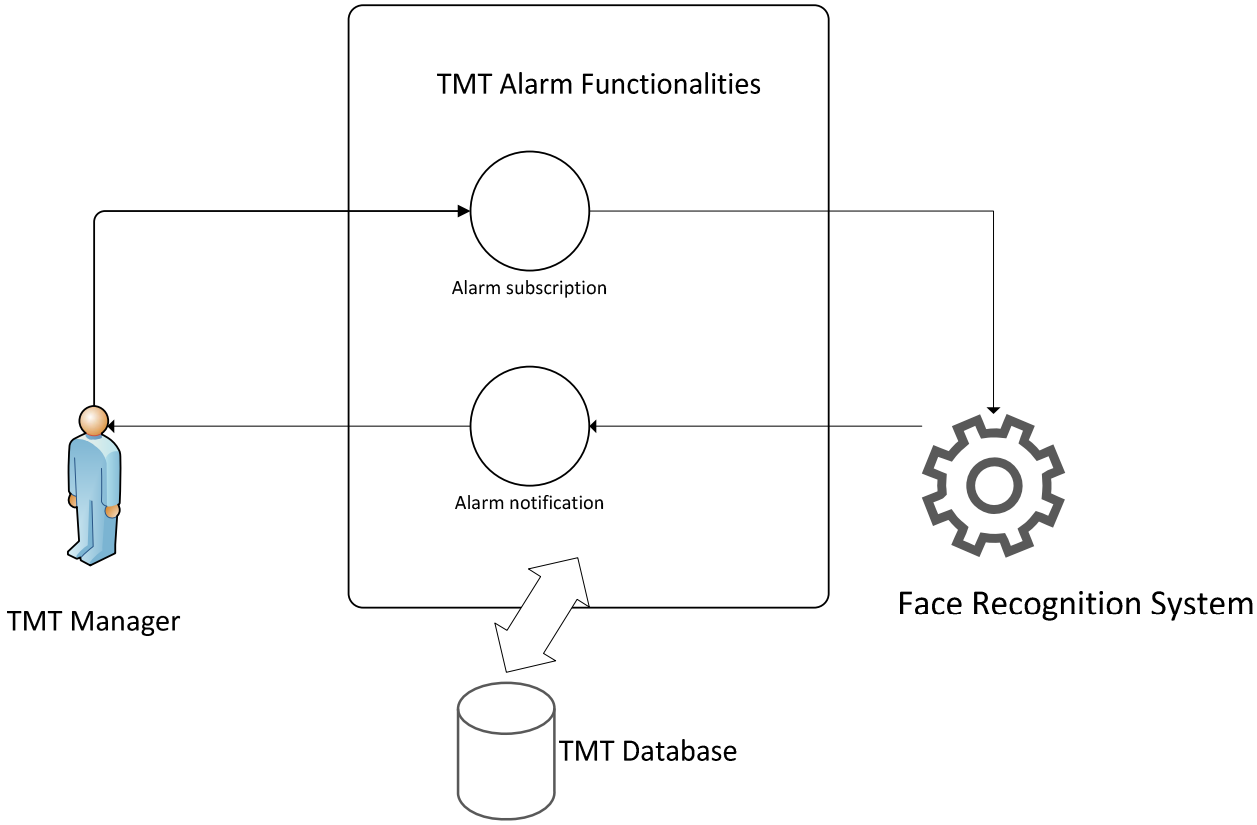

Figure 6-17: TMT Alarm functionalities use case diagram

A.3.4.1 Alarm Setup

Alarms are setup by means of modification of the camera features, setting the parameter 'is_started' to TRUE. The TMT manager can do it as explained in the previous sections or can perform this action in a more intuitive manner. To do the latter, TMT manager must be in the main screen of the application and select the face recognition geographic startup button, shown in the following picture.

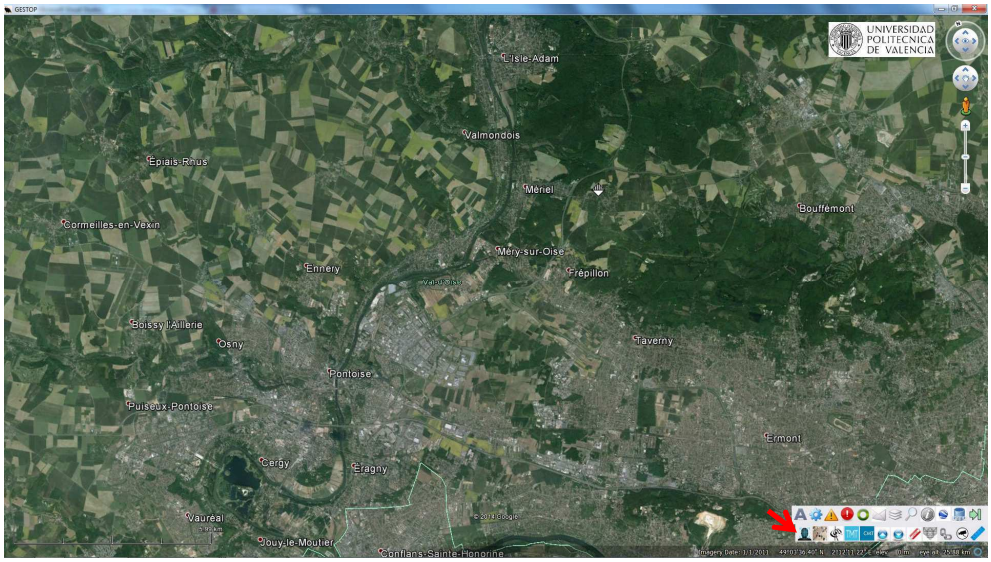

Figure 6-18: TMT face recognition access button

Once clicked, it leads the user to the following screen:

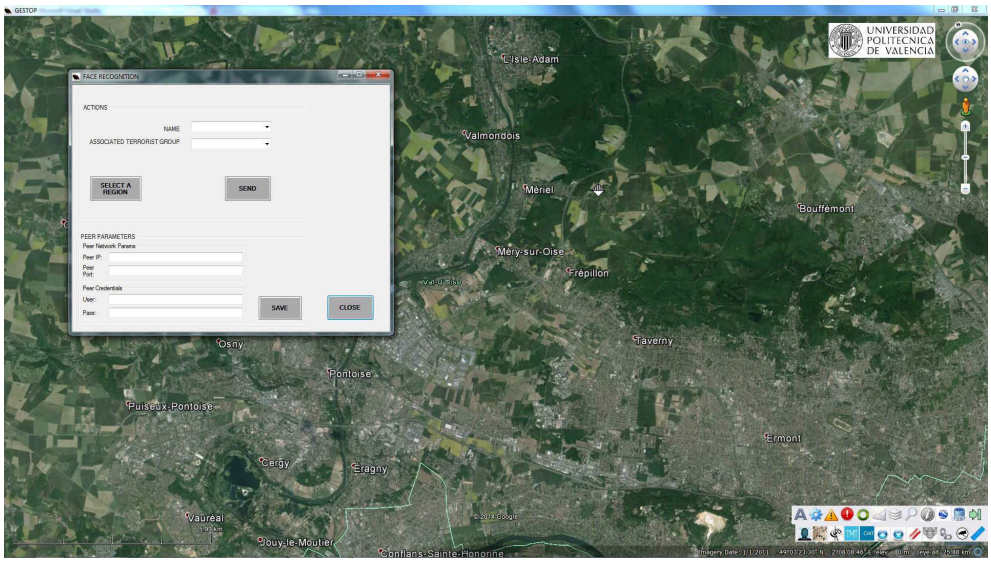

Figure 6-19: TMT face recognition screen

In this screen the user can modify the connection parameters to the FRS server (which are stored in a database) but the key functionality is to select the cameras in a geographic area, with the possibility of filtering those where there is a watchlist including a particular person (NAME) or a terrorist group. To select the area, the user must click in the button 'SELECT A REGION' and then click four points in the map to delimit the region, as can be seen in the following figure:

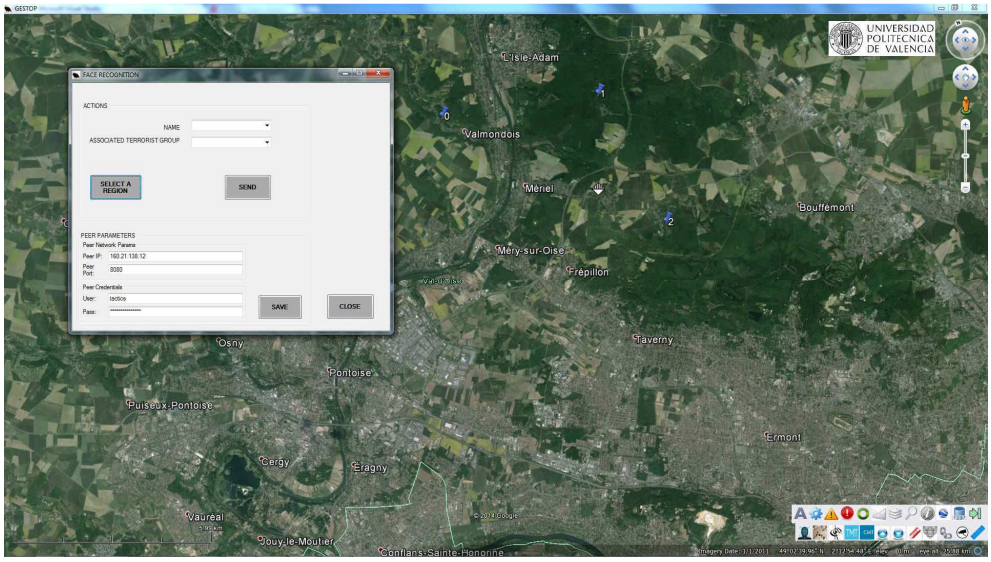

Figure 6-20: TMT FRS selection area I

Once the square is closed, the system will zoom in the area showing the corresponding cameras:

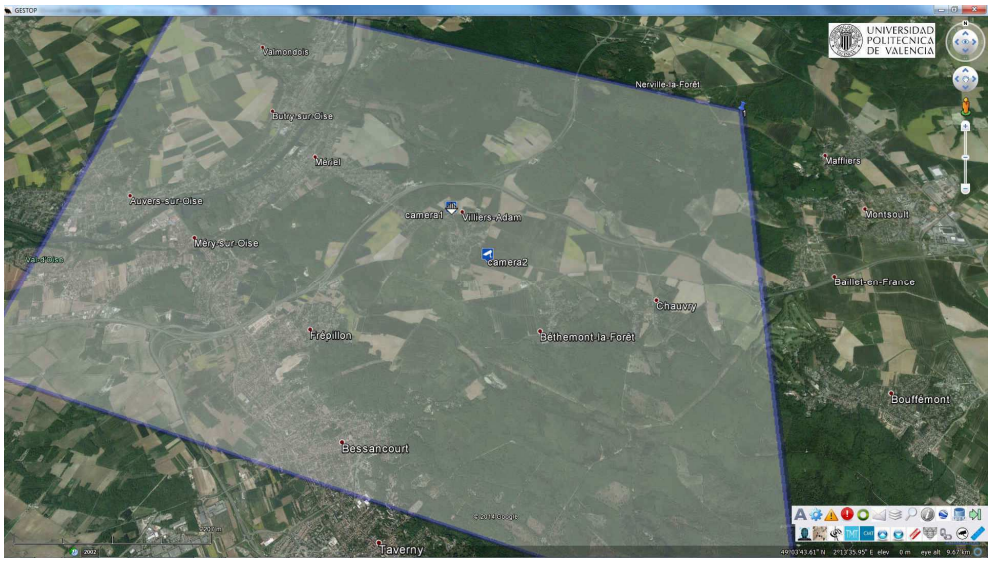

Figure 6-21: TMT FRS selection area II

At this point, the TMT manager must click on the info button (the one at the first row with a circled i) and then click at the map in the camera icon. A screen will pop-up like the following:

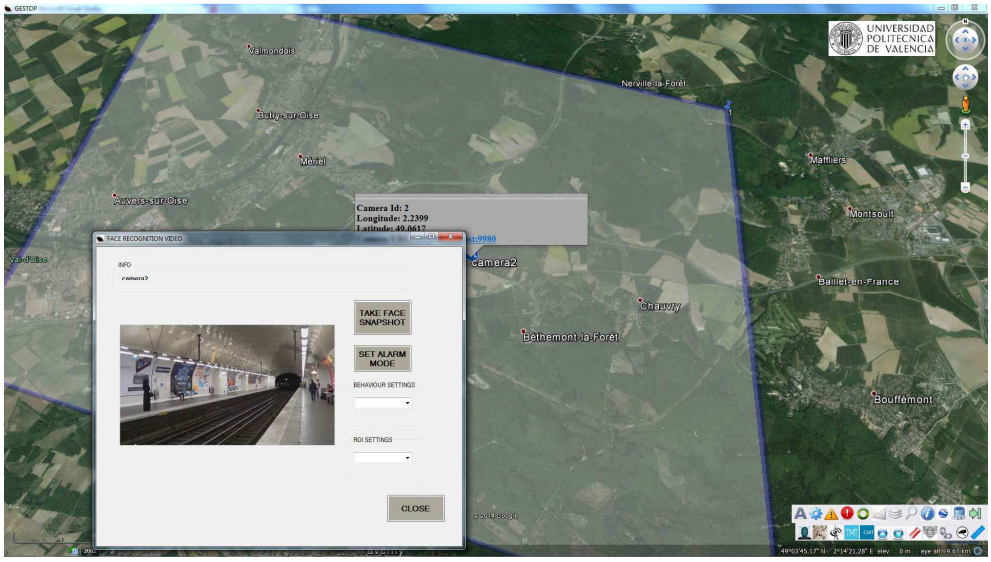

Figure 6-22: TMT FRS real time video flow and alarm subscription

At this point, we can see the real time video flow of the camera (if available) and also we can subscribe for an alarm by clicking the button ‘SET ALARM MODE’ for this particular camera. Moreover we can take a snapshot of the real time video to, for instance, create a reference for the FRS.

If we click on ‘SET ALARM MODE’, an HTTP POST will be sent to the FRS corresponding camera with the ‘is_started’ parameter to TRUE subscribing to the face recognition for this camera.

A.3.4.2 Alarm Reception

As explained in prior sections, TMT subscribes for FRS alarms and it is always passively listening for alarms sent by FRS on a previously agreed port. FRS will send a HTTP POST like the following to the TMT:

```
POST http://FRSserver:8989/alarms/faceRecognition
```

With the following JSON information:


```
{
  "timestamp" : "20140621T112400.000",
  "camera_id" : 0,
  "watchlist_id" : 0,
  "reference_id" : 1,
  "type" : "blacklist",
  "camera_picture" : "BASE64_STRING_CAM_PIC",
  "reference_picture" : "BASE64_STRING_REF_PIC"
}
```


Then, the TMT will show the alarm on screen informing the manager following the TACTICS end users recommendations (using a scrollable banner) as can be seen in the next figure:

Figure 6-23: TMT FRS alarm reception

A.4 Behaviour Analysis System Module interfaces

A.4.1 TNO API

The TNO API is a HTTP REST interface, providing access to several resources in the Behaviour Analysis System fusion module:

- Session
- Camera
- ROI
- Behaviour

Resources can be accessed using (a sub-set of) CRUD (Create, Read, Update and Delete) functionality. For passing information to the system JSON formatted messages are used. Information requested from the system is in plain text or HTML text by default, or JSON formatted on request.

The session resources describes the current session of the user, and contains links to the other resources. When creating a session, a return URI is specified, where alarms will be sent. Other resources are part of the session resource. Creating a session starts processing and provides access to the camera streams of the cameras used by the system. Deleting a session will stop all processing and remove all created resources.

A camera resource provides access to information on the cameras, to access a stream. As cameras are handled by the system, these resources cannot be created or deleted. A ‘Cameras’ container resource can provide a list of all cameras.

An ROI resources defines a region-of-interest, defined as a part of an image in a certain camera, at a certain time. It will mark persons of interest to the processing. Different ROIs may be created. ROI resources may be deleted, or are removed when a session is deleted. A ‘ROIs’ container resource can provide a list of created ROIs.

A Behaviour resource is linked to an ROI resource and defines the behaviour to be detected, for persons similar to persons in the ROI. Creating a behaviour will start the fusion process. If behaviour is detected with similar persons, an alarm will be sent to the return URI specified when creating the session that this behaviour belongs to. Fusion stops when the behaviour is deleted, i.e., no more alarms will be sent related to this behaviour and linked ROI. A ‘Behaviours’ resource can provide a list of currently existing behaviours.

All resources have a unique location specifier, i.e., a URI. It consists of the IP address of the BAS system, followed by a resource related specifier. These parts consist of a URI for the container of resources (i.e., sessions) followed by the ID of the resource. In general, resources are created by POSTing a HTTP message to the container, obtaining the id and or URI to the created resource in return. A list of resources is requested by sending a HTTP GET message to the container URI. HTTP GET messages to the resource URI provides information on the resource. A HTTP DELETE message to the resource URI removes the resource. If in the header the ‘Accept’ field is set to ‘application/json’, information on the created resource or requested description is returned as a JSON formatted message, else a short description in text, or HTML is returned.

A.4.2 API flow

An example of the process of using the Behaviour Analysis System is as follows, in case no errors are present:

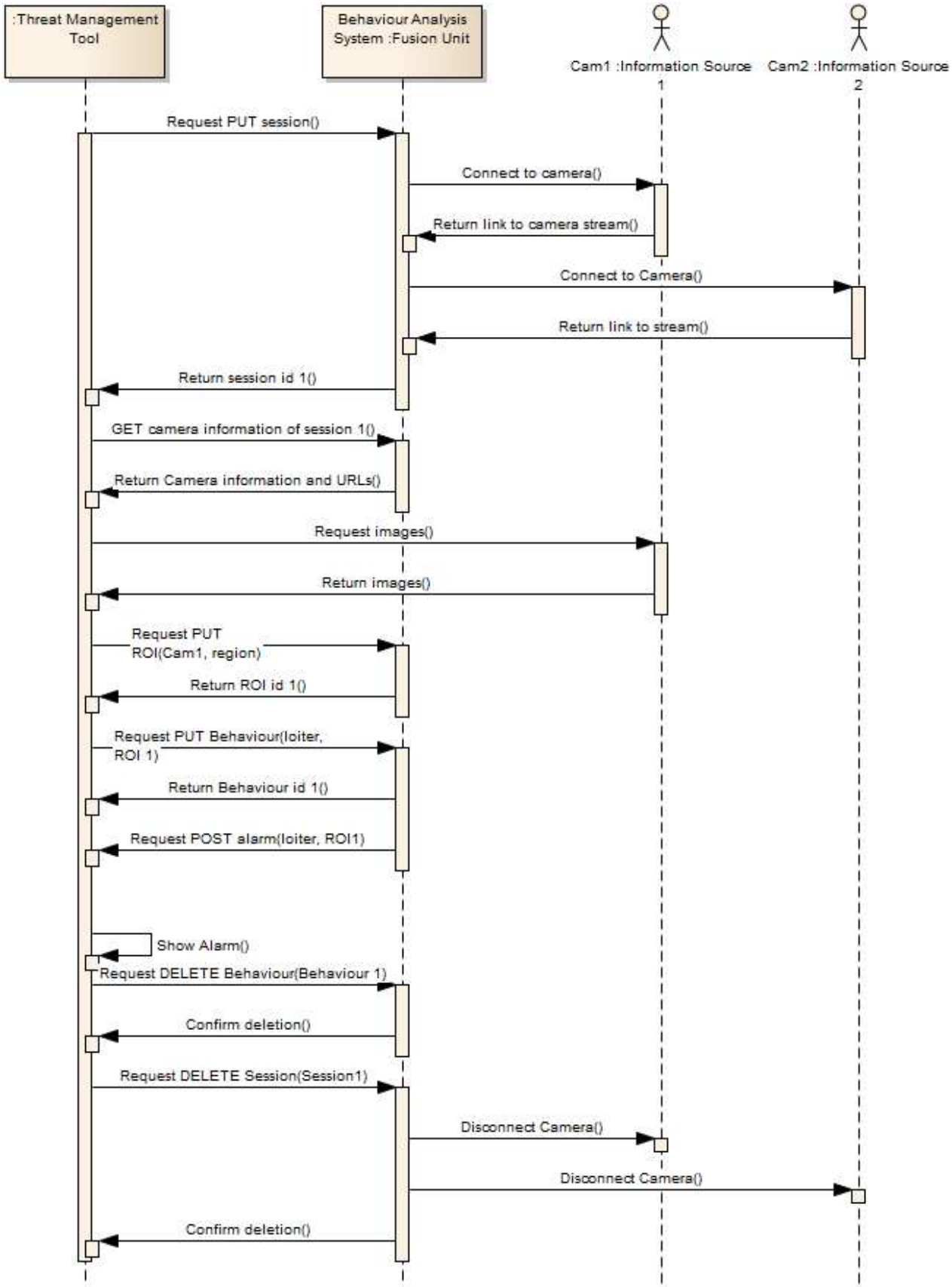

Figure 6-24: Example sequence of a session between the TMT and Behaviour Analysis System

The TMT creates a session, and in reply gets information on the camera streams. When creating the session, a URI is provided where alarms are expected to be sent. The camera streams can then be

accessed. When an event occurs that would make persons at a certain location (observed by a camera) suspect, such as an explosives sniffer giving an alarm, from the TMT an area in an image of a camera would be selected indicating these persons, and this information is provided to the BAS system by creating an ROI. The BAS system returns a reference to this created ROI. An expected suspect behaviour is then provided by the TMT, with a link to the created ROI. The BAS will then send alarms if the requested behaviour is detected for persons similar to those in the ROI, to the TMT at the return URI specified in the session. If a behaviour is deleted, related alarms will no longer be sent.

Standard HTTP responses are returned (i.e., 200 OK, 404 Not Found). If information was requested in JSON format, error replies will be in JSON format. Otherwise, (as default) an explanation is returned as text. In the specifications for the different resources in the next sessions, some will be indicated specifically. In addition, a '404 Not Found' indicates that no resource exists at the specified URI, and '500 Internal Server Error' indicates a problem in the BAS system, if such a problem occurs. When HTTP requests are made that are not allowed for the specific resource, a "405 Method Not Allowed" error is returned, specifying the allowed methods in the response header and in a text body.

A.4.3 Session

The session resource describes a processing session and contains (links) to the other resources. When creating the session, two timelabels have to be specified, defining the local time, and, in case replaying data for testing or demonstration, a start time of the stored data.

The table below lists the required data that describes a session.

Field	Type	Read / Write	Example
id	String	Read	"1"
URI	String	Read	"http://BAS.local/smartcameras/sessions/1"
localStartTime	String	Read & write	"123456789"
streamStartTime	String	Read & write	"111112222"
returnURI	String	Read & write	"http://TMT.local/alarms"

Table 6.5: Data describing a session resource.

Sessions can be created (POST), read (GET) and deleted (DELETE).

A.4.3.1 Create

A session is created by sending an HTTP POST request to the session container URI. The shown OK response is when header includes an 'Accept' field specifying 'application/json'. If not, a readable text response is returned. Error responses always are in text. Several possible error responses are shown.

request	POST http://BAS.local/smartcameras/sessions
	{
	"localStartTime":"123456789",
	"streamStartTime":"111122222",
	"returnURI":"http://TMT.local/alarms"
	}

OK response	HTTP 201 Created { "id":"1", "cameras":[

	{
	"id":"cam1",
	"streamURI":"http://BAS.local/smartcameras/cameras/5678",
	"URI":"http://BAS.local/smartcameras/sessions/1/cameras/cam1"
	},{
	"id":"cam2",
	"streamURI":"http://BAS.local/smartcameras/cameras/5679",
	"URI":"http://BAS.local/smartcameras/sessions/1/cameras/cam2"
	}
	],
	"URI":"http://BAS.local/smartcameras/sessions/1"
	}

Error response	HTTP 400 Bad Request {"message":"No data provided in session creation request'"} {"message":"Mediatype is not application/json but text/plain in session creation request"} {"message":"JSON information does not have returnURI in session creation request"}

The OK response provides the ID of the session, a URI to the session resource and information about the accessible cameras. For a description of the latter, see A.4.4.

A.4.3.2 Read

To get information on created session resources, a HTTP GET request can be send to the resources URI. This URI consists of the base URI to the session container, with the session ID appended, and is also returned when creating a session.

request	GET http://BAS.local/smartcameras/sessions/1 (no body)

OK response	HTTP 200 OK
	{
	"id":"1",
	"localStartTime":"123456789",
	"behaviours":[],
	"returnURI":" http://TMT.local/alarms ",
	"rois":[],
	"streamStartTime":"111122222",
	"cameras":[
	{
	"id":"cam1",
	"streamURI":"http://BAS.local/smartcameras/cameras/5678",
	"URI":"http://BAS.local/smartcameras/sessions/1/cameras/cam1"
	},{
	"id":"cam2",
	"streamURI":"http://BAS.local/smartcameras/cameras/5679",

	"URI":"http://BAS.local/smartcameras/sessions/1/cameras/cam2" }], "URI":"http://BAS.local/smartcameras/sessions/1" }
Error response	HTTP 404 Not found {"message":"Session 1 does not exist"}

This requests returns a full description of the session, including the values provided when creating the session, the accessible cameras, and the created ROI and Behaviour resources. The latter are included as arrays of JSON objects, each of which are the same as returned when reading these resources (see sections A.4.5 and A.4.6.

A.4.3.3 Delete

A session can be deleted by sending a HTTP DELETE request to the URI of the session resource. This will also remove all contained ROI and Behaviour resources, stop all related processing and remove access to the camera streams.

request	DELETE http://BAS.local/smartcameras/sessions/1 (no body)
OK response	HTTP 200 OK "Session 1 was removed."
Error response	HTTP 404 Not found {"message":"Cannot close non-existing session 1"}

A.4.4 Camera

The camera resources describe the actual cameras, providing a URI to the camera streams, and an ID for referencing to a camera. Camera resources are created when a session is created, and cannot be deleted by the client. Therefor the only allowed command is read.

The table below lists the data that describes a camera.

Field	Type	Read / Write	Example
id	String	Read	"cam1"
URI	String	Read	"http://BAS.local/smartcameras/sessions/1/cameras/cam1"
streamURI	String	Read	"http://BAS.local/smartcameras/cameras/5678"

Table 6.6: Data describing a camera resource.

A.4.4.1 Read

Information about the cameras can be requested by sending an HTTP GET request to the URI of the camera resource as provided by the session creation or read requests. Several possible error responses are shown.

request	GET http://BAS.local/smartcameras/sessions/1/cameras/cam1 (no body)

OK response	HTTP 200 OK
	{
	"id":"cam1",
	"streamURI":"http://BAS.local/smartcameras/cameras/5678"
	}

Error response	HTTP 404 Not found {“message”:“Session 1 does not exist”} {“message”:“Camera cam1 of session 1 does not exist.”}

An HTTP GET request can also be sent to the container URI, e.g., http://BAS.local/smartcameras/cameras , which will return a JSON array of JSON messages for each camera, including a URI field for each camera in addition to the above information.

A.4.5 Region of Interest

The ‘ROI’ resources describe one or more regions-of-interest, as defined by the client. ROIs can be created, read and deleted. Each ROI is linked to a camera, specifying its ID, and contains a timestamp referring to the specific image in which it is defined, and values specifying the rectangle (bounding box) in the image indicating where possible suspect persons are. All values are specified as strings, but some internal type checking may be done, i.e., whether a coordinate is an integer.

The table below lists the required data that describes an ROI.

Field	Type	Read / Write	Example
id	String	Read	"1"
URI	String	Read	“http://BAS.local/smartcameras/sessions/1/rois/1”
cameraId	String	Read & write	"cam1"
cameraTimeStamp	String	Read & write	“112233445”
boundingBoxX	String	Read & write	“111”
boundingBoxY	String	Read & write	“123”
boundingBoxWidth	String	Read & write	“321”
boundingBoxHeight	String	Read & write	“123”

Table 6.7: Data describing an ROI resource.

ROI resources can be created, read and deleted. The shown OK responses are when the request header includes an ‘Accept’ field specifying ‘application/json’. If not, a readable text response is returned. Error responses always are in text. Several possible error response texts may be shown in the examples.

A.4.5.1 Create

An ROI is created by sending an HTTP POST request with a JSON formatted message to the container URI for ROIs of an existing session (appending ‘/rois’ to the session URI).

request	HTTP POST http://BAS.local/smartcameras/sessions/1/rois { "cameraId":"cam1",

	"cameraTimeStamp":"112233445", "boundingBoxX":"111", "boundingBoxY":"123", "boundingBoxWidth":"321", "boundingBoxHeight":"123" }

OK response	HTTP 201 CREATED
	{
	id: "1",
	URI: “http://BAS.local/smartcameras/sessions/1/rois/1”
	}

Error response	HTTP 404 Not found {“message”:“Session 1 does not exist”} HTTP 400 Bad Request {“message”:“No data provided in ROI creation request.”} {“message”:“Mediatype is not application/json but text/plain in ROI creation request”} {“message”:“JSON information does not have boundingBoxY in ROI creation request”} {“message”:“Problem getting JSON information from ROI creation request: bounding box information not integer.”}

The OK response indicates the ID which behaviour responses have to refer to.

A.4.5.2 Read

Information about a ROI is obtained by sending an HTTP GET request to the ROI URI as returned in the creation response. ROI URIs consist of the session URI with ‘/rois/’ appended, followed by the ROI id, for example “http://BAS.local/smartcameras/sessions/1/rois/1”.

request	GET http://BAS.local/smartcameras/sessions/1/rois/1 (no body)

OK response	HTTP 200 OK
	{
	id: "1",
	cameraId: "cam1",
	cameraTimeStamp: "112233445",
	boundingBoxX: "111",
	boundingBoxY: "123",
	boundingBoxWidth: "321",
	boundingBoxHeight: "123"
	}

Error response	HTTP 404 Not found {“message”:“Session 1 does not exist”}

	{“message”:“ROI 1 of session 1 does not exist.”}

An HTTP GET request can also be sent to the container URI, e.g., http://BAS.local/smartcameras/sessions/1/rois , which will return a JSON array of JSON messages for each ROI, including a URI field for each ROI in addition to the above information.

A.4.5.3 Delete

An ROI can be deleted by sending an HTTP DELETE request to the URI of the resource. Removing an ROI has no further effects and does not currently remove Behaviour resources that might link to the ROI. It is therefore recommended not to use this functionality, but delete linked behaviours instead, or delete a session to remove all ROIs and Behaviours.

request	DELETE http://BAS.local/smartcameras/sessions/1/rois/1 (no body)
OK response	HTTP 200 OK “ROI 1 was removed from session 1”
Error response	HTTP 404 Not found {“message”:“ROI 1 of session 1 does not exist.”}

A.4.6 Behaviour

The Behaviour resource describes requests for alarms linking persons showing a specified behaviour and being similar to persons present in the image specified by an indicated ROI. As long as a behaviour resource exists, its corresponding processing is running, and alarms may be sent to the requested returnURI (specified for this session in the session creation request). As such, creating a behaviour resource will start processing and deleting a behaviour resource will stop processing.

The table below lists the required data that describes a Behaviour.

Field	Type	Read / Write	Example
id	String	Read	"1"
URI	String	Read	“http://BAS.local/smartcameras/sessions/1/behaviours/1”
roiId	String	Read & write	"1"
behaviour	String	Read & write	“loiter”

Table 6.8: Data describing a Behaviour resource.

Behaviour resources can be created, read and deleted. The shown OK responses are when the request header includes an ‘Accept’ field specifying ‘application/json’. If not, a readable text response is returned. Error responses always are in text. Several possible error response texts may be shown in the examples.

A.4.6.1 Create

A Behaviour is created by sending an HTTP POST request with a JSON formatted message to the container URI for Behaviours of an existing session (appending ‘/behaviours’ to the session URI). The provided ROI ID should be for an existing ROI. An error message is given if this is not the case.

request	HTTP POST http://BAS.local/smartcameras/sessions/1/behaviours { "roiId":"1", "behaviour":"loiter"

	}

OK response	HTTP 201 CREATED
	{
	id: "1",
	URI: “http://BAS.local/sessions/1/behaviours/1”
	}

Error response	HTTP 404 Not found {“message”:“Session 1 does not exist”} HTTP 400 Bad Request {“message”:“No data provided in behaviour creation request.”} {“message”: “Mediatype is not application/json but text/plain in behaviour creation request”} {“message”: “Problem getting JSON information from behaviour creation request”} {“message”: “JSON information does not have behaviour in behaviour creation request”} {“message”: “roiId 1 is not an existing ROI in session 1, in behaviour creation request””}

A.4.6.2 Read

Information about a Behaviour is obtained by sending an HTTP GET request to the Behaviour URI as returned in the creation response. Behaviour URIs consist of the session URI with ‘/behaviours/’ appended, followed by the Behaviour id, for example “http://BAS.local/smartcameras/sessions/1/behaviours/1”.

request	GET http://BAS.local/smartcameras/sessions/1/behaviours/1 (no body)

OK response	HTTP 200 OK
	{
	id: "1",
	behaviour: "loiter",
	roiId: "1"
	}

Error response	HTTP 404 Not found {“message”: “Session 1 does not exist”} {“message”: “Behaviour 1 of session 1 does not exist.”}

An HTTP GET request can also be sent to the container URI, e.g., http://BAS.local/smartcameras/sessions/1/behaviours , which will return a JSON array of JSON messages for each Behaviour, including a URI field for each behaviour in addition to the above information.

A.4.6.3 Delete

A Behaviour can be deleted by sending an HTTP DELETE request to the URI of the resource. Removing a Behaviour will stop processing for that behaviour and no more alarms will be sent related to that behaviour.

request	DELETE http://BAS.local/smartcameras/sessions/1/behaviours/1 (no body)

OK response	HTTP 200 OK “Behaviour 1 was removed from session 1”
Error response	HTTP 404 Not found {“message”: “Behaviour 1 of session 1 does not exist.”}

A.4.7 Alarm

For each existing behaviour resource, processing may cause detection of a requested behaviour for a person similar to a person in the linked region-of-interest. In this event, an alarm message is sent to the URI specified by the client when creating the session that includes the behaviour and ROI information. This information is sent as an HTTP POST request, containing a JSON formatted message with the following fields:

Field	Type	Read/Write	Example
cameraId	String	Read	"cam1"
cameraTimeStamp	String	Read	“112255555”
boundingBoxX	String	Read	“11”
boundingBoxY	String	Read	“12”
boundingBoxWidth	String	Read	“32”
boundingBoxHeight	String	Read	“12”
behaviour	String	Read	“loiter”
roiId	String	Read	“1”
behaviourId	String	Read	“1”

Table 6.9: Data composing an alarm resource

The cameraId and cameraTimeStamp together indicate a specific frame in a camera stream. The bounding box coordinates indicate where in this image the person is located. The behaviourId indicates for which behaviour resource this alarm is given. The roiId and behaviour field provide a direct link to the information in that request.

An example of an alarm is provided below.

request example	POST tmt/alarms/BAS
	{
	"cameraId":"cam1",
	"timeStamp":"112255555",
	"boundingBoxX":"11",
	"boundingBoxY":"12",
	"boundingBoxWidth":"32",
	"boundingBoxHeight":"12",
	"behaviour":"loiter",
	"roiId":"1",
	"behaviourId":"1"
	}

OK response example	HTTP 201 Created (no body)

A.5 TMT interfaces with Behaviour recognition system implementation

The TMT system uses the TNO API to connect to the TNO behaviour recognition system. The system exposes several HMI interfaces to the user in order to access the behaviour recognition system results as well as its management.

In the behaviour recognition system there are the following main entities:

- Session
- Region of interest (ROI)
- Behaviour
- Camera
- Alarm

A session is a container box that includes all the rest of the elements (ROIs, behaviours and cameras). The entity session has an alarm associated to it. A ROI is an area definition that the TMT manager uses to specify its interest in tracking and behaviour recognition. It includes also the time dimension.

A behaviour entity specifies the kind of behaviour we are interested to detect for a given ROI (or equivalently suspicious). It can be one of the following: loitering, scouting and meeting. A camera is an entity that refers to a real camera with a video flow to which applies the behaviour recognition analysis. Finally, an alarm is an entity, not specified explicitly in the API, that the user receives when the Behaviour Analysis System (herein after BAS) detects the occurrence of a behaviour.

Basically the user accesses the behaviour analysis system by doing the following actions, grouped by the entities they are related to:

- Session Actions
  - View existing Sessions
  - View a particular Session
  - Create a session
- ROI actions
  - View existing ROI
  - View a particular ROI
  - Create ROI and associate to a camera
- Behaviour actions
  - Associate a behaviour to a ROI
  - View existing behaviours
  - View a particular behaviour
- Camera actions
  - View existing cameras
  - View a particular camera
  - View camera video flow
- Alarm actions
  - Alarm setup
  - Alarm reception

To simplify the user interface, almost all the entity creation processes are performed while watching a video flow by just clicking on the image and setting some parameters while the data fusion screen remains mostly for configuration visualization purposes.

A.5.1 Session actions

In this section the session actions and their implementation in the TMT system are stated. Following we can see the UML use case diagram representing the actors involved and the TMT session actions:

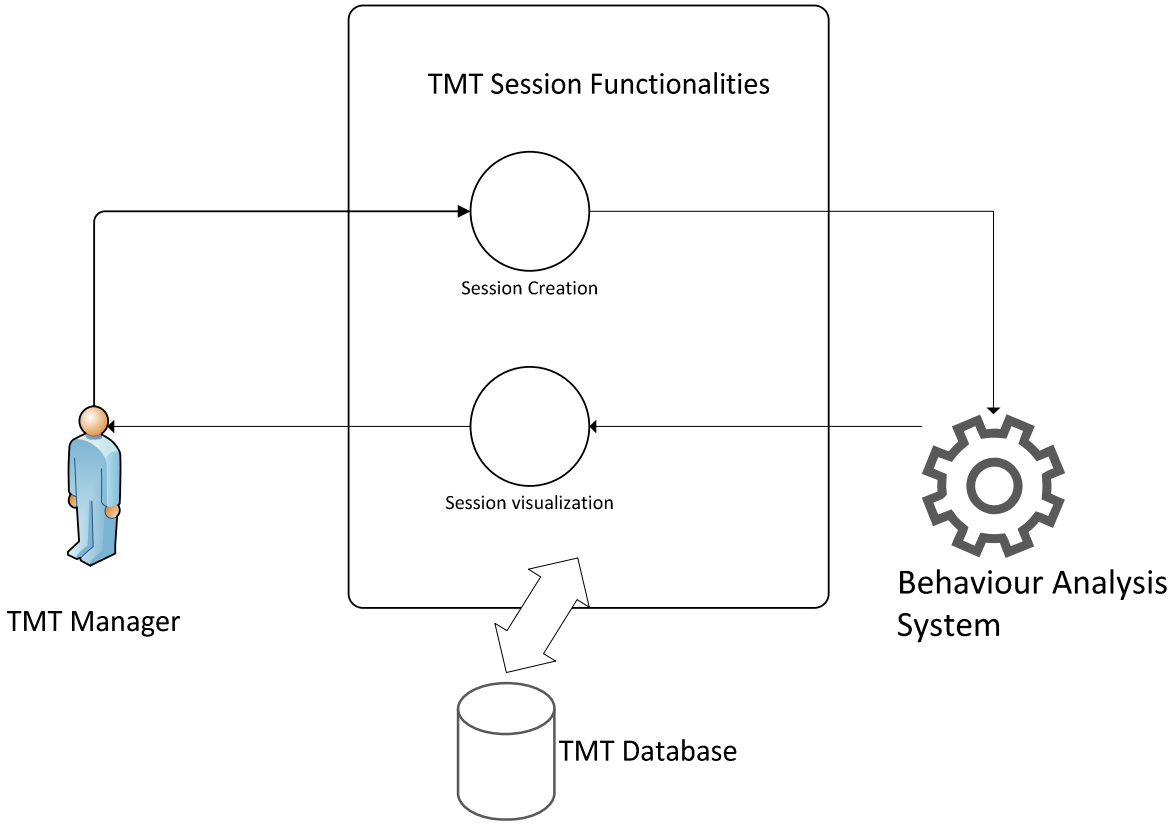

Figure 6-25: TMT Session functionalities use case diagram

A.5.1.1 Session Creation

The creation of sessions is integrated in the video flow visualization screen. Each time the TMT wants to register for a new alarm it must create a session, view the camera video flows and, and from them select rois and behaviours. This will be described in the alarms setup section.

A.5.1.2 Session Visualization

To see the features of a given session or a list of sessions the user must go to the data fusion management screen of the TMT. If the button 'LIST SESSIONS' is clicked, then TMT sends a HTTP GET request to the BAS system:

```
GET http://BAS.local/tactics/smartcameras/sessions
```

The BAS system replies with a JSON data message that includes relevant information of the sessions and that is shown in the following TMT screen:

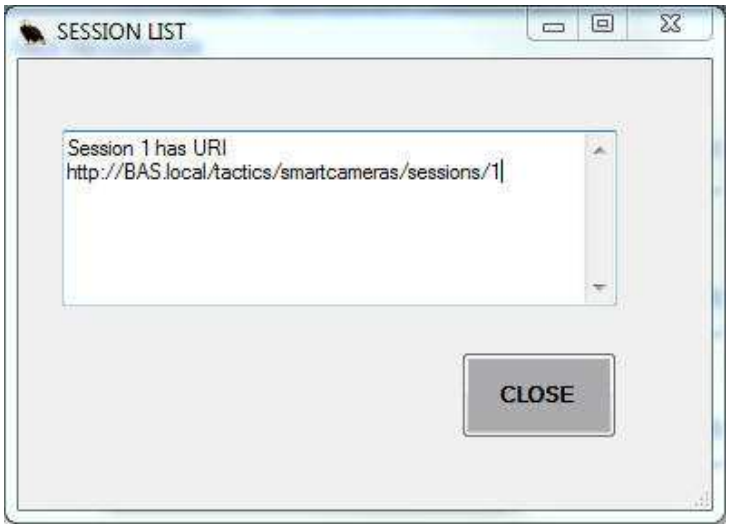

Figure 6-26: TMT Session list

If the user needs to check the features of a particular session, then he must select the session id in the corresponding combo box and click on the 'GET SESSION' button. A HTTP GET request is sent to the BAS system with the session id like the following:

```
GET http://BAS.local/tactics/smartcameras/sessions/1
```

The action at TMT can be seen in the following figure:

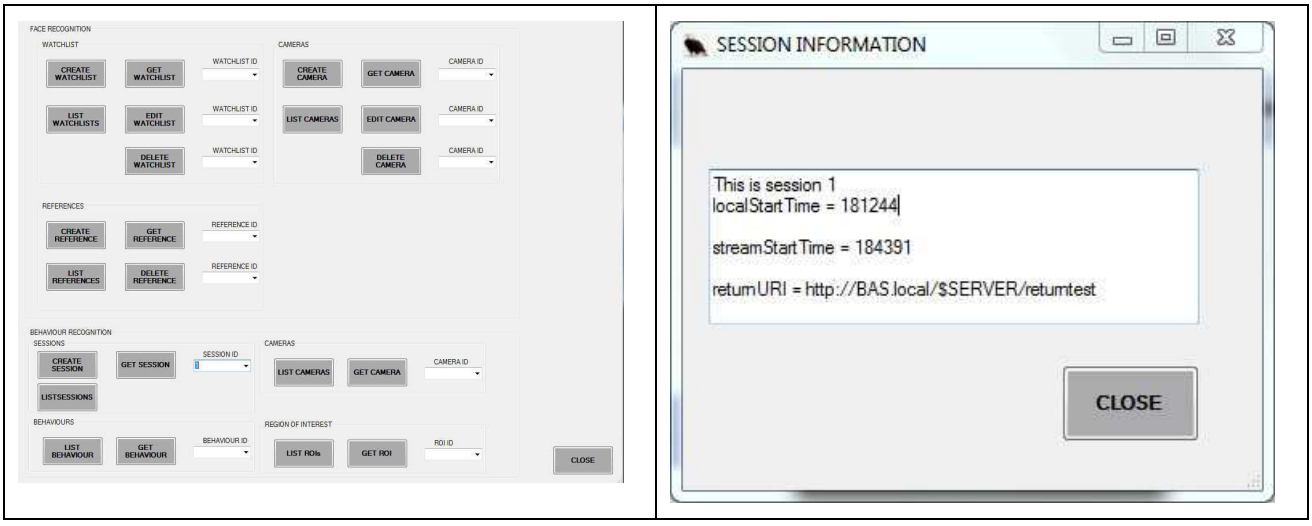

Figure 6-27: TMT Session information

A.5.2 ROI actions

In this section the ROI actions and their implementation in the TMT system are stated. Following we can see the UML use case diagram representing the actors involved and the TMT ROI actions:

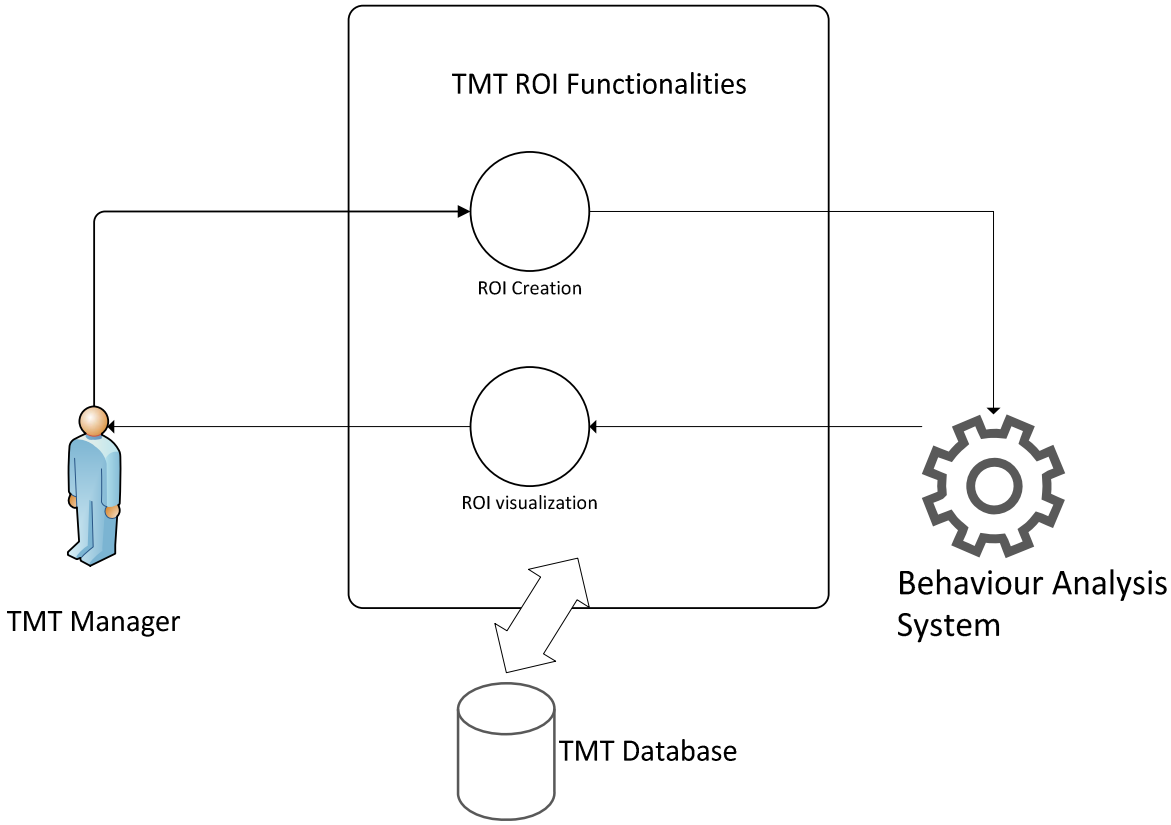

Figure 6-28: TMT ROI functionalities use case diagram

A.5.2.1 ROI Creation

The creation of ROIs is integrated in the video flow visualization screen. It will be described in the alarms setup section.

A.5.2.2 ROI Visualization

To see the features of a given list of ROIs, the user must go to the data fusion management screen of the TMT. If the button 'LIST ROIs' is clicked, then TMT sends a HTTP GET request to the BAS system:

```
GET http://BAS.local/tactics/smartcameras/sessions/1/rois
```

The BAS system replies with a JSON data message that includes relevant information of the ROIs and that is shown in the following TMT screen:

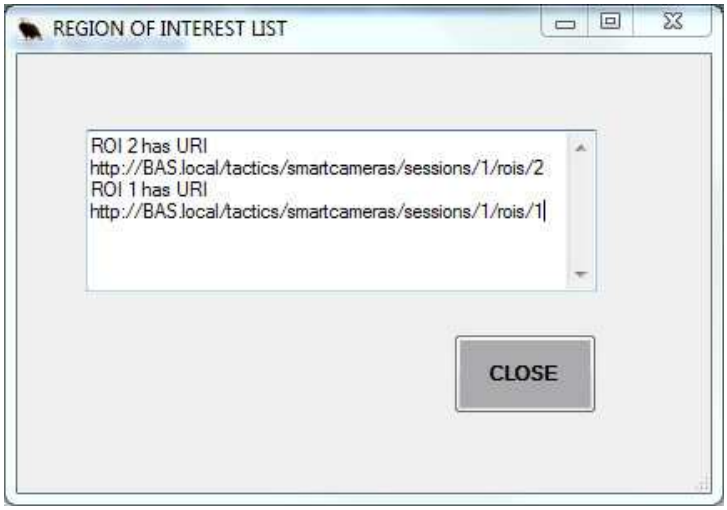

Figure 6-29: TMT List of ROI

If the user needs to check the features of a particular ROI, then he must select the ROI id in the corresponding combo box and click on the ‘GET ROI’ button. A HTTP GET request is sent to the BAS system with the ROI id like the following:

```
GET http://BAS.local/tactics/smartcameras/sessions/1/rois/3
```

The action at TMT can be seen in the following figure:

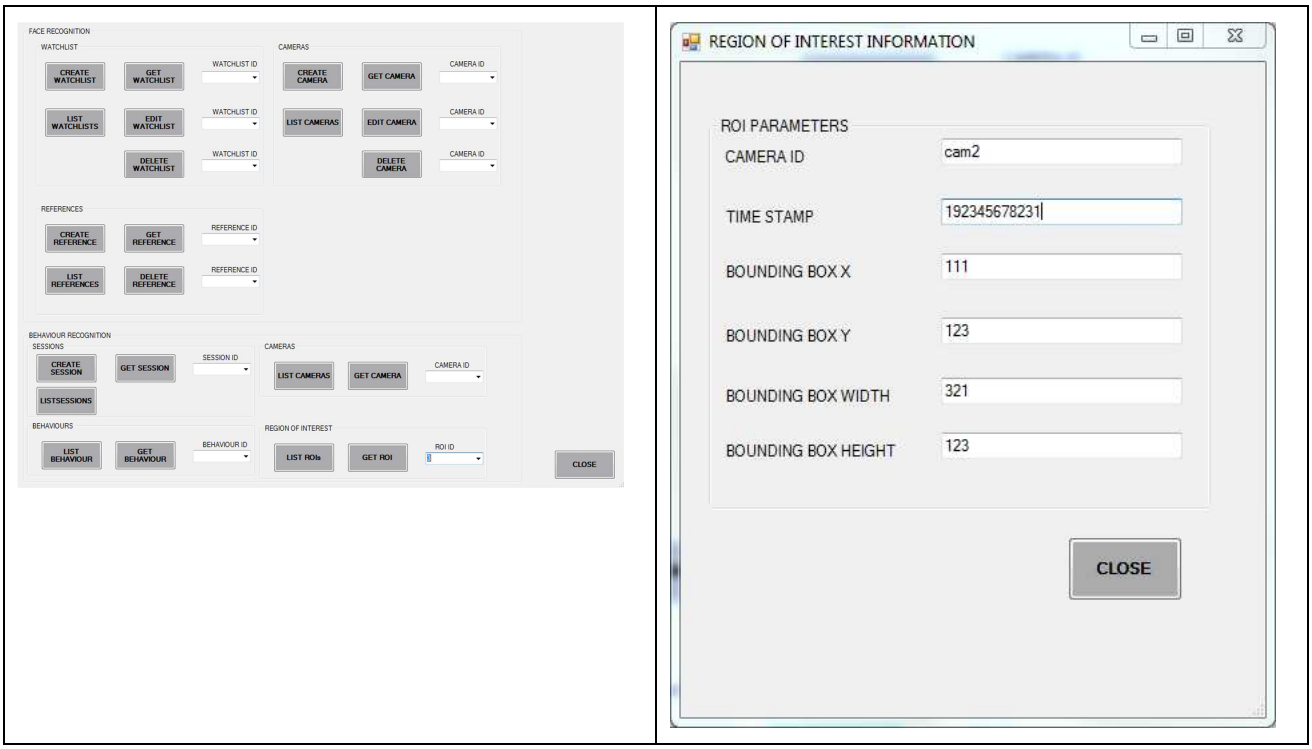

Figure 6-30: TMT ROI information

A.5.3 Behaviour actions

In this section behaviour actions and their implementation in the TMT system are stated.

Following we can see the UML use case diagram representing the actors involved and the TMT behaviour actions:

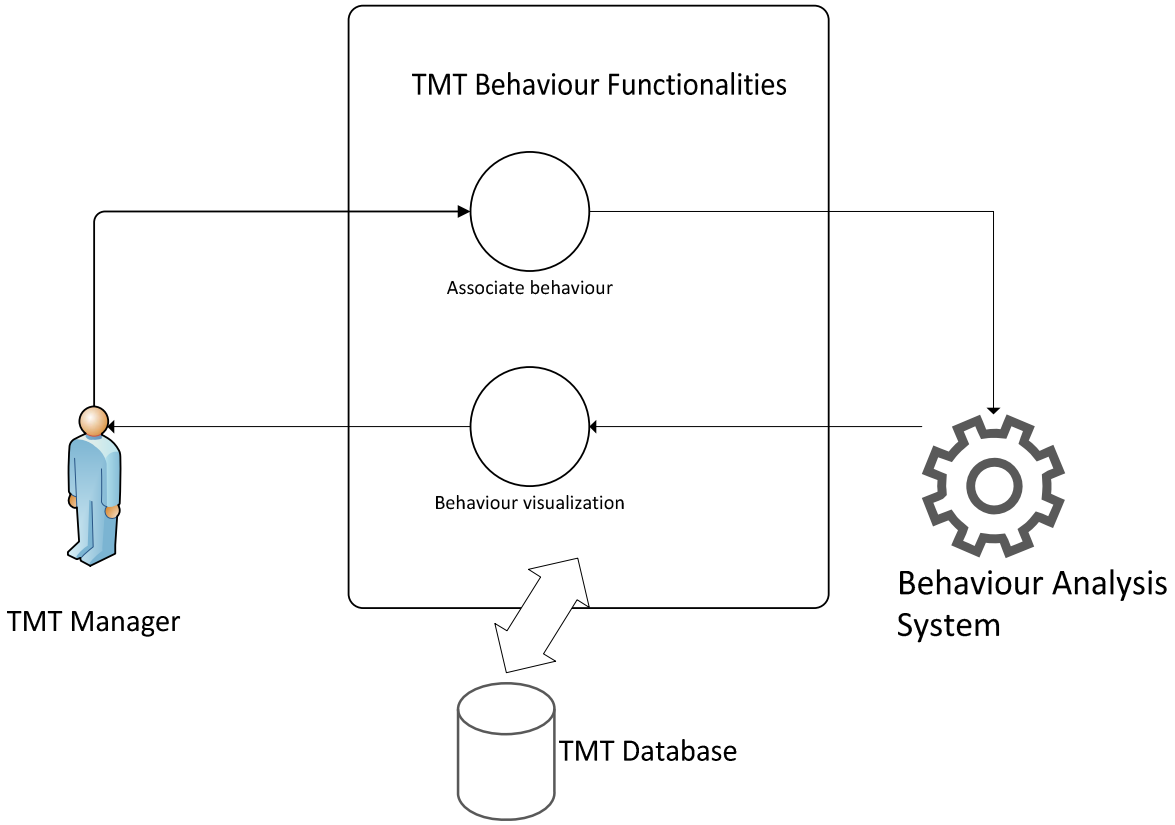

Figure 6-31: TMT behaviour functionalities use case diagram

A.5.3.1 Behaviour association

The association of behaviours to ROIs is integrated in the video flow visualization screen. It will be described in the alarms setup section.

A.5.3.2 Behaviour visualization

To see the features of a given list of behaviours the user must go to the data fusion management screen of TMT. If the button 'LIST BEHAVIOUR' is clicked, then TMT sends a HTTP GET request to the BAS system:

```
GET http://BAS.local/tactics/smartcameras/sessions/1/behaviours
```

The BAS system replies with a JSON data message that includes relevant information of the behaviours and that is shown in the following TMT screen:

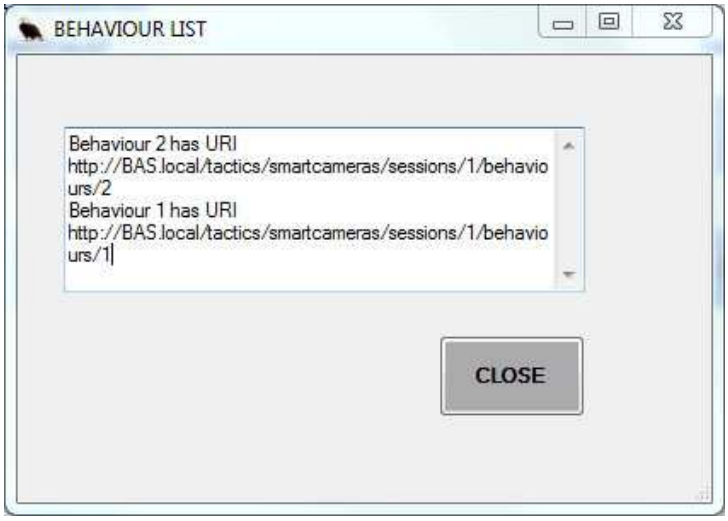

Figure 6-32: TMT Behaviour list

If the user needs to check the features of a particular behaviour, then he must select the behaviour id in the corresponding combo box and click on the 'GET BEHAVIOUR' button. A HTTP GET request is sent to the BAS system with the behaviour id like the following:

```
GET http://BAS.local/tactics/smartcameras/behaviours/3
```

The action at the TMT can be seen in the following figure:

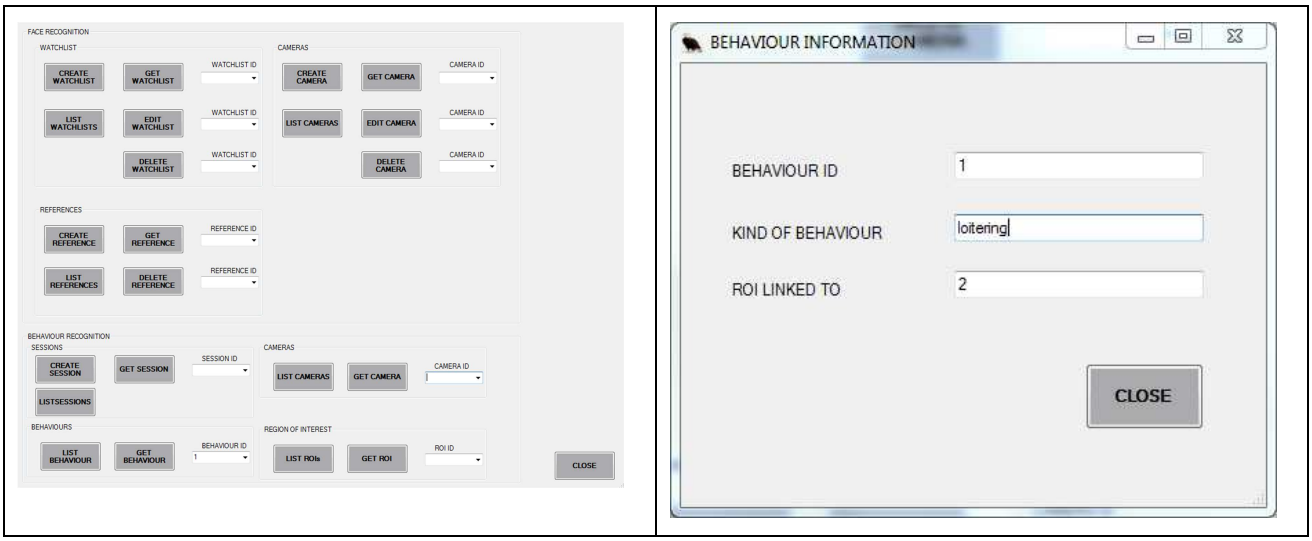

Figure 6-33: TMT Behaviour information

A.5.4 Camera actions

In this section camera actions and their implementation in the TMT system are stated.

Following we can see the UML use case diagram representing the actors involved and the TMT camera actions:

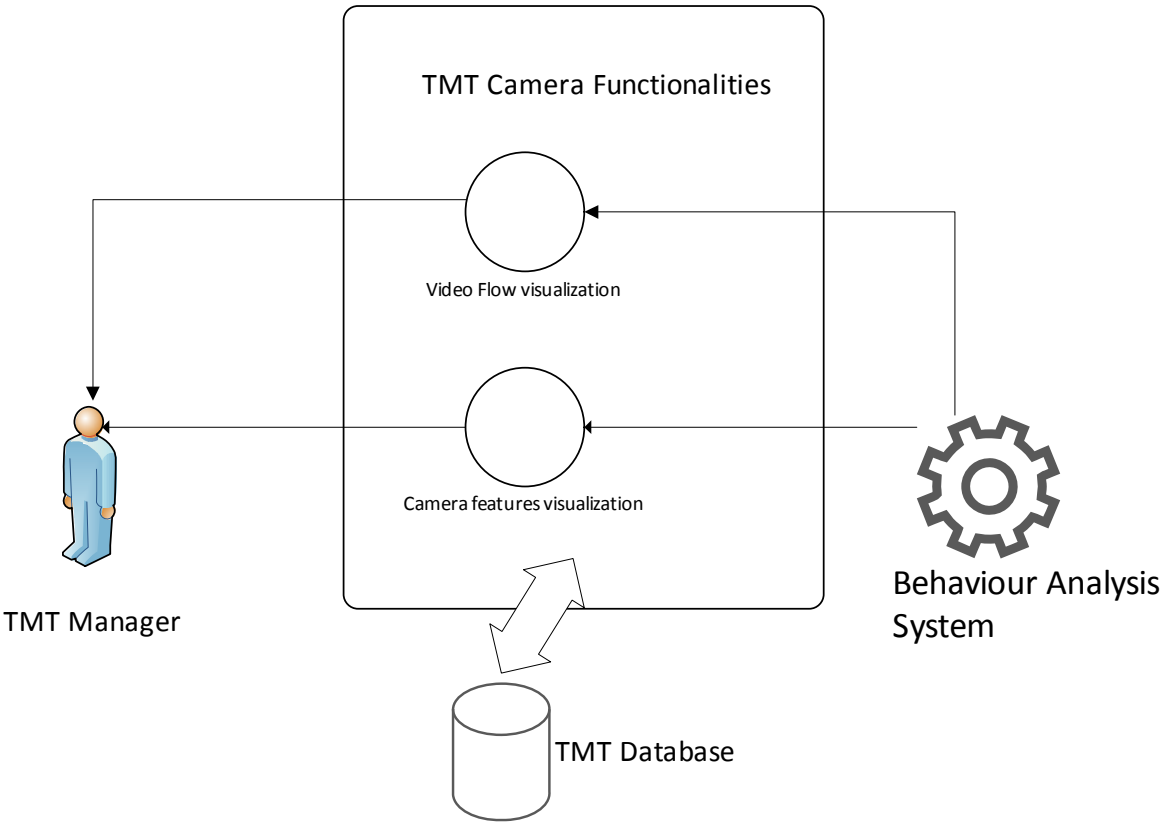

Figure 6-34: TMT BAS Camera functionalities use case diagram

A.5.4.1 Video flow visualization

Video flow visualization will be described in the alarms setup section.

A.5.4.2 Camera features visualization

To see the features of a given list of cameras the user must go to the data fusion management screen of the TMT. If the button 'LIST CAMERAS' is clicked, then TMT sends a HTTP GET request to the BAS system:

```
GET http://BAS.local/tactics/smartcameras/sessions/1/cameras
```

The BAS system replies with a JSON data message that includes relevant information of the cameras and that is shown in the following TMT screen:

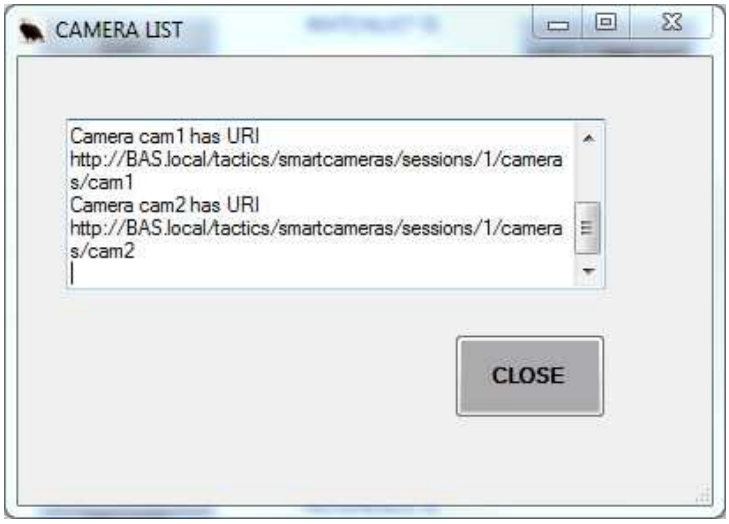

Figure 6-35: TMT Camera list

If the user needs to check the features of a particular behaviour, then he must select the behaviour id in the corresponding combo box and click on the 'GET CAMERA' button. A HTTP GET request is sent to the BAS system with the camera id like the following:

```
GET http://BAS.local/tactics/smartcameras/sessions/1/cameras/1
```

The action at the TMT can be seen in the following figure:

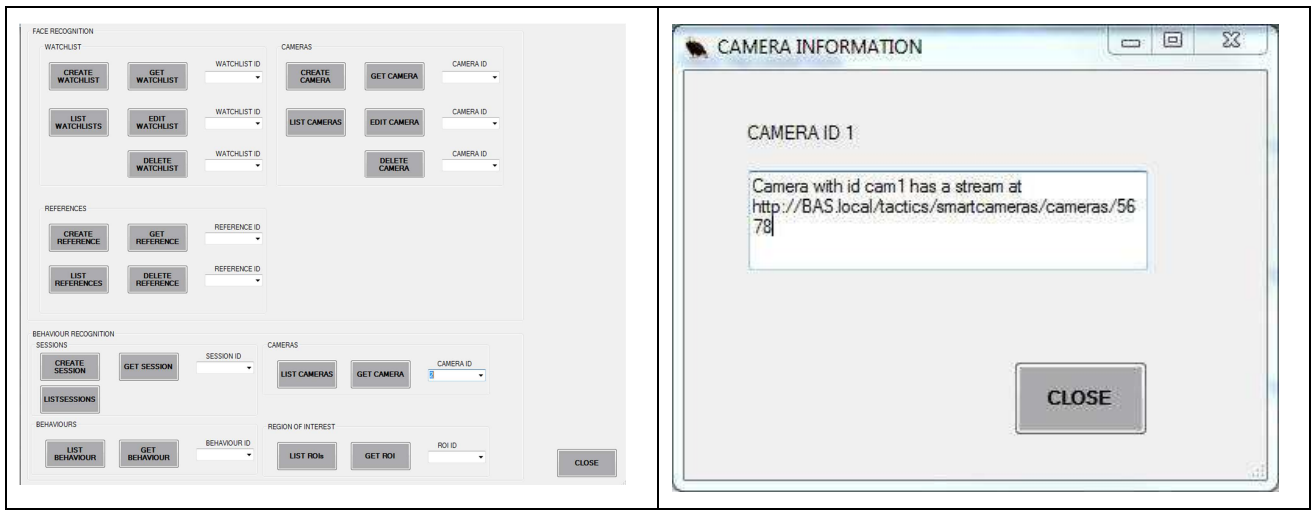

Figure 6-36: TMT Camera Information

A.5.5 Alarm actions

In this section behaviour actions and their implementation in the TMT system are stated.

Following we can see the UML use case diagram representing the actors involved and the TMT behaviour actions:

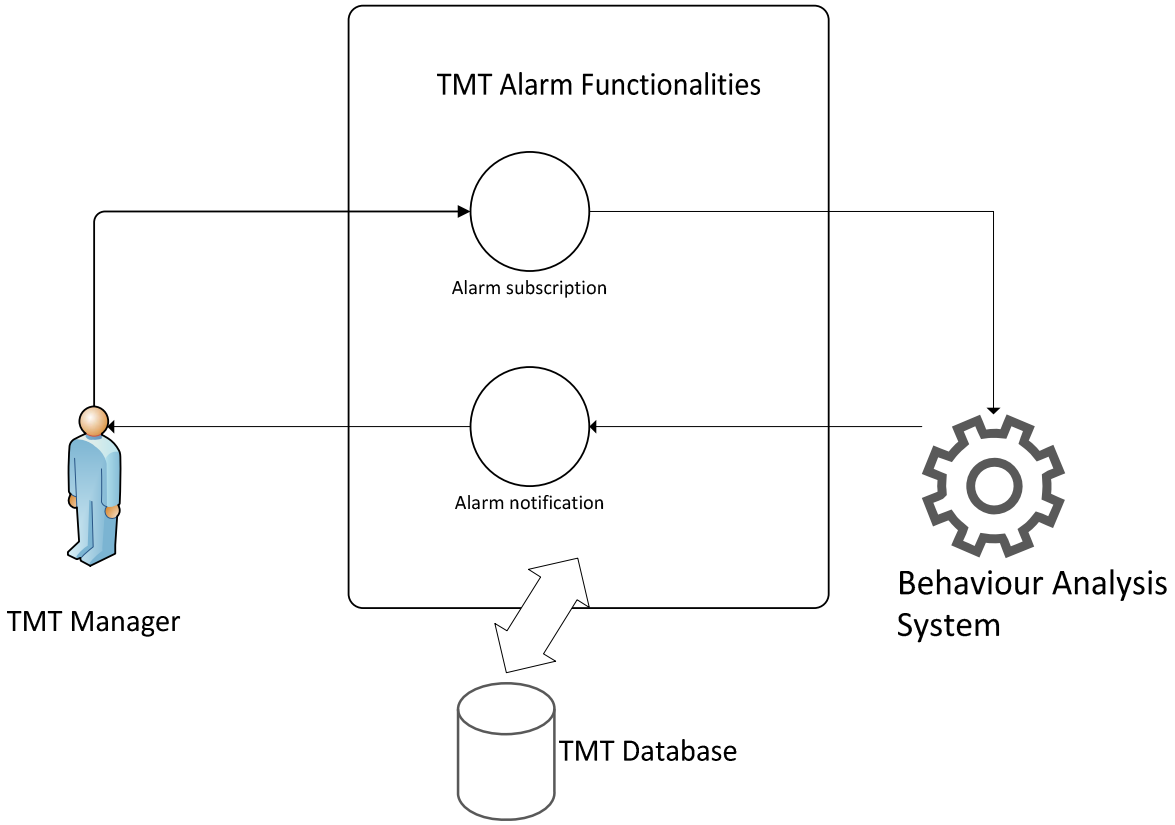

Figure 6-37: TMT BAS Alarm functionalities use case diagram

A.5.5.1 Alarm setup

The alarm setup action is a complex process that includes the creation of several BAS API entities such as ROI and behaviour but all those actions are encapsulated to the end user.

Basic steps to be taken in the overall BAS usage from the TMT point of view are the following:

- The TMT creates a session (currently replacing any existing one).
- Information about camera streams can be obtained from the link to the session. The TMT can then access the streams
- The TMT user defines one (or more) ROIs.
- The TMT user defines one (or more) behaviours. The moment the behaviour is posted to the BAS system, the analysis starts. The BAS will send alarms in case the behaviour is found linked to someone similar to a person in the ROI.
- When a behaviour is deleted, the analysis is stopped and no more alarms will be sent.
- If a session is deleted everything stops (streams, alarms).

First thing to be done is to create a new session. So when the TMT access is started, a session is sent (transparently to the user) to the BAS. To generate a session, TMT sends a HTTP POST as follows:

```
POST http://BAS.local/tactics/smartcameras/sessions
```

With JSON data:

```
{
"localStartTime":"87345","streamStartTime":"87355","returnURI":"http://192.168.21.14/receptionUrl"
}
```

The receptionUri is the endpoint where TMT expects the potential alarm from BAS.

So, to setup an alarm, first the TMT user must access a video flow. To select video flows from BAS in a particular region, the TMT manager must be at the main TMT screen and click in the behaviour analysis icon, as shown in the next figure:

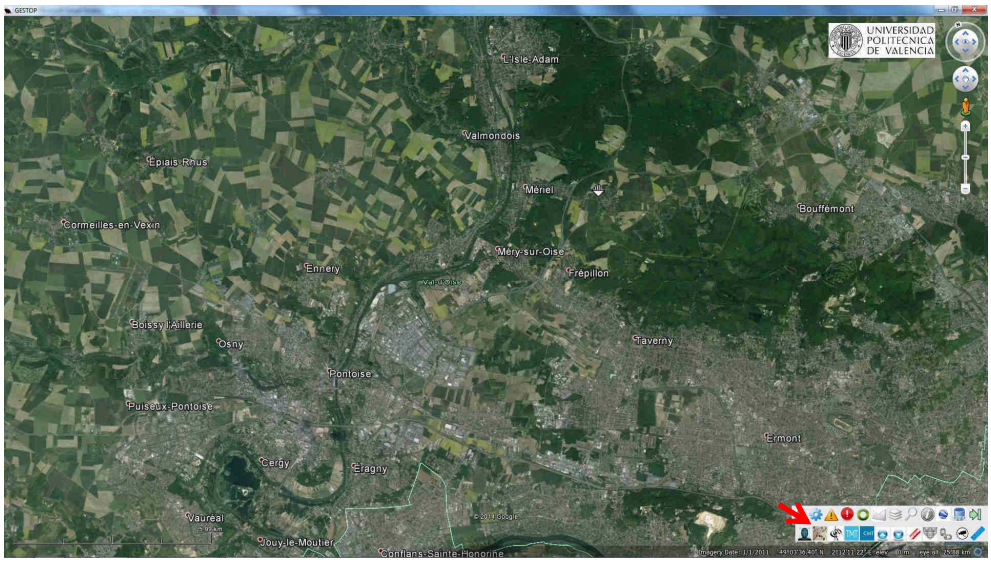

Figure 6-38: TMT Behaviour recognition button

Then, a new screen will pop-up with the following appearance:

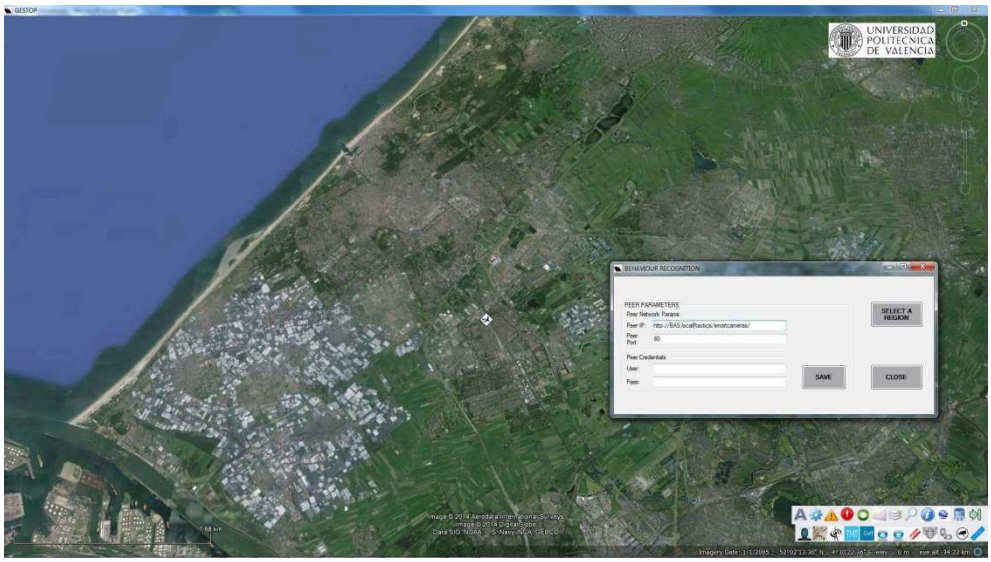

Figure 6-39: TMT BAS area selection I

We can adjust the connection parameters with the BAS system (just once as they will remain in the TMT database) and we have the 'SELECT A REGION' button which allows us to select a particular area to see the cameras within and apply BAS features to their video flows. Then, we click on that button and then select four points on the map as follows:

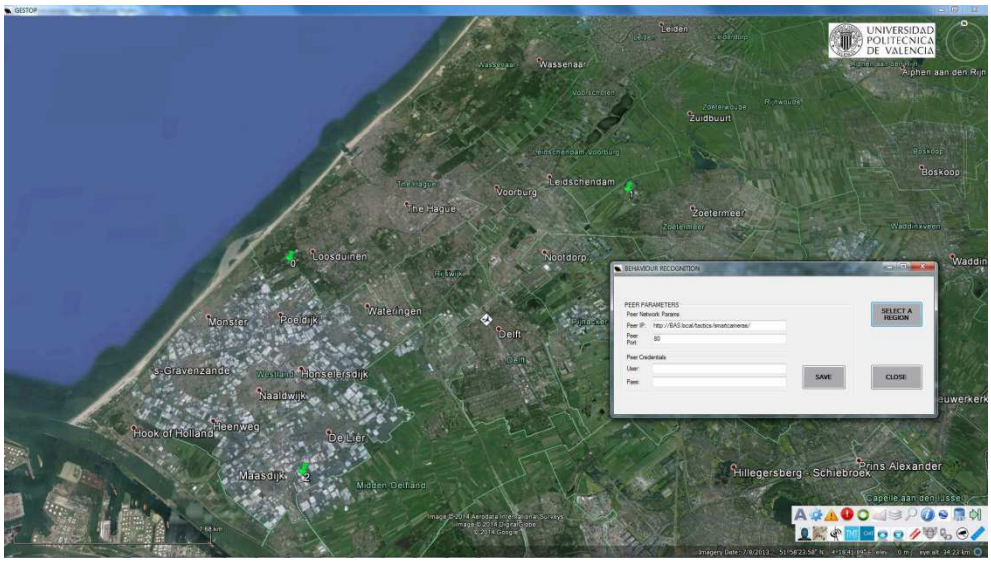

Figure 6-40: TMT BAS area selection II

Once the user selects four points, the system will zoom to the selected area shown the BAS-enabled cameras inside.

Figure 6-41: TMT BAS area selection III

At this point, the user can access to the cameras associated video flows by clicking on their icon after enabling the info mode (I button in the toolbar). If that action is done, the following screen will appear:

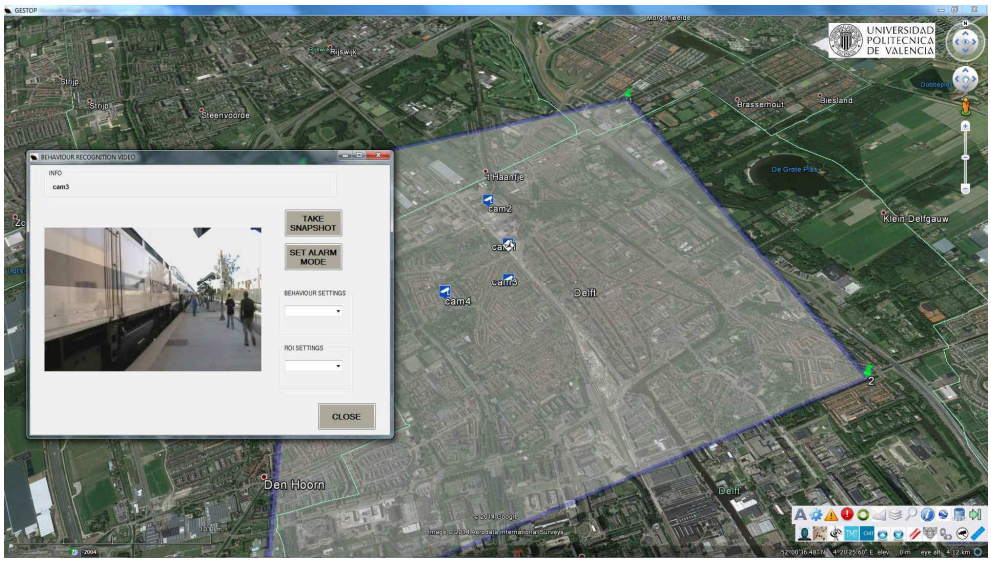

Figure 6-42: TMT alarm configuration I

There can be seen the real time video flow from that particular camera and also several BAS system actions that can be taken. As can be seen in the following figures, the user can specify the kind of behaviour to look for and the size of the ROI he is interested in.

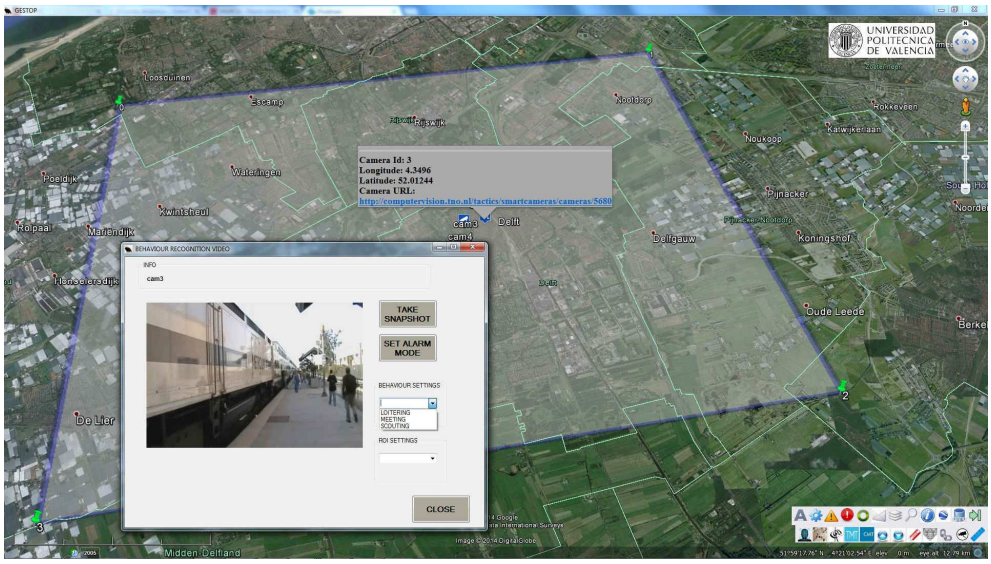

Figure 6-43: TMT alarm configuration II

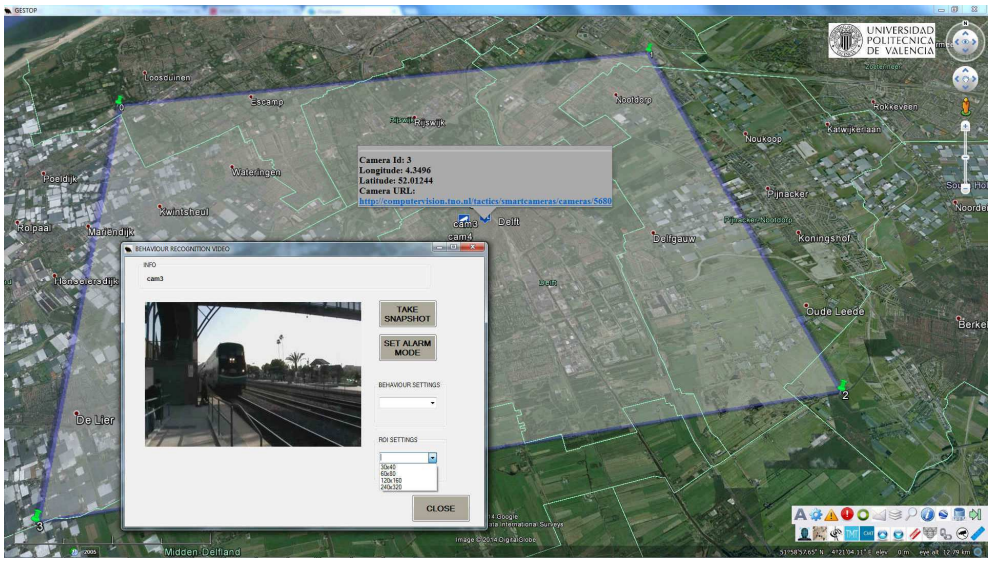

Figure 6-44: TMT alarm configuration III

Selected behaviour and roi size are stored for further usage at the time the query will be done to the BAS server. To get the system ready for on-video selection of the element to be analysed, the user must select the ‘SET ALARM MODE’ button as can be seen in the following figure:

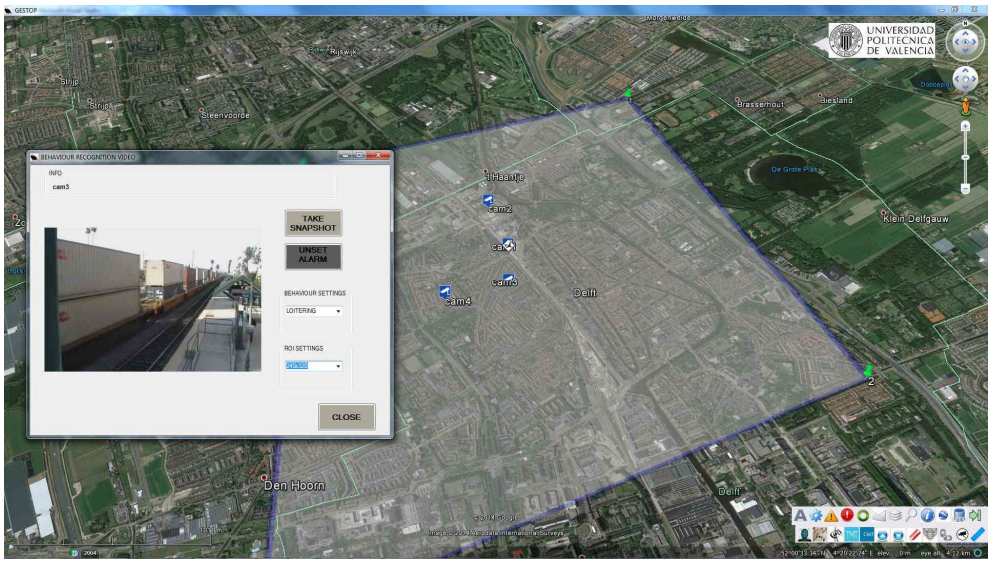

Figure 6-45: TMT alarm configuration IV

At this point, if the user clicks on the video flow over a suspect person, TMT will generate a request to the BAS system with all the previously configured parameters. In the next figure a debug-mode snapshot can be seen with some relevant information. In normal work this will not be shown to the user.

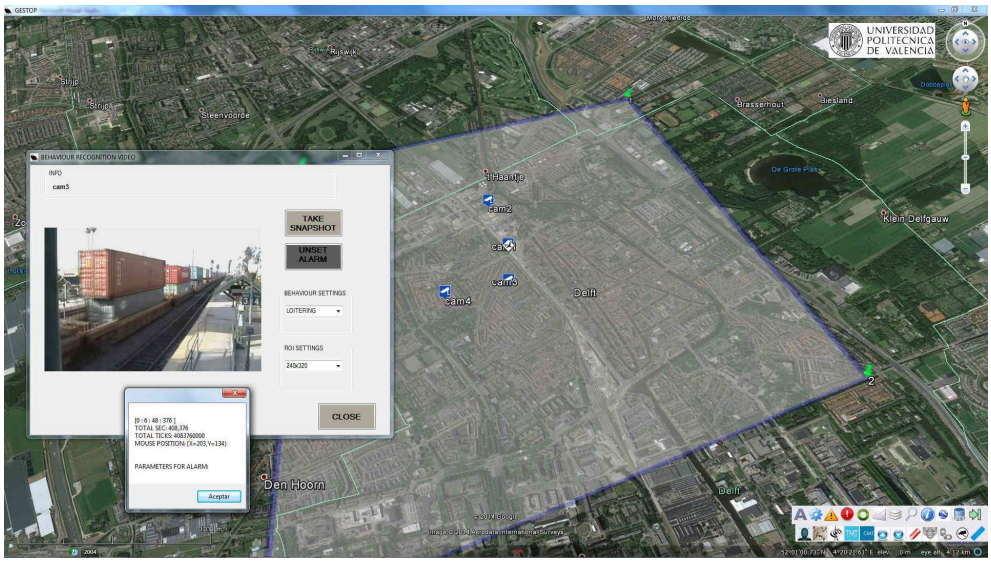

Figure 6-46: TMT alarm registration done

Particularly, once clicked on screen, TMT generates, following strictly this order:

- A ROI
- A behaviour

So to generate a ROI, TMT sends a HTTP POST to BAS like the following:

```
POST  http://BAS.local/tactics/smartcameras/sessions/1/rois
```

With JSON data:

```
{
  "cameraId":"cam3",
  "cameraTimeStamp":"238634569001",
  "boundingBoxX":"106",
  "boundingBoxY":"123",
  "boundingBoxWidth":"320",
  "boundingBoxHeight":"240"
}
```

After this process is done, a behaviour must be generated. TMT sends a HTTP POST like the following:

```
POST http://BAS.local/tactics/smartcameras/sessions/1/behaviours
```

With JSON data:

```
{
  "roiId":"1",
  "behaviour":"loiter"
}
```

Being ‘roId’ the id of the ROI recently generated. Once the behaviour is sent, the BAS system starts the analysis and will sent an alarm to TMT if something is detected.

A.5.5.2 Alarm reception

TMT system is passively listening for the reception of new alarms it previously subscribed for. In the subscription process the listening endpoint was specified. Then, whenever BAS detects an event, it sends the alarm to specified URL of TMT with the following HTTP POST:

```
POST http://192.168.21.14/receptionUrl
```

With JSON data:

```
{
"timeStamp":"234564931001","boundingBoxY":"123","behaviourId":"1","behaviour":"
loiter","boundingBoxX":"111","boundingBoxHeight":"123","cameraId":"cam 2",
"roiId":"4","boundingBoxWidth":"321"
}
```

Which is shown immediately to the TMT manager by means of the end-users specified interface, as shown in the next figure:

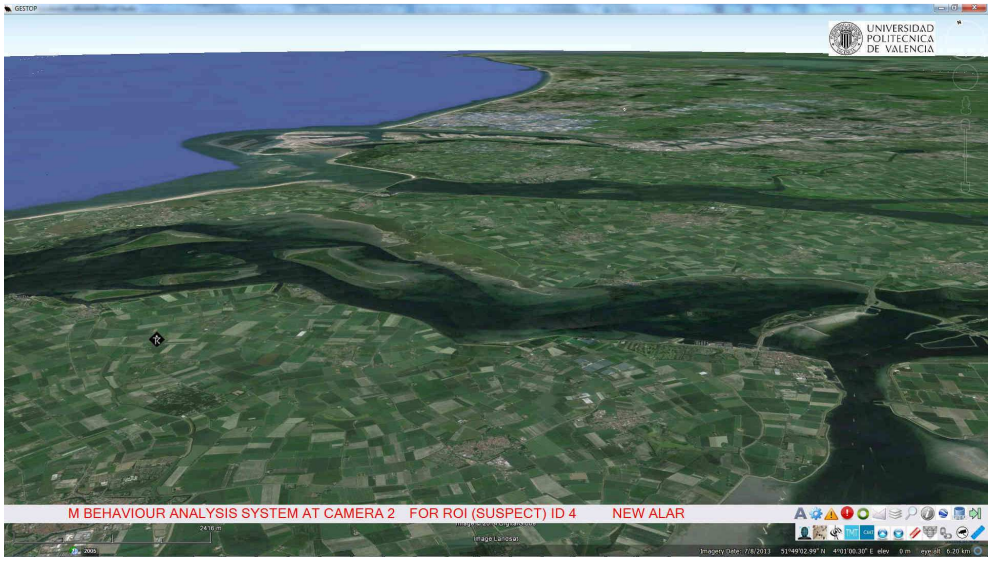

Figure 6-47: TMT BAS alarm reception