PrivAcy pReserving Infrastructure for Surveillance

Deliverable D4.2
SALT Compliant Processes Definition

Project:	PARIS
Project Number:	SEC-312504
Deliverable:	D4.2
Title:	SALT Compliant Processes Definition
Version:	v1.0
Date:	31/01/2014
Confidentiality:	Public
Authors:	Antonio Maña (UMA) Francisco Jaime (UMA) Fernando Casado (UMA) Marioli Montenegro (UMA) Zhendong Ma (AIT) Christophe Jouvray (Trialog) Daniel Le Métayer (INRIA) Denis Butin (INRIA) Víctor Manuel Hidalgo (Visual Tools) Fanny Coudert (KULeuven) François Thoreau (Namur)

Part of the Seventh
Framework Programme
Funded by the EC - DG INFSO

Table of Contents

Document History

Version	Status	Date
v0.4	Preliminary version of a SALTed design process	29/11/2013
v0.5	Refined version of the process	19/12/2013
v0.6	Addition of section 3.2	09/01/2014
v0.7	Addition of section 4.2	17/01/2014
v0.8	Addition of sections 4.4 and 5.2	28/01/2014
v0.9	Addition of section 4.1	29/01/2014
v0.10	Addition of remaining sections execpt for 5.1	30/01/2014
v1.0	Revision of the whole document	31/01/2014
v1.1	Modifications after internal revision	13/02/2014
v1.2	Final version	19/02/2014

Approval		
	Name	**Date**
Prepared	Francisco Jaime	19/02/2014
Prepared	Marioli Montenegro	19/02/2014
Reviewed	Mathias Bossuet, Fanny Coudert	11/02/2014
Authorised	Antonio Maña	19/02/2014

Circulation	
Recipient	**Date of submission**
Project partners	31/01/2014
European Commission	19/02/2014

Executive Summary

This document provides detailed information about engineering processes within the scope of the PARIS project, i.e focusing on accountability and privacy preserving engineering processes for surveillance systems. As explained in D2.2, the SALT "stamp" applies merely on an engineering process. In order to achieve this task, two surveillance system types have been studied and analyzed, covering both, video-surveillance and biometric systems. They have served as a basis for identifying possible SALT concerns that may apply to surveillance systems.

Then, a thorough description of the process follows, listing and describing the elements involved: modelling artifacts (models conceived to help in the development of surveillance systems), work-products, models, etc. At this point, a list of use cases helps to understand the operational functionality of the process. Moreover, the concept of "SALT compliance" is introduced, and why it should be desirable to be fulfilled by the surveillance systems under development.

Finally, this document also analyses how the engineering process impacts on privacy and accountability, and what type of products/documentation it could provide in order to evidence this impact and ensure that the resulting surveillance system design is SALT compliant.

List of Figures

List of Tables

Abbreviations and Definitions

Abbreviation	Definition
CODEC	COder DECoder
CORBA	Common Object Request Broker Architecture
DB	Data Base
DPA	Data Protection Authority
DPM	Domain Privacy Model
EPF	Eclipse Process Framework
FBI	Federal Bureau of Investigation
HTTPS	HyperText Transfer Protocol Secure
IBM	International Business Machines
IP	Internet Protocol
ISTPA	International Security, Trust and Privacy Alliance
IT	Information Technology
JPEG	Joint Photographic Experts Group
JSON	JavaScript Object Notation
LAN	Local Area Network
MDE	Model Driven Engineering
MPEG	Motion Picture Experts Group
NVR	Network Video Recorder
OASIS	Organization for the Advancement of Structured Information Standards
OCL	Object Constraint Language
OMG	Object Management Group
ONVIF	Open Network Video Interface Forum
OPEN	Object-oriented Process, Environment, and Notation
OPF	OPEN Process Framework
PARIS	PrivAcy pReserving Infrastructure for Surveillance
PbD	Privacy by Design
PbD-SE	Privacy by Design documentation for Software Engineers
PET	Privacy Enhancing Technology
PI	Personal Information
PIA	Privacy Impact Assessment
PII	Personally Identifiable Information
PIN	Personal Identification Number
PMA	Privacy Management Analysis
PML	Process Modelling Language

PMRM	Privacy Management Reference Model and Methodology
PTZ	Pan Tilt Zoom
QVGA	Quarter Video Graphics Array
QXGA	Quad Extended Graphics Array
RUP	Rational Unified Process
SALT	Social, ethicAl, Legal, Technological
SFI	SALT Framework Instance
SFMT	SALT Framework Management Tool
SFPI	SALT Framework Project Instance
SPEM	Software and Systems Process Engineering Metamodel
SPM	System Privacy Model
SysML	Systems Modelling Language
TC	Technical Committee
UMA	Unified Method Architecture
UML	Unified Modelling Language
WAN	Wide Area Network
XML	eXtensible Markup Language

1 Introduction

The main goal of the Workpackage 4 (entitled "SALT compliant processes") is to define and describe how an adapted engineering process will enable to ensure, as much as possible, that a determined surveillance system complies with a series of principles in order to ensure that security and privacy concerns are taken into account. When we refer to privacy concerns, we mean privacy concerns related to Social, ethicAl, Legal and/or Technological (SALT) categories, thus the categories are always implicit. In addition, specific attention is brought to accountability. These principles are expressed and hosted within a set of SALT Framework Instances (SFI), embedding considerations from the four SALT categories.

This document focuses on the presentation and definition of a SALT compliant process (next works may refine this initial definition). However, for the sake of clarity, we provide a list of SALT-related terms:

- SALT compliant process: here we can distinguish two types of subprocesses:
    - SALT knowledge building process: it covers the information acquisition process and the information representation process.
    - SALT knowledge use process: it covers the surveillance system design process.
- SALTed design process: this is how we call the process that will produce a SALT compliant surveillance system design (it corresponds to the SALT knowledge use process mentioned above). Within this process we can include system engineering rules, system deployment rules and system use rules.
- SALT compliant system: as a rough definition, by "SALT compliant" we mean that it fulfils with some minimum acceptable requisites regarding to privacy and accountability.

This deliverable mainly refers to the SALTed design process, which aims to support the design and development of a SALT compliant surveillance system. In particular, the process will provide ways to identify requirements and considerations related to the surveillance system under development, and it will help creating a SALT compliant design, deployment, usage and verification (considering the four SALT categories).

In the following text, we provide background information regarding engineering processes. In addition, we also analyse some processes and tools currently used in surveillance systems (for both, video-surveillance and biometric systems). Additionally, this document also identifies and defines relevant concepts that should be considered in order to preserve privacy and accountability in surveillance systems, making an effort to clarify these concepts for the different disciplines.

Deliverable 4.1 (entitled "Generic process for surveillance") presented generic processes applied to video-surveillance systems and biometric systems (sections 2 and 3, respectively). Here, we describe in detail examples for both types of systems, and they will help to highlight relevant concerns and to better understand how a SALTed design process must be. These concerns should be taken into account by system designers in order to comply with privacy aspects, and hence they serve as a basis for the definition of the SALTed design process, which will improve the system design in that respect. The SALTed design process is next resented in this document. In fact, not only the process is deeply described, but also the design objectives and the process fundamentals. Furthermore, use cases illustrate the operation of the process.

Finally, we would also like to mention the SALT framework management tool. The objectives and behavior of this tool are better covered by deliverable 3.1, although we remark that it has an impact and some consequences in the SALTed design process, since it is through this tool that the process gains access to the SALT framework instances, where privacy and accountability related information is stored.

The following text is structured as follows: section 2 provides some background information regarding engineering processes, their usefulness and important definitions related to them and the PARIS project; section 3 describes examples of processes currently used in video-surveillance and biometric systems; section 4 describes a SALTed design process; section 5 elaborates on the impact of the SALTed design process, from privacy and accountability viewpoints; and section 6 concludes the document.

2 Engineering processes background

Engineering processes cover an interdisciplinary field of engineering devised to design and manage complex engineering projects over their entire life cycles. Important issues such as reliability, requirements management and evaluation measurements among others, become more difficult when working with large and complex projects. Engineering processes ensure that all likely aspects of a project or system are taken into account and integrated into a whole, hence their great usefulness. The traditional scope of engineering processes embraces the design, development, production and operation of systems. In this sense, they also refer to the distinctive set of concepts, methodologies and structures that have been developed to meet the challenges of complex systems.

Engineering processes focus on analysing customer needs and required functionalities, i.e. system specifications, early in the development cycle, documenting requirements, and then proceeding with the system design, installation, validation, usage and verification while considering the complete problem, the system lifecycle. Moreover, engineering processes encourages the use of tools and methods to better comprehend and manage complexity in systems.

Interdisciplinary systems are inherently complex, since the behaviour of an interaction among system components is not always immediately well-defined or understood. Defining and characterizing such systems and subsystems, together with the interactions among them, is one of the goals of engineering processes. By achieving it, the existing gap between informal requirements from users, operators, organizations and technical specifications is successfully bridged.

A possible way to understand the motivation behind an engineering process is to see it as a method, or practice, to identify and improve common rules that exist within a wide variety of systems.

2.1 System development process

UML is a standard modelling language widely used to specify object-oriented systems. It defines notations to build several types of diagram, each one representing a particular view of a specific artefact to be modelled. The flexibility that UML provides, allows extending its modelling capacity using profiles, which facilitates the performance of model-based testing. Besides, it is also possible to add constraints to UML models by using OCL (Object Constraint Language), which specifies pre and post conditions to formalize operation behaviours.

Taking into account the UML adaptability and capability, we consider it is an excellent option for creating an engineering process and methodology. Therefore, UML is used as the core of the engineering process due to the following reasons:

- UML is an Object Management Group (OMG) standard that enjoys widespread use throughout industry.
- Tools supporting UML are widely available.
- Familiarity with UML class diagrams helps to mitigate the conceptual difficulty of the metamodelling language architecture.

- The UML metamodel helps us to define a modeling language satisfying the specific needs of a given SALT concern from a specific domain.
- The UML metamodel can be used to define the necessary diagrams describing a particular surveillance system.

With UML as the language to be used for the development process, we then have to find out how to model the different surveillance systems, in order to represent their characteristics, behaviours and domains. Depending on the domain, we may have different surveillance systems, therefore we need different models for each system domain. For this reason, our approach considers the use of metamodels due to their ability to generate models from another, more general, model.

2.2 Formal definitions

This section defines three important concepts that are going to be widely used regarding the SALTed design process.

2.2.1 Privacy by design

Privacy by design (PbD) commonly refers to the term made popular by Ann Cavoukian as the Information & Privacy Commissioner of Ontario, Canada. Cavoukian's PbD is a framework to ensure privacy and increase personal control over one's information. The PbD is defined by the following seven principles[1]:

1. Proactive not reactive: preventative not remedial. PbD advocates the proactive rather than reactive measures that anticipates and prevents privacy invasions before they happen, i.e., PbD comes before-the-fact, not after.
2. Privacy as the default. PbD aims at ensuring that personal data are automatically protected in any given IT system of business practice. Privacy is built into the system by default.
3. Privacy embedded into design. PbD should be embedded into the design and architecture of IT systems and business practices. Privacy is integrated into the core functionality of the system.
4. Full functionality – positive-sum, not zero-sum. PbD seeks to accommodate all legitimate interests and objectives in a positive-sum, i.e., a win-win situation.
5. End-to-end security – lifecycle protection. PbD should be embedded into the system prior to any information being collected, and extends throughout the entire data lifecycle.
6. Visibility and transparency. PbD seeks for visibility and transparency of the business practice or technology to assure all stakeholders the operations are according to the stated promises and objects, subject to independent verification.

[1] Ann Cavoukian, Privacy by Design: The 7 Foundational Principles, [Online] http://www.privacybydesign.ca/index.php/about-pbd/7-foundational-principles/

7. Respect for user privacy. PbD requires architects and operators to keep the interests of the user uppermost by offering strong privacy defaults, appropriate notice, and empowering user-friendly options.

Following Cavoukian's terminology, the following table summarizes the most commonly used practices associated with these principles.

PbD Principles	Fair Information Practices			
1. Proactive not reactive; Preventative not remedial	Privacy commitment	Recognize poor privacy design and practice		
2. Privacy as the default	Purpose specification	Collection limitation	Data minimization	User, retention, and disclosure limitation
3. Privacy embedded into design	Rely upon accepted standards and framework	Privacy impact and risk assessment	Minimize privacy impacts	
4. Full Functionality – Positive-sum, not zero-sum	Not impair full functionality	Embraces legitimate non-privacy objectives	Clearly documented interests and objectives	
5. End-to-End Security-Lifecycle Protection	Security	Applied security		
6. Visibility and transparency	Accountability	Openness	Compliance	
7. Respect for user privacy	Consent	Accuracy	Access	Compliance

Table 1. Fair Information Practices

2.2.2 Accountability by design

Accountability-by-design can be seen as complementary to privacy-by-design in the sense that it targets privacy goals by emphasizing verifiability, meaning the active demonstration of compliance with privacy requirements. It is of notable importance in the context of surveillance because a priori controls of system properties by individuals are mostly impossible; in particular, individuals have no guarantees in advance about the handling of their personal data by the users of the surveillance systems. Individuals possess little knowledge about the detailed workings of the surveillance infrastructure and associated data flows.

As a consequence, a posteriori control is needed. The core idea of accountability-by-design is first to take into account the requirement of a posteriori verification as soon as the design of the entire infrastructure starts. In practice, for systems recording, storing and processing personal data, it implies the demonstration by the data controller that all data handling operations are in line with obligations, which may either derive from regulation or be specific to the system under consideration, for instance following a negotiation with a representative of the public, defending the interests of the public affected by the surveillance system.

Technical considerations such as the design of logs, their secure storage (possibly using cryptography), and their (possibly mechanized) compliance check against machine-readable privacy policies all fall under the scope of accountability-by-design. The surveillance system should also be structured from the start to allow audits of the manual actions of its operators, with the decision of which audit information to exactly store carefully considered.

Accountability-by-design also means to develop and deploy the necessary organizational procedures as to ensure that the required controls are integrated into organizations' processes. Internal procedures should ensure that privacy concerns are taken into account in all business activities and provide assurance that practices comply with the organization's privacy policy. In that sense, it impacts the structuring of the privacy program, i.e. the privacy governance structure of the organization. It could amount to the appointment of a Data Protection Officer or the setting-up of periodical review of policies, procedures and practices.

2.2.3 SALT compliance definition

This document defines and describes how a “by-design” process will enable to ensure as far as possible that a surveillance system takes privacy aspects into account. The principles are themselves hosted and expressed within a SALT framework instance, embedding technical, legal, social and ethical considerations as the SALT acronym implies (Socio-ethicAl, Legal and Technological).

A surveillance system will be SALT compliant when its design and usage follows the SALTed design process, which means that it respects the following characteristics:

- Use of the SALT Framework instance for assisting decision-making.
- Involvement of the organizational, technical and human factors as depicted by the SALT Framework Instance.
- Exhibition of the required artifacts (i.e., for ensuring X-by-design) defined by the SALT Framework Instance.

3 Example of processes

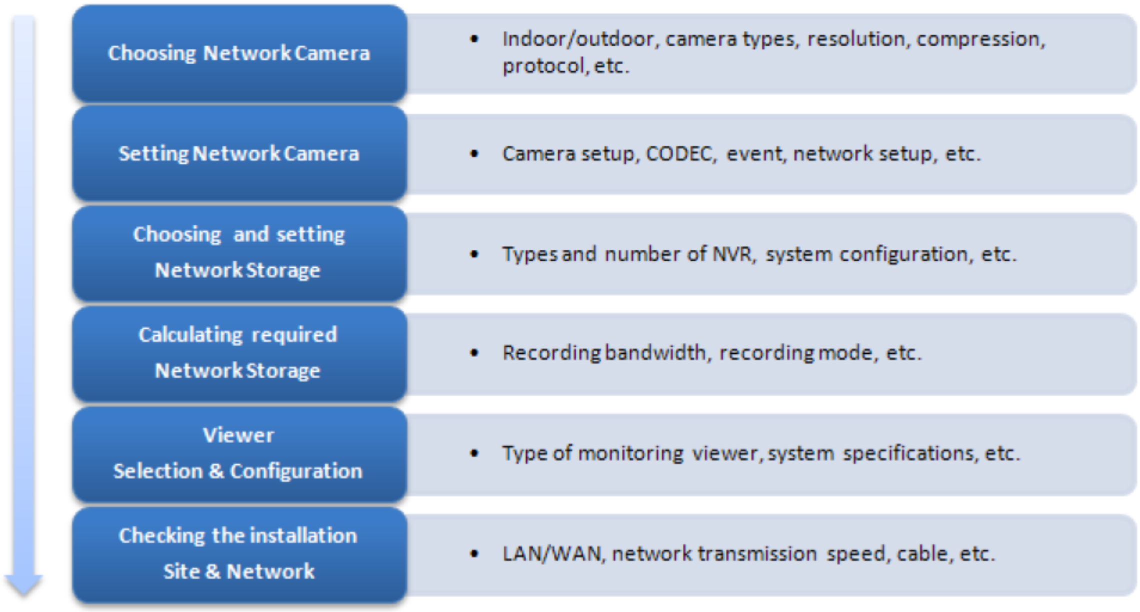
This section provides some examples of current processes for actual surveillance systems, differentiating between video-surveillance and biometric systems. They provide an insight of system designers about current surveillance systems, which helps to highlight a series of privacy and accountability concerns that should be taken into account in order to mitigate possible privacy risks. These concerns will serve as a basis for the SALTed design process described in section 4.

3.1 Example processes for video-surveillance systems

In the following, we use a design process (specified by Samsung[2]) to exemplify some typical considerations and procedures in developing digital network-based video surveillance systems.

3.1.1 Processes description

Figure 1 illustrates the design flow of a simple video-surveillance system (deliverable D2.2 describes the procurement and engineering for a complex video-surveillance syste, the Tabasco City surveillance system).



Choosing Network Camera	• Indoor/outdoor, camera types, resolution, compression, protocol, etc.
Setting Network Camera	• Camera setup, CODEC, event, network setup, etc.
Choosing and setting Network Storage	• Types and number of NVR, system configuration, etc.
Calculating required Network Storage	• Recording bandwidth, recording mode, etc.
Viewer Selection & Configuration	• Type of monitoring viewer, system specifications, etc.
Checking the installation Site & Network	• LAN/WAN, network transmission speed, cable, etc.

Figure 1. Design flow of a video-surveillance system

From an engineering point of view, the main steps involved in video surveillance system design include:

1. Choosing network camera
2. Setting network camera
3. Choosing and setting network storage
4. Calculating required network storage
5. Viewer selection & configuration
6. Checking the installation site & network

[2] Samsung Techwin, Networked Surveillance System Design Guide, 07/2012

1. Choosing network camera. There are a variety of cameras on the market. Being independent from camera manufacturers and cost, the decisions in this step are based on whether it is for indoor or outdoor usage, type of cameras (e.g., zoom cameras, dome cameras, or Pan-Tilt-Zoom (PTZ) cameras) and their corresponding field of view, image resolution from 320x240 (QVGA) to 2048x1536 (QXGA) pixels, video compression methods (e.g., Motion JPEG, MPEG-4, H.264), network security features (e.g., HTTPS authentication, IP address filtering, user access log, access control), and communication protocol supported by the cameras such as Open Network Video Interface Forum (ONVIF).

2. Setting network cameras. Camera setup might include the day and night function, backlight compensation, and noise compensation etc. for adjusting image quality to environment. Camera setup also includes the settings that specify how cameras compress and transfer video data, i.e., the CODEC setup. Camera setup further includes event setup, which notify a user by the time of event, search and save the video data of the event. The cameras are also set according to the network environment.

3. Choosing and setting the network storage. The Network Video Recorder (NVR) is a storage device attached to the network that stores video data from cameras and encoders. It provides both storage for video data and distribution of the stored data. In the design process, the system designer needs to decide on the number of NVRs depending on the data volume. As storage devices, one needs to decide the hard drive configuration of NVRs for data access speed and backup schemes.

4. Calculating required network storage. The designer needs to calculate the required storage capacity based on planned data retention period. Furthermore, one also needs to consider the input bandwidth to storage device, and the recording mode of storage device.

5. Viewer selection & configuration. Viewer provides surveillance operators and users the means to watch video footages from the cameras or the NVRs. Technology for viewer include Web viewer, integrated viewer for monitoring multiple surveillance channels, and mobile viewer. In large systems, the viewer can be provided from an independent software vendor.

6. Checking the installation site & network. This is the last step before the installation of the design. Some factors to be considered in this step include network infrastructure at the installation site, its transmission speed and physical components.

The above example only shows the design of video surveillance systems. It is clear that the design process is complex, involving several engineering fields, and it requires many decision-makings.

3.1.2 SALT concerns and processes

Since the SALT framework stores relevant information in order to support Privacy by Design in the engineering process of a surveillance system, one possible way to implement it is to integrate PbD and other necessary activities into the engineering process. Figure 2 gives a high level view.

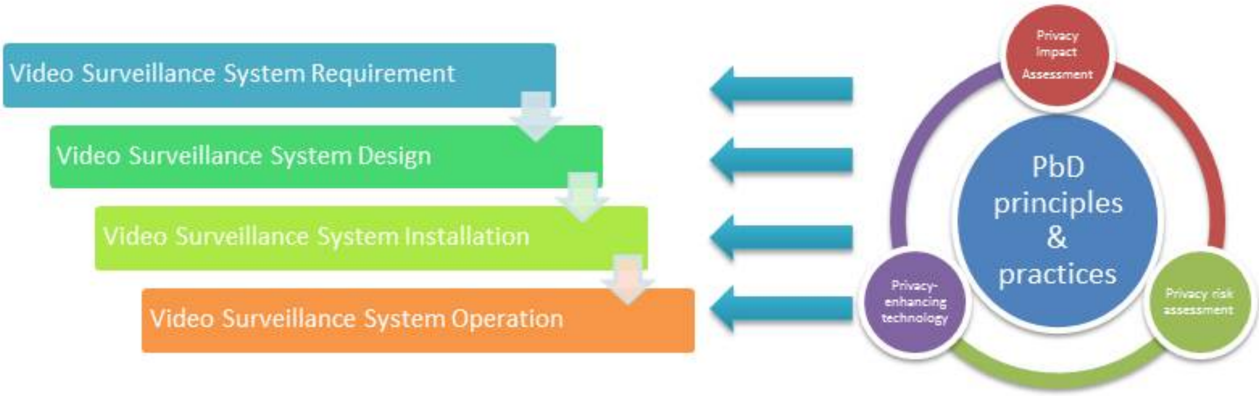

Video Surveillance System Requirement
Video Surveillance System Design
Video Surveillance System Installation
Video Surveillance System Operation
PbD principles & practices
Privacy Impact Assessment
Privacy risk assessment
Privacy-enhancing technology

Figure 2. Adding PbD to the engineering process

The left side of the figure is the usual engineering process of video surveillance systems. On the right side, the centre is PbD with its principles and practices. These principles and practices are very general. Therefore, three components, i.e., Privacy Impact Assessment (PIA), Privacy risk assessment, and privacy-enhancing technology, are the instruments that can be applied to the engineering process.

For example, PIA can be applied in the system requirement phase to assess the overall privacy impacts of surveillance. In the design phase, a system architect can integrate privacy risk assessment into the design phase (e.g., the one described in Section 3.1.1) for each of the technical decisions. Hence, we have two interactive processes: design process and risk assessment process. If any privacy risk is identified, the system architect can add privacy-enhancing technology to the design to mitigate or minimize these risks. All activities will be directly and indirectly guided by the PbD principles and practices. SALT framework instances will provide detailed and implementable guidelines and knowledge in these processes.

3.2 Example processes for biometric systems

After seeing the generic process for a biometric system on section 3 of deliverable *4.1 "Generic process for surveillance",* in this subsection we describe an example of process for a biometric system. First of all, we will see the processes description about a real installation. After that we will focus on the main parts of the system that can have a negative impact on the user.

3.2.1 Processes description

Below, we will introduce the process description for a specific example. The first step is to capture the requirements of the system to have a clear vision about the perspective of the system. For this reason, we need a deep description from the organization point of view. Now, we are going to summarize the context in which this example unfolds.

In this case, a specific customer contacted us for securing access to certain parts in some restaurants in Africa. The main problem was that they had had significant operational problems and economical losses due to the fact that the food was stolen. For this reason, they were interested in protecting the access to two specific areas of the restaurant:

- The restaurant kitchen, where only restaurant employees (waiters, cooks, etc.…) should be allowed to enter. Generally, people who go into this room, usually cooks, spend long periods of time, because waiters do not need to enter to this room to take the food. For this reason, they did not need a very quick response of the system.

- The second and most important area was the room where they keep the food. The desire of the owners was that only a reduced group of people, maybe 2 or 3, could access this place.

Once we have shown a general overview of the purpose of the system, we are in position to express the requirements on the system, and then propose a system design and a system operation mode. As we had two different areas to protect, we needed at least two systems. Both of these systems required an **enrollment phase**, since they needed to know beforehand the people who are going to be allowed to enter in the kitchen and the rooms where they store the food. About the **mode of operation**, as the customer did not specify whether the users need a kind of card, a username or a pin, we thought that the best option was to have these system working in **identification mode**, due to the fact that the number of employees of the restaurant would never be too large and the system could make the identification in a reasonable amount of time. This decision also prevents users from forgetting their pins or losing their cards.

It is also very important to describe how the system would **react to a positive or negative identification** of the system. In this case, it was clear that in a positive identification, the corresponding door would open. In the case of a negative response, we proposed to establish a communication between the system and the manager of the restaurant who would be responsible for taking the appropriate actions. However, they did not want to disturb the responsible so that he/she could focus on his/her tasks. That is why in a negative identification, they specifically asked to add the possibility to let the user introduce a personal PIN. Just in the case employees forget the PIN, they would contact to the manager. In that case, the manager would use the system to generate a new pin for the employee. Figure 3 shows a graphical representation for this process.

As regards the **data management**, the system stores the name of the employee, the pin in the case of a negative identification, and the specific template that represents his/her biometric characteristic. Of course, all these data are encrypted on the system. The stored template is a mathematical representation and not the raw data. About deleting these data, if one employee leaves the company, his/her personal data will be deleted of the system. In the same way, if one of the persons who is allowed to enter the kitchen or the room where the food is stored changes his/her functions in the restaurant, his/her personal data will be deleted, since from this point on he/she will have nothing to do with the kitchen and the room where the food is stored.

About the **operational requirements**, both systems do not exchange any kind of data with other systems or organizations. About the interaction between the employees and the system, the customer did not establish any constraints. Therefore, the users could directly interact with the system as long as the system response was not slow, especially in the room where food is stored, because of people with those permissions would access to it a large number of times a day. Furthermore, the system would not be available when the restaurant is closed. As a consequence, nobody could use the system and enter the different secured places during closing hours. In addition, the system records a weekly log where it stores information about how and when people enter in the specific areas. In the case authorities ask the restaurant for some kind of information, the manager/administrator of the restaurant would be in charge of providing the required information.

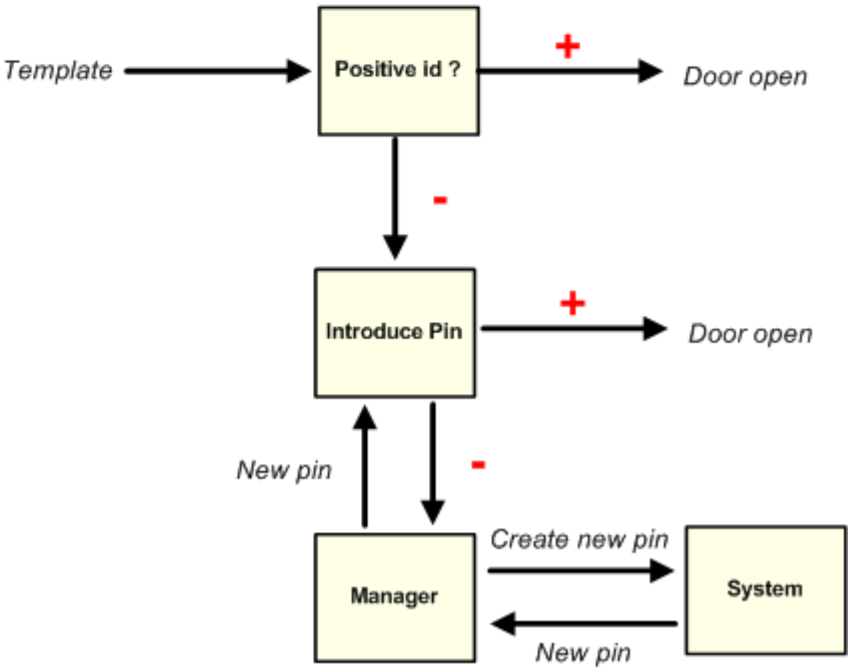


Figure 3. Graphical representation for a positive and negative identification of the system

From a **technical** point of **view**, since the client did **not require us to communicate** with **other system** and **organizations**, the **processing** could be done for a specific device such as retina scan or in the case that sensors only provide the raw data, such as a camera, it would be done by a specific computer. In the same way, **data analysis** and **storage** would be done in a specific server and a central database. About the **communications**, as these systems were not going to have any kind of communication with external systems, in the case that different sensors used the network to communicate with each other, for example to establish a communication between the retina scan and the central server, they would use an isolated LAN.

It was also important to know the **number of people** who would **undergo to the system**. For the system that has to protect the room where the food is stored, it was clear that the number of people would be very low. For the other system, which has to protect the kitchen, the number of people would not be high. Based on the data that the customer gave us, none of the restaurants would exceed 30 employees. In addition, we had to take into account that the number of employees could grow in the future. Anyway, this number will never be very large. In addition, the customer did not require integrating the system with another system.

Regarding the **performance,** we needed to focus on the **accuracy**, **error rate** and **time of response** of the system. For the system that has to protect the kitchen we needed an accurate system with a low error rate. However, if the system fails, the consequences would not be critical, since it is highly probable that there would be more people inside the kitchen and if the possible suspects would want to steal food, they would need to pass other access control. In this case, as we explained before, the customer did not require a big time of response, since people who go into this room spend long periods of time inside the kitchen. On the contrary, the system that protects the room where the food is stored was critical. Therefore, it needed a high accuracy system with a very low error rate. About the time of response, it was necessary that the system had an acceptable response time due to the fact that these employees would often enter the room.

About the **environment conditions**, the global system was going to be installed in several restaurants and the conditions could change a lot. For instance, the level of light could change between different restaurants. In this environment, the level of noisy, temperature and humidity is high.

About the **privacy measures**, all employees should be informed about the kind of personal data that are stored. In addition, all employees should know when the system is processing and analyzing their personal data. Plus, the only person who could access to the logs of the systems is the administrator, possibly the restaurant manager. Besides, the system has a log to store the people how access to the information of the system.

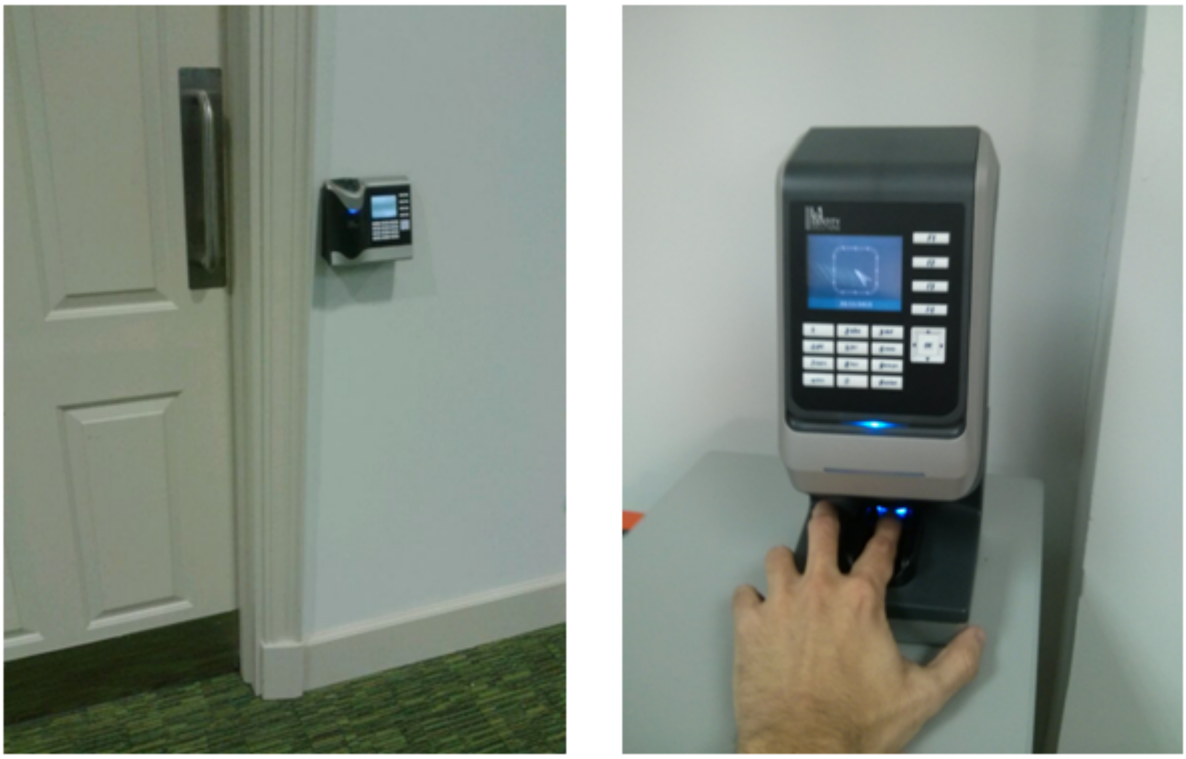


Figure 4. Finger print recognition device and finger vein recognition device

Once, we have described the requirements of the whole system, we pass to the **design & development phase**. Based on the requirements previously captured, we selected a fingerprint recognition device for the system that protects the kitchen, because it is a system that works very well for small and medium scale systems. On the contrary, for the system that protects the room where the food is stored, we selected finger vein recognition, because it is a very accurate system with a low error rate, which perfectly fits with the requirements. Furthermore, the response time of the system is acceptable for this scenario. Concerning the architecture of the system, the system is composed of the fingerprint recognition, the finger vein recognition, a central server and a database. These devices are responsible for calculating the mathematical representation of the template, and then templates are sent to a central server. This server stores the data in the enrollment phase. On the matching phase, the devices deliver the corresponding template to the central server, and then the central server makes the comparison between the current template and the one previously stored.

Finally, it is important to highlight that we carried out different **tests**. Initially, we did some test in a laboratory environment. Next, we tested the whole system in one of the restaurants where we had controlled the population. And finally, we executed the final test in one of the restaurants without controlling the population. In Figure 4 we can see the fingerprint recognition devices installed on one of the restaurants, and the finger vein recognition during the tests in a laboratory environment.

3.2.2 SALT concerns and processes

Once we have seen an example of one of our deployments for a biometric system, we are going to analyze the different parts of the system that could impact on the privacy of the users in order to take them into account for the SALT compliance.

Almost all biometric systems require an enrollment phase, where the system **stores** the **biometrical characteristics** of individuals. Therefore, when we talk about biometric systems, it is almost impossible to understand them as such without storing personal data in the enrollment phase. They could even have systems that store information in the matching phase, but as a general rule, this should be avoided, since the system already has templates to make the comparison. Generally, regarding the storage, we have to be especially careful with the **collection, use, retention and disclosure** in order to avoid the **misuse of the data**.

When a system stores personal information, people might prefer to **keep** their biometric **data private**, since the organization could extract more information about them. For example, if we had installed an iris recognition device instead of a fingerprint recognition one, the restaurant could use the biometric data to extract other kind of information such as illness. In this way, not only the information would not be private, but the organization could also be using the information for **another purpose** (function creep).

Another important point has to do with the knowledge of the users about their personal data. In some cases, this information must be restricted, for instance the use of a fingerprint in a criminal investigation. The organization should normally inform to their employees what and why their data is stored and who access to this data. In the previous example, everybody knows that there is a system that is saving their fingerprints and their finger-vein shapes to enter to some specific areas of the restaurant, and the manager of the restaurant is the only one who can access to this information. So, the users of the systems should have the **rights of information and access**.

From the technical and operational point of view, it is very sensitive that the systems are **integrated** with **other systems or organizations** to **exchange different data**. The interoperability between organizations generates higher data exchange. As a consequence, the likelihood to use the biometric data for **other purpose** is greater. For example, collecting information from different sources makes possible that people can be tracked, or the organization even creates profiles about their behaviors or activities to predict actions or categorize persons. In the restaurant example, the security system is not integrated with other systems or organizations, therefore we avoid the possibility to use the personal information of the employees.

In addition to the misuse of data, biometric systems can **damage** the **personal identity** and body integrity. A good example that can damage a person's identity has to do with the **mode of operation** of the system. For example, if the system works in the categorization mode, it can classify people as legal or illegal, low or high risk, which could damage the identity of a person. As you know, biometric systems are not perfect. So if we work in the identification mode, an error of the system could be very harmful to the person. A good example that represents this type of error is the case of Brandon Mayfield. He was arrested over two weeks in 2004 in connection with the attack on some trains in Madrid. The fact that led to his arrest was that the fingerprint found by the Spanish police and the one analyzed by the FBI coincided. After that, Spanish authorities found out that the fingerprints actually belonged to someone else. As it was explained in the D4.1 *"Generic process for surveillance"* section*,* the identification mode compares 1 template against N stored in the database. Thus, the more people undergo the system, the greater the possibilities to get an error. Regarding the verification mode, here it is less probable to make a mistake, due to the fact that the comparison is 1 to 1.

Furthermore, the **accuracy** and the **error rate** are two parameters very important to consider. The greater the consequences are for a person, systems have to be more accurate and with a minor error rate to minimize the impact on the person. In the restaurant example, the system that protects the kitchen has less impact on the person than the system that protects the room where the food is stored. In the case that the consequences for a person could be disastrous, it could be a good idea to use a multimodal system in order to minimize the likelihood of getting an error.

4 General SALT compliant process

This section thoroughly describes a general SALT compliant process, focusing on the SALTed design process, i. e., the modified engineering process followed by system designers in order to develop a SALT compliant surveillance system (SALTed surveillance system).

4.1 Design goals for the process

4.1.1 Objectives

The SALT Framework is a composition of tools, methods, activities and processes (see Figure 5). Different stakeholders will be involved either for adding to or using the knowledge.

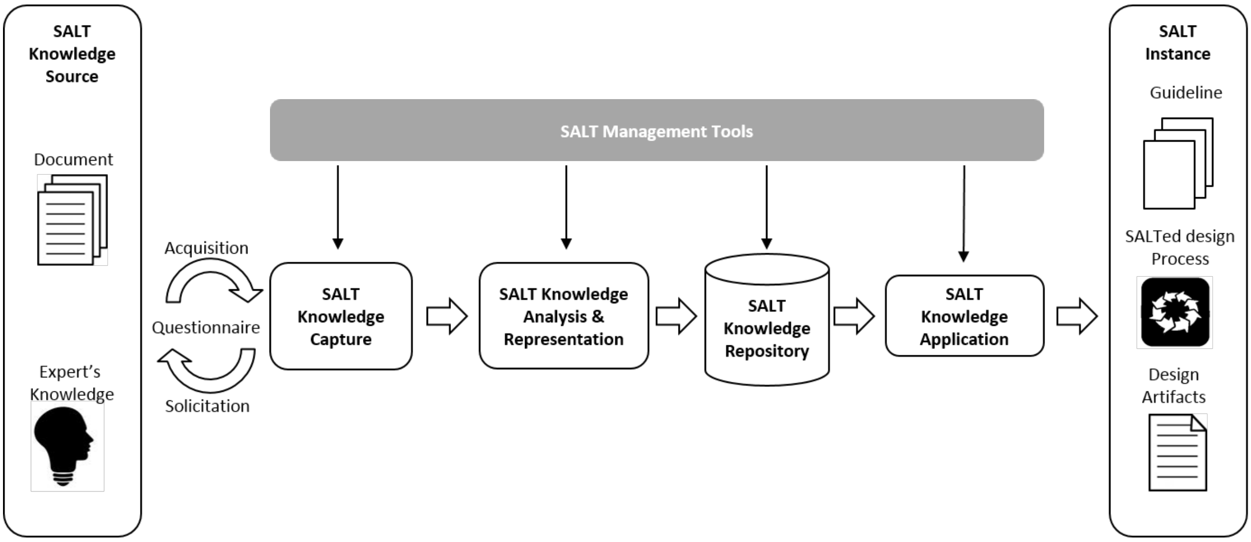


Figure 5. The SALT Framework in a nutshell

According to the principles defined in Figure 5, the project considers that the SALT Process is finally decomposed of three independent processes, namely:

- Information acquisition process (concerns to the SALT knowledge building process).
- Representation process (concerns to the SALT knowledge building process).
- System design process (concerns to the SALT knowledge using process).

For all of these elements, defining a process consists in decomposing all tasks that must be performed and pointing out the stakeholders and their respective contributions.

4.1.2 Overview of existing process modelling

In the context of PARIS, model driven engineering techniques are used. The state of the art proposes a number of solutions for modelling processes such as SPEM, OPF, UMA, or RUP. All of them are briefly presented in the next paragraphs.

SPEM (Software and systems Process Engineering Metamodel) is standardized by the OMG (Object Management Group which leads also other standards like UML, SysML or CORBA). As argued in its name, SPEM was initially designed as a standalone metamodel (see Figure 6). After the emergence of UML, they decided to instantiate SPEM through a UML profile. A UML profile corresponds to the extension mechanism of UML in order to take into account domain specific representations or constraints. SPEM is focused on the basic elements required for specifying a process. An issue can be raised regarding the usage of SPEM in PARIS. Indeed, SPEM cannot cover all independent processes composing the PARIS SALT Compliant Process (e.g., how the project management can be impacted by the SALT Framework).

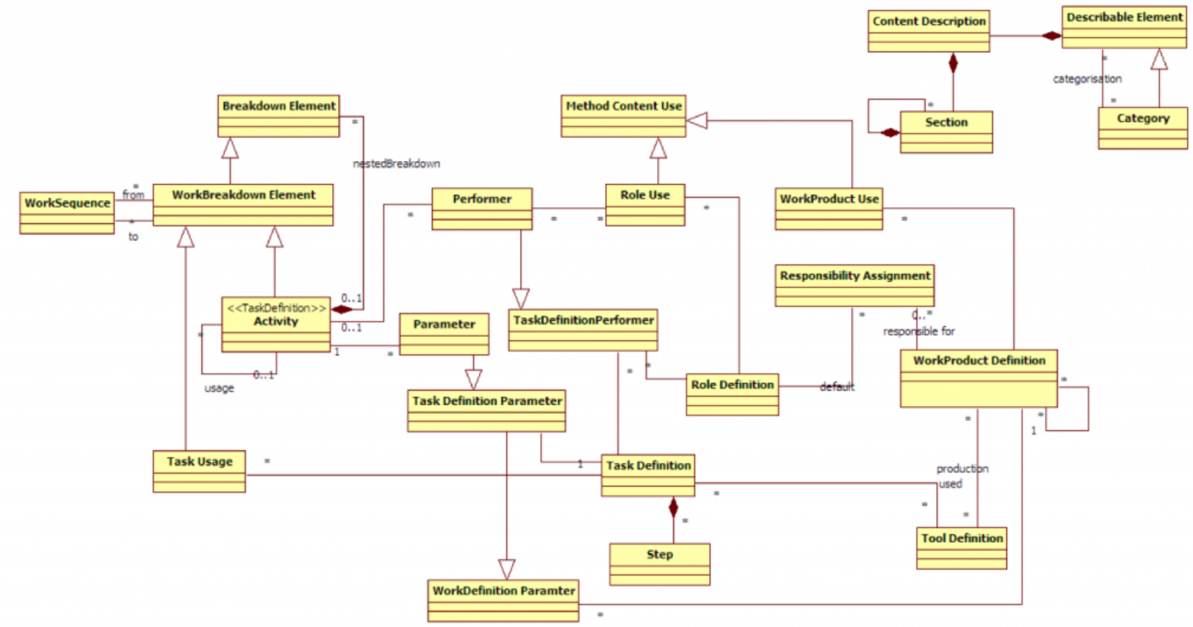


Figure 6. Overview of the SPEM Metamodel

OPF (OPEN Process Framework), initiated by OPEN, covers more aspects than the process modelling. OPF is a framework that defines (i) a process, (ii) a repository where the metamodel elements are stored and (iii) guidelines (see Figure 7). Even if the principles can be reused in PARIS, this framework is not maintained anymore.

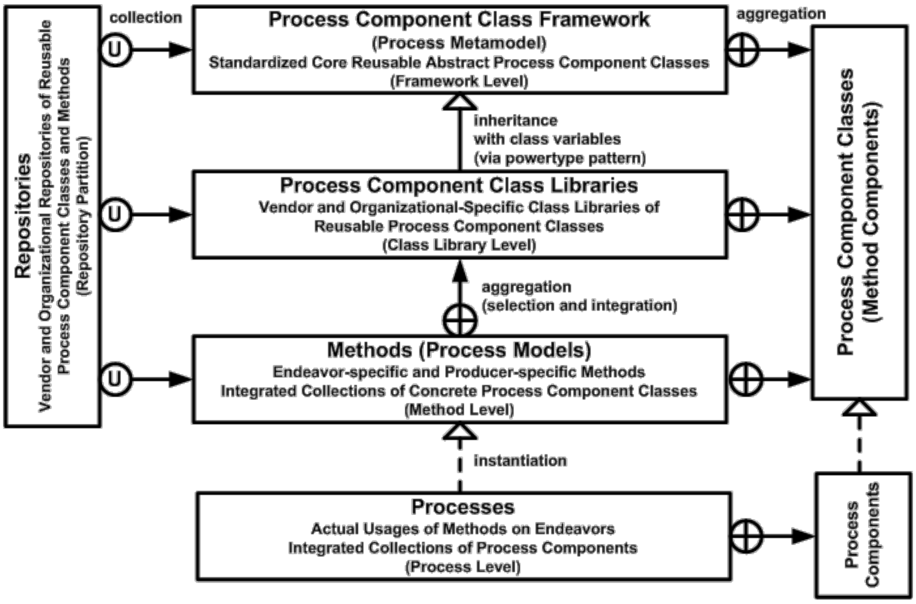

Repositories
Vendor and Organizational Repositories of Reusable Process Component Classes and Methods
(Repository Partition)
collection
U
Process Component Class Framework
(Process Metamodel)
Standardized Core Reusable Abstract Process Component Classes
(Framework Level)
aggregation
inheritance
with class variables
(via powertype pattern)
Process Component Class Libraries
Vendor and Organizational-Specific Class Libraries of Reusable Process Component Classes
(Class Library Level)
aggregation
(selection and integration)
Methods (Process Models)
Endeavor-specific and Producer-specific Methods
Integrated Collections of Concrete Process Component Classes
(Method Level)
instantiation
Processes
Actual Usages of Methods on Endeavors
Integrated Collections of Process Components
(Process Level)
Process Component Classes
(Method Components)
Process Components

Figure 7. Overview of the OPF Framework

UMA (Unified Method Architecture), as UML objective, aims at unifying existing methods for process specification. This solution was proposed by IBM and is used for EPF composer (integrated in the Eclipse platform), which is an open source tool platform for publishing, tailoring, exchanging methods and processes. The metamodel is shown in Figure 8.

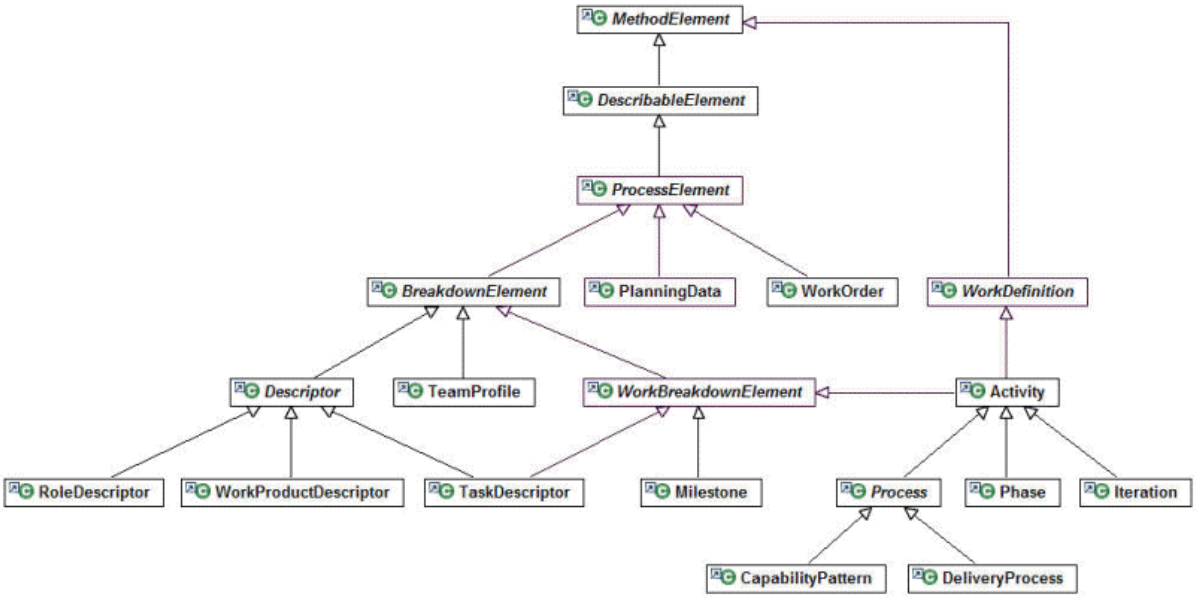


Figure 8. Overview of the UMA Metamodel

RUP metamodel, namely IBM Rational Unified Process, is designed for modelling processes. RUP has defined a UML profile like SPEM but also a specific language called PML (Process Modelling Language). RUP was very popular at the beginning of MDE (Model Driven Engineering), however this solution is less used by the community. Figure 9 highlights the main elements composing the RUP metamodel.

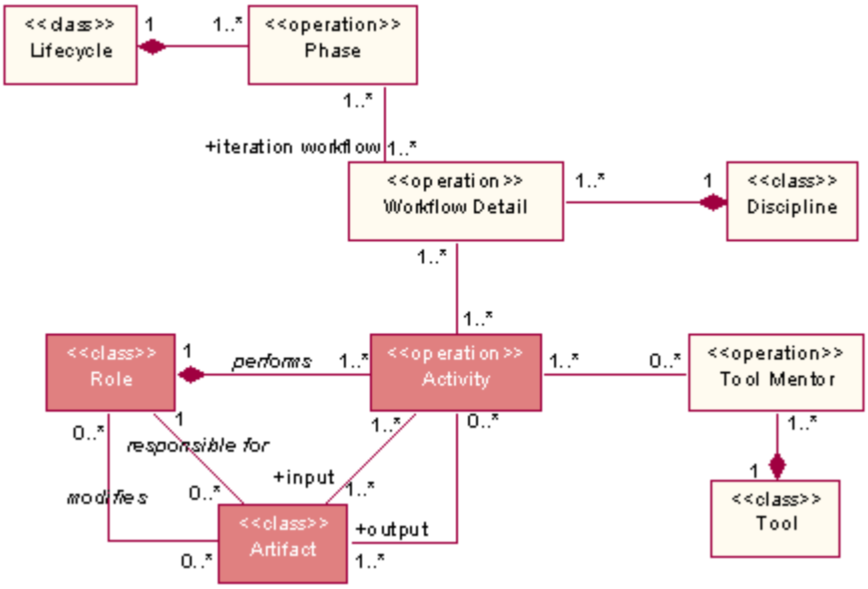


Figure 9. Overview of the RUP Metamodel (core only)

4.1.3 Towards the SALT compliant process model

In Section 4.1.2, some process modelling metamodels have been presented. We can note that in general, the main artefacts for defining a process are the following:

- Phase is a basic element composing a process. It represents a significant period that is characterized by milestones or checkpoints, and the delivery of a set of deliverables.
- Activity or Task, which is a sub-element composing a phase.
- Role, which corresponds to the performer of an activity. For instance, in PARIS, we have defined several roles e.g., SALT expert, DPA, System designer, Surveillance system Operator.
- Input and output work products, which are the artefacts (i) used at the beginning of an activity and (ii) produced at the end of an activity.
- Tools used during an activity. For instance, the SALT Framework Management Tool is part of the toolset used in the SALT Compliant Process.

The next sections will present how the PARIS project defines the SALT Compliant Process, which will make use of a set of domain and system models, already described in the deliverable D2.1.

4.2 Process fundamentals

This section describes the key elements that take part within a SALTed design process, considering the input elements as much as the output ones.

4.2.1 Modelling artefacts

Modelling artefacts comprise a set of models conceived to help in the development of surveillance systems during the application of a SALTed design process. These models are entities that capture common characteristics at the domain or system level, and hence some of them could match the system under development. We distinguish several kinds of modelling artefacts:

- **Domain Privacy Model (DPM):** this model shows problems, requirements and solutions that are bounded to a determined domain. It provides general information concerning a specific domain such as a school, a supermarket, an airport, a hospital, etc. This information refers to possible general requirements or problems of a given domain, together with the solutions that could be used to solve such problems or to fulfil the domain requirements.
- **System Privacy Model (SPM)**: we can see this model as a final model, which instantiates and represents all characteristics and aspects of a given surveillance system.

For a better functionality and understanding, we observe other three modelling artefacts that represent different states of an uncompleted SPM:

- **Requirements model:** it covers possible problems and requirements that may appear during the development of a surveillance system according to technological, social, legal and/or ethical constraints.
- **Solutions model:** it represents the solutions for the requirements and constraints shown in the requirements model.

- **Deployment model**: this model enhances the solutions model by taking into account the characteristics of the deployment scenario.

We can see these three models as a natural evolution within an SPM: we first have the system requirements, then we have the solution, and finally we integrate the scenario properties. However, it is important to remark that these three models are traceable, which means that they are not cascaded, with one model following another in a sequential way. Instead of that, we can move back and forth among the three types of model. E.g., we may be working with a deployment model, then find out new system requirements, and go back to the requirements model again.

Additionally, these models are not pure until the end of the process, which means that they can be mixed. E.g., at some point, we can find a solution for some requirements, but not all of them, leading to a model containing not only requirements but also solutions.

For the sake of clarity, we remark the difference between a DPM and a requirements model or a solutions model. A DPM is a model for general scenarios and that can be used during the SALTed design process. On the other side, a requirements model and a solutions model are models the system designer uses to develop an actual surveillance system. Hence, a given project may match a scenario represented in a DPM, which is stored in a repository, and in this case, some requirements and solutions of the current surveillance system will coincide with the general ones represented in the DPM.

4.2.2 Process lifecycle

Figure 10 shows the main phases for creating a surveillance system following the SALTed design process. The lifecycle starts when a stakeholder defines the (informal) system specifications, together with the context where it is going to be used. Using the SALTed design process, the system designer creates a SALT compliant surveillance system based on that information, in addition to legal, ethical, social and technological information related to the context and provided by the SALT instances.

Once the stakeholder specifies the system specifications and the context, the informal system requirements and the technological components are settled through a negotiation process, and corresponding DPMs are selected. Then, the surveillance system goes through four different models within the process lifecycle. These models are not cascaded, i. e., they do not go one after the other, but we can move forward and backward among them.

The requirements model is made from the informal system requirements, the technological components and the DPMs, and it covers the problem presented in the system. This first model, requirements model, does not take into account any legal, social, ethical or technological considerations.

The design model, which is a SALT compliant model, is made from the requirements model considering SFIs (SALT Framework Instances containing social/ethical/legal/technological information provided by experts) and SFPIs (SALT Framework Project Instances containing constraints and agreements specific to a given system and provided by stakeholders of that particular system). It provides proper solutions for the problems presented in the requirements model. Once the design model is made, it could be necessary to modify the requirements model in order to properly match with the current design model.

The deployed model is the final version obtained from the design model, also taking into account the deployment scenario characteristics. In this manner, it considers SFIs and SFPIs related to the scenario and the specification of the system. With the aim of adapting the surveillance system to the real scenario, it is possible to modify the previous design model to fit with the deployed model.

The Verified SPM (System Privacy Model). This is the deployed model after a verification of its SALT compliance by an external entity. This external verification can be a certification by a SALT authority or an informal verification made by an authorized entity.

The SALT framework management tool aids the designer during the SALTed design process. A SALT compliant SPM may become invalid when a change of context or specifications occurs. In such a case, the SALTed design process starts again.

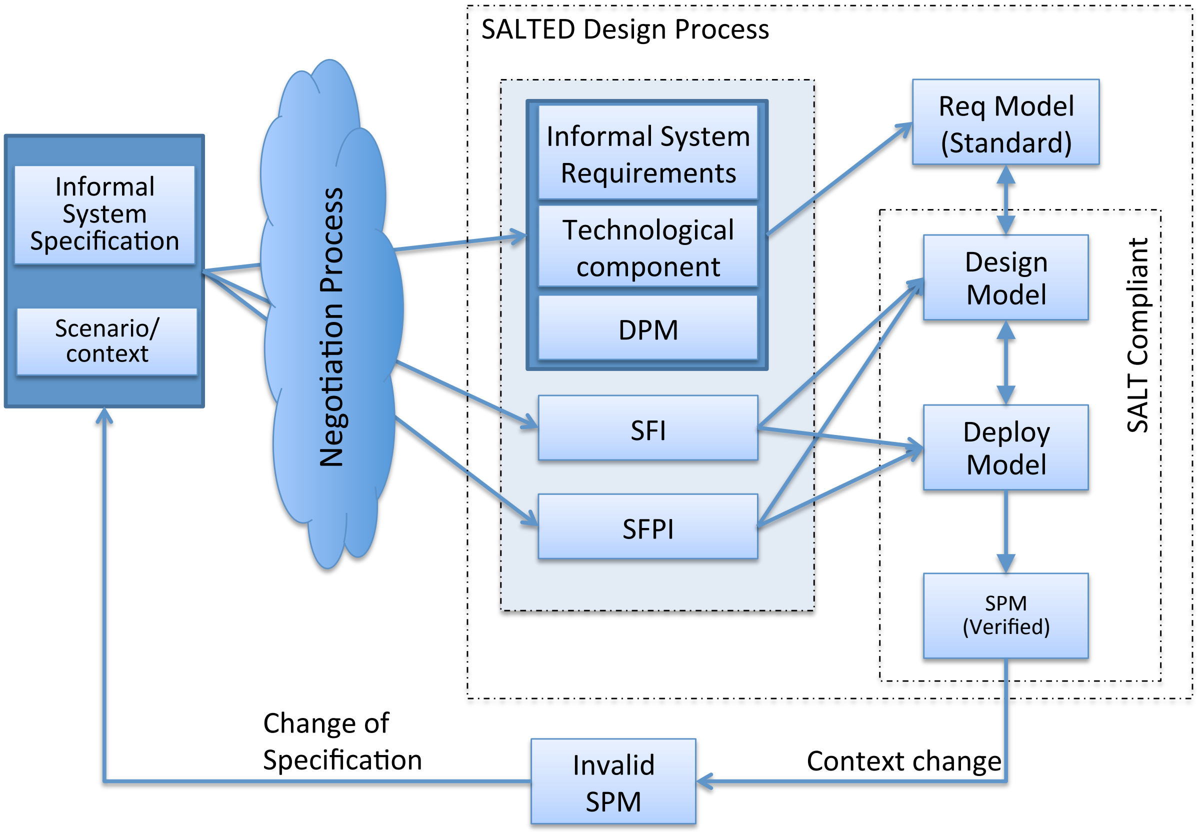


Figure 10. Process lifecycle

4.2.3 Process work products

The SALTed design process will produce four different types of work products, as it is depicted in Section 4.3:

- **SFI:** SALT Framework Instances containing privacy and accountability aware information regarding to surveillance systems.

- **System requirements design:** an initial system design following current industry workouts. It includes the system requirements.
- **SALTed system design:** a fully SALT compliant system design, which guarantees that this system fulfils a certain level of privacy and accountability. It includes the solutions used to accomplish with the requirements listed in the previous design.
- **Deployed system design:** this is the realization design, i. e. the final version that takes into account the deployment scenario characteristics.

Selected SFIs will be available for each surveillance system, depending on each particular system needs and requirements. The SALT framework management tool (SFMT) is in charge of providing such SFIs according to a set of filters given by the system designer, and they will contribute with valuable information. Following the SALTed design process, the system designer will use this information to provide a SALT compliant system. But before obtaining the final design, ready to be implemented, two more previous versions are also produced: an initial system requirements design and the SALTed system design.

The system requirements design is the initial and most simple design. It is a typical design according to nowadays industry standards and workouts, which the system designer creates according to the initial system specifications and with the help of a design tool. This design does not guarantee the privacy and accountability requisites needed for being SALT compliant. Figure 11 shows an example for this type of design.

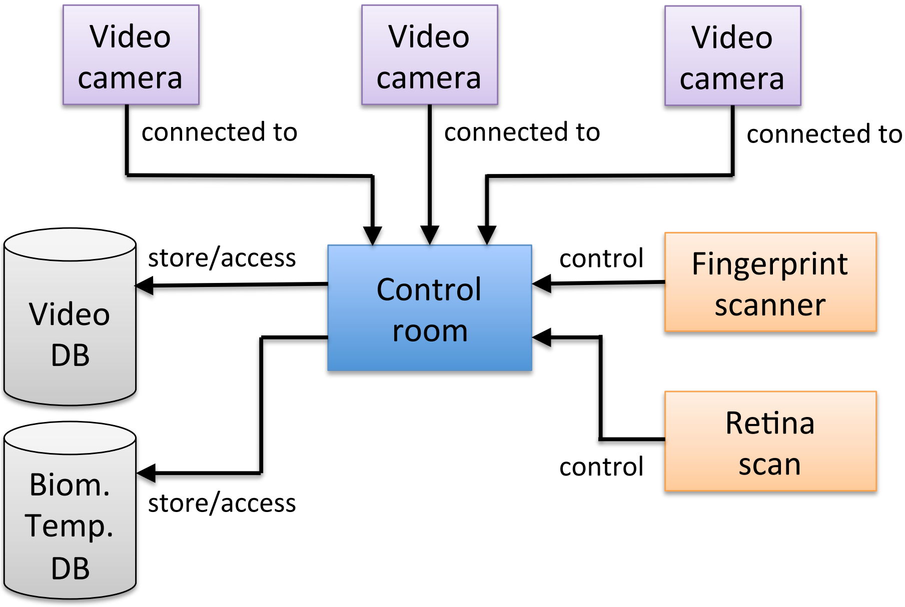


Figure 11. Example of system requirements design

In a next stage, and following the SALTed design process, privacy and accountability related information is incorporated to the system. This task is performed by taking into account selected SFIs retrieved from the SALT repositories according to a set of filters given by the system designer. These filters will vary depending on the current surveillance system under development. In this way, the initial system design previously produced is converted into a SALT compliance system design (also called a SALTed system design), which incorporates the solutions chosen to fulfil the system requirements. Figure 12 shows an example of a SALTed system design.

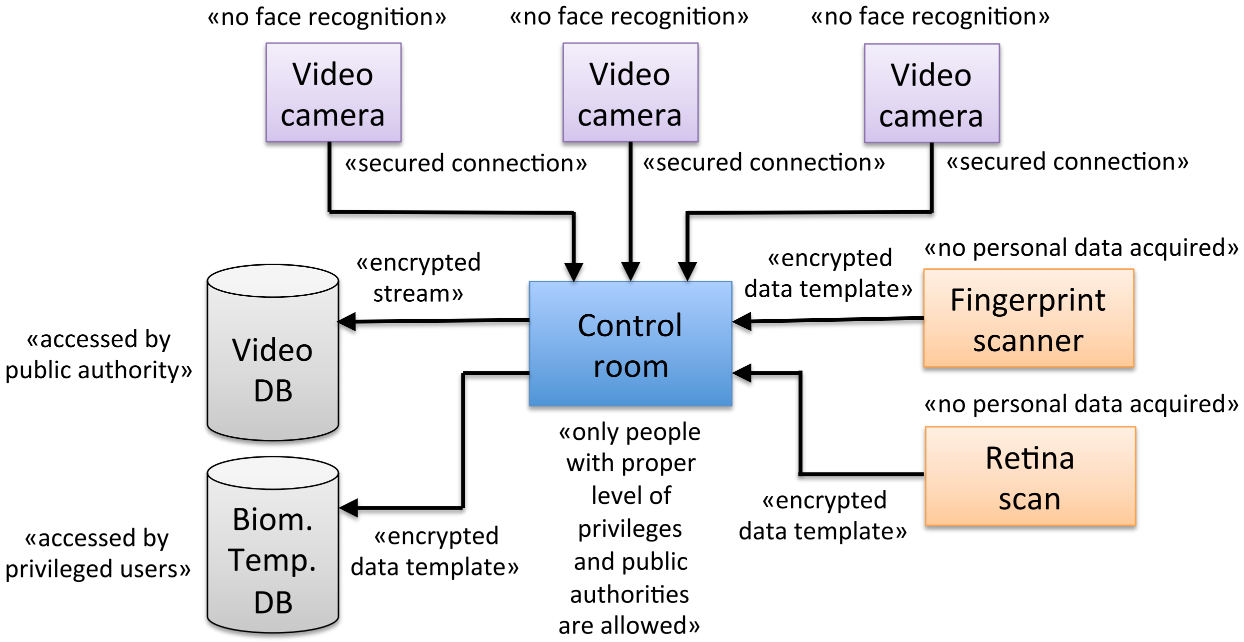

«no face recognition»
Video camera
«no face recognition»
Video camera
«no face recognition»
Video camera
«secured connection»
«secured connection»
«secured connection»
«encrypted stream»
«encrypted data template»
«no personal data acquired»
Fingerprint scanner
«accessed by public authority»
Video DB
Control room
«no personal data acquired»
Retina scan
«accessed by privileged users»
Biom. Temp. DB
«encrypted data template»
«only people with proper level of privileges and public authorities are allowed»
«encrypted data template»

Figure 12. Example of SALTed system design

Finally, all characteristics specific to the real scenario where the system is going to be deployed are also considered, such as camera positions, doors control access, light and humidity conditions, type of subjects under surveillance, etc. Therefore, the SALTed system design is converted into the so called deployed system design, which is the design ready for its actual implementation (including the actual realization for the solutions listed in the previous design). Figure 13 shows an example of a deployed system design.

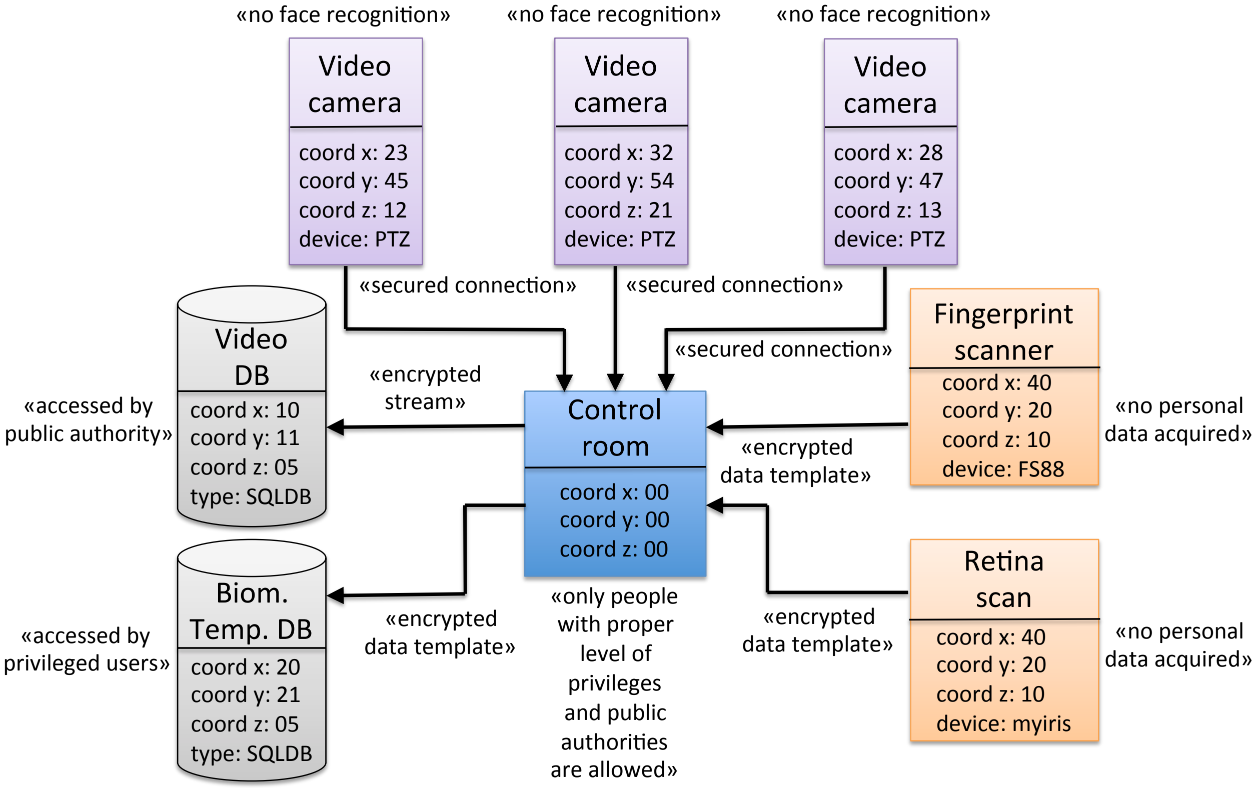


Figure 13. Example of deployed system design

4.3 Detailed process description

This section describes the whole SALTed design process, which is the reference process in order to create a SALT compliant surveillance system, starting from the SALT instances creation and selection, and ending with the surveillance system deployment. For a better clarity and understanding, the SALTed design process is presented through the different use cases, according to the actors related to the system. This approach helps not only to visualize the system architecture, but also its main functionalities.

To describe use cases, we use the formatting guides and rules originally depicted by Alistair Cockburn[3].

Use case 1: Modify SALT template.

- **Context of use:** modification of the SALT template, i. e. the formatting structure used to store SALT instances (privacy and accountability aware information) into a given repository. At this point of the project, this structure is not yet defined (it could be an xml-schema, a JSON-schema, a wiki structure, etc.).
- **Scope:** component.
- **Level:** sub-function.
- **Primary actor:** SALT authority. At the beginning, the PARIS consortium will adopt this role.
- **Stakeholders and interests:** project stakeholders and standards bodies (persons who have the sufficient knowledge to create an SFI) will be the entities mainly interested, since they will access the resulting SALT template to create or extend SFIs (SALT Framework Instances) following the proper representation format.
- **Precondition:** a SALT template already exists and it does not correctly address the information contained within a SALT instance.
- **Minimal guarantees:** the SALT template must not be ambiguous and avoid the inclusion of redundant information.
- **Success guarantees:** a precise format that allows an accurate representation and storage of SALT instances.
- **Trigger:** appearance of SALT instances whose information does not properly fit into the current SALT template.

Figure 14 shows the corresponding UML use case diagram.

Since we need the flexibility of being able to dynamically adapt the SALT framework to future changes and evolutions, we have to provide the engineering process with the proper mechanisms in order to fulfil this requirement. Therefore, we have to consider that not only the actual information contained within the SALT instances (privacy and accountability concerns) may change, but also the structure/format that the SALT framework needs to represent this information. For this reason, we have included the role of the SALT authority as the entity that has the privileges to modify the structure/format for the information representation, i.e. the SALT template.

[3] Alistair Cockburn, *Writing effective use cases* (Addison-Wesley, 2001).

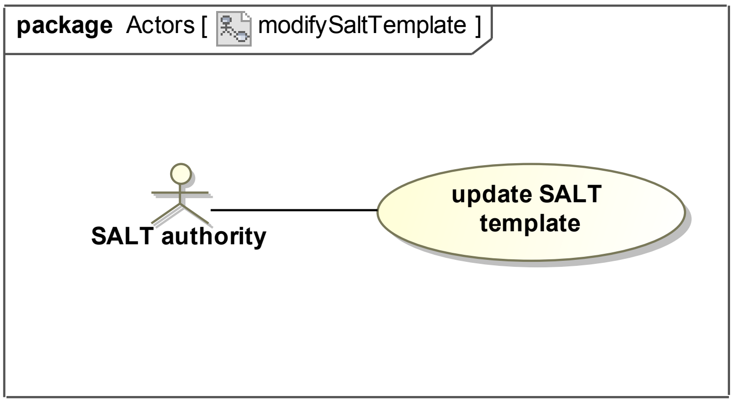


Figure 14. SALT template modification use case diagram

Use case 2: Create a SALT framework instance (SFI).

- **Context of use:** creation and storage (within a repository) of a standard SALT framework instance.
- **Scope:** system.
- **Level:** user.
- **Primary actor:** standards body. It could also be called "standards committee". With this name we refer to persons who have the sufficient knowledge to create an SFI. They can be considered as experts regarding to social, ethical, legal and technological areas.
- **Stakeholders and interests:** system designers in first place and project stakeholders in second place. The former because they will directly use SFIs for the system design, and the latter because they may take into account SFIs when defining the system specifications.
- **Precondition:** a SALT template exists and the SALT framework management tool (SFMT) is operational.
- **Minimal guarantees:** the information within the SALT instance fulfils with the current SALT template.
- **Success guarantees:** new social, ethical, legal or technological information related to surveillance systems is added to the SALT repositories.
- **Trigger:** a standards body has the will (and the knowledge) to create a new SFI.

Use case 3: Extend a SALT framework instance.

- **Context of use:** extension and storage (within a repository) of a standard SALT framework instance (SFI).
- **Scope:** system.
- **Level:** user.
- **Primary actor:** standards body. It could also be called "standards committee". With this name we refer to persons who have the sufficient knowledge to create an SFI. They can be considered as experts regarding to social, ethical, legal and technological areas.
- **Stakeholders and interests:** system designers in first place and project stakeholders in second place. The former because they will directly use SFIs for the system design, and the latter because they may take into account SFIs when defining the system specifications.

- **Precondition:** an SFI and a SALT template exist. The SALT framework management tool (SFMT) is operational.
- **Minimal guarantees:** the information within the extended SALT instance fulfils with the current SALT template.
- **Success guarantees:** an already existing SFI is extended with updated information (from the implementation point of view, this may mean the creation of a new SFI, which includes the data of the SFI meant to be extended).
- **Trigger:** a standards body has the will (and the knowledge) to update an existing SFI.

Figure 15 shows the UML use case diagram corresponding to these two use cases.

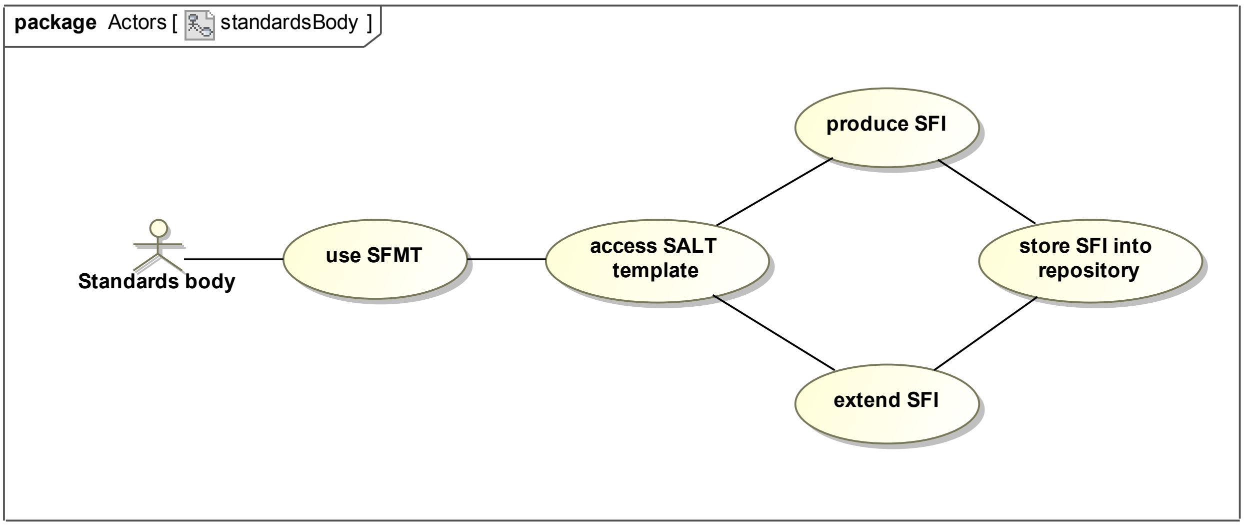


Figure 15. SFI creation and extension use case diagram

A standards body comprises the persons with the appropriate knowledge regarding to the four categories covered by the PARIS project: social, ethical, legal and technological. They use the SALT framework management tool to produce standard SALT instances (SFIs), which, of course, will be created (and extended) according to the SALT template. These SFIs are stored into the SALT repositories, hence they can be accessed later on, whenever it is required for a given surveillance system.

Use case 4: Create a SALT framework project instance (SFPI).

- **Context of use:** creation and storage (within a repository) of a SALT framework project instance.
- **Scope:** system.
- **Level:** user.
- **Primary actor:** project stakeholder.
- **Stakeholders and interests:** system designers, since they will use SFPIs for the system design.
- **Precondition:** at least, a minimum set of system specifications is clear. A SALT template does exist. The SFMT is operational.
- **Minimal guarantees:** the information within the SFPI fulfils with the current SALT template. The SFPI may also refer to some already existing SFIs.

- **Success guarantees:** an SFPI, which expresses system specifications, is created and stored within the SALT repositories.
- **Trigger:** project stakeholders want to define/implement a new surveillance system.

Figure 16 shows the corresponding UML use case diagram.

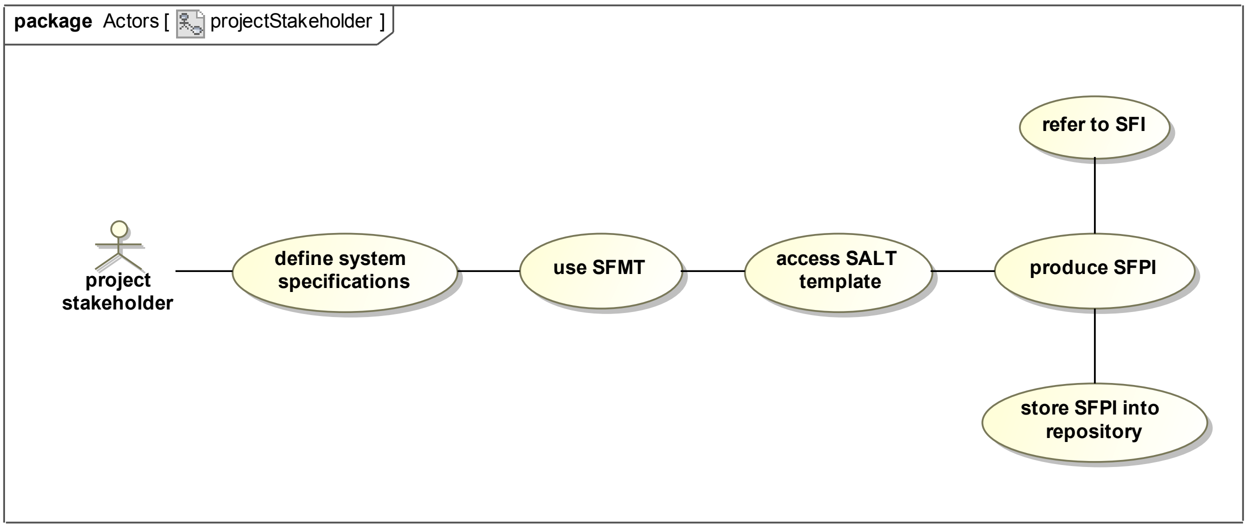

package Actors [projectStakeholder]
project stakeholder
define system specifications
use SFMT
access SALT template
produce SFPI
refer to SFI
store SFPI into repository

Figure 16. SFPI creation use case diagram

Project stakeholders make use of two different tools: the specification tool and the SALT framework management tool. The specification tool helps them to define the requirements for a given system, together with the compliance information in case they want the system to adhere to some specific regulations or standards already defined.

They use the SALT framework management tool for two different tasks. The first one deals with the creation of project specific SALT instances, which are called SFPIs. These instances express system requirements and agreements directly addressed by stakeholders, who have to follow a SALT template in order to provide a valid SFPI. The SALT template defines the structure/format that a SALT instance must have for computer-assisted tools being able to operate with them (it can be an xml-schema, a JSON-schema, a wiki structure, etc.). Once the SFPIs, which are valid for a concrete system, are created, they are stored into the SALT repositories. With this action we make them available for future systems that may need similar requirements or may adhere to similar initial agreements. An SFPI may refer to a standard SFI in order to avoid the inclusion of redundant information that it is already contained within another SFI.

SALT repositories are the place where SALT instances are stored. They can be seen as a database, though we do not consider here the type of implementation (centralized, distributed...), nor even if there are several repositories or just one.

The second SFMT usage carried out by project stakeholders is more direct. In this case they use the SFMT to access the SALT repositories and select those standard instances (SFIs) that may apply to the surveillance system we are working with.

Use case 5: Provide technological components capabilities.

- **Context of use:** delivery of technical components capabilities.
- **Scope:** component.
- **Level:** sub-function.
- **Primary actor:** technology provider.
- **Stakeholders and interests:** system designers. They will take into account components specifications for the system design.
- **Precondition:** at least one technological component must be available.
- **Minimal guarantees:** DPMs (Domain Privacy Models) refer to these components specifications.
- **Success guarantees:** technological components capabilities are provided to the design tool.
- **Trigger:** technology providers have components, required by the current surveillance system, whose capabilities have not yet been delivered.

Figure 17 shows the corresponding UML use case diagram.

By technology provider we mean the entity (person or company) that provides the capabilities of the different components that can be used in a surveillance system: video cameras, fingerprint scanners, retina scan devices, etc.

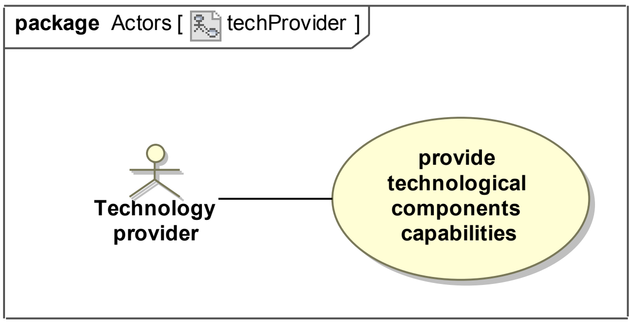

package Actors [techProvider]
Technology provider
provide technological components capabilities

Figure 17. Technological components capabilities delivery use case diagram

Use case 6: Design surveillance system.

- **Context of use:** design of a whole surveillance system.
- **Scope:** system.
- **Level:** user.
- **Primary actor:** system designer.
- **Stakeholders and interests:** project stakeholders. Any person who will directly interact with the final system design (which will take into account the specific deployment scenario characteristics).
- **Precondition:** SFPIs and SFIs related to the surveillance system must be available.
- **Minimal guarantees:** an early system design, valid, but not SALT compliant, is also provided.
- **Success guarantees:** a complete SALT compliant system design specifically designed for the current deployment scenario is provided.

- **Trigger:** system designers receive the appropriate documentation to design/implement a given surveillance system.

Figure 18 shows the corresponding UML use case diagram.

The system designer is the person who provides the design of the surveillance system to be deployed. To achieve this task the designer retrieves the system requirements and the compliance information previously generated by the stakeholders, hence they know what requirements the surveillance system has to fulfil.

By using the design tool, the system designer gets access to the SFPIs and SFIs generated by the stakeholders. Besides, due to their knowledge and experience regarding to surveillance systems, they can also ask the SALT repositories for other SFIs that can be helpful for the system. How the design tool is connected to SALT instances and repositories may differ depending on the actual scenario, but it is fair to think that they will normally be accessed through the Internet.

The design tool also has access to the DPMs (Domain Privacy Models) repository and the components specifications (provided by technology providers). DPMs provide system information (one or several models) according to the current domain. Then, in first place, a general system design is developed, which will convert to a SALT compliant system after taking into account the corresponding SFIs. Finally, the design tool will be able to provide a SALT compliant deployed surveillance system for a specific scenario (which will use video surveillance technology, biometrics devices, or both).

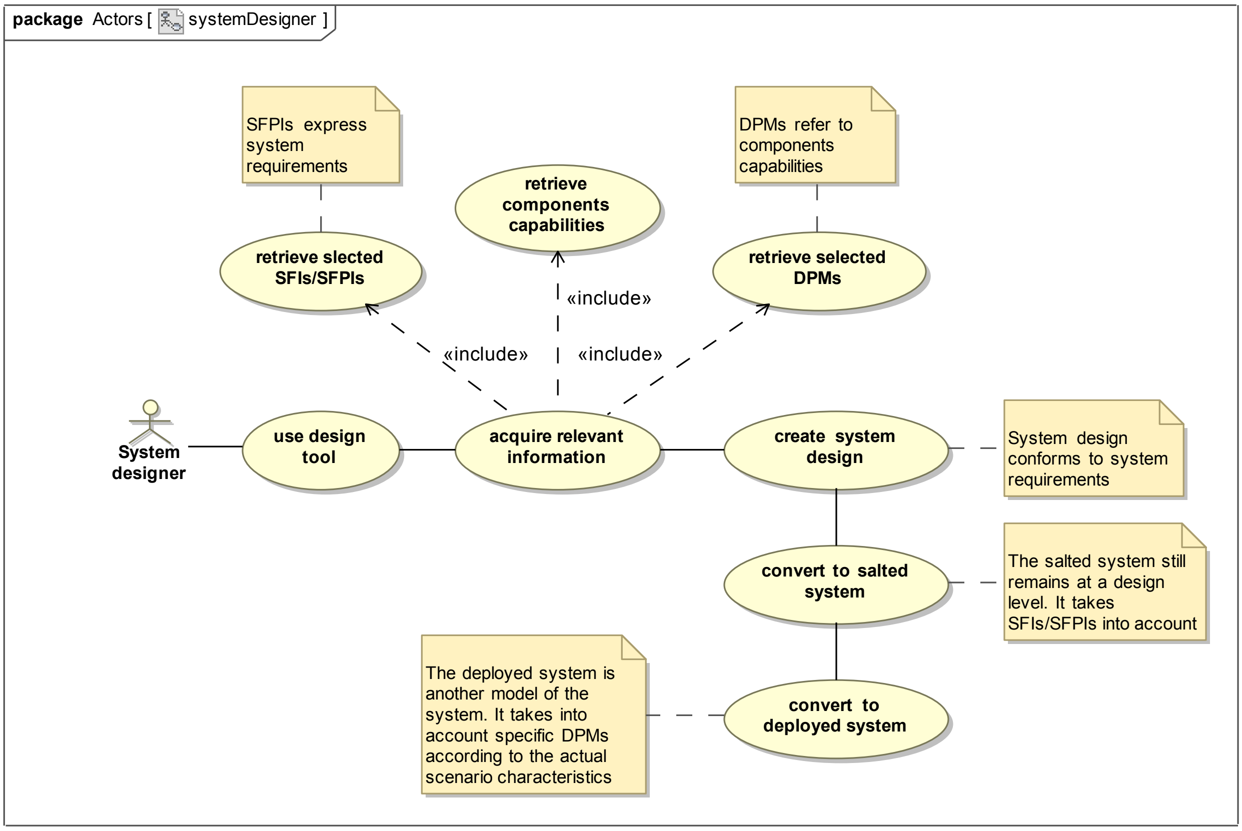

package Actors [systemDesigner]
SFPIs express system requirements
retrieve components capabilities
DPMs refer to components capabilities
retrieve slected SFIs/SFPIs
retrieve selected DPMs
«include»
«include»
«include»
System designer
use design tool
acquire relevant information
create system design
System design conforms to system requirements
convert to salted system
The salted system still remains at a design level. It takes SFIs/SFPIs into account
The deployed system is another model of the system. It takes into account specific DPMs according to the actual scenario characteristics
convert to deployed system

Figure 18. System design use case diagram

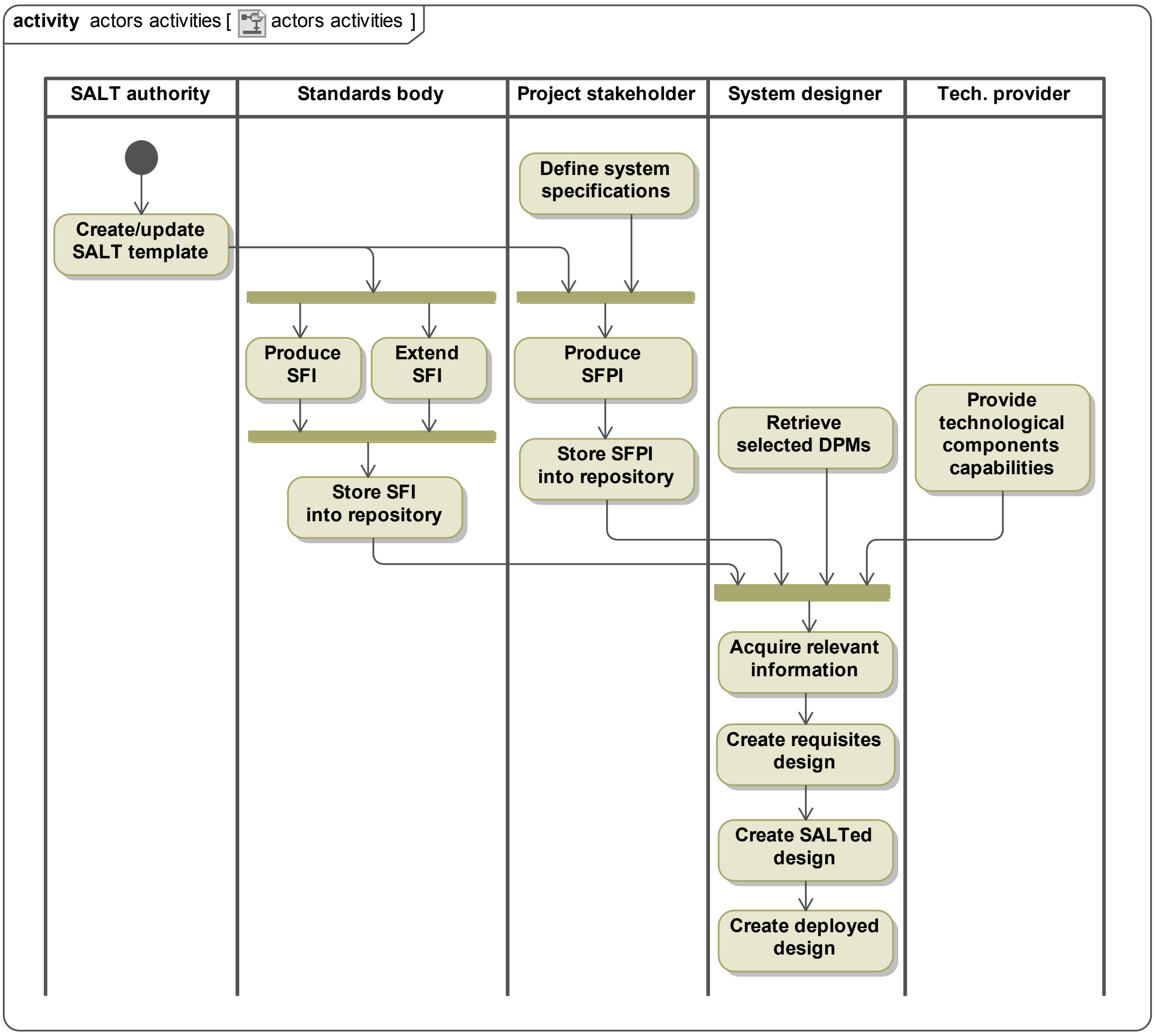

activity actors activities [actors activities]
SALT authority
Standards body
Project stakeholder
System designer
Tech. provider
Create/update SALT template
Define system specifications
Produce SFI
Extend SFI
Produce SFPI
Store SFI into repository
Store SFPI into repository
Retrieve selected DPMs
Provide technological components capabilities
Acquire relevant information
Create requisites design
Create SALTed design
Create deployed design

Figure 19. Activity diagram for a SALTed design process

Once all these use cases have been detailed, we propose an activity diagram showing the interactions between actors and their actions within the SALTed design process, which can be seen in Figure 19. This diagram depicts all actors and dependencies among them. In general the proposed process will fulfil the following steps:

1. A SALT template is created and kept up to date.
2. Information is provided.
    a. SALT repository is populated.
        i. Project stakeholders define system specifications and store SFPIs.
        ii. Standards bodies create, update and store SFIs.
    b. Technology providers give and store technological components capabilities.
3. System designer acquires information relevant to the system.
    a. DPMs.
    b. Components specifications.
    c. SFPIs and SFIs.
4. System designer creates a system design.
    a. In first place a requirements system design is produced.

b. Taking into account SALT information (in the form of SFIs), the requirements design is converted to a SALT compliant design.
c. A final deployed system design is produced according to components capabilities and scenario characteristics.

4.3.1 Process phases

We can distinguish three different phases within a general SALT compliant process:

1. Project definition phase.
2. Project design phase.
3. Certification, deployment and operation phase.

Project definition phase: Figure 20 depicts a general overview of the project definition phase.

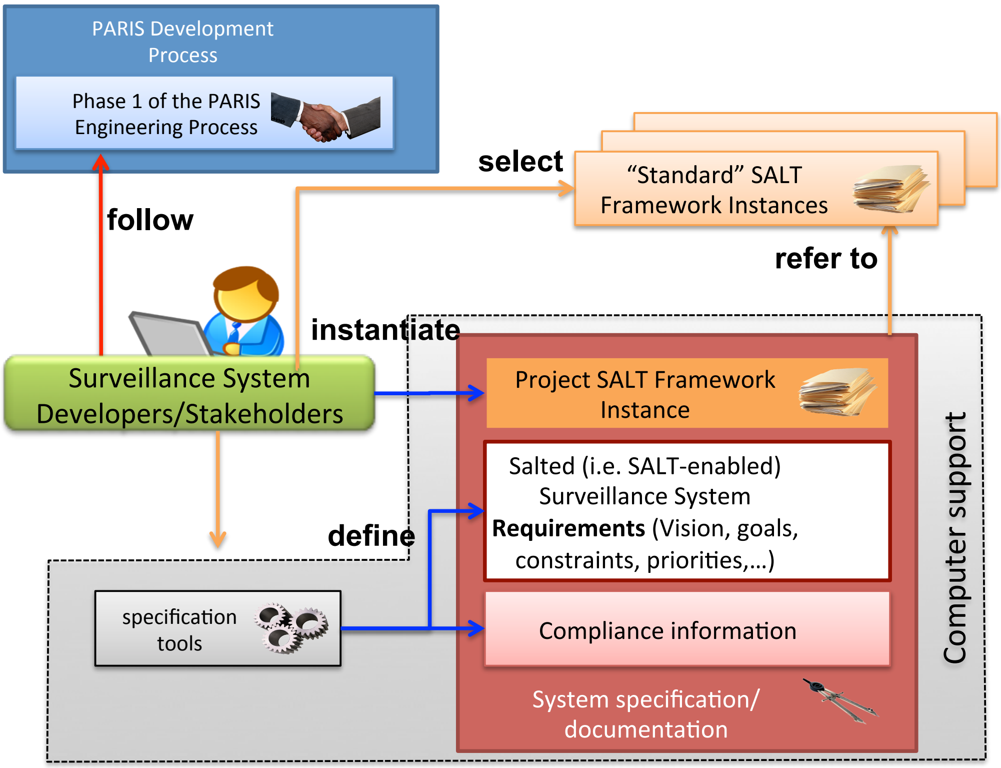

PARIS Development Process
Phase 1 of the PARIS Engineering Process
select
"Standard" SALT Framework Instances
follow
refer to
instantiate
Surveillance System Developers/Stakeholders
Project SALT Framework Instance
Salted (i.e. SALT-enabled) Surveillance System **Requirements** (Vision, goals, constraints, priorities,...)
define
specification tools
Compliance information
System specification/ documentation
Computer support

Figure 20. Project definition phase

In this phase, the surveillance system stakeholders define the specifications and the agreements the system must fulfil, i.e. the system requirements and compliance information (regulations and/or standards to follow). To accomplish this task they use specification tools.

These agreements and requirements, which are specific for the current project, are provided with a representation compatible with standard SALT framework instances, and hence they are called SALT framework project instances. This fact guarantees that they can be stored in repositories and accessed in the same way as the rest of instances, allowing future projects with similar requirements to reuse them.

Project design phase: Figure 21 depicts a general overview of the project design phase.

In this phase, the system designer/developer produces a SALT compliant system design. To achieve this task, he uses the system specification and documentation produced by the stakeholders in the previous phase, together with some other standard SALT instances that may be applicable to the system he is dealing with.

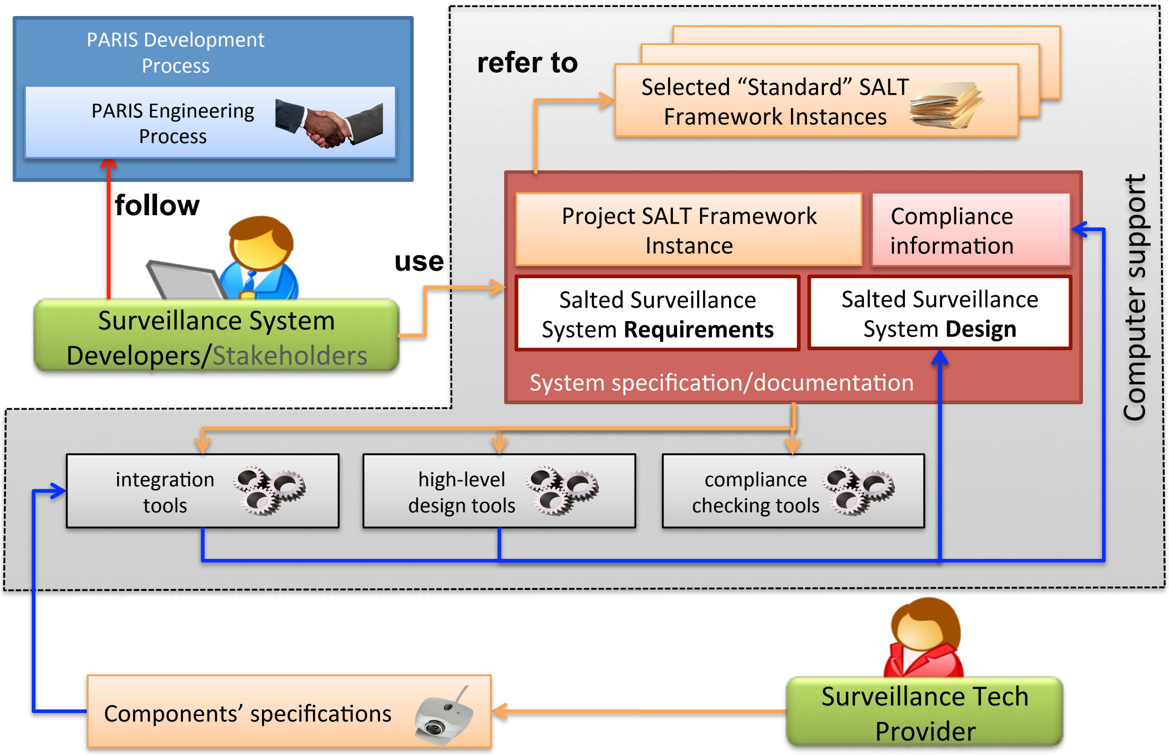

PARIS Development Process
PARIS Engineering Process
follow
use
refer to
Selected "Standard" SALT Framework Instances
Project SALT Framework Instance
Compliance information
Salted Surveillance System Requirements
Salted Surveillance System Design
System specification/documentation
Computer support
Surveillance System Developers/Stakeholders
integration tools
high-level design tools
compliance checking tools
Components' specifications
Surveillance Tech Provider

Figure 21. Project design phase

A set of tools helps the system designer in his work:

- Design tool: aids the designer in his task to produce a proper system design.
- Compliance checking tool: aids the designer to recognize whether the system fulfils, or not, with certain regulations and standards that are required for a proper system operation.
- Integration tool: it helps to integrate surveillance components and propagate their characteristics throughout the whole system. Specifications of such components are given by the surveillance technology providers involved in the project.

Certification, deployment and operation phase: Figure 22 depicts a general overview of this phase.

In this phase the system is deployed, but it also addresses the operation and certification tasks. The certification is performed by a certification authority, who will have to be designated. This entity requires the outputs of both previous phases, i.e. the SALT compliant system produced by the designer and the system specification and documentation produced by stakeholders, and then it generates a PARIS privacy certificate specific for the current system. This certificate is the element that guarantees the SALT compliance of the system (the system accomplishes a minimum of privacy requirements).

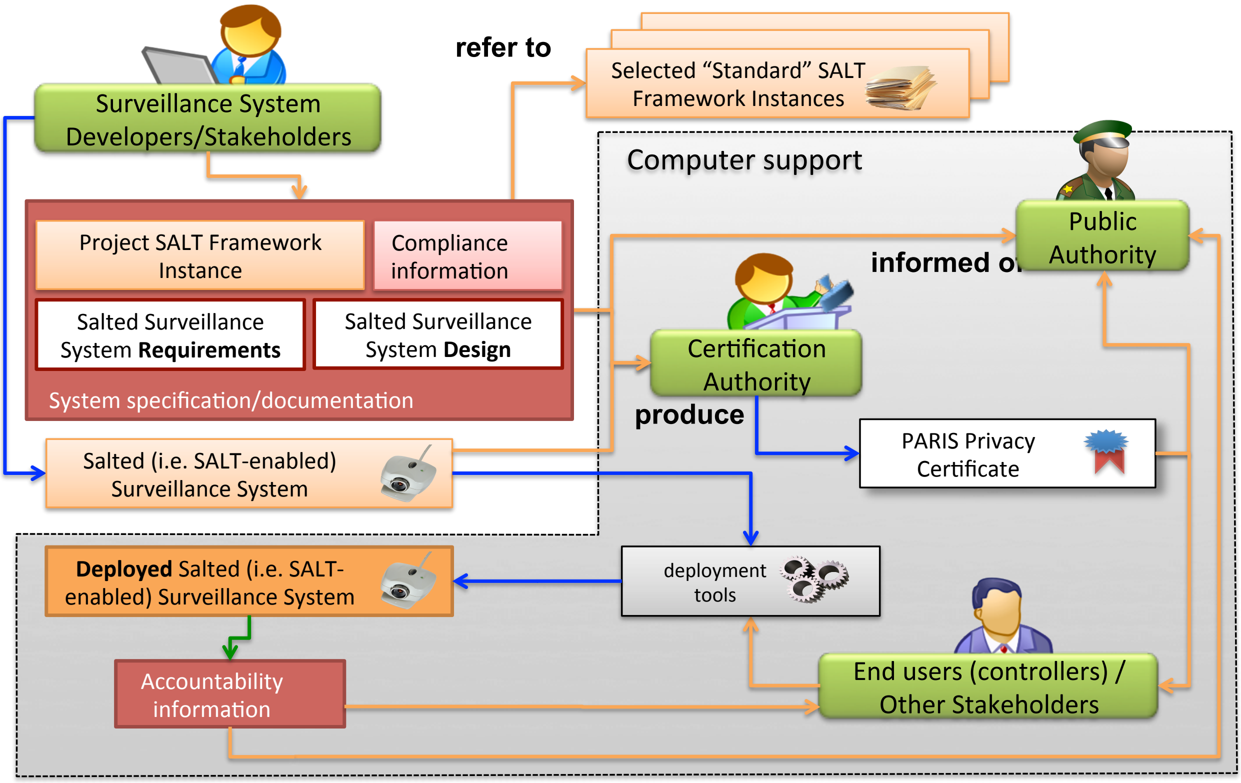

refer to
Selected "Standard" SALT Framework Instances
Surveillance System Developers/Stakeholders
Computer support
Project SALT Framework Instance
Compliance information
Salted Surveillance System Requirements
Salted Surveillance System Design
System specification/documentation
Public Authority
informed of
Certification Authority
produce
PARIS Privacy Certificate
Salted (i.e. SALT-enabled) Surveillance System
Deployed Salted (i.e. SALT-enabled) Surveillance System
deployment tools
Accountability information
End users (controllers) / Other Stakeholders

Figure 22. Certification, deployment and operation phase

The deployment is also based on the SALT compliant system produced in the previous phase, and it is aided by some sort of deployment tools managed by the system end users (system controllers) and stakeholders. They have to bear in mind the characteristics of the environment where the surveillance system is going to be used.

Once the surveillance system is deployed, it will generate the accountability information derived from its regular operation. This information is accessed by public authorities when needed. Public authorities may also access the certificate provided by the certification authority and the system specification, including SALT instances, which will help them to look in the appropriate direction when searching within the accountability information.

4.3.2 Automated support

In first place, we have to remark that tools and applications developed within the scope of the PARIS project are not intended to be "decision makers", i. e., they will not indicate the user what to do in order to obtain a SALT compliant surveillance system. Their mission is to help the user (stakeholder, system designer, operator, technology provider, etc.) by providing useful information and guidelines. At this point, there are some tasks that can be automated (computer support). Let us consider them from the type of user point of view.

Automated support for systems stakeholders:

Stakeholders can use tools that help them create the documentation containing the surveillance system specifications:

- They can use computer assistance to define the system requirements for it to be SALT compliance, such as: objectives, constraints, priorities, etc.
- Compliance information can also be created through a tool that may help to identify and represent existing regulations and standards, which may be required by the surveillance system to be fulfilled.
- Automated support is also provided in order to represent project specific SALT instances. In this way, stakeholders are aided in the task of providing these instances with an appropriate format (since they must be able to be stored within the SALT repositories).

Automated support for system developers/designers:

In this case, system designers can use a tool to access the surveillance system information that has been previously created by stakeholders. It also assists them to select and retrieve standard SALT instances from the repositories that may be applicable to the current surveillance system under development. With all this information at hand, system designers are ready to produce a SALT compliant system design.

Apart from designing the system, designers also have to check that their system design fulfils the compliance information (standards/regulations) and whether it is SALT compliant or not. This task may also be supported by some automated tool, enlightening the designer workload.

Finally, an automated mechanism could help to integrate technological components into the surveillance system and to propagate their characteristics/specifications.

Automated support for the certification authority:

The certification authority counts on automated support for issuing PARIS privacy certificates. To achieve this task, access to the surveillance system information (characteristics) must be granted, thus the authority has the required knowledge to ensure the SALT compliance of the system (with a certificate).

Automated support for end users (controllers):

These are the people who will operate the surveillance system in a regular basis. They will rely on an automated tool that will assist them with the system functionality. Besides, it will also guide them in the process of accessing the recorded accountability information (probably with some restrictions according to the user privileges), and in the interaction with public authorities when required (commonly regarding access to the accountability information too).

Automated support for public authorities:

Public authorities could probably be the users with the lowest knowledge of the surveillance system itself. Therefore, they undoubtedly require some guidance when dealing with the system. Besides, we have to keep in mind that they will mainly use the system regarding to accountability issues (typically because something went wrong or because they were asked to demonstrate compliance with the legal framework).

In this way, public authorities not only count on the system controllers help, but also with some kind of automated support for accessing accountability information (also according to some privileges). They may also need support for accessing the system documentation and specification, and the PARIS privacy certificate (in order to check the system SALT compliance).

5 Impact of SALT frameworks

This section analyses the impact of SALT frameworks on privacy and accountability aspects, which are the key concerns of the PARIS projects. It also studies what types of evidences can be produced by the SALT compliant process from the four PARIS categories perspective.

5.1 Evidences produced by the process

5.1.1 From a social and ethical perspective

The status of evidence provided by socio-contextual and ethical knowledge is not hard binding compliant rules of behavior, which is impossible to achieve. In other words, the socio-contextual and ethical expert is not in charge of telling what is good or bad, fair or unfair, reasonable or not.

Instead, the kind of evidence produced is process dependent, hence it relies on inputs from the system designer and relevant stakeholders. Thus, as for socio-contextual and ethical dimensions, experts invite the designer of a system to take into full consideration a variety of socio-contextual and ethical dimensions while designing the system. Depending on the specificities of the system, the designer is the best person to answer practical as well as ethical questions, and can justify his/her own choices according to some ethical insights.

Doing so allows for a full-fledged contextualization of the technologies that are being designed. Henceforth ethics appears as a “know-how”. Ethics cannot be transferred or learned as a theoretical knowledge but has to be practiced in order to be genuinely appropriated by those who face an ethically challenging situation, e.g. the installation of a video-surveillance system or a biometrics system. Such specific situations occur at a certain time, in a certain space, and falls under a specific configuration, depending on the targeted public, the purpose of the system, its material setting and its location.

Thus, evidence produced by the process depends on the quality of the socio-contextual and ethical reflection along the process of developing the technology, based on the knowledge insight produced by the expert. In this respect, a general guiding principle regarding the quality of outcomes is that “the broader is the better”, i.e. the system designer might very well consult and/or delegate the treatment of specific questions or choices to persons which are more able to deal with them. This results from Ladrière idea that *“nobody has a privileged competency in ethics. This is why an ethical approach could only be a collective process through which the different positions have to be confronted, with the hope of a convergence of these positions justified by the believe of the universality of the human reason”* (Ladrière, 1997).

Materially, as stated in D3.1, one of the SALT requirements is core to the kind of socio-contextual and ethical evidence that can be produced. The SALT management tool must document the purposes and reasons for all decisions made in the design process. In that sense only socio-contextual and ethical responsibility can be achieved. Since “responsibility” means “answer to”, answering the ethical questionnaire is a way to achieve responsibility by explicitly stating the goals and aims of the system together with its purposes.

For this reason, compliance is considered as reflection over the whole process of socio-contextual and ethical undertaking of the system design. If the approach has been closely

followed, implemented and resulted in clear choices and entrenched options, and that the latter are documented, then the approach can be said compliant.
It cannot be otherwise because then the reflection would fall short of empirical flesh and interest, hence providing empty and not genuine “social acceptability”. According to Brunson, the term "social acceptability" refers to aggregate forms of public consent whereby judgments are shared and articulated by an identifiable and politically relevant segment of the citizens. In this perspective the norms emerge from a democratic exercise involving all the concerned actors.

In short, evidence produced by following the SALT process results in documentation about the choices and options which have been decided upon. The quality of the process needs itself a case-to-case appraisal depending on how well the socio-contextual and ethical articulations have been thought of, justified and documented.

5.1.2 From a legal perspective

As described in D2.2, section 2.1, the SALT methodology is based on a three steps process. The outcome of each of these phases will have to be validated by a legal expert. The SALT process should thus produce sufficient documentation as to allow this legal validation.

The first phase is concerned with checking the legal opportunity of the envisioned system, i.e. its purpose and beneficence. From a legal point of view, this amounts to assessing the necessity and the legitimacy of the system in relation to its stated purpose. A first assessment of the impact of the technology on individuals’ fundamental rights (such as on the right to privacy) is realized at that level. The SALT process will guide decision makers through a series of aspects that should be taken into account in order to make this assessment. The outcome of such reflexive process phase should be documented and should contain the different elements reviewed in order to allow the legal expert to validate, from a legal point of view, the grounds on which the decision to use or discard a given technology has been taken.

The second phase is oriented to the design process. A questionnaire will lead the designer through the legal requirements stemming from the data protection framework. The legal framework often leaves some margin of appreciation (in order to allow sufficient generality and flexibility of the Law in its application to the variety of situations it is expected to regulate). The options taken by the system designer should be documented as to enable the legal expert to check the compliance of system requirements with the data protection framework.

The third phase consists of an overall check. The legal expert will review the overall documentation produced during the process (Legal/ Socio-Ethical/Technological) in order to check the overall compliance of the system to be deployed with the applicable legal framework. No additional documentation should be produced at that stage.

5.1.3 From a technological perspective

In computer science, non-functional properties imply the usage of specific process. Evidences are often requested in order to prove that requirements are well managed. For instance, a safety certification process will follow a standard (i.e., IEC-61508 standard) since scheduling

analysis will be provided for real-time aspects. Security or privacy is newer for computer science however we can already mention some initiatives like the common criteria[4].
The common criteria is an international standard for security evaluation. The process covers all design phases and is concluded by an assurance report. Even if the common criteria is not privacy oriented, it is important to note that data confidentiality is on the scope.

The CNIL (French DPA) also proposes privacy oriented guidelines. A risk-based analysis is also considered and can conclude on an evidence document. Threats are identified and the occurrence risk (combination of likelihood and impact) is processed. In the guidelines, they suggest several issues that can trigger to an evidence document:

- Protecting primary assets. Minimizing the amount of personal data and using security mechanisms are of course a main concern. However, other aspects like obtaining the consent of data subjects are also taken into account.
- Addressing the impacts. Accountability is considered in particular by tracing the activity on the IT system.
- Addressing risk sources.
- Protecting supporting assets.

5.1.4 Next xteps

At each phase of the compliant process, evidence documents can be generated. In the deliverable D4.3, some specific use cases will be studied. For all use cases, specific evidences will be identified.

5.2 Impact on privacy

The literature provides some methodologies in order to manage privacy like PMRM made by OASIS. This section provides an analysis of PMRM and states how the SALTed design process can be used.

5.2.1 Privacy Management Reference Model and Methodology

PMRM (Privacy Management Reference Model and Methodology) is a specification that was released by OASIS in July 2013. OASIS (Organization for the Advancement of Structured Information Standards) defines itself as *a non-profit consortium that drives the development, convergence and adoption of open standards for the global information society*.

5.2.1.1 Previous Work

The objective of PMRM (pronounced pim-rim) is to provide a guideline for developing operational solutions to privacy issues. It is the result of preliminary work carried out within ISTPA (International Security, Trust and Privacy Alliance), an organisation that is no longer existing.

ISTPA initially defined a privacy framework[5] in the early 2000. The framework explains how to map privacy principles[6] into 10 privacy services:

[4] http://www.commoncriteriaportal.org/

- Audit – independent, verifiable accountability
- Certification – credentials, trusted processes
- Control - only permissible access to data
- Enforcement - redress when violation
- Interaction - manages data/preferences
- Negotiation – of agreements, rules, privileges
- Validation - checks accuracy of personal information
- Access - subject can correct/update information
- Agent – software that acts on behalf of data subject
- Usage – data use, aggregation, anonymization

ISTPA then defined a subsequent version called privacy management reference model in 2009[7]. It enhanced the initial privacy framework into a reference model by integrating considerations on the lifecycle. The resulting model is shown in Figure 23.

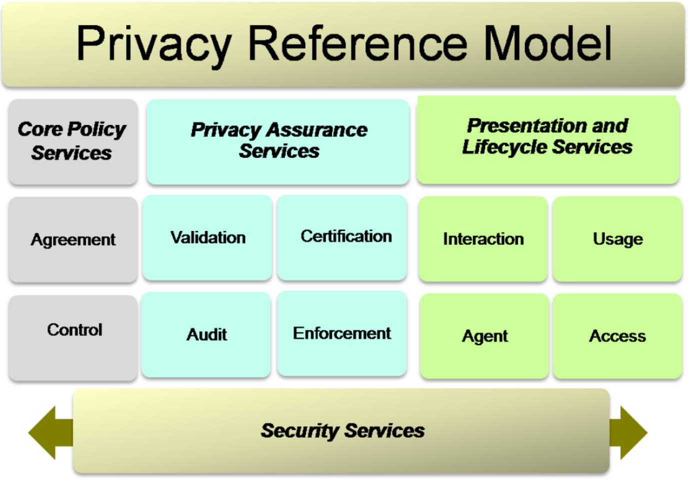


Figure 23. Privacy Reference Model (from ISTPA documents)

The reference model provides an abstract specification on how privacy in terms of policy and business requirements are mapped onto operational services, as showed in Figure 24. The reference model is targeted to various stakeholders (policy makers, business managers, privacy officers, systems architects, software developers).

[5] A version can be found there: http://emoglen.law.columbia.edu/LIS/archive/privacy-legis/ISTPA-FrameworkWhitePaper013101.pdf

[6] Listed as disclosure, relevance, participation, collection limitations, use limitations, accountability, security, verification

[7] http://xml.coverpages.org/ISTPA-PrivacyManagementReferenceModelV20.pdf

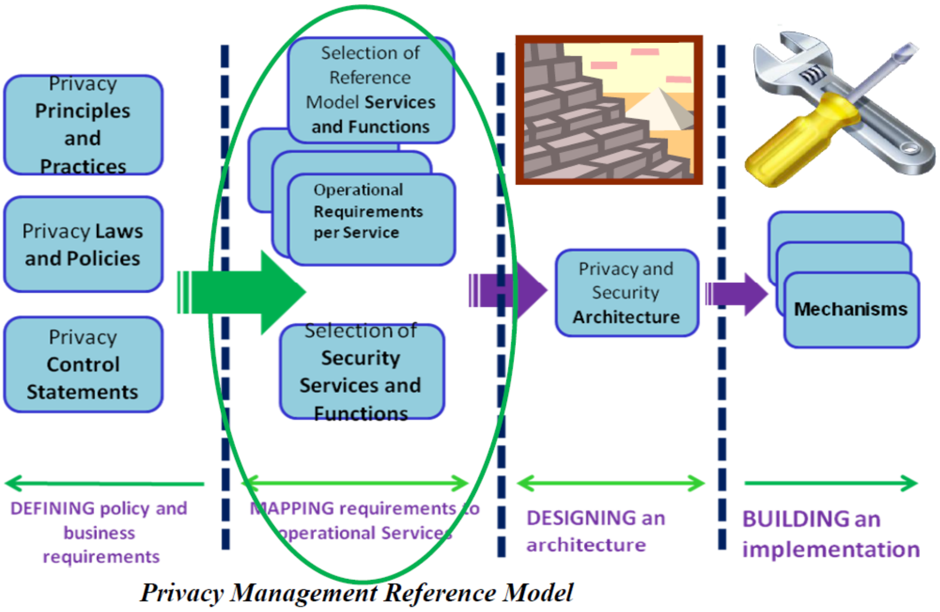

Privacy Principles and Practices
Privacy Laws and Policies
Privacy Control Statements
Selection of Reference Model Services and Functions
Operational Requirements per Service
Selection of Security Services and Functions
Privacy and Security Architecture
Mechanisms
DEFINING policy and business requirements
MAPPING requirements to operational Services
DESIGNING an architecture
BUILDING an implementation
Privacy Management Reference Model

Figure 24. Privacy Reference Model in the Life Cycle (from ISTPA documents)

5.2.1.2 Overview of PMRM

The current specification further extends ISTPA works by integrating considerations and issues related to complex systems (e.g. cloud, health, smart grid, social network…) that include networked, interoperable capabilities, applications, devices and that involve multiple jurisdictions. PMRM includes the following:

- A conceptual model of privacy management
- A methodology for analysing privacy use cases
- A set of operational services

5.2.1.2.1 PMRM Conceptual Model

The PMRM conceptual model provides a common conceptual framework and vocabulary to help people cooperate across disciplines and organizational boundaries. It is displayed in Figure 25. Note that PI and PII stand for Personal Information and Personally Identifiable Information.

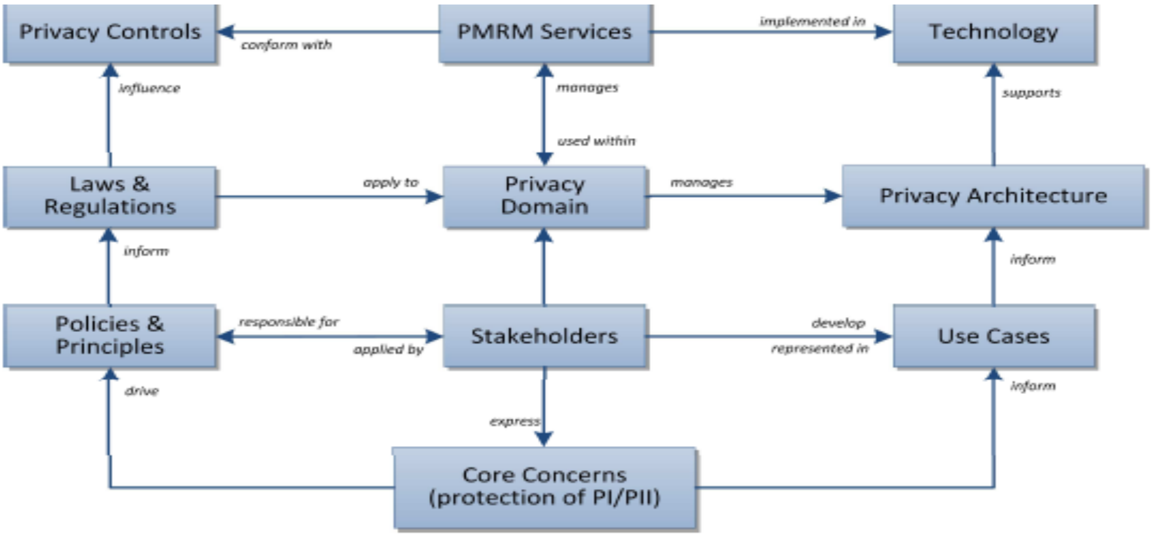


Figure 25. PMRM Conceptual Model (from OASIS specification)

The PMRM conceptual model is not viewed as prescriptive, i.e. one may choose to partially instantiate the conceptual model.

5.2.1.2.2 PMRM Methodology

The PMRM methodology defines a number of tasks to allow for a privacy management analysis that will lead to a privacy architecture. The methodology is depicted in Figure 26.

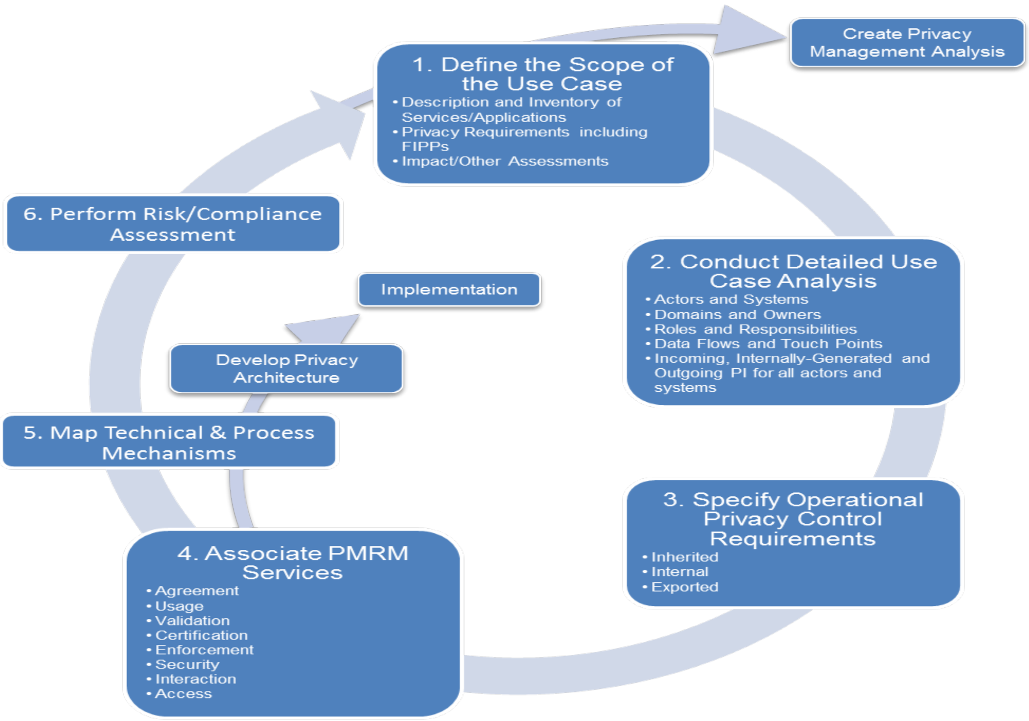

Create Privacy Management Analysis
1. Define the Scope of the Use Case
• Description and Inventory of Services/Applications
• Privacy Requirements including FIPPs
• Impact/Other Assessments
2. Conduct Detailed Use Case Analysis
• Actors and Systems
• Domains and Owners
• Roles and Responsibilities
• Data Flows and Touch Points
• Incoming, Internally-Generated and Outgoing PI for all actors and systems
3. Specify Operational Privacy Control Requirements
• Inherited
• Internal
• Exported
4. Associate PMRM Services
• Agreement
• Usage
• Validation
• Certification
• Enforcement
• Security
• Interaction
• Access
5. Map Technical & Process Mechanisms
Develop Privacy Architecture
Implementation
6. Perform Risk/Compliance Assessment

Figure 26. PMRM Methodology (from OASIS specification)

The methodology includes 20 tasks:

- Use case description and high-level privacy analysis
    - Task 1: Use Case Description
    - Task 2: Use Case Inventory
    - Task 3: Privacy Policy Conformance Criteria
    - Task 4: Assessment Preparation
- Detailed privacy analysis
    - Task 5: identify participants
    - Task 6: identify systems
    - Task 7: identify privacy domains and owners
    - Task 8: identify roles and responsibilities within a domain
    - Task 9: Identify Touch Points. Touch points are the intersections of data flows with privacy domains or systems within privacy domains
    - Task 10: Identify Data Flows
    - Task 11: Identify Incoming PI
    - Task 12: Identify Internally Generated PI
    - Task 13: Identify Outgoing PI

  - Task 14: Specify Inherited Privacy Controls
  - Task 15: Specify Internal Privacy Controls
  - Task 16: Specify Exported Privacy Controls
- Identify functional services necessary to support privacy controls. The lists of services are showed in Figure 23
  - Task 17: Identify Services that conform to identified privacy controls
- Define technical functionality and business processes supporting the selected services
  - Task 18: Identify functions that satisfy selected Services
- Performance risk and/or compliance assessment
  - Task 17: Conduct Risk Assessment
- Initiate iterative process
  - Task 18: Iterate the analysis and refine

The result of applying PMRM methodology is a PMA (Privacy Management Analysis). The methodology application can be flexibly used:

- The order of tasks is not prescriptive
- The process of establishing a privacy architecture can be started at any stage
- The process of determining when and how technology implementation will be carried out can be started at any stage

5.2.2 Relationship of OASIS PMRM and the SALTed design process

5.2.2.1 Some Definitions

It is useful at this point to provide some definitions:

- Management[8]: the process of dealing with or controlling things or people.
- Design[9]: a plan or drawing produced to show the look and function or workings of a building, garment, or other object before it is made.
- By Design[10]: as a result of a plan; intentionally.
- Analysis[11]: detailed examination of the elements or structure of something.
- Architecture[12]: the complex or carefully designed structure of something.
- Reference model[13]: an abstract framework for understanding significant relationships among the entities of some environment, and for the development of consistent standards or specifications supporting that environment. A reference model is based on a small number of unifying concepts and may be used as a basis for education and explaining standards to a non-specialist. A reference model is not directly tied to any standards, technologies or other concrete implementation details, but it does seek to

[8] oxford online dictionary

[9] oxford online dictionary

[10] oxford online dictionary

[11] oxford online dictionary

[12] oxford online dictionary

[13] from http://en.wikipedia.org/wiki/Reference_model

provide a common semantics that can be used unambiguously across and between different implementations.

- Methodology[14]: a body of practices, procedures, and rules used by those who work in a discipline or engage in an inquiry; a set of working methods.

5.2.2.2 Usage of PMRM in PARIS

- **Privacy management includes PbD.** Design processes are one of the multiple activities that need management. Therefore a privacy-by-design process is part of a privacy management system. This means that the PMRM document could be used as a reference document to a SALTed design process for designing a surveillance system.
- **Liaison with the FP7 PRIPARE project.** The PRIPARE project aims at providing a methodology for privacy by design. The PRIPARE methodology will extend PMRM methodology. Figure 24 shows where the PMRM privacy management reference model is located (mapping privacy requirements to operational services). We believe that the SALTed design process and the knowledge stored in the framework include the mapping as well as the other phases (the design of a privacy and security architecture, and the selection of mechanisms).
- **Architecture in the PMRM conceptual model is not clearly highlighted.** The PMRM conceptual model in Figure 25 includes an entity called *privacy architecture*. This entity is defined as a collection of proposed policies and practices appropriate for a given domain resulting from use of the PMRM. Therefore the architecture of the system being designed is not directly exhibited in the conceptual model. We believe that it is integrated in the entity called technology. All of these technologies can also be stored as knowledge in the SALT Framework.
- **Compliance and consistency between the SALTed design process and the PMRM conceptual model.** At this point of the PARIS project it does not exist a metamodel for the SALTed design process, although the process seems consistent with the PMRM model. Assuming that the technology entity in Figure 25 integrates architecture and PETs, the PARIS privacy-by-design methodology could be based on the PMRM conceptual method.
- **PbD-SE complements PMRM.** OASIS is currently working on a standard on Privacy-by-Design documentation for software engineers. The mission of the TC is to enable privacy to be embedded into IT system design and architecture. It is at this point too early to characterise the content of this specification but it would make sense to state that the specification will likely focus on the software documentation that would complement PMRM methodology.

5.3 *Impact on accountability*

The principle of accountability is a transparency mechanism whose goal is to increase trust in the design and use of information systems. The principle aims at increasing the transparency of

[14] from http://www.thefreedictionary.com/methodology

policies, procedures and practices of the organization in terms of data processing activities and by doing so to create a climate of trust supported by evidence. In the specific field of surveillance, accountability is expected to limit the imbalance of power inherent to surveillance by compelling the user of surveillance technology to account for its practices to a trustworthy third party.

Accountability should be understood not only as reporting and auditing mechanisms but also as "data governance". It follows that the goals of accountability mechanisms are threefold:

- Increase legitimacy and transparency of the decision-making process.
- Ensure a "responsible information use" (ensuring that only appropriate use occur) - concept of data governance.
- Provide evidence of compliance - concept of "proven trust".

Accountability is defined within PARIS[15] as a demonstrable acknowledgment and assumption of responsibility for having in place (i) appropriate policies, procedures and the promotion of good practices that include correction and remediation for failures and misconduct and (ii) appropriate accounts of actual practices to make it possible to demonstrate a posteriori that responsibilities have been exercised consistently with all legal, organizational and ethical requirements. It is a concept that has governance and ethical dimensions. It envisages an infrastructure that fosters responsible decision-making, engenders answering mechanics, enhances transparency and considers liability. It encompasses that organization will report, explain and be answerable for the consequences of decisions about the protection of data. Accountability promotes implementation of practical mechanisms whereby legal requirements and guidance are translated into effective protection of data.

This definition makes explicit the key elements of an accountability approach. Being accountable means that:

- The organization acknowledges its commitment to be accountable either internally (getting support from top management levels) or externally (advertising its commitment)
- The organization is able to demonstrate the organization complies with its commitments (voluntary) and obligations (mandatory)
- The organization implements appropriate policies, procedures and practices to ensure they are in line with the organizational commitment
- The organization accepts to remedy harms originated by organizations' practices to data subjects
- The organization should be ready to correct its policies, procedures and practices whenever proven not being appropriate anymore (by internal audit, failure, etc.). Accountability is a dynamic process.

Having clarify the goals and key elements of accountability schemes within an organization, the benefits of such schemes for data controllers are the following:

- Under the forthcoming Data Protection Package (General Data Protection Regulation and Law Enforcement Data Protection Directive), comply with a legal obligation. Setting

[15] See PARIS Deliverable D.2.2.

up an accountability scheme will enable the organization to provide sufficient evidence that its practices are compliant with the provisions of the data protection framework, whenever requested by the Data Protection Authority (or competent Supervisory Authority);

- Raise stakeholders' trust, most particularly data subjects, provided the measures taken under the scheme are adequately communicated. In that sense, the forthcoming General Data Protection Regulation provides for two possibilities: to apply for a Data Protection Seal, granted by Data Protection Authorities that will certify legal compliance; and to publish the measures implemented in activity reports. Organization can however opt for other communication channels;
- Improve the acceptability of the organizations' solutions towards data subjects showing it takes into account users' privacy;
- Improve the organization's branding;
- Improve capacity for resilience in case one employee or service provider mishandles the personal data by helping the organization to assign responsibilities, to react more quickly and adequately;
- Get a better overview and understanding of the organization's data processing activities, data handling procedures, data flows inside and outside the organization;
- Create a data protection culture within the organization by ensuring that privacy concerns are integrated at all stage of business activities.

The benefits of such schemes for data subjects are the following:

- Reduce the imbalance of powers in surveillance relationships by increasing the transparency over the use of surveillance technologies;
- Empower citizens over the personal data processing activities linked to surveillance technologies, enabling them to exercise their rights (of access, rectification, deletion, etc.) under the data protection framework.

Opting for not implementing accountability mechanisms within an organization could thus amount to:

- Legal infringement of accountability related obligations when the new European Data Protection Package is accepted;
- Mistrust amongst individuals monitored because of the opacity of the surveillance technology producing a chilling effect (individuals adjust their behaviour to match what is supposedly expected from them) and a feeling of loss of control over one's personal data amongst data subjects and of lack of protection against data abuses;
- Difficulties in assigning responsibilities in case of privacy breach and to react efficiently and adequately;
- A low level of awareness of privacy requirements within the organization, resulting in potential legal sanctions, bad branding, or difficulties to deal with the competent supervisory authority.

6 Conclusion

This deliverable focuses on the definition of an engineering process and the steps to follow for the development of surveillance systems with a certain level of privacy and accountability from a social, ethical, legal and technological point of view. It begins with an introductory chapter, which describes the current situation of surveillance systems regarding privacy and accountability concerns. It also justifies the need of a dedicated engineering process intended to design SALT compliant surveillance systems.

State of the art modelling tools dedicated to engineering processes is also studied, together with existing supporting tools. At this point, we feel the necessity of a privacy-by-design and accountability-by-design approach, which will finally lead to a SALT compliant system. And hence, definitions for these concepts are provided.

The engineering process that leads to a SALT compliant surveillance system is what we have called a SALTed design process. However, before describing such a process, we show and analyse two current surveillance systems, which will serve as a basis to identify phases and concepts for the SALTed design process. The example systems under study cover both, video-surveillance and biometric systems (the two types of systems covered by the PARIS project). Therefore, real systems are presented in detail, showing the process flow starting from the client's specifications and finally reaching the phase where design decisions are taken. Additionally, we also identify possible SALT concerns (regarding to privacy and accountability) that can be taken into account by the SALTed design process.

Then, we find what we could call the core of this document, where a SALT compliant process is presented and explained in detail, covering not only the SALTed design process, but also the processes in charge of information acquisition. The process' targets and fundamentals are exposed, stating what the work products produced by the process are, together with the modelling artefacts used by the SALTed design process. We also provide a lifecycle description, which it helps for a better understanding of the process ins and outs. Moreover, the inclusion of a list of possible use cases clarifies the way the process operates, i. e., what type of user can perform what type of action.

Finally, we analyse the impact of SALT frameworks on key points such as privacy and accountability. At this point, we also show what could be the evidences produced by the SALTed design process in order to state the importance of this impact.

Nonetheless, we still have much work to face in the remaining time of the PARIS project. For the closest future of the engineering process, we plan to focus on: (i) the development of a UML profile that will help system developers to create SALT compliant surveillance systems, even though they are not UML experts, (ii) refining the SALT compliant process, which will require the collaboration of several project partners, especially regarding the information acquisition processes, and (iii) creating guidelines that can be used for the development of SALT compliant surveillance systems.

7 References

[1] A. Cavoukian, "Privacy by Design: The 7 Foundational Principles". http://www.privacybydesign.ca/index.php/about-pbd/7-foundational-principles/

[2] Samsung Techwin, "Networked Surveillance System Design Guide", July 2012.

[3] A. Cockburn, "Writing effective use cases", Addison-Wesley, 2001.

[4] PARIS FP7 Project Deliverable D2.2 "Structure and Dynamics of SALT Frameworks".

[5] PARIS FP7 Project Deliverable D2.1 "Contexts and concepts for SALT Frameworks".