PrivAcy pReserving Infrastructure for Surveillance

Deliverable D3.1
SALT Framework Management Tool Requirements

Project:	PARIS
Project Number:	SEC-312504
Deliverable:	D3.1
Title:	SALT Framework Management Tool Requirements
Version:	v1.0
Date:	21/02/2014
Confidentiality:	Public
Authors:	PARIS Partners

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

Table of Contents

Document History

Version	Status	Date
V0.1	Proposal of the structure	16/10/2013
V0.2	Integration of all partner contributions to Section 3	31/01/2014
V0.3	Integration of partner contributions to Section 4	04/02/2014
V0.4	Enhancement round of all sections	10/02/2014
V0.5	Provision of comments by Namur and VT	14/02/2014
V1.0	Integration of internal reviewer comments	21/02/2014

Approval		
	Name	**Date**
Prepared	Christophe Jouvray	21/02/2014
Reviewed	All Project Partners	21/02/2014
Authorised	Antonio Kung	day/month/year
Circulation		
Recipient	**Date of submission**	
Project partners	21/02/2014	
European Commission	21/02/2014	

Executive Summary

The PARIS project contributes on privacy and surveillance reconciliation. For this reason, we define a framework called SALT (Social, ethicAl, Legal and Technical). A tool will implement all concepts defined in SALT. This document aims at describing the objectives of the SALT Management Framework Tool and its requirements. Moreover, an analysis of several representations will be presented.

List of Figures

List of Tables

Abbreviations and Definitions

Abbreviation	Definition
GUI	Graphical User Interface
ICT	Information and Communication Technologies
MDE	Model Driven Engineering
PARIS	PrivAcy pReserving Infrastructure for Surveillance
PbD	Privacy by Design
PIA	Privacy Impact Assessment
PoC	Proof of Concept
SALT	Social, Anthropological, Legal, Technical
SFI	SALT Framework Instances
SFMT	SALT Framework Management Tool
XML	eXtensible Markup Language

1 Introduction

The work package WP3 is focused on the SALT Framework Management tool which allows for a digital reference and representation of a SALT framework. Stakeholders involved in the SALT process need adapted access to the SALT framework as a reference, i.e. detailed information on their domain of interest and high level consistent information on other aspects. The tool is based on a unified common digital representation of the elements of a SALT framework which is structured in a way that such separation of concern can be ensured. The benefit of this tool is the assurance that a common understanding is reached between stakeholders.

The tool includes mechanisms for creating and updating a SALT framework. Therefore the same tool can be used for the creation of different frameworks. A given framework could also evolve since its social-contextual, ethical, and legal references change over time. The surveillance, privacy and accountability capability could also evolve. Thus, the tool should also allow for updates.

The tool will integrate information and guidelines to allow for reasoning, based on the provision of the rules specified to reach the proper public / private reconciliation and the rules for the potential interplay between SALT frameworks. It could also include some automated reasoning capability (to be investigated during the project).

The tool is used as a reference for the design of surveillance systems which comply with a given SALT framework. Designers of surveillance systems and stakeholders in the entire value chain, from technology providers to system integrators, should be able to use a SALT framework as a reference throughout the design and subsequent deployment and operation of surveillance systems. Specific information can be provided such as rules, constraints, and guidelines on the balance between surveillance, privacy, and accountability capability.

1.1 Objective of the Document

The objective of this document is twofold. On one hand, partners are identifying the requirements for a SALT framework management tool. For this reason, we will rely on use cases which cover the edition and update of knowledge and the usage of the knowledge. On the other hand, a study of SALT Framework representation is performed. In particular, we will take care to the appropriate representation for the different stakeholders.

1.2 Structure of the Document

Section 2 addresses the dynamics of the SALT Framework management tools. In particular, we will identify the objectives of the tools, the stakeholders who will use the tools and finally how the tools will be used during the life cycle. Section 3 is focused on the identification of requirements according to the objectives defined previously. Finally, Section 4 aims at proposing some representations for the SALT Framework management tools.

2 Dynamics of the SALT Framework Management Tools

One of the main objectives of the deliverable consists in defining the tools requirements. For this purpose, it is necessary to understand:

- Who are the users?
- When do they plan to use it?
- How do users plan to use it?

Some use cases (scenarios) have been identified in the deliverables D2.2 and D4.1 (in particular, see Section 3.3 of D2.2). These story boards correspond to the keystone for this document. Based on these inputs, this section aims at providing answers to the questions listed above. Section 2.1 highlights the stakeholders who will use the tools and their main expectations (scenarios). Section 2.2 points out the usage of the tool. Section 2.3 defines the main objectives of the tools. Finally, according to this inputs, the requirements will be defined in Section 3.

2.1 Identification of the Stakeholder

The concept of stakeholder is here understood as a person, group or organization that has interest or concern in the SALT Framework. Stakeholders can affect or be affected by the tool. Not all stakeholders are equal, they are driven by specific needs. To ensure that the SALT Framework is both useful and used, stakeholders' needs should be identified and met.

SALT Experts. SALT Experts are users (individuals or organizations) who will feed the SALT knowledge source. At this stage, they are appointed by the consortium as having sufficient expertise as to be able to provide reliable content. They can be philosophers, lawyers, academics, members of DPAs, experts in technologies.

Their expectation lies with the usability of the tool (it is easy to include content/they know how to formulate the content into useful language for the tool). It follows that the SALT Framework, as a tool, should be:

- Versatile: the tool is flexible enough to allow SALT experts to express their specific knowledge)
- Trustworthy: the information format allows SALT Experts to represent what they want to express.
- Adaptable: SALT Experts should be able to update the content they have provided.

System designers. System designers can be classified under two categories: (1) system owners, responsible for defining system's requirements (decision-maker) and (2) system developers, responsible for the realization of the system. The transition from system requirements to the realization of the system is most often a contract resulting from a tendering process.

System owners (requirements level) are driven by the following needs:

- Obligation to comply with regulation
- Protection of goods and persons mainly in public areas
- Political motivation: public satisfaction / privacy optimization to satisfy citizens' expectations

Business motivation: provision of a good quality service which generates benefits of any kind (economic, strategic, social, etc.)They expect that the SALT Framework will provide them with:

- Improved assessment of the need for a security system, i.e. to provide them with criteria to make the right balance between surveillance and privacy needs and to provide them with guidance on accountability mechanisms in order to increase transparency.
- State of the art overview of possibilities offered by security systems : performance and privacy impacts
- Better definition of security and privacy requirements

System Developers or Designers (realization level) are driven by the needs:

- To comply with system requirements at minimum cost
- To comply with applicable regulations
- To optimize long-term fruitful collaboration

They expect that the SALT Framework will provide them with:

- Definition of verification steps in engineering process
- Identification of key choices that impact privacy and optimize the cost linked to the integration of privacy enhancing features.

Whenever system developers have a certain margin of manoeuvre, privacy-enhancing features can be a differentiator on the market and give them a competitive advantage.

Operators of the system. While not direct users of the SALT Framework, operators are affected by the SALT Framework in that the system built on the guidance of the SALT Framework should be able to inform them about the rules to follow and to check that they comply with the rules. Their actions should be auditable (accountability requirement). The SALT Framework should thus produce guidelines and integrate into the design process these additional requirements, namely:

- Need to find appropriate ways to inform the operators
- Need to authenticate them
- Need to log the appropriate information to make audits possible (not too much, not too little)
- Need to ensure the security (confidentiality, integrity) of this information

Surveillance System Maintenance Operator. A surveillance system maintenance operator is a legal entity that is responsible for maintaining the system, meaning performing needed corrective actions on its components to ensure nominal working of the system along its lifetime. The real person acting for a surveillance system maintenance operator is called **Surveillance System Maintainer**.

Legal Counsel/Privacy Office. Legal counsels of the organization are not direct users of the SALT Framework but they participate to the SALT process at different stages. The SALT Management tool should thus produce sufficient documentation as to ease their assessment of the

lawfulness and legitimacy of the system designed. Legal counsels expect to obtain relevant information for the following purposes:

- to draft/update the Privacy Impact Assessment,
- to assess the compliance of the system being designed with applicable laws (during the whole life cycle of the system),
- to report to organizations whom the data controller is accountable to (business partners/DPAs/surveillance authority/etc)

They expect that the system designed should also include additional requirements in relation to their obligations towards data subjects (drafting of information notice, privacy policies, addressing complaints, etc.)

Supervisory Authorities. Supervisory authorities are not users of the SALT management tool. However they can interact with the SALT Framework in two ways:

- they can use the SALT Framework as a body of knowledge and thus as a source of information. In that case, they expect this body of knowledge to be reliable and to provide them with state-of-the-art information about a specific topics in its legal, socio-ethical and technical aspects
- they can use the accountability features of the system deployed to assess the level of compliance with the legal framework of the data processing activities carried out by the data controller. This means that the SALT Management Tool should produce sufficient documentation about the process followed, the options taken and introduce into the design process accountability requirements that make the system auditable.

Surveillance System Contractor. A surveillance system contractor is a legal entity (or several legal entities generally in consortium and represented by a prime) assuming the production of the surveillance system. Its responsibility is to fill the testable requirements attached to the system.

Data Subjects. Data subjects are not users of the SALT Framework. They are affected by the SALT Framework insofar as the system designed should enable the organization to adequately inform them of the data processing activities carried out, and to deal with complaint and redress requests. In addition, whenever the organization decides to involve data subjects in the design of system requirements or in the choice of technology to be used in order to increase the legitimacy of the system designed, the SALT Framework should guide system designers in the process.

2.2 The Tool in the Life Cycle

As its name points out, the SFMT (SALT Framework Management Tool) is a tool devised to manage the information contained within the SALT framework and its format. To achieve this goal, the tool interacts with the stakeholders of a given surveillance system, and depending on the type of stakeholder, it will perform a series of activities or another, i. e., legal experts and systems designers will use it in different ways, just to mention one example.

As a requisite for a proper functionality of the tool, we assume that a SALT template already exists, which sets the structure and format for the SALT instances that are going to form the SALT framework. These instances store the privacy and accountability related information regarding to surveillance systems for both: video-surveillance systems and biometric systems.

With this in mind, in this section we describe the role and the behaviour of the SFMT inside the lifecycle of a general SALT compliant process (described in PARIS project deliverable 4.2). Therefore, we provide a list showing the natural flow of actions provided by the tool and what types of stakeholders are related to each of them, together with some characteristics that must be fulfilled.

1. In first place, in order to always ensure a valid and useful SFMT, the tool must be flexible enough to adapt itself to different SALT templates. This property is very important because the SALT template will typically evolve according to new types of knowledge for any of the considered categories: socio-contextual, ethical, legal and technological. For example, new laws may arise whose representation would not be accurate when using the current SALT template. In this case, the template has to evolve to a new version with an updated representation format that allows including the knowledge from the new laws. If this happens, the SFMT must be able to work with the new template.
2. Experts from the different categories (surveillance systems technicians, sociologists, philosophers, lawyers, etc.) make use of the SFMT to introduce their knowledge into the SALT framework. This task can be achieved in two ways:
    a. Creation of a SALT instance: the tool will assist experts in this task by creating new SALT instances that will store their knowledge.
    b. Update of a SALT instance: in order to update some information that has been previously stored, first the tool hast to find and access the old SALT instance with the original information. Then, this information is shown to the expert, who will update it according to his new knowledge. Finally, the SFMT will store a new SALT instance with the updated information, but it will also keep the old one, which will be deleted or not depending on its current validity.
3. System designers use the SFMT in order to get access to the relevant information that may apply to the surveillance systems they are developing. For this purpose, the tool must allow system designers to introduce the specifications and context of the systems under development. Then, the tool will connect to the repositories where the SALT framework resides and will select those SALT instances whose information may apply or may be of importance for the current surveillance system. Once the instances are found, the SFMT will show this information to the system designers in an understandable way (we have to keep in mind that system designers will not surely be lawyers or sociologists, for example). With this information, and following the SALTed design process (described in PARIS project deliverable 4.2), system designers will be able to produce SALT compliant surveillance systems.

As a consequence, it is clear that the SFMT performs an important role within the whole lifecycle of the SALT compliant process, since it is the one that helps system stakeholders to interact with the information contained in the SALT framework. Such information is a key element for the SALT compliant process itself because its use is mandatory to design a final SALT compliant process.

2.3 Purpose and Objectives of SMFT

Introduced in the deliverable D2.2, Figure 1 presents where the SALT Management Framework tool (SMFT) takes places. The following paragraphs summarize the main objectives of the tool for each purpose.

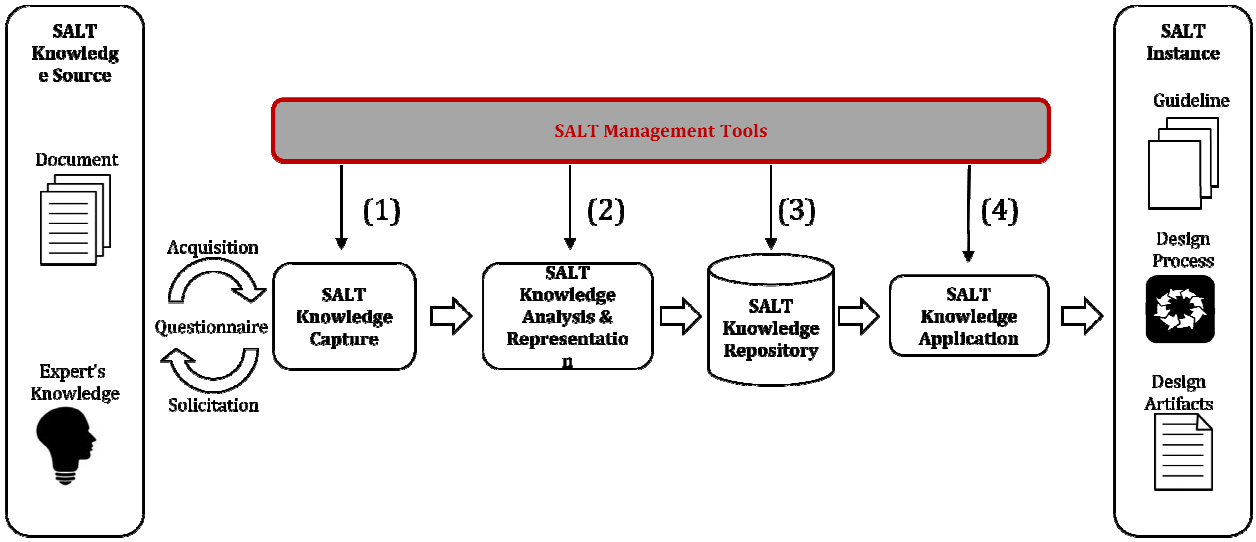

Figure 1 SALT Management Tools in the SALT Framework

Managing the SALT Knowledge Capture (KC Phase)

The SALT framework relies on literature and domain experts as the knowledge source. As a collection of knowledge from various sources, the SALT framework should be accessed and edited by different users in a cooperative way. Some editors have to be provided to the experts in order to enable the capture of knowledge. Since SALT addresses several domains e.g. legal and technical, it is crucial to adapt the editor according to the expert concern. The tool should also provide a way for customizing the representation for improving the flexibility and the user experience.

Finally, the tool has to allow the update of knowledge already captured by experts (e.g., due to modification of the legal framework).

Managing the SALT Knowledge Analysis & Representation (KAR Phase)

The SALT knowledge is selectively captured in various ways that are deemed relevant. In this stage, experts' effort will be needed to analyse the captured knowledge to decide its relevance. Since SALT knowledge comes from different disciplines and persons, efforts are also needed to identify the links and to synthesize the knowledge in coherence. Therefore, the analysed knowledge is transferred from textual description to defined models which facilitate the automatic storage and processing of captured knowledge. By consequence, the tool has to transform the text into a model and to provide mechanisms in order to check the consistency of several knowledge inputs.

Managing the SALT Knowledge Repository (KR Phase)

The SALT knowledge repository stores all models from various sources. The storage is done with an internal representation which allows a maximum of flexibility. In particular, it will be necessary to export models into an appropriate way according to the user needs. Finally, it is

important to be able to search models with keywords or properties (e.g., domain, expert name).

Managing the SALT Knowledge Application (KA Phase)

Although the SALT knowledge repository itself can already be used as a knowledge base or a reference for privacy in surveillance systems, a more valuable contribution of the SALT framework will be the knowledge application, in which system designers and other stakeholders, typically experts in a specific field, can leverage and apply the knowledge to solve similar problems in an efficient and correct way. Since SALT knowledge is represented as models, the knowledge application is analogous to model instantiation, i.e., given system specification and scenario characteristics, the SALT framework will provide instances corresponding to the particular context. Thus an instance can be seemed as a specific view of the SALT framework corresponding to a particular filter provided by the user, i.e., a subset of information from the whole framework. This specific information entitles designers to take proper design decisions to develop surveillance systems. These SALT instances include design guidelines, processes, as well as design artefacts.

Role of SFMT

Thus the role of the SALT Framework Management tools is to provide tool support for the creation, edition, search and extraction of the knowledge in the SALT framework. In other words, the SALT management tools are a set of computer programs that enable a user including domain experts and stakeholders to interact with concepts and information stored in the computer. The SALT management tools will also provide the capability to transform and process knowledge represented as computer models. In summary, Table 1 provides the main objectives of the SMFT.

Objective ID	Phase	Description
O1	KC	Allow to capture knowledge by an expert according to the usual domain representation
O2	KC	Allow to customize the way
O3	KC	Enable the update of knowledge already capture by experts (e.g. due to modification of the legal framework)
O4	KAR	Provide a feature in order to (semi) automatically transform expert knowledge text into model
O5	KAR	Provide algorithm in order to help experts to check the consistency of different inputs
06	KR	Provide a way to store models in a repository
O7	KR	Provide a way in order to search models according to some properties and/or keywords
O8	KA	Export some knowledge as an instance with all necessary information for using it

Table 1 Summary of the SFMT Objectives

It is important to note that the management of the SALT dynamic is a main characteristic of SFMT. Actually, the content of the SALT framework will be continually updated. It is also possible to update the structure of the SALT framework in order to be compliant with new

concepts or regulations. To update the structure will have a strong impact on all knowledge already stored. Indeed, it is necessary to have mechanisms in order to check if some contents must be changed.

3 Requirements for a SALT Framework Management Tools

In the previous section, we have presented who and how the SFMT (SALT Framework Management Tools) will be used. In particular, the SFMT objectives have been identified. According to these inputs, this section presents the requirements of the SFMT.

3.1 Presentation of the Template

Every requirement will be presented through the same template as depicted in the following.

REQUIREMENT_ID	***<Name of the requirement>***
Description: <Description of the requirement>	
Source: <Source of the requirement>	
Purpose: <Mention to one arrow of Figure 1>	
Rational: <Explain the rational of the requirement>	
Type: <*Mandatory or Optional*>	**Validation criteria:** <*Inspection, Test*>
Date: dd/mm/yy	**Version:** X.Y

Table 2 Template for Describing a Requirement

3.2 List of Requirements Related to Knowledge Capture

REQ_KC_1	***Creation of the questionnaire***
Description: Creation of the questionnaire through which the SALT framework will capture knowledge	
Source: SALT experts	
Purpose: SALT knowledge capture	
Rational: The SALT experts create a questionnaire in order to capture information which will later serve to generate the recommendations to the system designers.	
Type: *Mandatory*	**Validation criteria:** *Inspection*
Date: 29/01/14	**Version:** 0.1

REQ_KC_2	***Edition of the questionnaire***
Description: Edition of the questionnaire through which the SALT framework will capture knowledge.	
Source: SALT experts	
Purpose: SALT knowledge capture	
Rational: Once the questionnaire has been created, it is possible that the SALT experts want to modify it. For example, due to the fact that the existing questions are not considering certain information.	
Type: *Mandatory*	**Validation criteria:** *Inspection*
Date: 29/01/14	**Version:** 0.1

REQ_KC_3	***Information acquisition.***
Description: The information through which the instances will be created is captured by the SALT framework.	
Source: SALT experts knowledge, SALT experts opinion based on their expertise, documents and reports.	
Purpose: SALT knowledge capture.	
Rational: The salt management tool acquires the information through the questionnaire, which is filled in by the SALT experts, the opinion of the SALT experts based on their expertise, different comments and reports. The SALT management framework should provide the proper tools to let the SALT experts introduce the corresponding information.	
Type: *Mandatory*	**Validation criteria:** *Inspection*
Date: 29/01/14	**Version:** 0.1

REQ_KC_4	***Create profile of video surveillance requirement***
Description: The SALT management tool should allow video surveillance user (e.g. public security authority) and system designer to create entries in the SALT knowledge base on surveillance requirements. Example requirements include public risk analysis, protection goal, and area definition. The requirements should be generalised and captured in a set of high level profiles.	
Source: Surveillance design process description	
Purpose: (1) SALT Knowledge Capture and (2) SALT Knowledge Analysis & Representation	
Rational: The knowledge on surveillance requirements will guide stakeholders when initiating a project for the installation and operation of video surveillance system.	
Type: *Mandatory*	**Validation criteria:** *Inspection*
Date: 29/01/2014	**Version:** 0.1

REQ_KC_5	***Create profile of biometrics surveillance requirement***
Description: The SALT management tool should allow biometrics surveillance user (e.g. public security authority) and system designer to create entries in the SALT knowledge base on surveillance requirements. Example requirements include public risk analysis, protection goal, and area definition. The requirements should be generalised and captured in a set of high level profiles.	
Source: Surveillance design process description	
Purpose: (1) SALT Knowledge Capture and (2) SALT Knowledge Analysis & Representation	
Rational: The knowledge on surveillance requirements will guide stakeholders when initiating a project for the installation and operation of video surveillance system.	
Type: *Mandatory*	**Validation criteria:** *Inspection*
Date: 29/01/2014	**Version:** 0.1

REQ_KC_6	***Create video/biometrics surveillance technology body of knowledge***
Description: The SALT management tool should enable video/biometrics surveillance designers and engineers to create a basic set of knowledge on system design, such as, for instance for video surveillance, choosing and configuration of camera, decision on storage technology, choosing of video analytic capabilities, decision on video management system, rules and technology for data access, and interface specification etc.	
Source: Video surveillance system design guide	
Purpose: (1) SALT Knowledge Capture and (2) SALT Knowledge Analysis & Representation	
Rational: The technical concerns in the SALT framework	
Type: *Mandatory*	**Validation criteria:** *Inspection*
Date: 29/01/2014	**Version:** 0.1

REQ_KC_7	***Create common practice of video/biometrics surveillance operation***
Description: The SALT management tool should enable engineers and operators to create knowledge based on common practice in video/biometrics surveillance system operations including general process, human-to-human and human-to-machine interactions, oversight, and other related activities.	
Source: AIT	
Purpose: (1) SALT Knowledge Capture and (2) SALT Knowledge Analysis & Representation	
Rational: The operation of video surveillance system is privacy relevant.	
Type: *Mandatory*	**Validation criteria:** *Inspection*
Date: 29/01/2014	**Version:** 0.1

REQ_KC_8	***Create privacy-enhancing measures for video surveillance system***
Description: The SALT framework management tool should enable system designer and privacy engineer to create a knowledge base for existing privacy-enhancing measures for video surveillance system, such as video masking, data encryption, and access control. This knowledge might be linked to knowledge on video surveillance technology, since advanced video analytics can be used to either harm or protect privacy.	
Source: AIT	
Purpose: (1) SALT Knowledge Capture and (2) SALT Knowledge Analysis & Representation	
Rational: Privacy measure is the main source for technical means to enhance privacy in system design.	
Type: *Mandatory*	**Validation criteria:** *Inspection*
Date: 29/01/2014	**Version:** 0.1

REQ_KC_9	***Privacy policy language specification***
Description: The SALT management tool should make it possible to add SALT knowledge entries containing technical details about the privacy policy language used to ensure compliance of data handling operations of a system.	
Source: Document	
Purpose: SALT Knowledge Capture	
Rationale: Knowledge of the privacy policy language in use facilitates compliance checking for accountability.	
Type: *Mandatory*	**Validation criteria:** *Inspection*
Date: 30/01/2014	**Version:** 0.1

REQ_KC_10	***Data handling log format specification***
Description: The SALT management tool should make it possible to add SALT knowledge entries containing technical details about the format of the logs containing data handling operation traces of a system (accountability issue).	
Source: Document	
Purpose: SALT Knowledge Capture	
Rationale: Knowledge of the log format in use facilitates compliance checking for accountability.	
Type: *Mandatory*	**Validation criteria:** *Inspection*
Date: 30/01/2014	**Version:** 0.1

REQ_KC_11	***Data handling compliance specification***
Description: The SALT management tool should make it possible to add SALT knowledge entries containing the specification of data handling compliance, defined with respect to logs and one or more privacy policy language(s), for a system.	
Source: Document	
Purpose: SALT Knowledge Capture	
Rationale: Knowledge of data handling compliance rules facilitates compliance checking for accountability.	
Type: *Mandatory*	**Validation criteria:** *Inspection*
Date: 30/01/2014	**Version:** 0.1

REQ_KC_12	***SALT collaborative tools***
Description: The SALT management tool may be able to propose a voting/comment mechanism dedicated to its users	
Source: THALES	
Purpose: SALT Knowledge Capture	
Rational: collaborative process about evaluation of SALT instance topics relevance	
Type: *Optional*	**Validation criteria:** *Test*
Date: 31/01/2014	**Version:** 0.1

REQ_KC_13	***Flexible representation***
Description: The SALT management tool may have several front-ends in order to be compliant with the concerns of the users	
Source: SALT experts	
Purpose: SALT Knowledge Capture	
Rational: The tool will be used by different users who have various backgrounds. To adapt the editor to the user expectations is important in order to sell the SALT tool chain. For instance, according to the login information, a specific profile can be selected.	
Type: *Optional*	**Validation criteria:** *Test*
Date: 31/01/2014	**Version:** 0.1

REQ_KC_14	***Traceability of modifications***
Description: The SALT management tool may store the name of the user who provides new input	
Source: SALT experts	
Purpose: SALT Knowledge Capture	
Rational: Trust in the knowledge store by the framework is a key issue. By storing the user names, we should raise a responsibility feeling by the user.	
Type: *Optional*	**Validation criteria:** *Test*
Date: 31/01/2014	**Version:** 0.1

REQ_KC_15	***Knowledge qualification***
Description: The SALT management tool should require some keywords when new input is stored.	
Source: SALT experts	
Purpose: SALT Knowledge Capture	
Rational: Help to find information in the repository	
Type: *Optional*	**Validation criteria:** *Test*
Date: 31/01/2014	**Version:** 0.1

3.3 List of Requirements Related to Knowledge Analysis & Representation

REQ_KAR_1	***Creation of the template***
Description: Creation of the template through which the instances of the SALT framework are represented.	
Source: SALT experts.	
Purpose: SALT knowledge analysis and representation.	
Rational: All the information which is captured using the SALT framework must be represented. This representation makes that all the instances of the SALT framework follow the same structure, making the information more manageable for later decision-making of a surveillance system.	
Type: *Mandatory*	**Validation criteria:** *Inspection*
Date: 29/01/14	**Version:** 0.1

REQ_KAR_2	***Edition of the template***
Description: Edition of the template through which the instances of the SALT framework are represented.	
Source: SALT experts	
Purpose: SALT knowledge analysis and representation.	
Rational: Once the template has been created, it is possible that the users or entities responsible want to modify the template. For example due to any change in law that makes them have to modify certain representation of the instance, or simply because it was not initially taken into account any information.	
Type: *Mandatory*	**Validation criteria:** *Inspection*
Date: 29/01/14	**Version:** 0.1

REQ_KAR_3	***SALT hash tags***
Description: The SALT management tool must propose a capability to tag each knowledge inserted using sets of pre-defined keyword.	
Source: THALES	
Purpose: SALT Knowledge Analysis & Representation	
Rational: navigation/search within SALT instances	
Type: *mandatory*	**Validation criteria:** *Test*
Date: 31/01/2014	**Version:** 0.1

REQ_KAR_4	***SALT automatic semantic analysis tool***
Description: The SALT management tool might propose a semantic analysis tool capable of perform sorting/grouping/searching within SALT knowledge repository	
Source: THALES	
Purpose: SALT Knowledge Analysis & Representation	
Rational: navigation/search within SALT instance	
Type: *mandatory*	**Validation criteria:** *Test*
Date: 31/01/2014	**Version:** 0.1

3.4 List of Requirements Related to Knowledge Repository

REQ_KR_1	***Storage of the instances***
Description: Storage of the instances which have been represented according to the template.	
Source: Knowledge represented as an instance/instances	
Purpose: SALT Knowledge Repository.	
Rational: Once the knowledge has been captured, analysed and represented based on the template, this representation must be stored, so that the instances/references can be generated subsequently.	
Type: *Mandatory*	**Validation criteria:** *Inspection*
Date: 29/01/14	**Version:** 0.1

REQ_KR_2	***SALTed System Export***
Description: The SALT management tool may be able to propose an export format of a SALT instance usable to populate/initiate another instance. This export format may be itself encrypted to avoid unauthorized access to a SALT instance.	
Source: THALES	
Purpose: SALT Knowledge Repository	
Rational: exchange of SALT instances between SALT management tools	
Type: *Mandatory*	**Validation criteria:** *Test*
Date: 31/01/2014	**Version:** 0.1

REQ_KR_3	***Edition of the instances.***
Description: The SALT expert should be able to change instances stored in the SALT repository.	
Source: SALT experts	
Purpose: SALT Knowledge Repository.	
Rational: The SALT experts should be able to provide the different experts a mechanism to change the stored instances, due to different reasons (i.e, due to a specific law have changed).	
Type: *Mandatory*	**Validation criteria:** *Inspection*
Date: 29/01/14	**Version:** 0.1

REQ_KR_4	***Model based Repository***
Description: The knowledge stored in the repository should be transformed into a model.	
Source: SALT experts	
Purpose: SALT Knowledge Repository.	
Rational: Knowledge stored has to be exported later via an instance. For this reason, model-based techniques will be used.	
Type: *Mandatory*	**Validation criteria:** *Test*
Date: 29/01/14	**Version:** 0.1

3.5 List of Requirements Related to the Framework Use

REQ_FU_1	***SALTed instance versioning***
Description: The SALT management tool shall be able to host incremental versions of a SALT instance, with identification of contributors, and versions difference management	
Source: THALES	
Purpose: SALT Knowledge Repository	
Rational: capability to track changes within a SALT instance	
Type: *Mandatory*	**Validation criteria:** *Test*
Date: 31/01/2014	**Version:** 0.1

REQ_FU_2	***Interface for context information and design requirement***
Description: The SALT management tool should provide an interface to system designer for describing context information of design requirements.	
Source: AIT	
Purpose: SALT Knowledge Application	
Rational: The context information is needed for instantiate a SALT instance.	
Type: *Mandatory*	**Validation criteria:** *Test*
Date: 29/01/2014	**Version:** 0.1

REQ_FU_3	***Documentation of design decisions***
Description: The SALT management tool might document the purposes and reasons for all decisions made in the design process.	
Source: AIT	
Purpose: SALT Knowledge Application	
Rational: The documentation will provide evidence for compliance check.	
Type: *Mandatory*	**Validation criteria:** *Inspection*
Date: 29/01/2014	**Version:** 0.1

REQ_FU_4	***Compliance check and warning***
Description: The SALT management tool might be able to verify a system design is SALT-compliant or prompt warnings if the technical decisions harm privacy.	
Source: AIT	
Purpose: SALT Knowledge Application	
Rational: The tool will assists system designers, similar to spell checking.	
Type: *Optional*	**Validation criteria:** *Test*
Date: 29/01/2014	**Version:** 0.1

REQ_FU_5	***Specification of the Surveillance system***
Description: The Surveillance system designer introduces in the SALT management tool the specification of the Surveillance system.	
Source: Surveillance system designer.	
Purpose: SALT knowledge application.	
Rational: The Surveillance system designer, who represents to the Surveillance system owner, introduces the proper specification with the aim to get the recommendation of the SALT framework for creating a Salted surveillance system. Hence, The SALT management tool must facilitate the introduction of the specification of the system.	
Type: *Mandatory*	**Validation criteria:** *Inspection*
Date: 29/01/14	**Version:** 0.1

REQ_FU_6	***SALT instance selection by the SALT management tool.***
Description: The SALT management tool has to select a proper instance or instances based on the specification done by the system designer.	
Source: Specifications of the surveillance system.	
Purpose: SALT knowledge application.	
Rational: Once the SALT management tool has received the specification of the surveillance system from the system designer, the SALT management tool will use an internal engine to select the one or several instances which match with the specifications. It is important to highlight that if the system does not find any instance that matches the specification, no instance will be selected.	
Type: *Mandatory*	**Validation criteria:** *Inspection*
Date: 29/01/14	**Version:** 0.1

REQ_FU_7	***SALT instance recommendations***
Description: The SALT management tool shows in a proper way the recommendation to the system designer for making the new system based on the SALT instance.	
Source: SALT instances.	
Purpose: SALT knowledge application.	
Rational: Once the SALT management tool has selected the instance or instances based on the specifications of the system designer, the information or recommendations must be presented to the system designer in the simplest and easiest way to understand it.	
Type: *Mandatory*	**Validation criteria:** *Inspection*
Date: 29/01/14	**Version:** 0.1

REQ_FU_8	***Access to technical privacy policies by auditors***
Description: Auditors (for instance accredited third parties) mandated to check the privacy compliance of the surveillance system should have access to technical privacy policies, logs and compliance rules in the SALT knowledge repository through the SALT management tool.	
Source: Document	
Purpose: SALT Knowledge Repository	
Rationale: Access to privacy policies, logs and compliance rules is needed to perform compliance checking.	
Type: *Mandatory*	**Validation criteria:** *Inspection*
Date: 30/01/2014	**Version:** 0.1

REQ_FU_9	***Pointers to existing compliance checking mechanisms***
Description: The SALT management tool should provide pointers to existing compliance checking mechanisms to users of the framework, depending on the privacy policy language used, if any exist.	
Source: Expert's Knowledge	
Purpose: SALT Knowledge Application	
Rationale: In case compliance checking mechanisms for a given privacy policy language already exists, such pointers may encourage reuse of existing solutions.	
Type: *Optional*	**Validation criteria:** *Inspection*
Date: 30/01/2014	**Version:** 0.1

REQ_FU_10	***Pointers to information about common privacy policy languages***
Description: The SALT management tool should provide pointers to relevant information, such as official specifications, in case the privacy policy language is a commonly used one.	
Source: Document	
Purpose: SALT Knowledge Application	
Rationale: Information about the relevant privacy policy language facilitates the accountability-by-design process.	
Type: *Optional*	**Validation criteria:** *Inspection*
Date: 30/01/2014	**Version:** 0.1

REQ_FU_11	***SALT certificate***
Description: The SALT management tool might be able to issue a certificate guarantying the design process has been SALTed. This certification can be included with the knowledge export as an appendix.	
Source: THALES	
Purpose: SALT Knowledge Application	
Rational: Capability to prove the application of SALT instance on a system design	
Type: *Optional*	**Validation criteria:** *Test*
Date: 31/01/2014	**Version:** 0.1

REQ_FU_12	***SALT compliance verification over time***
Description: The SALT management tool might be able to propose a check list that enables to check periodically that the system privacy level has not been modified. Some warning can be generated in order to alert designers.	
Source: THALES	
Purpose: SALT Knowledge Application	
Rational: Capability to prove SALTed compliance over time	
Type: *Optional*	**Validation criteria:** *Test*
Date: 31/01/2014	**Version:** 0.1

REQ_FU_13	***Guidelines for SALTED system updates***
Description: The SALT management tool may be able to propose light guidelines enabling fast SALT compliance checking when slight modifications are realized	
Source: THALES	
Purpose: SALT Knowledge Application	
Rational: capability to assist system evolutions during the system lifecycle	
Type: *Optional*	**Validation criteria:** *Test*
Date: 31/01/2014	**Version:** 0.1

4 Analysis of Possible SALT Concepts Representation

The PARIS project will develop the SALT Framework management tool in the task 3.4 which will start at the middle of the project. However, in order to facilitate the communication among the partners, some proofs of concept were developed by the technical partners. The advantage of the tool sketch allows illustrating how non technical concepts can be managed. This section provides some representation approaches which are compliant with the objectives of the SALT Framework management tool.

Section 4.1 presents the knowledge capture with wiki whereas Section 4.2 uses XML as structured way.

4.1 Analysis of a Wiki-based Representation through a PoC

This section presents a sketch of the SALT tools based on wiki for capturing knowledge and creating an instance.

4.1.1 Principle

Wikipedia is a well-known community-based web portal. Everyone is able to add knowledge in a very easy way. During the project discussions, we are faced to varied practices due to our different background and domains. For instance, in computer science, modelling techniques are often used in order to abstract complex problems. It seems impossible to use this usual way in order to interact with legal or ethical domains. For this reason, we decide to try wiki as knowledge capture editor. The project does not aim to reinvent a new wiki technology but to adapt this technology according to our requirements presented in Section 3.

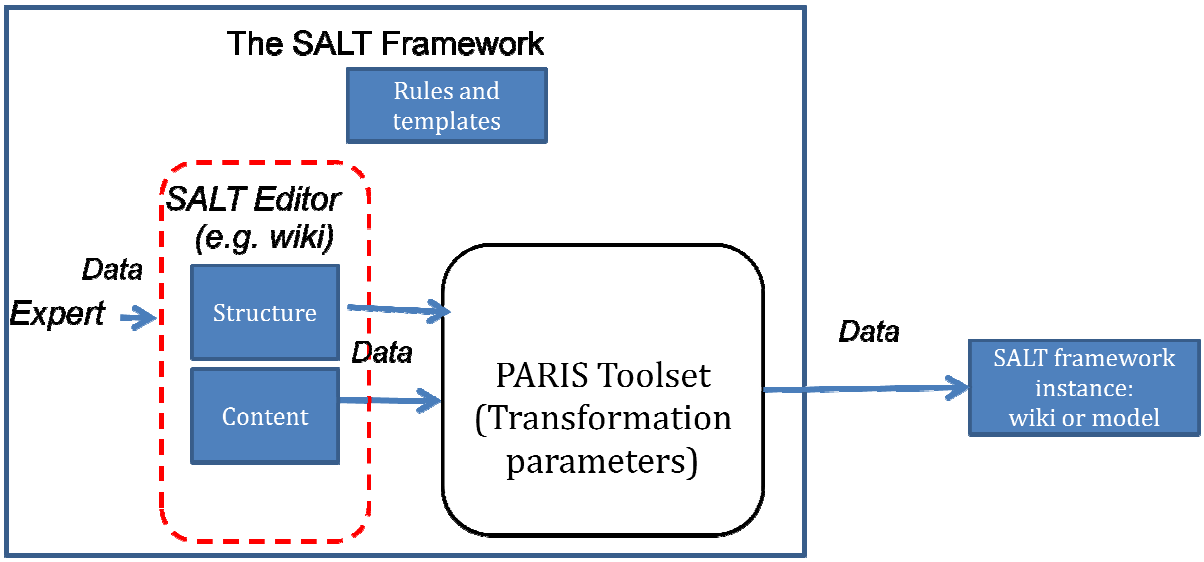

Figure 2 Principle of a Wiki-based Representation in PARIS

Figure 2 illustrates the approach for adding information and using it in an instance. The SALT editor is the tool for capturing knowledge by SALT experts. Section 4.1.2 is an analysis of the usage of a wiki as knowledge capturing tool. Section 4.1.3 illustrates (i) how it is possible to define and customize a SALT instance, and (ii) how a wiki can be used as a SALT instance viewer.

4.1.2 Wiki as Knowledge Capturing Tool

Several wiki frameworks are available in the market. We decided to use mediawiki since it is mature and well maintained by the community. A wiki has a main page. In the context of PARIS, we decided to use this main page for structuring the knowledge capturing as shown in Figure 3.

As describer in Figure 3, we divide the main page in two sections. The first one corresponds to the structure of the SALT Framework where templates are described. In Figure 3, templates for defining taxonomies and privacy harms are specified. According to the deliverable D2.2, others like questionnaire templates will be defined. The second section represents knowledge provided by the experts.

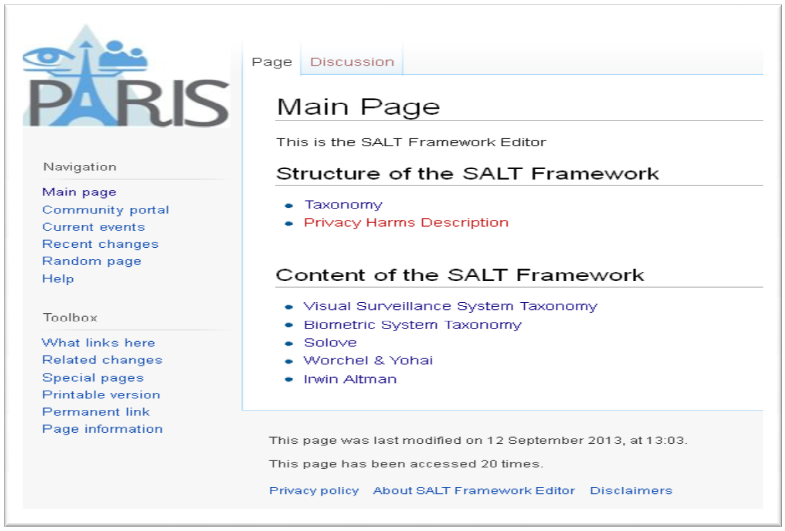

Figure 3 Organization of the main page

Figure 4 is an example of a taxonomy template. Some brief guidelines explain how to fill out the template.

Figure 4 Overview of the Taxonomy Template

Figure 5 is an example of knowledge provided by an expert. In this example, we provide the description of a visual surveillance system taxonomy. This taxonomy was described in the deliverable D2.1.

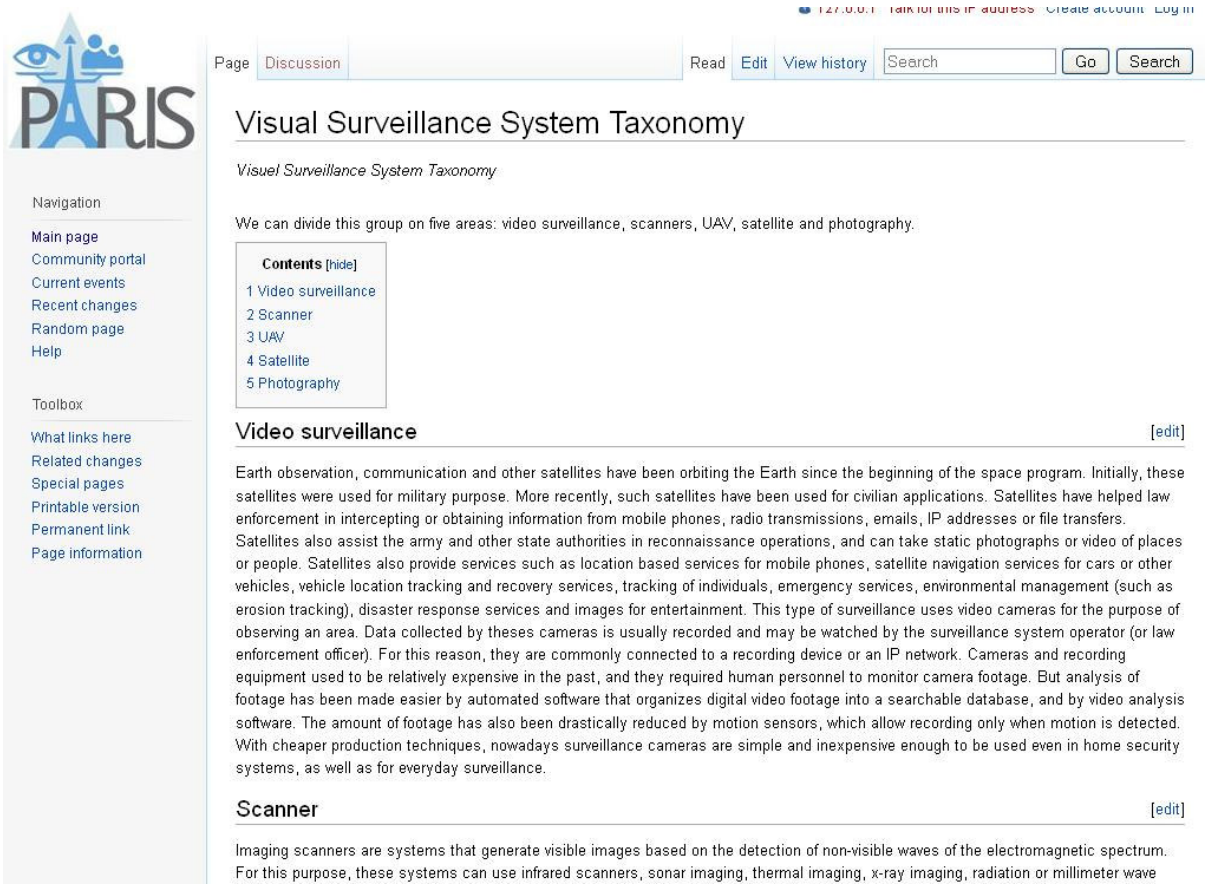

Figure 5 Visual Surveillance Taxonomy Description

4.1.3 Defining and Customizing a SALT Framework Instance

The SALT Framework knowledge is an aggregation of data available for specific context. At the starting of a surveillance system design, it is necessary to setup which information is useful for the project. In the PARIS project, this step consists in defining a SALT Framework instance (see Figure 6).

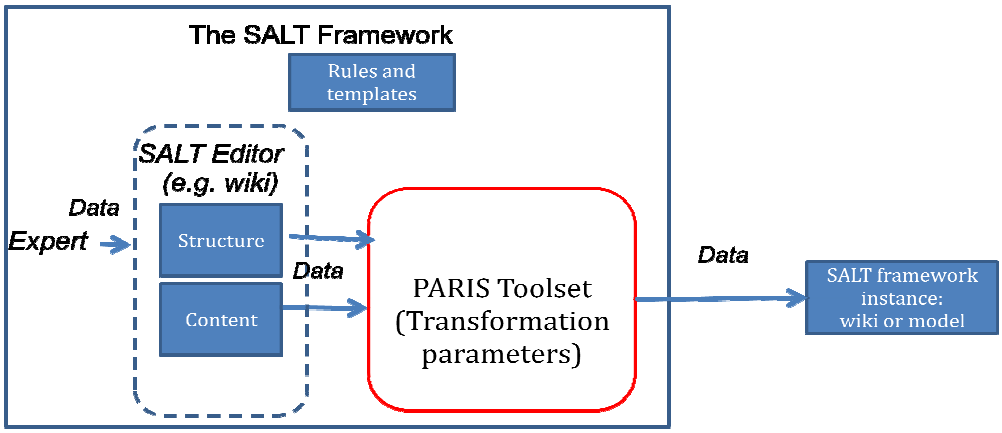

Figure 6 Highlight of Toolchain for Creating the Instance

In order to select the item composing the SALT instance, we decide to develop a very small graphical interface which screens the wiki. Then, the user selects the elements needed for the instance. Basically, the tool provides the following features:

- Enabling the knowledge composing the instance
- Customizing the instance according to the user concerns
- Generating automatically the instance

4.1.3.1 Selection of the SALT Instance Knowledge

Figure 7 is the GUI developed in PARIS for selecting the items which will be integrated in the SALT instance. The GUI has been developed in C++ with QT Creator.

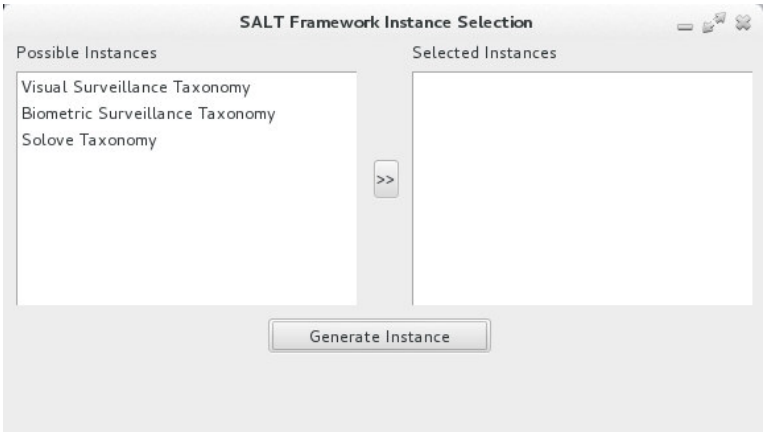

Figure 7 Graphical User Interface for setting up a SALT Instance

On the left of Figure 7, all elements defined in the wiki are shown. In this proof of concept, all inputs are at the same level. Of course, at the end, a structure way will be selected. When one item is selected and the ">>" button is pushed, it is selected in the instance.

When all desired knowledge is selected in the instance, the user can generate the instance by pushing the "Generate Instance" button.

4.1.3.2 Customization of the SALT Instance

An instance has to be conformed to the stakeholder requirement. For this reason, we decide to implement some mechanisms in order to customize the generated instance (see Figure 8).

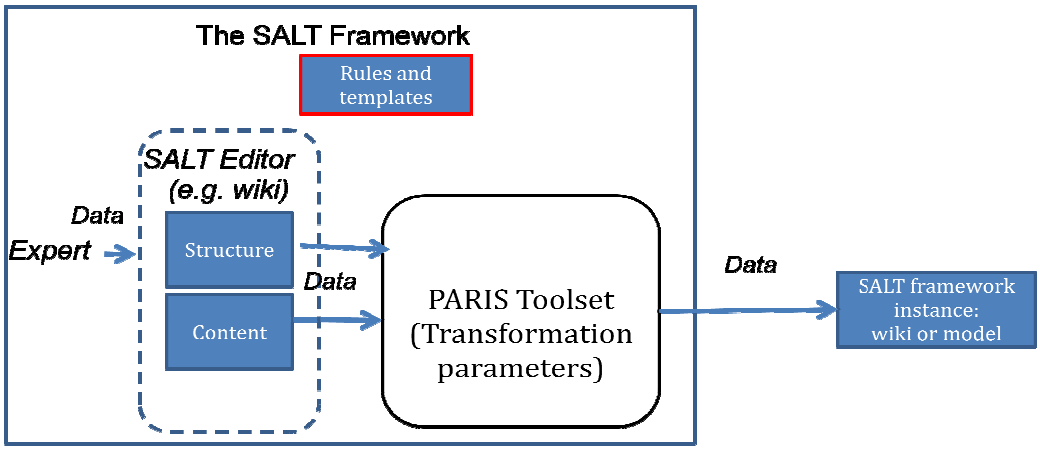

Figure 8 Rules and Templates

According the stakeholder concerns, different outputs can be generated. In the context of this sketch, we decided to generate a new wiki but with a different look. For generating this new

wiki, some specific parameters have to be specified such as the database name, the server address, and so on.
During the generation process, it is necessary to define how the data is being transformed. Figure 9 is an example for a wiki. On the left of the figure, we represent the hierarchy in the SALT Framework reference. On the right, we point out how it will be managed in the instance.

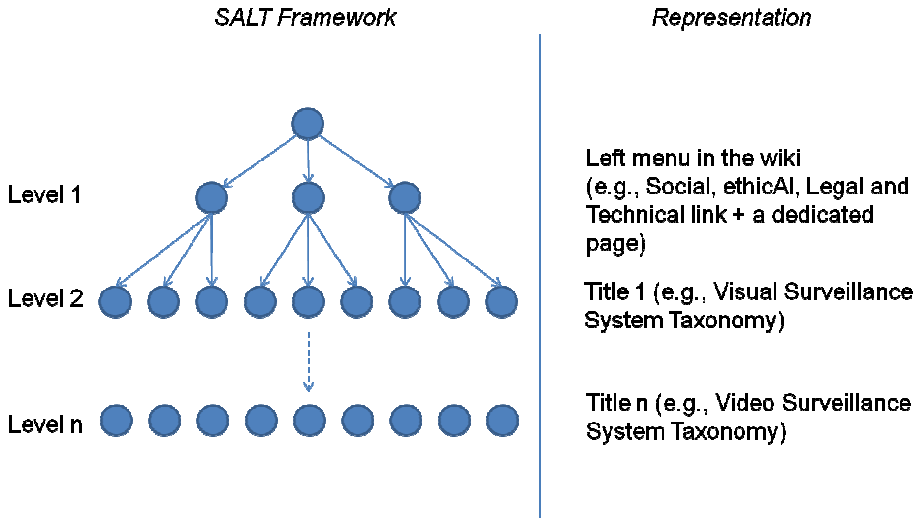

Figure 9 Mapping of the Representation between the SALT Knowledge Reference and the Instance

4.1.3.3 Automatic Generation of a SALT Instance

The SALT reference has to be stored in a repository. The storage is conformed to an internal representation which will be never exhibit to the outside. In the PARIS project, we decided to use model-based techniques for this purpose (see Figure 10).

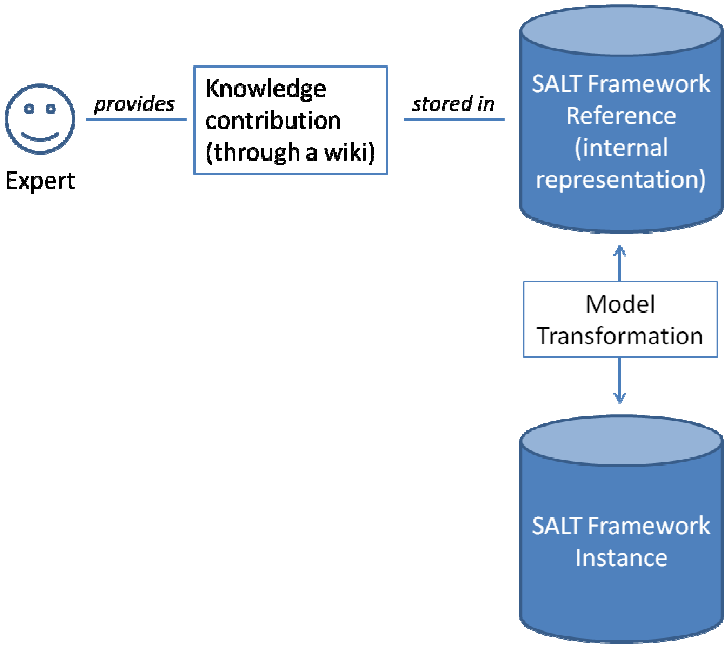

Figure 10 Generation of an Instance by Model Transformation

Partners have already implemented the model transformation specified in Figure 10. From the SALT Framework Reference, we are able to generate a new wiki.

The principle used in the sketch consists in skirting a wiki management bot for our purpose. A bot is a piece of software which perform jobs. In our case, we ask to the bot to update the SALT Framework instance with the elements selected with the GUI. We selected piwikipediabot which is a bot able to manage different kind of wikis. Thus, the model transformation consists in the following steps:

1. Generate the wiki pages
2. Generate the script which will be executed by the bot

For both steps, model to text transformations are used. In the sketch we used Acceleo which is on of the main reference in this domain. Figure 11 and Figure 12 shows some screenshots of the generated wiki.

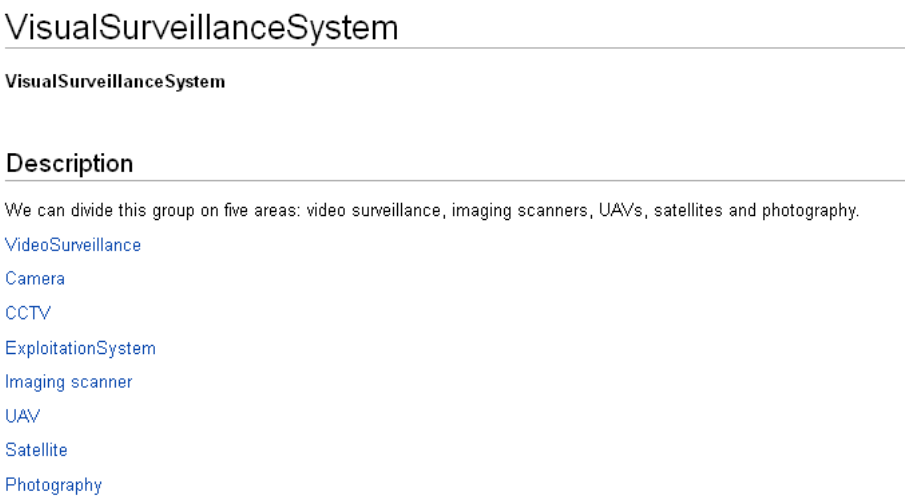

Figure 11 Wiki page generated for the visual surveillance taxonomy

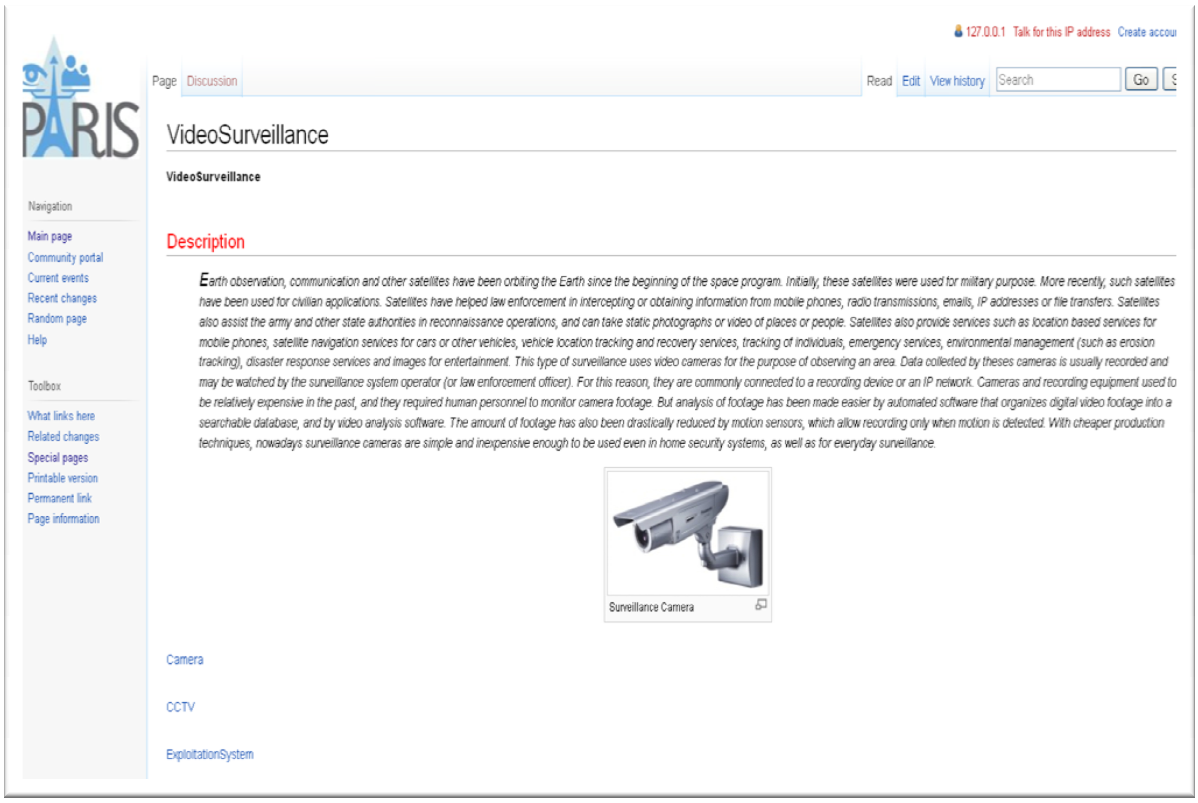

Figure 12 Page which describes one element of the taxonomy

4.2 Analysis of XML-based Graphical Representation through a PoC

The SFMT is devised to provide a series of objectives and tasks. Among them, the tool is also responsible for presenting the contents of SFIs (SALT Framework Instances), thus in addition to providing an interface that allows for the creation and modification of such SFIs, it must also show a correct representation of the information, in a way that any average user (not necessarily an expert) can understand and manage it. This means that the tool should fulfil the following properties:

- **Friendly:** several types of users will interact with the tool and we cannot assume that all of them will be technological experts. Therefore, the tool should integrate an interface apt for all types of users.
- **Adaptable:** the tool interface should not allow performing the same actions, neither shows the same information, to all users. Hence, the interface should be different for each type of users depending on their privileges.
- **Intuitive:** it is also desirable to provide an interface as simple as possible, thus the tool usage is intuitive enough. Thanks to this, even a user who has never used the tool could do it with ease.

According to these properties, it seems logical to implement an interface directly related to the type of user and his usage. For this reason, it is necessary some sort of user recognition, e. g. an access control in order to identify the current user privileges. Figure 13 shows a possible presentation for user identification.

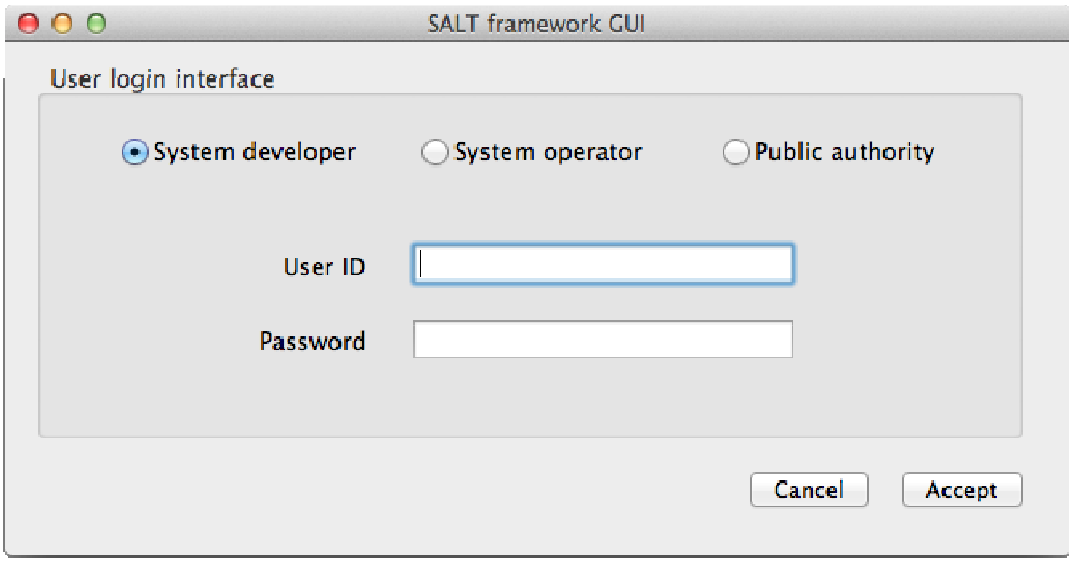

Figure 13 User identification interface

Moreover, the SFMT will also present different interfaces depending on the type of action that is going to carry out:

- **Creation:** this interface will help users (typically experts in one of the following areas: social, ethical, legal, technological) to create SFIs containing relevant information regarding their area of expertise. It must show a proper representation of the fields and types of information that have to be stored within an SFI, indicating what information is mandatory and what it is not, making clear to the users the process of introducing their concepts and concerns to the SALT framework. At this point, the tool can also provide mechanisms to integrate certain policies that could be requested by given repositories when creating an SFI, e. g., a repository may require to provide an author for every SFI that it holds, thus there is no anonymous information stored.
- **Search:** this is a very important interface. It will guide the user in the task of locating the SFIs that may apply to a corresponding surveillance system. To achieve this goal, it must provide an input mechanism allowing the user to introduce some system specifications, context and scenario characteristics. The entered information will determine how the SFMT will access the corresponding SFIs.
- **Modification:** the interface that helps the user to modify an already existing SFI is a mix of the two previous interfaces, since it first has to search the SFI that has to be updated, and then it creates a new one with the modified information. The deletion of the old one will depend on its current validity.
- **Presentation of information:** this possibly is the most important interface, since it is the one in charge of presenting to the user all SALT concepts and concerns that are stored within the SFIs that have been selected. We remark again that the SFMT does not take any design decision by itself, it just presents to the user relevant information that he may take into account for the development of a determined surveillance system. According to this information, the system designer will take the responsibility of taking any decision. As a consequence, the representation of this information and how it is shown is a key element within the SFMT, which will have to show a format attainable for every type of user.

Figure 14 and Figure 15 show two different possible interfaces that allow a user to search the appropriate SFIs. The first one is based on several lists of parameters where the user has to select the proper values; the second one is destined to those users who already know the exact value of the possible parameters that define a surveillance system. Here, we remark that the SFMT is still in an early state of development, and hence these interfaces still need of refinement.

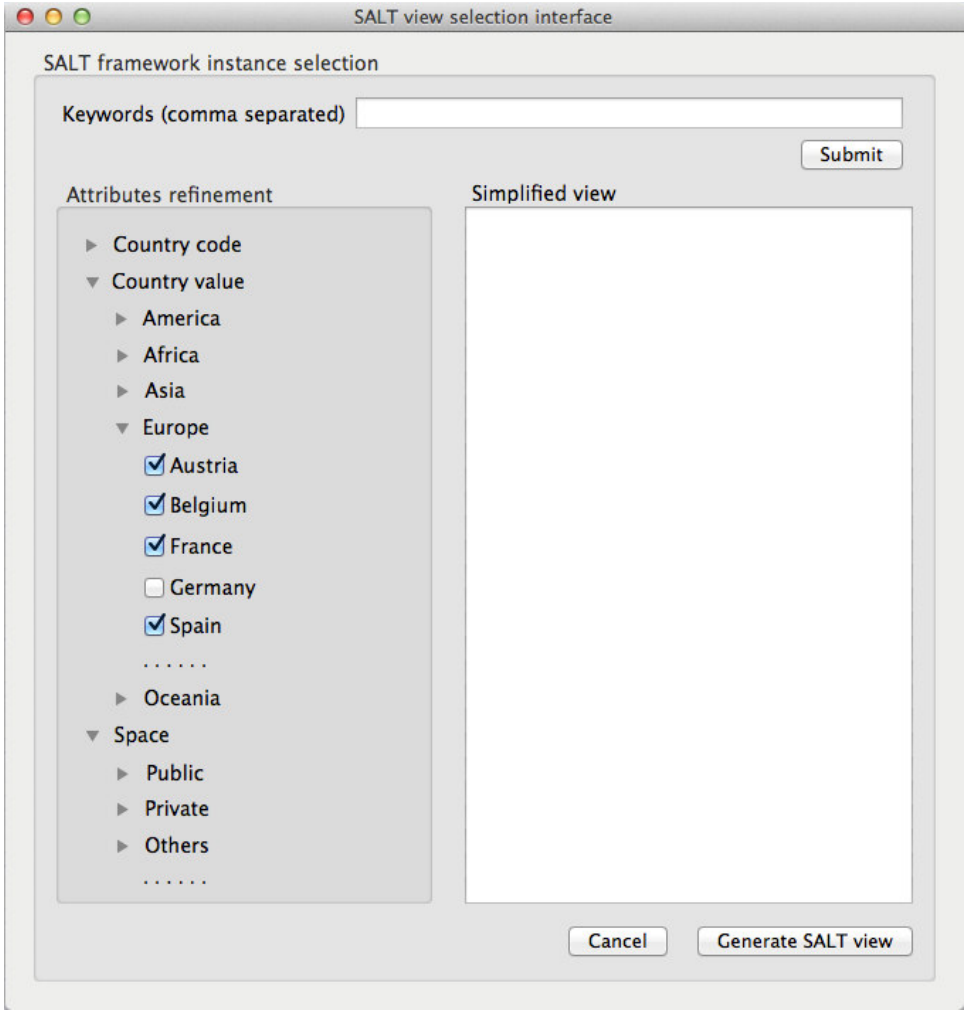

Figure 14 Searching interface with unfolding options

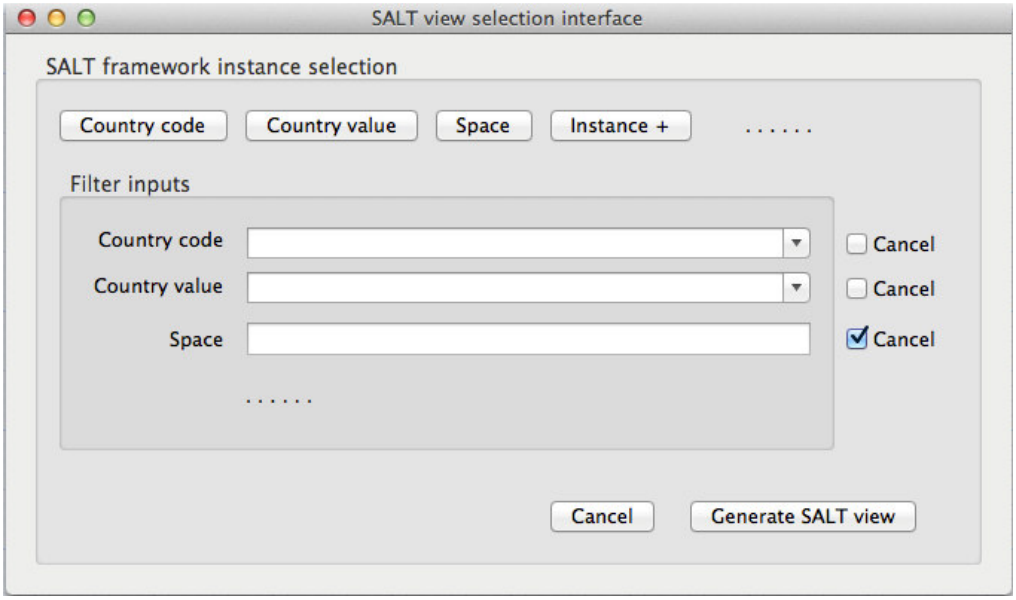

Figure 15 Searching interface using known parameters

4.3 Future steps

The sketches presented in this section were initial works allowing partners to understand other domains. For instance, technical people wish a tool able to capture in digital representation all legal information in structured way (i. e. some templates with keywords and the legal expert can fill out). In the opposite way, the tools developed at the early phase in PARIS provide some ideas to non-technical people of the computer science limitations.

For the next steps, the project will continue to develop some sketches in order to provide direct feedbacks from the experts. In particular, some concepts defined in the document D2.2 will be instantiated (e.g., questionnaire). A traditional tool development process will be used. The architecture and implementations will be defined and realised during the next tasks of work package WP3.

5 Conclusion

All concepts defined in the SALT Framework have to be implemented through some tools, called SFMT (SALT Framework Management Tools). This document provides a status on the stakeholder who will use them and their expectations. Basically, SFMT has 4 meta-functionalities which can be decomposed in smaller functions:

- Managing the SALT Knowledge Creation
- Managing the SALT Knowledge Analysis & Representation
- Managing the SALT Knowledge Repository
- Managing the SALT Knowledge Application

For all functionalities defined above, some requirements have been identified. These requirements will be the basis for defining the architecture of SFMT.

Finally, we made an analysis on the representations which can be used for implementing SFMT. Models will be used in order to store the knowledge in a repository. However, it is necessary to implement different front-ends. In particular, we used two approaches: (i) a wiki which is very simple to use without capturing constraints and (ii) a GUI which is conformed to the schema used for storing data in a repository. Both solutions are complementary and have different users (e.g., easier to convince non-technical people to use a wiki instead of more structured tool which are usual for technical users).