Multi-layered Security Technologies

for hyper-connected smart cities

D3.4: M-Sec Architecture: Functional and technical specifications – final version

Grant Agreement No. 814917

Multi-layered Security technologies to ensure hyper-connected smart cities with Blockchain, BigData, Cloud and IoT

Project acronym	M-Sec
Deliverable	D3.4 M-Sec Architecture: Functional and technical specifications – final version
Work Package	WP3
Submission date	30 June 2020
Deliverable lead	Orfefs Voutyras (ICCS) / Kenji Tei (WU)
Authors	Orfefs Voutyras (ICCS), Antonis Litke (ICCS), George Palaiokrassas (ICCS), Koumoto Takafumi (NII), Arturo Medela (TST), Akira Tsuge (KEIO), Tadashi Okoshi (KEIO), Jin Nakasawa (KEIO), Xavier Cases (WLI), Vanessa Clemente (WLI), Aamir Bokhari (YNU), Kenji Tei (WU), Mathieu Gallisot (CEA), Levent Gurgen (CEA), Keiko Doguchi (NTTE), Sonia Sotero Muñíz (AYTOSAN)
Internal reviewer	Xavier Cases (WLI), Koumoto Takafumi (NII)
Dissemination Level	Public
Type of deliverable	R

The M-Sec project is jointly funded by the European Union's Horizon 2020 research and innovation programme (contract No 814917) and by the Commissioned Research of National Institute of Information and Communications Technology (NICT), JAPAN (contract No. 19501).

Version history

#	Date	Authors (Organization)	Changes
v0.1	12 March 2020	Orfefs Voutyras (ICCS)	Initial ToC and Content
v0.2	20 March 2020	Orfefs Voutyras (ICCS)	Section 1 finalised
v0.3	30 April 2020	Orfefs Voutyras (ICCS)	Section 2, 4 finalised
v0.4	13 May 2020	All technical & UC partners	Section 3 content provided
v0.5	10 June 2020	All technical & UC partners	Section 3: final results extracted and confirmed
v0.6	17 June 2020	Orfefs Voutyras	Executive Summary, Conclusions finalised
v0.7	22 June 2020	Orfefs Voutyras (ICCS)	Version for internal review
v0.8	25 June 2020	Xavier Cases (WLI)	Internal Review
v0.9	26 June 2020	Takafumi Koumoto (NII)	Internal Review
v1.0	30 June 2020	Orfefs Voutyras (ICCS)	Version ready for submission

Table of Contents

Version history 3
Table of Contents 4
List of Tables 5
List of Figures 5
Glossary 6
Executive Summary 7
1. Introduction 8
1.1 Scope of the document 8
1.2 Relation to other WPs and Tasks 9
2. Methodology & Individual Results 10
2.1 Overall Methodology followed 10
2.2 Step 1: What – Functionalities and Functional Groups 13
2.3 Step 2: Where – Infrastructure Layers 16
2.4 Step 3: When – Designing, Deployment and Operational Phase 17
2.5 Step 4: Who – Partners and Tasks 17
2.6 Step 5: How – Links and Functional Flows 17
2.7 Step 6: Why – Requirements, Use Cases and Objectives 18
3. Final Results – Functional View 19
3.1 Assets' categorisation & Overall Architecture 19
3.2 Architectural view for the pilots 22
3.3 Architectural view in the case of external systems 26
4. Final Results – Secondary Models & Views 27
4.1 Domain Model 27
4.2 Physical Entity & Context Views 30
4.3 Information Model 33
5. Conclusions 34

List of Tables

Table 1: Assets' details summary – What, Who, Where and When 19
Table 2: Assets' details summary – Why 20

List of Figures

Figure 1—1: Relation of T3.1 and D3.2 to other WPs and Tasks 9
Figure 2—1: M-Sec Requirements Management methodology 10
Figure 2—2: M-Sec System Architecture Definition lifecycle 12
Figure 2—3: IoT ARM "native" Functional View (from IoT-A) 14
Figure 2—4: M-Sec components mapped to the IoT ARM "native" FGs 14
Figure 2—5: M-Sec components mapped to the M-Sec FGs 15
Figure 2—6: M-Sec components mapped to Layers (and the M-Sec FGs) 16
Figure 3—1: M-Sec System 5W1H analytical global Architecture View 21
Figure 3—2: M-Sec System 5W1H analytical pilot 1 Architecture View 22
Figure 3—3: M-Sec System 5W1H analytical pilot 2 Architecture View 23
Figure 3—4: M-Sec System 5W1H analytical pilot 3 Architecture View 24
Figure 3—5: M-Sec System 5W1H analytical pilot 4 Architecture View 25
Figure 3—6: M-Sec System 5W1H analytical extended Architecture View 26
Figure 4—1: UML Classes diagram of the M-Sec Domain Model 29
Figure 4—2: M-Sec instantiated IoT Domain Model for Pilot 1 32
Figure 4—3: IoT Information Model as defined in the IoT ARM 33

Glossary

Acronym	Description	Acronym	Description
5W1H	Who, What, When, Where, Why, How	RA	Reference Architecture
AE	Augmented Entity	RF	Reference Model
ARM	Architectural Reference Model	RFID	Radio-frequency identification
D	Deliverable	r-IoT	resource-centric IoT
FC	Functional Component	RM	Reference Model
FG	Functional Group	SotA	State of the Art
FV	Functional View	T	Task
GVE	Group of Virtual Entities	UC	Use Case
i-IoT	Integrated IoT	UML	Unified Modelling Language
IoT	Internet of Things	UNIs	Unified requirements
MVP	Minimum Viable Product	VE	Virtual Entity
PE	Physical Entity	ve-IoT	VE-centric IoT
RDF	Resource Description Framework	WP	Work Package
QR code	Quick Response Code	XML	Extensible Markup Language

Executive Summary

The work described in this deliverable (D3.4) was carried out in the framework of WP3 – "Requirements, architecture, hyper-connected smart city". The report presents the final version of the document describing the overall architecture of M-Sec (analysis, design and specification).

The output of this deliverable is of high importance for the other WPs, but also for the project itself, since it associates the users' requirements and the use case scenarios with specific functionalities and building blocks and provides a common framework for all the technical tasks of the project to work on, in order to provide the final M-Sec system. Therefore, from the beginning of the project, WP3 decided to follow, at the level that this is possible, well-established and successful methodologies for the design of concrete IoT Architectures.

All technical partners involved in this task collaborated and provided input regarding the expected architecture to meet the objectives set out in the project, especially with regard to novel Security aspects in IoT contexts. Every partner focuses on the individual models that they are responsible for during the implementation phase of WP4 and supports the integration activities, while following the common framework set by D3.4.

The structure of this document is simple and mainly follows the one presented in Section 2. Section 1 gives an introduction to the scope of this document and its relation with other WPs and Tasks. Section 2 extensively presents the overall methodology followed to produce the final results of this deliverable (M-Sec Architecture) as well as the specific individual steps followed to acquire all the required information. Section 3 is the most important section of the document, as it summarises the main results of the document and Task 3.2 in general. Section 4 provides a view to some secondary results of the task. Finally, Section 5 concludes the document.

Regarding the differences between 'D3.3 M-Sec Architecture: Functional and technical specifications – first version' and 'D3.4 M-Sec Architecture: Functional and technical specifications – final version':

- Section 1 has remained more or less the same.
- Section 2 is considerably extended and presents the overall methodology followed for the extraction and presentation of all the information required for the functional views of the M-Sec system. The corresponding Section 2 of D3.3 is now part of Section 2.2. Some indicative results are presented (mainly depicting specific steps of analysis).
- Section 3 corresponds to Section 6 of D3.3. This section provides an updated view of the functional flow of the M-Sec system and includes a lot of extra information that was not included in Y1 (specified FGs, layered view, etc.)
- Finally, Section 4 corresponds to Sections 3, 4 and 5 presented during Y1 and have no new content.

All in all, the deliverable is considered to have provided a concrete framework that has been supporting all the design, development, and integration activities of the project.

1. Introduction

1.1 Scope of the document

The current document is the deliverable D3.4 'M-Sec Architecture: Functional and technical specifications – final version' and comprises the final (and major) outcome of Task T3.2 – M-Sec Architecture. The report presents the final version of the document describing the overall architecture of M-Sec and its basic building blocks.

The main scope of the document is to enable the consortium to agree on a common vocabulary, framework and methodology to rely on for specifying the M-Sec system. That way, all partners adopt the same language and have for instance all components described in the same manner, utilising the same viewpoints. This helps in the designing process and in reaching a common understanding for the project's needs and expected results, as well as the means to achieve them. To that end, well-established and successful methodologies for the design of concrete IoT Architectures were used, such as the IoT Architectural Reference Model (ARM) from the IoT-A FP7 project[1].

This approach ensures the continuous alignment with technical and stakeholders' requirements, conformance with state-of-the-art technologies and the proper definition of the full appropriate functionality of M-Sec. It also ensures interoperability between different systems that have to be integrated, allows for openness of the interfaces and data formats and exchange approaches, and ensures a high level of Security and Privacy, so that the system is trustworthy by the stakeholders and users.

This deliverable contains all the Architectural Views required for the coordination of the partners and also presents extensively the process through which those views were extracted. As such, the primary audience of this document consists of the members of the consortium that participate in the design and development of the components and modules of the M-Sec pilots as well as of the M-Sec core system per se. Additionally, the document is of wider interest to stakeholders that are active in the domains of Smart Cities, IoT, Security and Big Data, including researchers participating and contributing to H2020 projects under the aforementioned topics, especially to the ones that are expected to use or extend the results of M-Sec in the future.

[1] https://cordis.europa.eu/project/id/257521

1.2 Relation to other WPs and Tasks

The following figure gives an overview of the relations of this deliverable and the corresponding Task (T3.2) to other WPs, Tasks and deliverables. The figure is focused on T3.2, so not all relations between the rest of the tasks are presented.

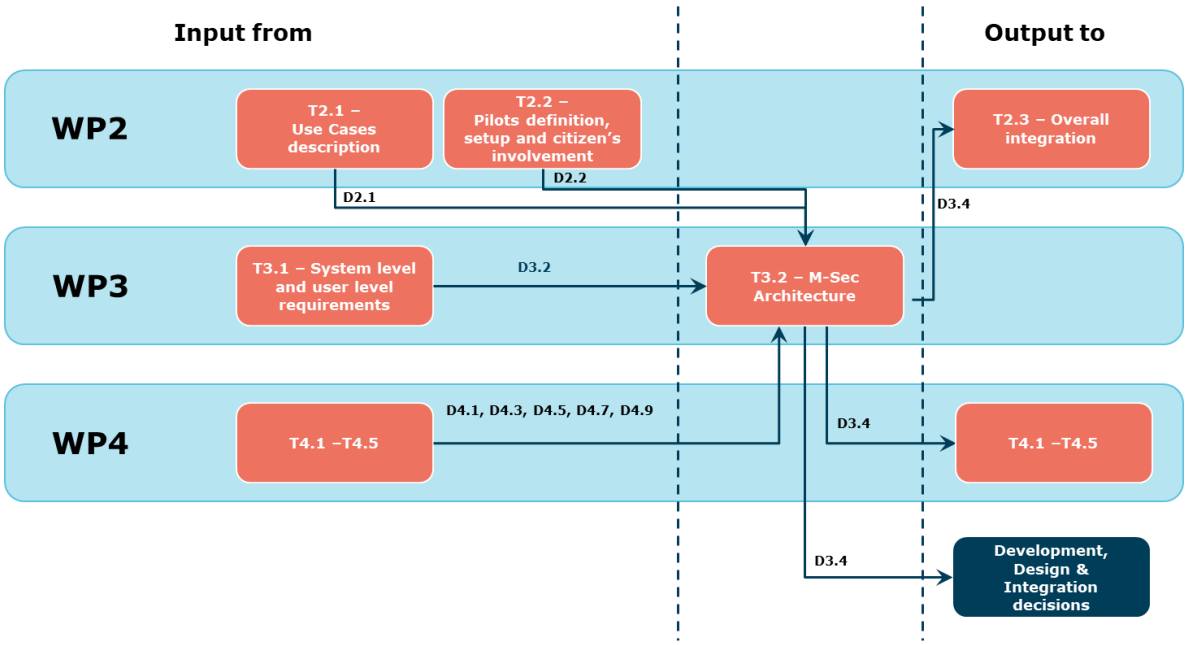

Figure 1—1: Relation of T3.1 and D3.2 to other WPs and Tasks

Task 3.2 receives input from WP2 and in particular from Tasks "T2.1 - Use case description" and "Task 2.2 - Pilots: definition, setup and citizens involvement" through the corresponding deliverables (D2.1 and D2.2). More specifically, these deliverables provide in a holistic way an overview of the use cases description along with particular details on their implementation within the pilots. To do so, D2.1 provides a full analysis of the use cases covered in the M-Sec project. D2.2 then describes the different actors/stakeholders involved, the functions and conditions that need to be offered, how, when and where the pilots will be set up, and the plan for engaging and committing the participation of the citizens and the stakeholders. Input is also received from "Task 3.1 - System level and user level requirements analysis" which focuses on the requirements analysis from potential end-users of the M-Sec platform, including both corporate users and citizens. Also, T3.2 takes as input from WP4 information about the capabilities and technical specifications of the available assets, information that can be used to extract technical requirements (focused on assets constraints) but also ideas about the fulfilment process of requirements.

On the other hand, the results of this deliverable will be used within WP4 which will be focusing on the development and adoption/adaptation of tools upon which the implementation of the M-Sec IoT Security layer, cloud/data Security layer, application level Security layer and blockchain framework will be based on. Naturally, the same results will be used in order to achieve end-to-end Security by "Task 2.3 - Overall Integration" which is focused on the integration of components and modules developed and deployed in WP4 and will generate an integrated system. Finally, some parts of the document are related to the exploitation too (see D5.7).

2. Methodology & Individual Results

2.1 Overall Methodology followed

The following figure gives an overview of the overall **System Architecture Definition methodology** devised and followed in order to complete Task 3.2 successfully and provide valuable results for other Tasks.

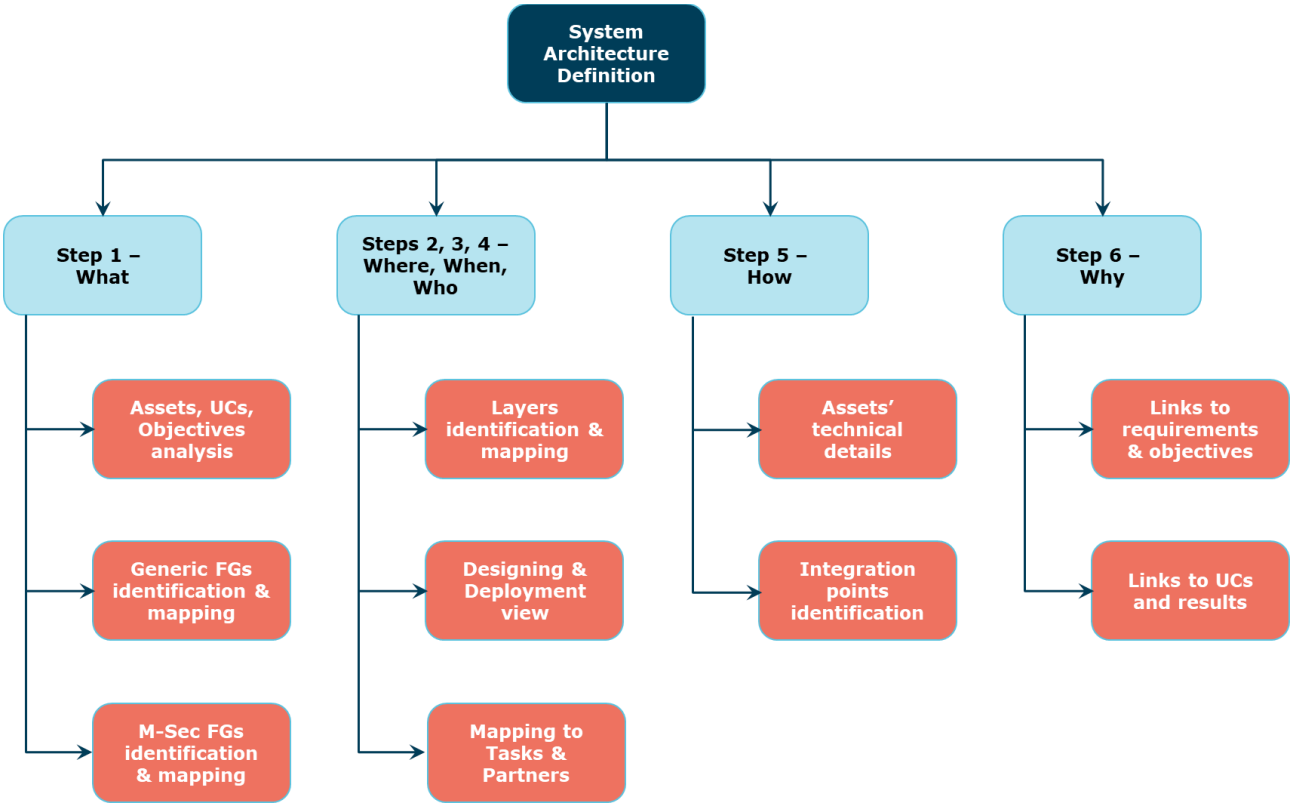

Figure 2—1: M-Sec Requirements Management methodology

The methodology is based on the **5W1H approach**. The 5W1H (Five Ws and How, 5W1H, or Six Ws) are questions whose answers are considered basic in information gathering or problem solving. They are often mentioned in journalism, research and police investigations. According to the principle of 5W1H, a report can only be considered complete if it answers these questions starting with an interrogative word: Who, What, When, Where, Why, and How.

The "problem" to be solved in the case of this task is the definition of the M-Sec System Architecture. The five Ws and 1 H are asked for each and every identified component (asset), and thus the corresponding steps also take place, having on their centre each component (and in some cases, specific groups of components and sub-systems). The aggregation of the answers to those questions provide the final results for the definition of the M-Sec system as a whole.

Some details about each step are presented below. More details about each individual step are presented in the sections to follow.

- **Step 1 – Answering to WHAT the components and the sub-systems can do:** This step includes all the activities required to define the functionalities that the M-Sec components provide. Such activities include the study of the SotA, identification of the assets available by the partners of the consortium, description of the said assets, identification of their functionalities, categorisation of assets based on those functionalities, and identification of groups of assets with similar functionalities (resulting to the identification of Functional Groups and sub-systems). Further details about the methodology followed for this step are presented in Section 2.2.
- **Step 2 – Answering to WHERE the components "run":** In the scope of M-Sec, we transform this question to "At which (infrastructure) layer does each component run?" and identify elements which will act as the baseline for the definition of a layered architecture. This step includes all the activities required to define such a layered system (such as the study of the corresponding literature and the identification of the actual infrastructure expected to be used within the scope of the project) and mapping the components to it. Further details about the methodology followed for this step are presented in Section 2.3.
- **Step 3 – Answering to WHEN the components "run":** This step is pretty straightforward compared to the previous ones and focuses on identifying at what phases of the project (and the lifetime of the corresponding system) each component is expected to be used or be active. Thus the only activity of this step is the categorisation of the components based on the said phases. Further details about the methodology followed for this step are presented in Section 2.4.
- **Step 4 – Answering to WHO is responsible for a component:** This step is focused on assigning each component to a specific partner of the consortium and to a specific technical Task under which the said component will be developed and be integrated into the overall system. Further details about the methodology followed for this step are presented in Section 2.5.
- **Step 5 – Answering to HOW the components and the sub-systems work:** This step includes an analysis on a components-level and on a system-level. In the case of the components-level analysis, how each component actually works is identified. Such technical details are provided in the deliverables of WP4. In the case of the system-level analysis, how the components are link with each other is identified. In other words, in this step, the several sub-systems and functional flows make their appearance. The identification of the links between the components are the main subject of Task 2.3 'Overall Integration' (in the deliverables of which these links are also mentioned as integration points). Further details about the methodology followed for this step are presented in Section 2.6.
- **Step 6 – Answering to WHY the components must do what they do–** This step is the main one that links the technical solution presented through the results of the previous steps with the actual scope of the project. It includes all those activities that link the requirements of the use cases and the overall scope and objectives of the project to the specific solution presented in the architecture. Further details about the methodology followed for this step are presented in Section 2.7.

The order of the steps and activities described above is not indicative of their chronological order. Most of the activities and the progress towards the completion of these steps have been taking place in parallel during the whole lifetime of the project. Moreover, even though the activities are presented under different steps, most of them are highly interlinked with each other, being dependent and providing continuous feedback to each other.

The following figure presents an indicative overview of several stages (activities that actually took place in chronological order) implemented in T3.2 to complete the **System Architecture Definition Lifecycle**.

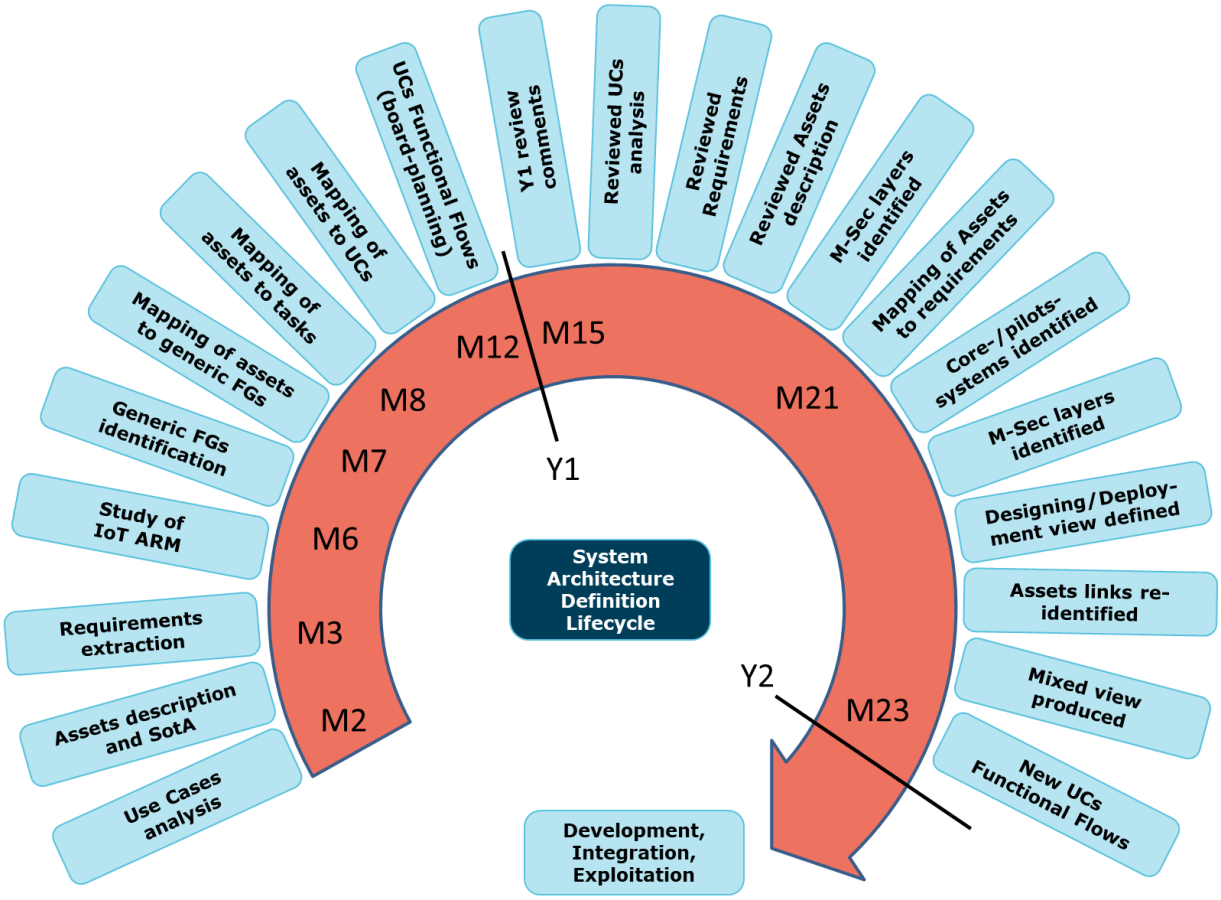

Figure 2—2: M-Sec System Architecture Definition lifecycle

During Y1, the groundwork for the System Architecture Definition was set and the first outcomes of the Task were produced. All assets were mapped to specific UCs and their links between them were also identified. This supported the WP4 development activities that were initiated in M10, as well as the integration activities that followed up.

During Y2, the activities of T3.2 got solidified even more and the interlinkage between all the related WPs, tasks and activities became possible due to other parts of the project reaching a point of maturity. During this year, the M-Sec specific Functional Groups (FGs) were identified, partly through the new categorisation schema of Requirements (see Groups of Requirements in presented in D3.2). This made it possible not only to directly connect the results of T3.1 to T3.2, but also to expand the architectural views already available and redefine the integration points between assets. Several other ways to categorise the assets were identified.

All those results are summarised in Figure 3—1, the main outcome of this deliverable.

2.2 Step 1: What – Functionalities and Functional Groups

As it was mentioned in the previous section, to identify WHAT the M-Sec system does, it is necessary to study it from both a components-centric and a groups-centric point of view. The description of individual components is already well documented in other deliverables.

Grouping components together based on their common capabilities results in the extraction of Functional Groups. By identifying these FGs and mapping the components to them, the answer to what the system provides as a whole is completed.

Initially, during Y1, several architectural reference models were studied in order to find a grouping system that could be adapted into the M-Sec concept. As stated in Section 1, the IoT Architectural Reference Model (ARM) from the IoT-A FP7 project[2]. Through this model, generic FGs were identified, providing a first approach towards grouping the several M-Sec components.

The IoT-A ARM proposes the following list of generic FGs:

- **IoT Process Management FG:** The purpose of the FG is to allow the integration of process management systems with the IoT platform;
- **Service Organisation FG:** This FG is responsible for composing and orchestrating services, acting as a communication hub between other FGs. It contains the Service Choreography Functional Component which "offers a broker that handles Publish/Subscribe communication between services";
- **Virtual Entity FG:** This FG relates to Virtual Entities (see Section 4) containing functions such as discovering VEs and their associations with Resource-centric IoT-services. Through this FG can be accessed also the VE-centric IoT Service like for instance the ones related to experience sharing.
- **IoT Service FG:** The IoT Service FG contains functions relating to r-IoT Services. Those services expose the resources like sensors and actuator and provide a mean for reading sensor value or actuating. It also contains storage capabilities functionality. More specifically the ARM states that "A particular type of IoT Service can be the Resource history storage that provides storage capabilities for the measurements generated by resources";
- **Communication FG:** The Communication FG is used to abstract the communication mechanisms used by the Devices. Communication technologies used between Applications and other FGs are out of scope of this FG, as these are considered to be typical Internet technologies;
- **Security FG:** The Security FG *"is responsible for ensuring the Security and privacy of IoT-A-compliant systems"*;
- **Management FG:** The Management FG contains components dealing with Configuration, Faults, Reporting, Membership and State. It should be mentioned here that this FG works in tight cooperation with the Security FG.

Figure 2—3 depicts these FGs, their typical Functional Components (FCs) and the relations between them (nearby blocks are considered linked), while Figure 2—4 depicts how the available M-Sec assets map to these generic various FGs.

[2] https://cordis.europa.eu/project/id/257521

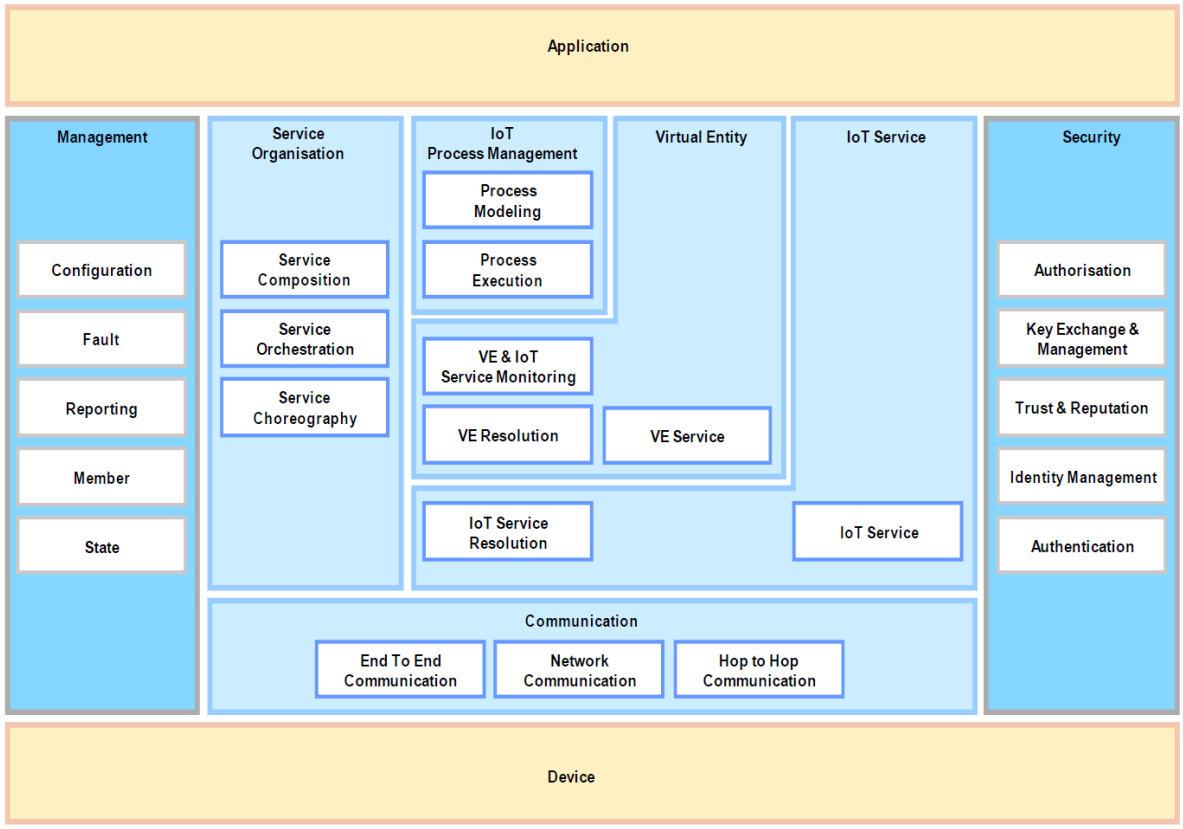

Figure 2—3: IoT ARM "native" Functional View (from IoT-A)

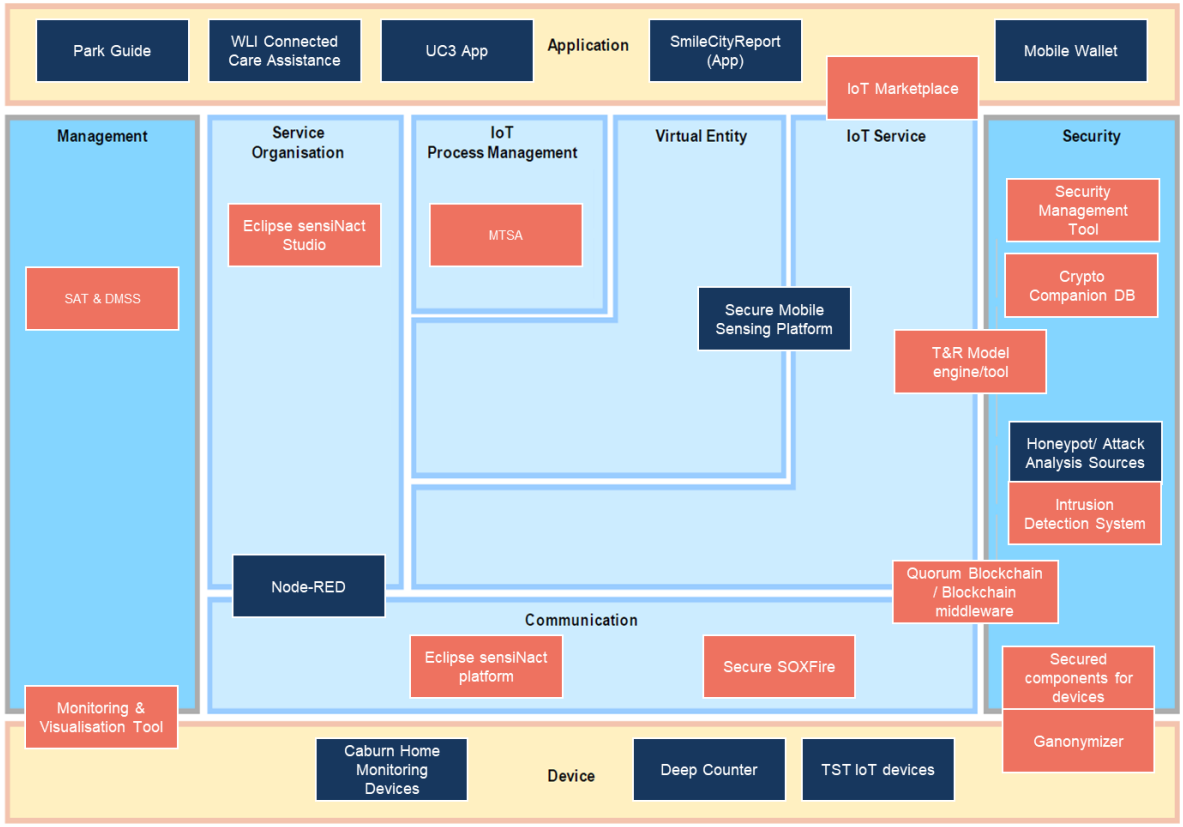

Figure 2—4: M-Sec components mapped to the IoT ARM "native" FGs

The “generic” IoT Functional View proposed by IoT-A (Figure 2—3) is very important as it proposes a “layered” model of Functional Groups (FG), which maps to most of the concepts introduced in the IoT domain, together with a set of essential Functional Components (and associated interfaces) that an IoT system should provide. Also, by studying Figure 2—4, it can be noticed that the all of the Functional Groups are covered by M-Sec components (the colour-coding of the components is explained in Section 2.7). This gives an indication of the fact that the M-Sec architecture to be presented later on is not lacking any components or generic capabilities and can be readapted to most IoT systems. This figure is an updated version of the one presented during Y1 in D3.3, Section 6.3.

During Y2, the main focus was given on going beyond generic FGs and actually extracting M-Sec “native” FGs, based on the specific needs and capabilities of the project (Figure 2—5). The identified FGs are: Applications FG, Development & Designing Tools FG, Devices FG, IoT Data Marketplace FG, Storage FG, Development & Security Designing Tools FG, Devices Security FG, Privacy Management Tools FG, Secure & Trusted Storage FG, Secure City Data Access FG, End-to-End Security.

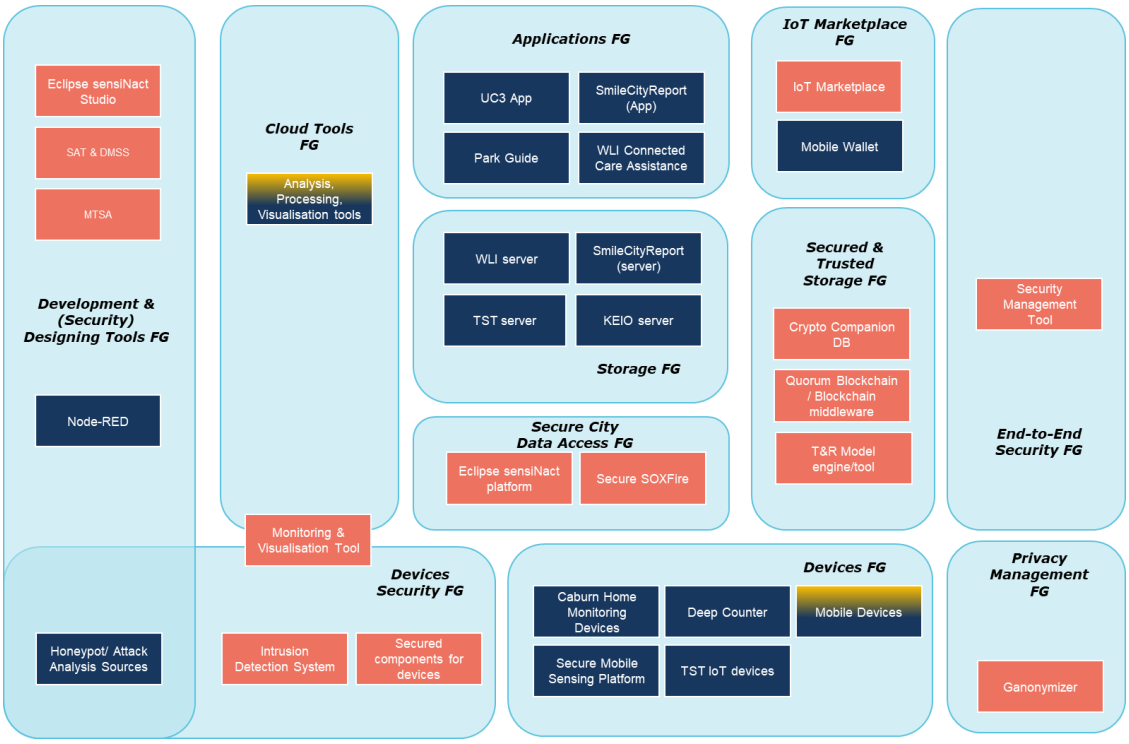

Figure 2—5: M-Sec components mapped to the M-Sec FGs

The identification of these FGs became possible in two ways: First of all, the several components were “naturally” grouped in specific FGs based on their nature and common capabilities they offer. Secondly, the rationale behind identifying these specific FGs was further supported through the Requirements Consolidation and the Requirements Fulfilment processes (presented in D3.2, Sections 4 and 6). For example, Requirement R2.2-6.6-1 states that “Data collected from home sensors MUST be sent over a secure channel and MUST not be tampered”. This requirement’s Group is Security/Privacy and its Sub-Group is “Data Transfer”. As such, it can be easily realised that Secure City Data Access FG is necessary for the project and that the corresponding components belonging to this category (based on their description) are Eclipse sensiNact platform and Secure SOXFire. Following a similar approach for all assets and requirements makes it possible to identify all of the required FGs and the components mapped to them.

2.3 Step 2: Where – Infrastructure Layers

This step focused on identifying at which specific layers the several components of M-Sec run, so as to provide a layered view of the project. During Y2, a lot of practical security and non-security architectures of other projects and initiatives (non-generic ones, unlike the IoT ARM) were studied. After taking under consideration several 3-, 4-, and 5-layered IoT architectures[3], it was decided to follow a 4-layered architecture, the layers of which are extracted roughly from an infrastructure point of view. The layers are:

- **IoT Layer:** This layer includes all the devices, sensors, and their corresponding low-level networks that are met in IoT ecosystems. Similar concepts/ names found in the literature are "Perception Layer" and "Sensor Layer". It also includes the "edge".
- **Cloud Layer:** This layer, as the name suggests, is the layer that includes all the infrastructure upon which cloud services are based.
- **Application Layer:** This layer defines all the use the IoT technology or in which IoT has deployed.
- **Middleware Layer:** The layer in between the IoT and the Cloud one. It connects the two layers. In the literature, a similar/related concept found is the Network Layer and the Communication Layer. Within M-Sec, this layer includes the infrastructure needed for both Peer-to-Peer and IoT-to-Cloud approaches.

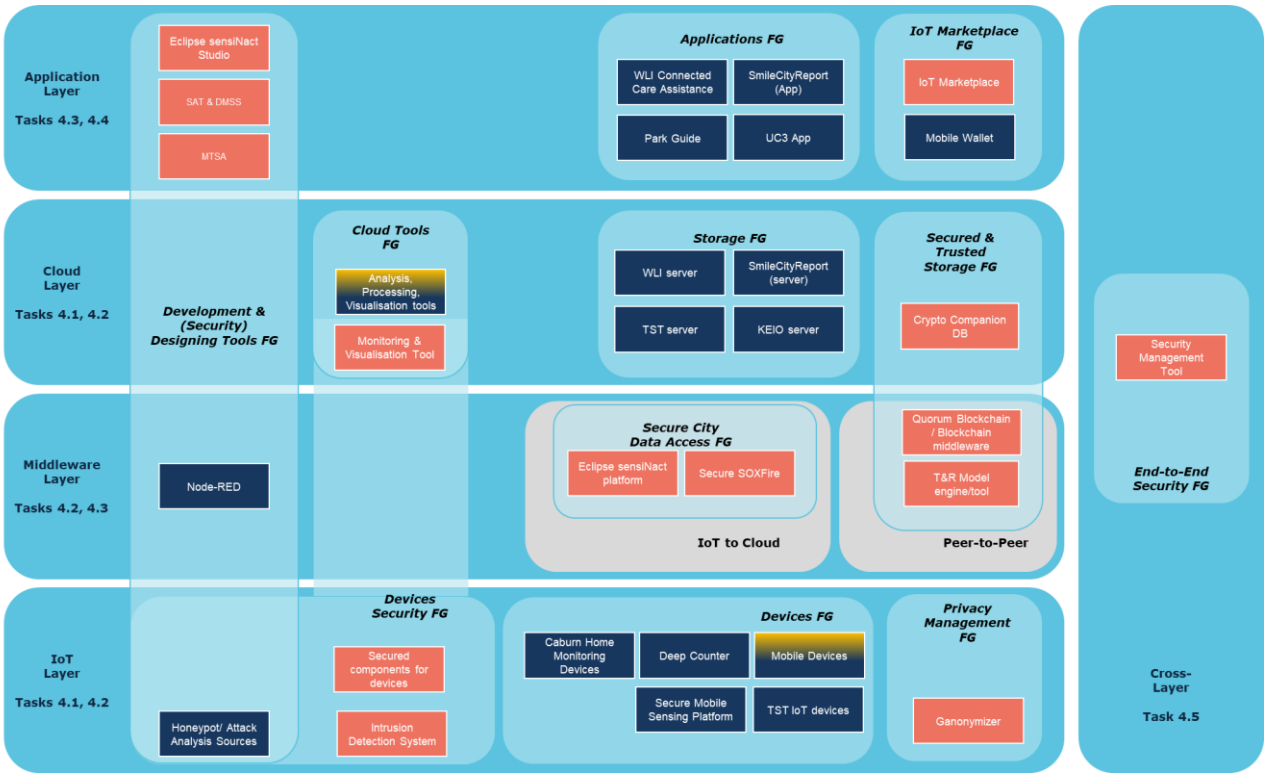

Figure 2—6: M-Sec components mapped to Layers (and the M-Sec FGs)

It should be noted that hints about this categorisation of components based on such layers have been given since the beginning of the project (see layered "architecture" in WP4 deliverables during Y1). However, the grouping of the M-Sec components in these specific layers as defined/identified here is now official.

[3] https://www.mdpi.com/1424-8220/18/9/2796/htm

2.4 Step 3: When – Designing, Deployment and Operational Phase

Three time phases are identified, during which specific assets can be used: the designing phase (which includes tools used of the overall design of the final system), the deployment phase (which includes tools used only to facilitate the deployment of the final system), and the operational phase (which includes all the tools that are active run-time, while the final system is active).

In the context of M-Sec, only one FG has been identified to be part of the designing phase, while no FGs have been identified that are used only in the deployment phase. The FG used only during the designing phase appears in Figure 3—1 with a red border.

2.5 Step 4: Who – Partners and Tasks

This step consists of two parts: identifying the consortium partner that is mainly responsible for each component and mapping each component to the corresponding joint development and integration activities (in other words, mapping the component to a task that belongs to one of the technical work packages). By default, each technical partner is responsible for the assets that they bring to the project. Regarding the distribution of components among the technical tasks, this can be seen in detail Table 1.

Core-system assets (see Section 2.7) belong to one of the Tasks of WP4 (T4.1 – IoT Security, T4.2 – Cloud and data level security, T4.3 – P2P level security and blockchain, T4.4 – Application level security, T4.5 – Overall end-to-end security), while technical work on pilot-system assets takes place in T2.3– Overall Integration. It can be noticed that the WP4 tasks follow approximately the layered approach introduced in Section 2.3. However, regarding the security assets present in the project, it should be noted that the Tasks to which the assets belong identifies the Layer for which the assets are used, but not necessarily the layer at which the assets run. For example, the Monitoring and Visualisation tool runs on the Cloud Layer, but also secures the IoT Layer.

2.6 Step 5: How – Links and Functional Flows

In this step, it is identified how the overall system does what it does. To provide this level of information, it is necessary to identify how the several components are linked with each other, forming sub-systems and the overall system (analytical functional flow view). Links can also be identified between FGs (generic functional flow view). For the components level, the description has been provided. For the system level though, it is necessary to identify the connections, the **links** between the components and between the FGs.

There are several ways through which the links between the components have been identified. One is through studying each UCs and identifying the relevant components and their links through “board-planning”. Another one is by studying the requirements of each UC, identifying which asset fulfils each requirement and the recognising dependencies between components by searching for the corresponding dependencies between the requirements (see Annex in D3.2). Finally, a last method is by using the requirements spreadsheet (see D3.2) in order to identify for each requirement to which asset the requirement is relevant to and which asset covers the requirement. The link in this case will be between the “Related to” asset and the “Covered by” asset. All links can be identified by studying that spreadsheet. Likewise, links between FGs are identified. The results of this Step are shown in Figure 3—1.

2.7 Step 6: Why – Requirements, Use Cases and Objectives

While Step 1 (What) includes all the activities required to define the functionalities that the M-Sec components **can** actually provide, Step 6 (Why) includes all the activities required to identify all the functionalities that the M-Sec components **must** provide. It goes without saying that these two steps are interlinked and actually take place mostly in parallel, while a lot of their activities actually overlap. Both steps give feedback to each other, by Step 1 identifying at all times what the system can do and Step 6 providing the rationale behind the selection of these capabilities, by defining what the system must do.

The background information in Step 6 is provided by an analysis of the general scope and the objectives of the project, the Use Cases, and the corresponding requirements. Although all the previous Steps get input mainly from the technical deliverables of WP4 (see Figure 1—1), this Step is the main one which exploits the input provided by WP2 and Task 3.1.

The "why a component is used" question can be viewed from several viewpoints. One is related to explaining the necessity of an asset. This is done by proving the connection of the asset to the fulfilment of a requirement. This process is described in the previous section and corresponds to the Requirements Fulfilment process described in D3.2, Section 6.

Another way to answer to the "why" question is by showing why each asset (or sub-system) is used in a specific UC. The mapping of components to UCs is shown in Table 2.

Finally, another way to answer to the "why" question is by identifying to which M-Sec objectives the asset is linked. Three main objectives are identified, and thus three different categories of assets:

- **Core-system asset:** The M-Sec core-system is considered the system that will be the "minimum viable product" (MVP) of M-Sec, the system that will be freely available to users and other projects and initiatives. It is the exploitable part of the project and it contains all the main features that will deem M-Sec successful as a project. Those assets are linked to the core-system requirements presented in D3.2, Section 4. Those assets are coloured with red in Figure 3—1.
- **Pilot-system asset:** The M-Sec pilot-systems are considered the systems that are developed for the pilots of the project. Pilot-system assets are considered the assets that are used only for the sake of completing successfully the objectives of the pilots per se (and not of the project as a whole). These assets will not be freely available or exploitable at the end of the project. Characteristic examples are the specific devices used in the pilots. Those assets are linked to the pilot-system requirements presented in D3.2, Section 4. Those assets are coloured with dark blue in Figure 3—1.
- **External-system asset:** An external system is a generic system introduced in the study of the architecture, used to show the linkage of the core-system with other systems in the future, after the end of the project. The corresponding assets are coloured with yellow in Figure 3—1.

3. Final Results – Functional View

3.1 Assets' categorisation & Overall Architecture

Following the methodology of the previous section, Table 1 provides a summary of all the details for answering the "What", the "Who", the "Where" and the "When" questions related to the assets.

Table 1: Assets' details summary – What, Who, Where and When

Asset	Partner	FG	Layer	Task
SmileCityReport (App)	KEIO	Applications	Application	2.3
UC3 APP	KEIO	Applications	Application	2.3
Park Guide	TST	Applications	Application	2.3
Worldline Connected Care Assistance	WLI	Applications	Application	2.3
Node-RED	WLI	Development & Designing Tools	Middleware	2.3
Deep Counter	KEIO	Devices	IoT	2.3
Secure Mobile Sensing Platform	KEIO	Devices	IoT	2.3
TST IoT crowd-counting devices	TST	Devices	IoT	2.3
TST IoT environmental devices	TST	Devices	IoT	2.3
Caburn Home Monitoring Devices	WLI	Devices	IoT	2.3
Mobile Wallet	WLI	IoT Data Marketplace	Application	2.3
SmileCityReport (Server)	KEIO	Storage	Application	2.3
Honeypot (IoTPOT)	YNU	Development & Security Designing Tools	IoT	4.1
Secured components for devices	CEA	Devices Security	IoT	4.1
Intrusion Detection System (IDS)	YNU	Devices Security	IoT	4.1
Ganonymizer	KEIO	Privacy Management Tools	IoT	4.1
Monitoring & Visualization Tool	YNU	Devices Security	Cloud	4.1
Crypto Companion Database	WLI	Secure & Trusted Storage	Cloud	4.2
Eclipse sensiNact platform (and Studio)	CEA	Secure City Data Access	Middleware	4.2
Secure SOXFire	KEIO	Secure City Data Access	Middleware	4.2
IoT Marketplace	ICCS	IoT Data Marketplace	Application	4.3
Quorum Blockchain/ Blockchain middleware	ICCS	Secure & Trusted Storage	Middleware	4.3
T&R Model engine/tool	ICCS	Secure & Trusted Storage	Middleware	4.3
MTSA	WU	Development & Security Designing Tools	Application	4.4
SAT & DMSS	NII	Development & Security Designing Tools	Application	4.4
Security Management Tool	CEA	End-to-End Security	Cross-Layer	4.5

Similarly, Table 2 provides a summary of all the details for answering the "Why" question related to the assets. The table splits the assets in core-system and pilot-system ones and also identifies to which UC each asset is used.

Table 2: Assets' details summary – Why

Asset	System	UC1	UC2	UC3	UC4	UC5
SmileCityReport (App)	Pilot	no	no	no	yes	yes
UC3 APP	Pilot	no	no	yes	no	no
Park Guide	Pilot	yes	no	no	no	no
Worldline Connected Care Assistance	Pilot	no	yes	no	no	no
Node-RED	Pilot	yes	yes	yes	yes	yes
Deep Counter	Pilot	no	no	yes	no	yes
Secure Mobile Sensing Platform	Pilot	no	no	yes	no	yes
TST IoT crowd-counting devices	Pilot	yes	no	no	no	yes
TST IoT environmental devices	Pilot	yes	no	no	no	no
Caburn Home Monitoring Devices	Pilot	no	yes	no	no	yes
Mobile Wallet	Pilot	no	no	no	yes	yes
SmileCityReport (Server)	Pilot	no	no	no	yes	yes
Honeypot (IoTPOT)	Pilot	no	no	yes	no	no
Secured components for devices	Core	yes	no	no	no	no
Intrusion Detection System (IDS)	Core	no	no	yes	no	no
Ganonymizer	Core	no	no	yes	yes	yes
Monitoring & Visualization Tool	Core	no	no	yes	no	no
Crypto Companion Database	Core	yes	yes	yes	yes	yes
Eclipse sensiNact platform (and Studio)	Core	yes	yes	no	no	no
Secure SOXFire	Core	no	no	yes	no	no
IoT Marketplace	Core	yes	no	yes	yes	yes
Quorum Blockchain/ Blockchain middleware	Core	yes	yes	yes	yes	yes
T&R Model engine/tool	Core	yes	no	no	yes	yes
MTSA	Core	yes	yes	no	no	no
SAT & DMSS	Core	no	no	yes	no	no
Security Management Tool	Core	yes	yes	yes	yes	yes

Figure 3—1 aggregates all the above information and also provides details about the "How" through the presentation of the actual links between the assets (analytical view). The final result is a **mixed** view, since it combines the layered view (Where) with the FGs one (What). It is also a global, since it provides the overall architecture of M-Sec as a project, rather than providing just an instantiation of it (as in the case of pilots). Since the diagram includes all the information presented in the previous sections about all of the Steps, this architectural view is coded as the 5W1H one.

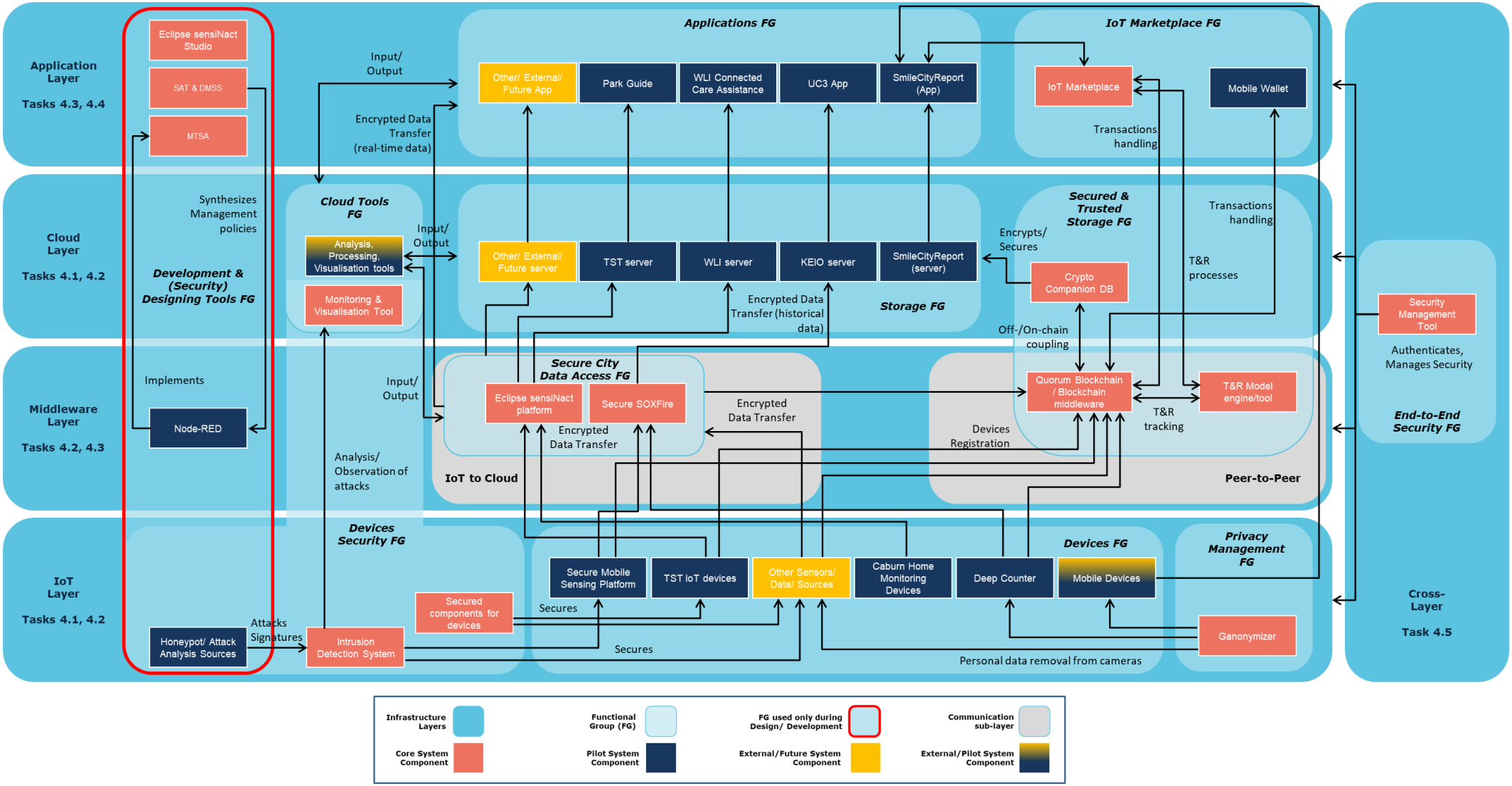

Figure 3—1: M-Sec System 5W1H analytical global Architecture View

3.2 Architectural view for the pilots

The following diagrams depict the mapping of the aforementioned assets to the Use Cases (as components), as well as the dependencies and relations between those assets.

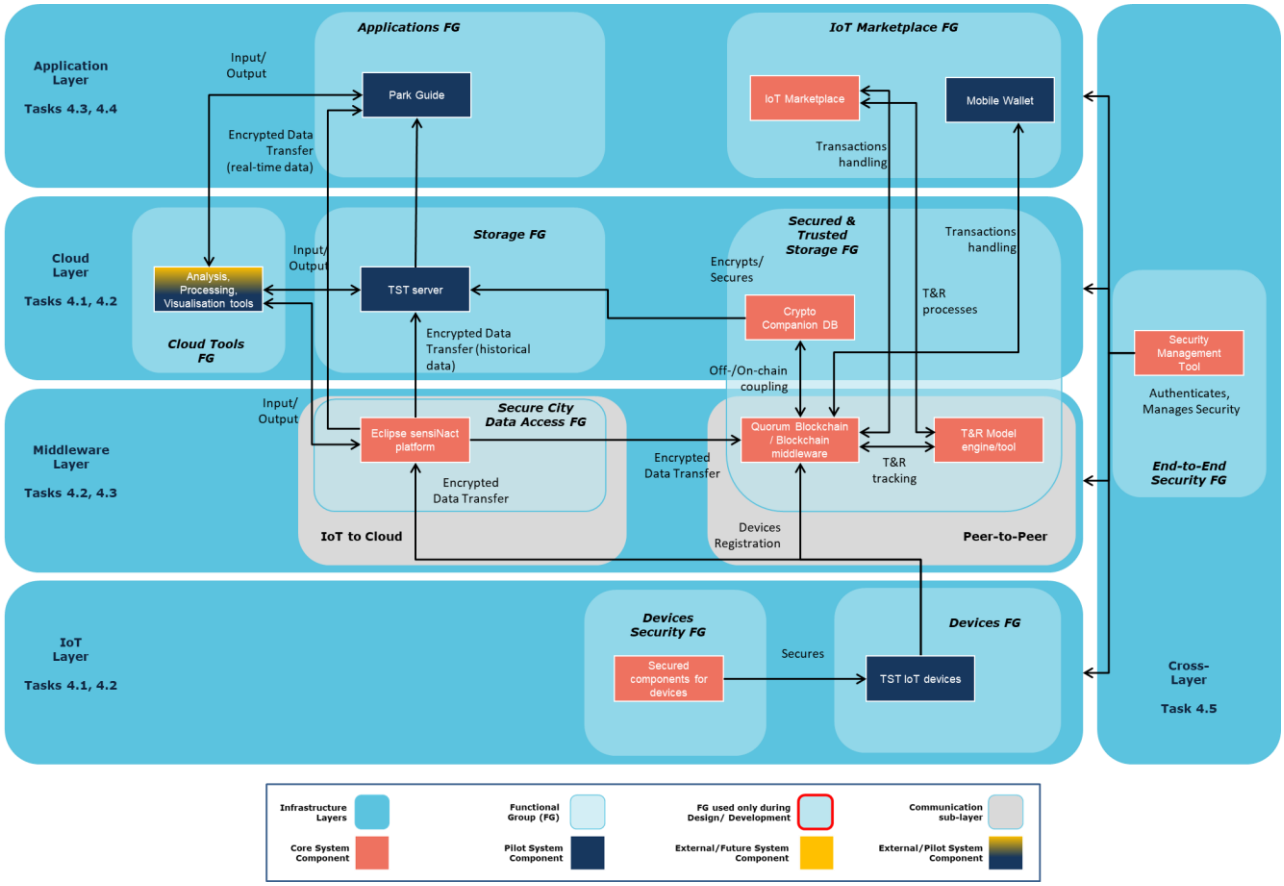

Figure 3—2: M-Sec System 5W1H analytical pilot 1 Architecture View

UC1 – Secured IoT Devices to enrich strolls across Smart City parks (SAN-UC1) focuses on a Smart City scenario based on providing data that will improve QoL (Quality of Life) for its citizens. IoT environmental and crowd-counting devices (Devices FG) are set in various spots of Santander. These devices are secured by an M-Sec core component, the "Secured components for devices and gateways" (Devices Security FG). The data are transferred from the devices to a server (Storage FG) through an M-Sec core component, "Eclipse sensiNact platform" (Secure City Data Access FG). From the server, the data can be enhanced through analysis and visualisation tools (Cloud Tools FG). The final results are shown through an application to the citizens. Both historical and real-time data (visualised and non-visualised) can be offered. The environmental data and privacy sensitive information stored in the server are encrypted and secured through an M-Sec core-system component, "Crypto Companion DB" (Secured & Trusted Storage). The aforementioned storage assets are coupled with a blockchain and the corresponding middleware, to which the devices have been registered. This way it is possible for interactions on the data level to be stored in the blockchain and be shared, e.g. with other Smart Cities, through the IoT Marketplace, which exposes the "services" of the sensors and the corresponding datasets. A T&R engine can receive feedback from end-users (through the marketplace or the App) and rank the data sources based on the trust on them. The Security Management Tool provides end-to-end security.

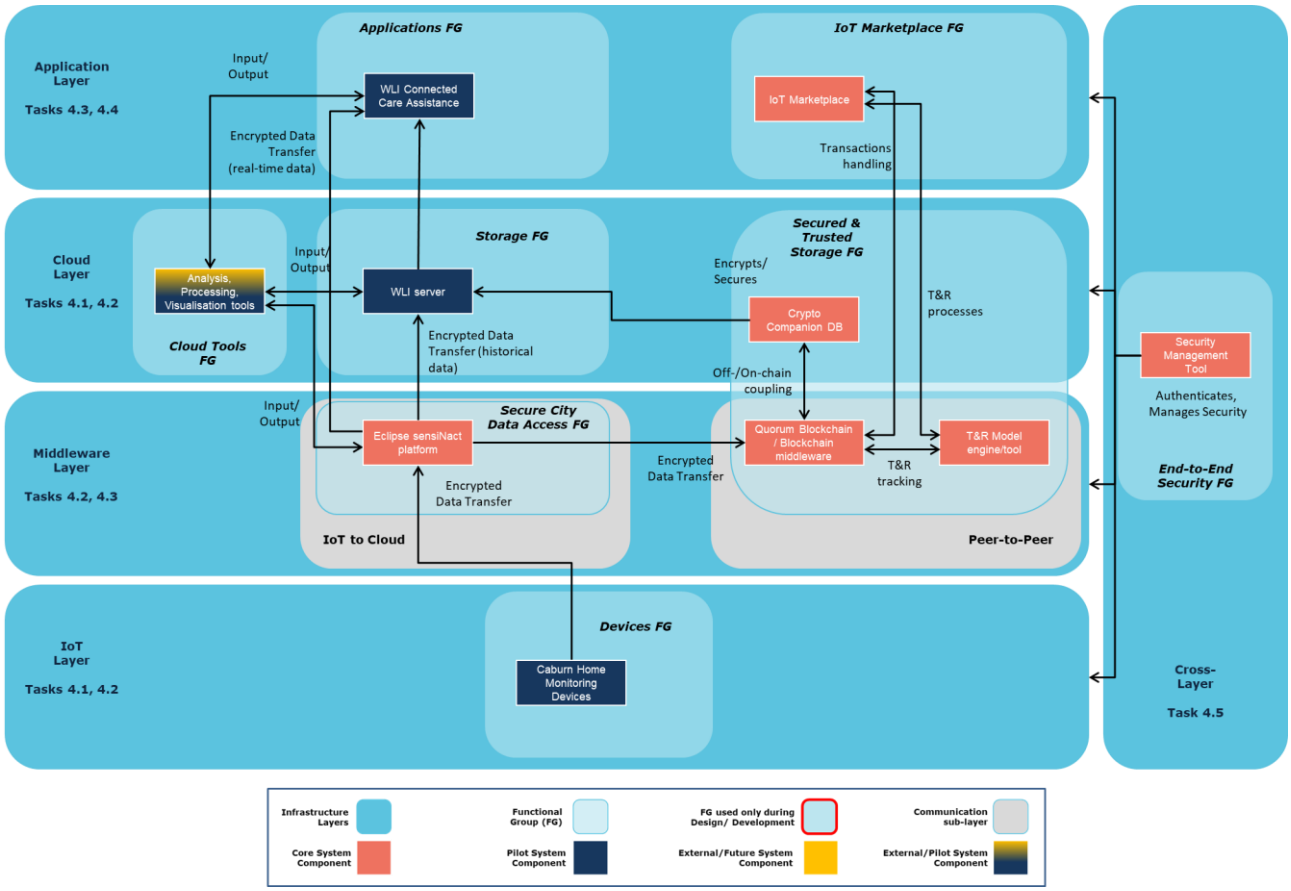

Figure 3—3: M-Sec System 5W1H analytical pilot 2 Architecture View

UC2 – Home Monitoring & Wellbeing Tele-assistance for active and independent ageing people (SAN-UC2) focuses on a Smart Home scenario based on home activity monitoring through the use of sensors for the assistance of elder people. This pilot has the aim to digitalise some of the current analogic-based, tele-assistance services provided by the Social Services department of the Santander City Council through a third-party operator. Home Monitoring devices include sensors such as presence sensors, bed occupancy sensors, window/door open sensors and smart plugs. Since “Secured components for devices and gateways” (Devices Security FG) is demonstrated in the previous UC, it does not reappear here. The data are transferred from the devices to a server (Storage FG) through an M-Sec core component, “Eclipse sensiNact platform” (Secure City Data Access FG). From the server, the data can be enhanced through analysis (Cloud Tools FG). The final results are shown through an application to the citizens. Both historical and real-time data can be offered. The monitoring data and privacy sensitive information stored in the server are encrypted and secured through an M-Sec core-system component, “Crypto Companion DB” (Secured & Trusted Storage). The aforementioned storage assets are coupled with a blockchain and the corresponding middleware, which can keep log files related to the Access Control of the services, data and components. Due to the nature of the data being captured, IoT Marketplace is not used in this scenario. The Security Management Tool provides end-to-end security and is one of the main components to be demonstrated in this use case, as it offers security management capabilities and authentication mechanisms at all layers.

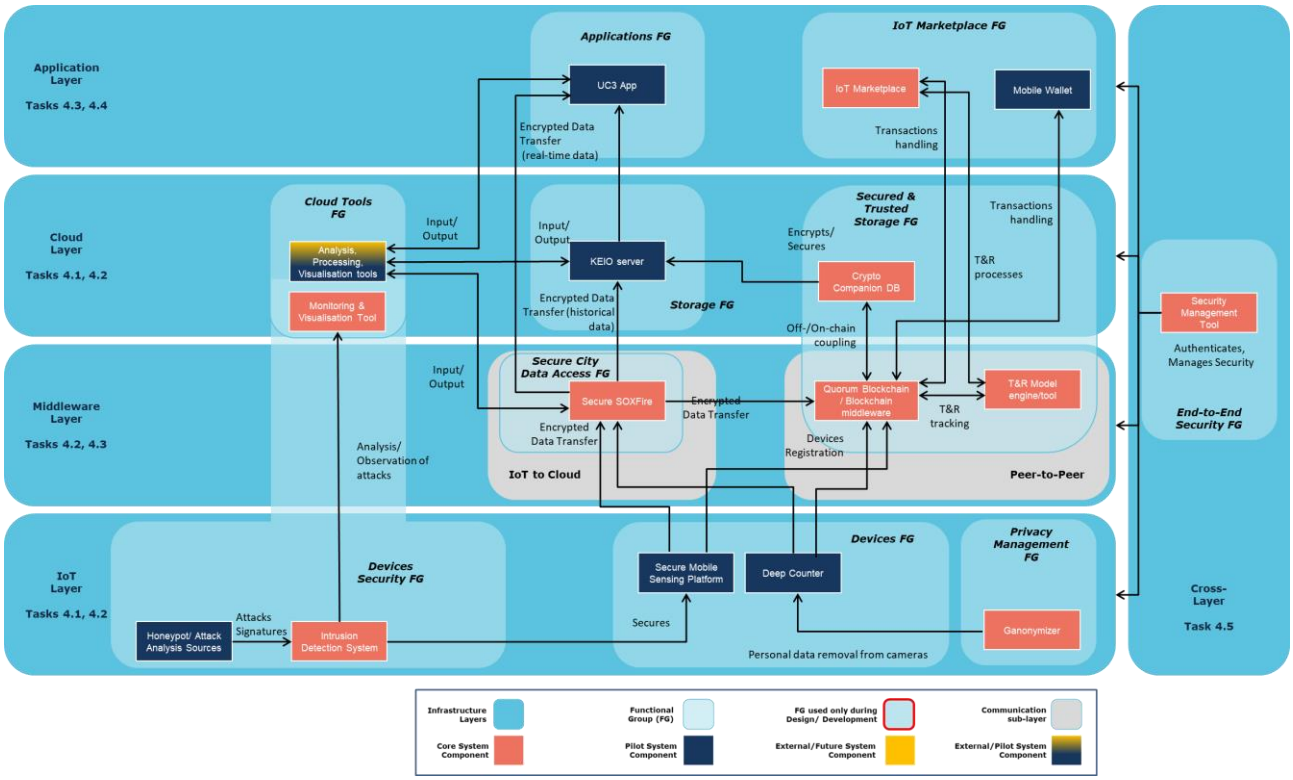

Figure 3—4: M-Sec System 5W1H analytical pilot 3 Architecture View

UC3 – Secure and Trustworthy Mobile Sensing Platform (FUJ-UC3) focuses on a Smart City scenario where urban environment monitoring entities (for example local governments) are able to visualise spatially and temporarily dense environmental and citizens-activity related data. The Secure Mobile Sensing Platform provides environmental data through the sensors of several trucks "scanning" all the roads of the Fujisawa day. Although the platform has been extended so as to be more secure, the component is further secured by using Intrusion Detection System (enhanced through input from Honeypot and other Attack Analysis Sources). Through a Monitoring and Visualisation Tool (Cloud Tools FG), it is possible to monitor the security-related "health" of the system, from the level of the devices up to that of the cloud. Deep Counter (Devices FG) is using cameras on the trucks to count the amount of garbage each house or neighbourhood produces. At this stage, an M-Sec core component, GANonymizer, removes personal information from the camera photos/videos (e.g. peoples' faces) on the edge, thus ensuring the protection of the citizens privacy. Both data sources (mobile sensing platform and cameras) can be registered as to the blockchain. Their data is encrypted and securely transferred to a server through Secure SOXFire. At this stage, other tools can analyse the data and provide visual statistics to the city authorities through an App or a web-page. The server per se is encrypted and secured by Crypto Companion DB. The non-private data can be grouped in datasets and become available to other users (e.g. other smart cities or authorities). Depending on the business model, the data can be acquired or exchanged for monetary value. In any case, the Blockchain can be used for storing the log-files of various interactions between stakeholders (access related data, etc.).

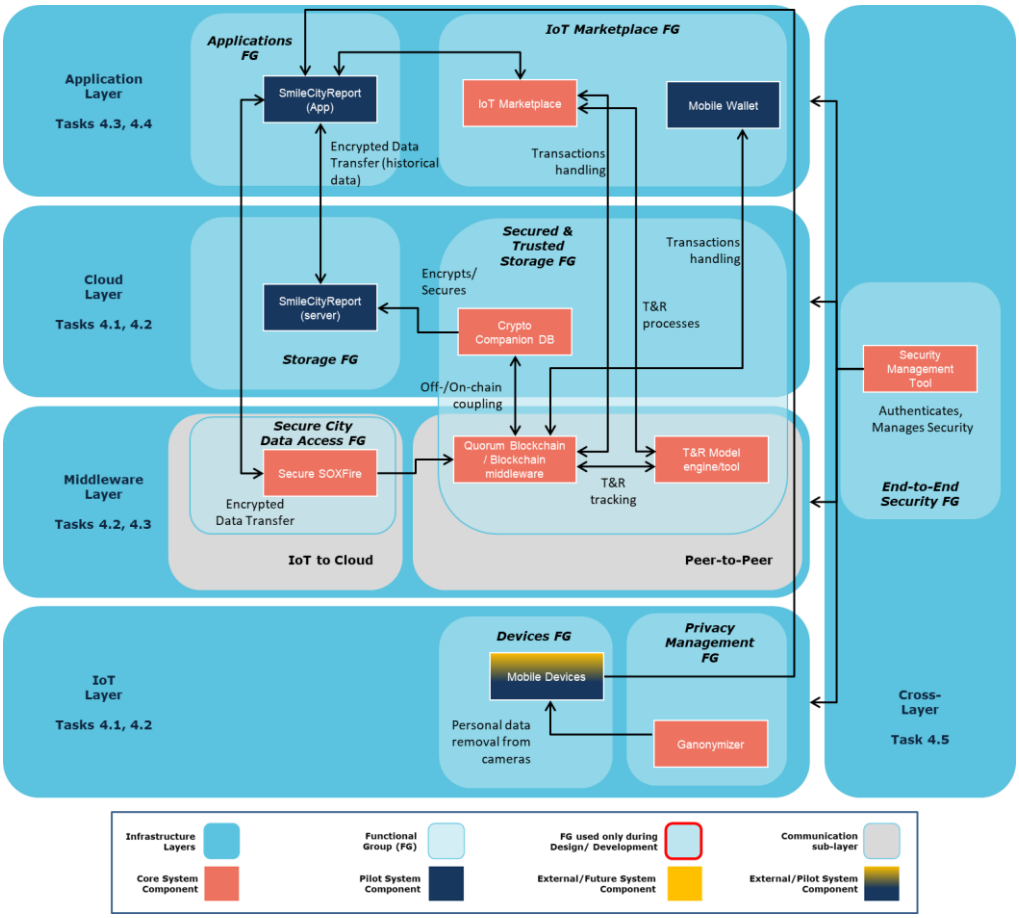

Figure 3—5: M-Sec System 5W1H analytical pilot 4 Architecture View

UC4 – Secure Affective Participatory Sensing of City Events (CB-UC4) explores the possibility of secure sharing on citizen's affective information and information on the city, by using "SmileCityReport" application on the M-Sec platform. This use case is related to topics such as mobile participatory sensing, edge-(mobile-)side computation for privacy protection, secure data sharing of sensed information, etc. Unlike UCs 1, 2 and 3, this use case will not focus on IoT devices for smart homes or cities, but will rely on mobile phones and devices used by citizens. This difference actually changes the flow of information considerably: instead of having a bottom-up flow of the data, in this case, the Application at the top is the actual starting point (after the Mobile Devices) of the flow. Through the application, citizens can take photos, share them with others, post them under specific themes and sell them through the Marketplace. The Secured & Trusted Storage FG and the Secure City Data Access FG keep the same role they have in the previous UCs. The Security Management Tool provides end-to-end security, as it offers security management capabilities and authentication mechanisms at all layers.

3.3 Architectural view in the case of external systems

The following figure gives an indicative view of a future system created after the adaptation/reusage of the M-Sec core-system to other domains and applications.

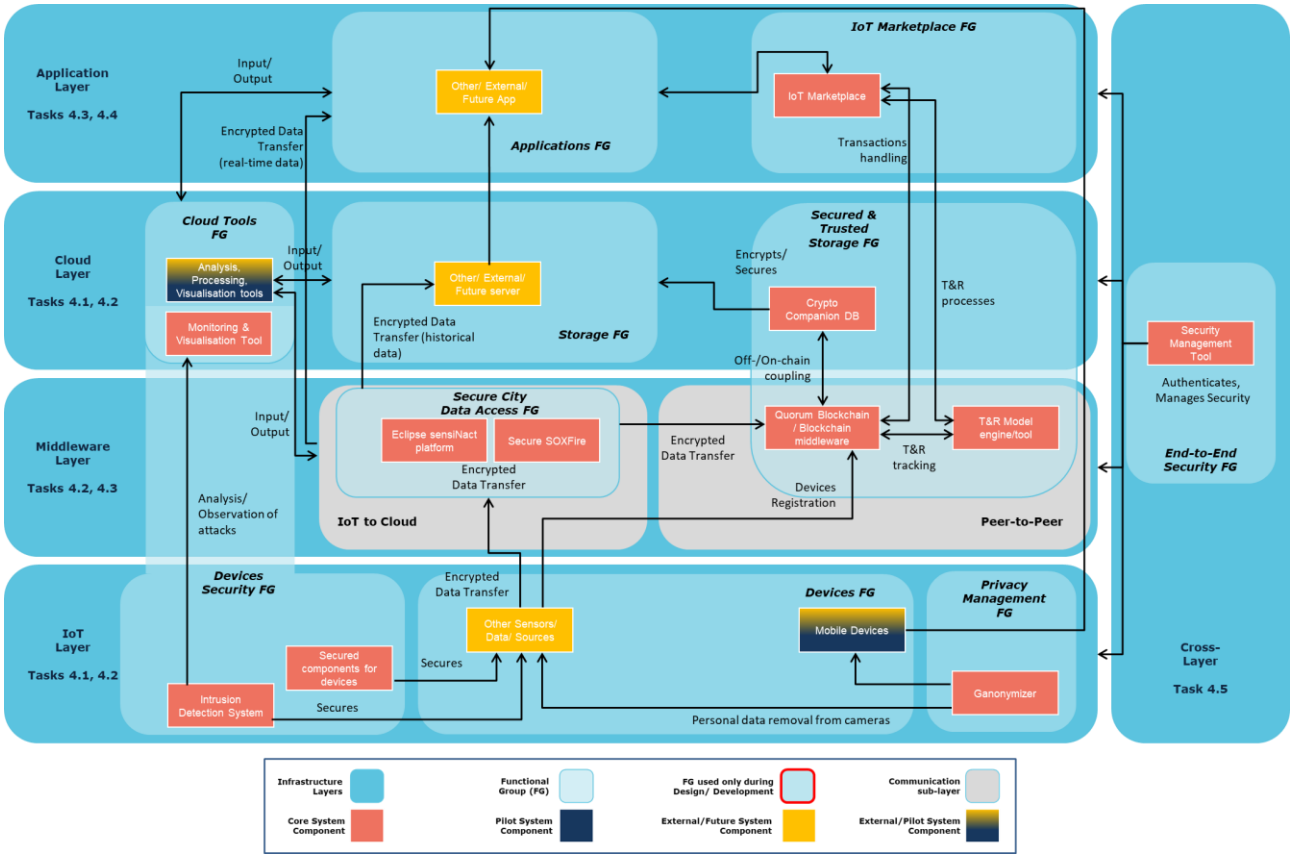

Figure 3—6: M-Sec System 5W1H analytical extended Architecture View

4. Final Results – Secondary Models & Views

4.1 Domain Model

Introduction, Definition of Terms, and Relations

One of the most important models proposed by the ARM is the **IoT Domain Model**. It can be considered to some extent as a formal definition of what the IoT domain is made of, in terms of IoT concepts and relationships between those concepts. In the following subsection we therefore introduce all concepts and terms considered in the M-Sec project and we put them in perspective with those referenced within the IoT Domain Model. As a result of this exercise, we get an M-Sec Domain Model that eventually fits into the framework introduced by IoT-A.

Physical Entities (PEs): Physical Entities are the objects from the Real-World that are virtualised in the cyber-space using Virtual Entities. In M-Sec, the PEs are smart objects in the sense that they have perceptions (provided by the attached Sensors), react to changes in their environment, take decisions and enjoy a certain level of autonomy and can be acted on using attached actuators. Of course, all these features are embodied within the concept of Virtual Entities.

Virtual Entities (VEs): VEs are at the heart of an IoT system. They are a representation of physical objects in the Cyber-World. Properties of the VEs can be populated/instrumented using tags, sensors (including tag readers, positioning sensors, temperature sensors, etc.) and actuators which are devices. Devices and hosted IoT resources are therefore used for bridging the gap between the Cyber and the Physical world; Sensors do report information about the physical entity while actuators are used to modify the state or properties of the Physical Entity. VEs are therefore the digital extension of the Physical-Entities that are used to manage, monitor, and control their physical counterparts (PEs). VEs get perception through accessing sensors readings via IoT Services (Resource-centric IoT Service) and can impact their environment or undertake physical actions using actuators (triggered via other IoT Services). Finally VEs may interact with other VEs for various purposes like collaboration, giving access to VEs properties/attributes, giving access to raw data (via access to r-Centric IoT Service), offering actuation services, etc.

Group Virtual Entity (GVEs): M-Sec makes a distinction between individual VEs (representing single objects) and Groups of VEs (GVEs) that aggregate/encompass a potentially large number of VEs. The IoT Domain Model does not make such difference as it allows VEs to be aggregations of various VEs. Like VEs, GVEs have their own properties (mainly based on properties of embedded individual VEs) and have their own objectives (often management and optimisation functions). Finally, a Group of VEs can embed other Group of VEs, if necessary.

Augmented Entity (AE): Although not directly used in M-Sec, it is worth mentioning that the IoT-A Domain Model introduces the concept of Augmented Entity (AE). The AE is the composition of a VE and the corresponding PE and highlights the fact that the two entities are linked with each other and make a whole entity. Augmented Entities are the “things” in the term Internet of “Things” as an AE represents both the physical and digital aspect of the thing.

Virtual Entities are **Digital Artefacts**: Active Digital Artefacts are running software applications, agents and Services that may access other Services or Resources. Passive Digital Artefacts are passive software elements such as database entries that can be digital representations of a physical entity.

IoT Devices: In M-Sec, IoT Devices are the hardware supporting the sensing and actuation functions. Micro-controllers, batteries, ROM memory, etc. are just **Devices** (without the IoT prefix).

IoT Resources: The software embedded in IoT Devices that provides the raw readings (for sensors) and actuations. In that sense, IoT Resources are not accessed directly; instead the IoT Resources are accessed (exposed) through the corresponding IoT Services. An IoT Service could for instance expose with a standardised interface the raw reading provided via the IoT Resource enriched with meta-data.

IoT Services: IoT Services are associated with Service descriptions that can be used to discover particular Sensing/Actuation capabilities. We can consider different kinds of IoT services depending on their level of abstraction:

- Resource-centric IoT services (r-IoT Service) are exposing the IoT Resources using standardised interfaces and possibly adding meta-data to the raw readings available at the resource level. They all connect to a sole resource (sensor or actuator).
- VE-centric IoT Services (ve-IoT Service) are associated to the VEs and are used for accessing VEs attributes/status or to access VE-level services, not directly connected to VEs attribute or situation.
- Integrated IoT Service (i-IoT Service) are combinations of the two above when combining different readings from different sensors (e.g. “Secured” room can depend on lock/unlock status, presence indicators and light status).

VE properties: All of the above mentioned service types are wrapped under the VE property concept. This provides a unified way of describing, retrieving and accessing these services. VE properties allow a richer and uniform semantic description despite the differences between these services. This is guaranteed by the mandatory attributes required to be filled in during the services’ endpoints description.

Services: Services (without IoT prefix) are associated to VEs and GVEs but do not relate to specific Resources. Services are not part of the IoT Domain Model but could be added to the global picture for the sake of clarity.

M-Sec User: A user of the M-Sec platform will mainly interact through IoT Services and Services. A user can be a human person or a Digital Artefact (VE, Application, and Service). It is worth noting that the services offered by the platform itself are not represented in the Domain Model as they are not constrained to the IoT Domain, but are general enablers.

Sensors provide data or information about the Physical Entity they monitor.

Tags are used to identify Physical Entities to which the Tags are usually physically attached. The primary purpose of Tags is to facilitate and increase the accuracy of the identification process. This process can be optical, as in the case of QR codes, or RF-based, as in the case of RFID.

Actuators can modify the physical state of a Physical Entity, like in the case of changing the state (rotate, stir, inflate, switch on/off, …) of simple Physical Entities or activating/deactivating functionalities of more complex entities.

The M-Sec Domain Model

The following Figure 4—1 shows a first version of the M-Sec Domain Model. It adds Digital Artefacts and Applications to the VEs (individual VEs and Group VEs), IoT Resources, Services (IoT Services and "classic" Services and Devices) and shows details about how those key concepts relate to each other.

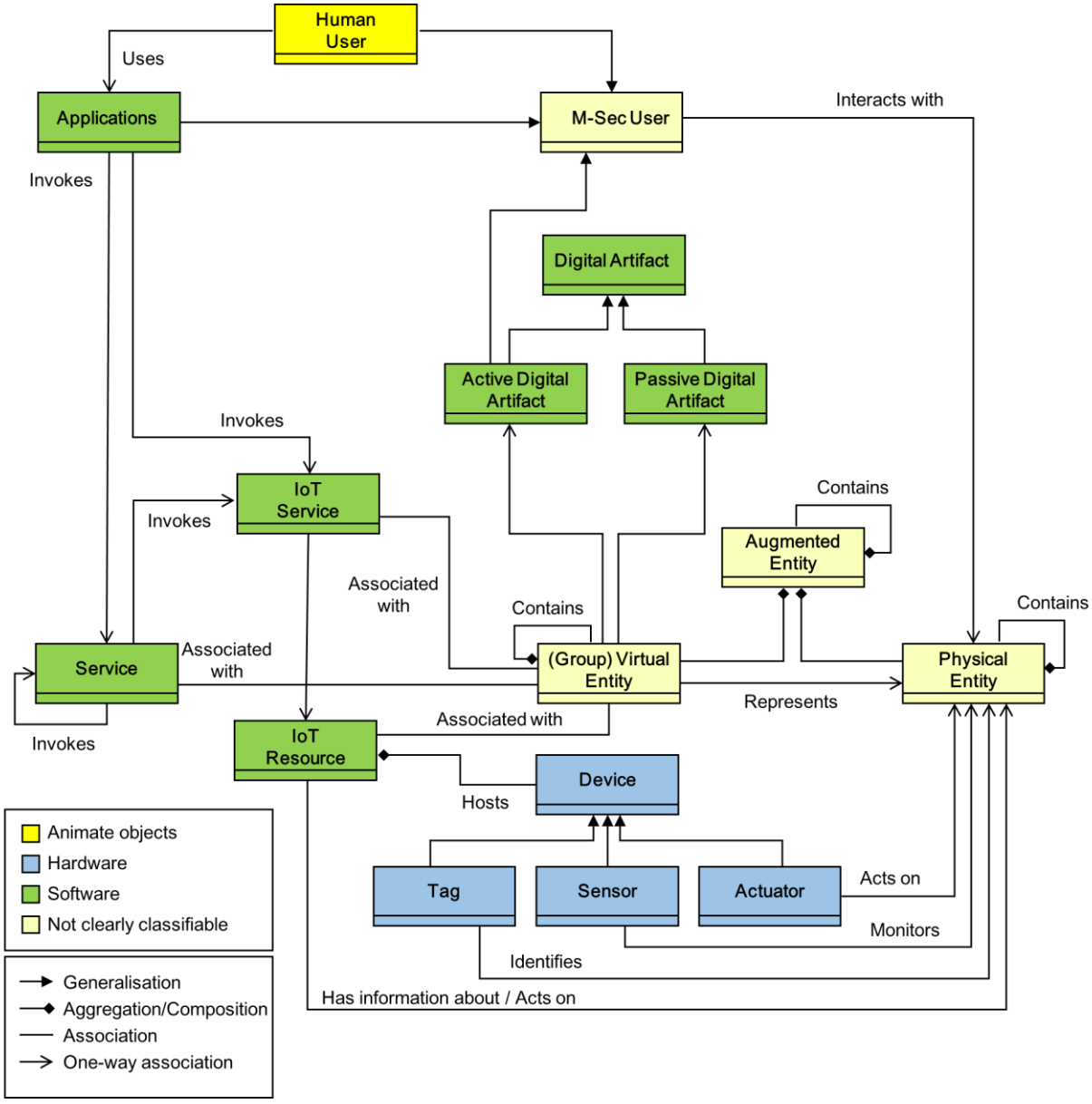

Figure 4—1: UML Classes diagram of the M-Sec Domain Model

4.2 Physical Entity & Context Views

Introduction to IoT Physical-Entity and IoT Context View

The ARM methodology introduced by IoT-A covers a number of Models, Views and Perspectives and gives details on how to build and use those. However, there are two views, namely the IoT Physical-Entity View and the IoT Context View, which are very important (considering the whole architecting process). Those views relate very much to the intrinsic characteristic of the specific IoT case being designed and consequently it is very difficult to extract a generic way of handling them, via a generic Physical Entity Model for instance (for the PE View).

As such, in this section we briefly describe the process needed to derive the Physical-Entity and Context Views. The Use Case 1 "Reliable IoT devices with multi-layered security for a smart city" is used as an example.

Physical-Entity View

The IoT PE View relates to the Physical Entities (sometimes also called Entities of Interest) which are in the scope of the IoT system under design, and which will be virtualised in the form of Virtual Entities (as shown and explained in the Domain Model). This view describes the properties of the PEs (and how they are captured in the VEs) and depicts which devices are used and what are their relationships to the PE (and associated properties). It also explains where those devices are situated compared to the PE (touching, fixed, in proximity, etc.). The PE also describes the nature of the information captured by devices.

IoT Context View

According to the ARM, the IoT Context View consists of two different parts: the IoT Domain Model and the Context View. The Context View describes *"the relationships, dependencies and interactions between the system and its environment (the people, systems, external entities, with which it interacts)"* either by using plain text or UML-like notations.

Applied to one of the M-Sec use-cases, e.g. Use Case 1, the context view will have to identify in the details all actors involved and all external components (e.g. all VEs) with which the M-Sec platform is interacting. It will also describe the nature of their interactions. Such concerns may cover the following aspects:

- System scope and responsibilities
- External entities and services and data used
- Nature and characteristics of external entities
- Identity and responsibilities of external interfaces
- Nature and characteristics of external interfaces
- Other external interdependencies
- Impact of the system on its environment

Indicative Context and Physical-Entity Views

Description of Pilot 1

This pilot will be carried out in one of the parks of the city of Santander, the **park** of Las Llamas. Within this park, there are specific **routes** that the visitors can follow. Along these routes, a series of **IoT sensing devices** in selected locations will be deployed. These sensing devices will include **temperature sensors**, **humidity sensors**, **noise sensors** and **luminance sensors**. The readings from these sensors will be forwarded to a secure **data delivery platform**. Thus, a series of **stops** will be created in each route where the users will be able to acquire visualised environmental information as well as other relevant data about the scenery of the specific stop (such as flora, fauna, or iconic places information). This information will be available to the citizens through an **environment monitoring and route guiding application**. The users will be able to access this information by scanning with their mobile device **QR codes** stuck close to the place where every IoT device is located to.

The technological developments will focus on the creation of novel IoT devices through the integration of different sensors and will incorporate in parallel novel security layers, both from the hardware and the software standpoint, ensuring high levels of reliability and trustworthiness. This pilot could also be linked to the M-Sec marketplace, based on the blockchain, and store IoT data produced by the sensors developed.

Physical-Entity View for Pilot 1

Based on the above description and following the Domain Model introduced in Section 3.2, we now describe briefly the main entities related to this specific pilot.

M-Sec-user:

- ParkVisitor: The visitors of the park can use their mobile devices to scan the QR-codes close to the smart devices and get information about the route stops and the route *per se* through an application.

Physical Entities (PEs):

- RouteStop-PE: These stops are scattered among the park and can be shared by more than one Route; lots of information can be available for them like position, environmental data, etc.
- Route-PE: A route consists of a set of stops sharing an itinerary; accessing a Route allows to access routes and individual spots positions and status.
- ParkVisitor-PE: This physical entity corresponds to the smart device (smart phone, tablet, etc.) that a Park Visitor will use to get access to the Routes' or Routes-Stops information through ParkVisitorAPP.

Virtual Entities:

- RouteStop-VE: It represents a RouteStop-PE.
- Route-VE: It represents a Route-PE. It is actually a **Group VE**, since it consists of an aggregation/composition of several RoutStop-VEs. It should be noted that the information related to this entity is greater than just the sum of the corresponding RouteStop-VEs, since this GVE will include other Properties too, such as RouteLength.
- ParkVisitor-VE: It represents a ParkVisitor-PE.

Devices:

- StopDevice: The deployed IoT sensing devices deployed along the Routes.
- SmartPhone: The smart mobile devices used by the park visitors to scan the StopDevice QR codes and get access to their data/information.

Applications:

- ParkVisitorAPP: The application running on the smart device to get the Route/RouteStops data/information.

Figure 4—2 shows a fragment of the M-Sec IoT Instantiated Domain Model applied to this scenario.

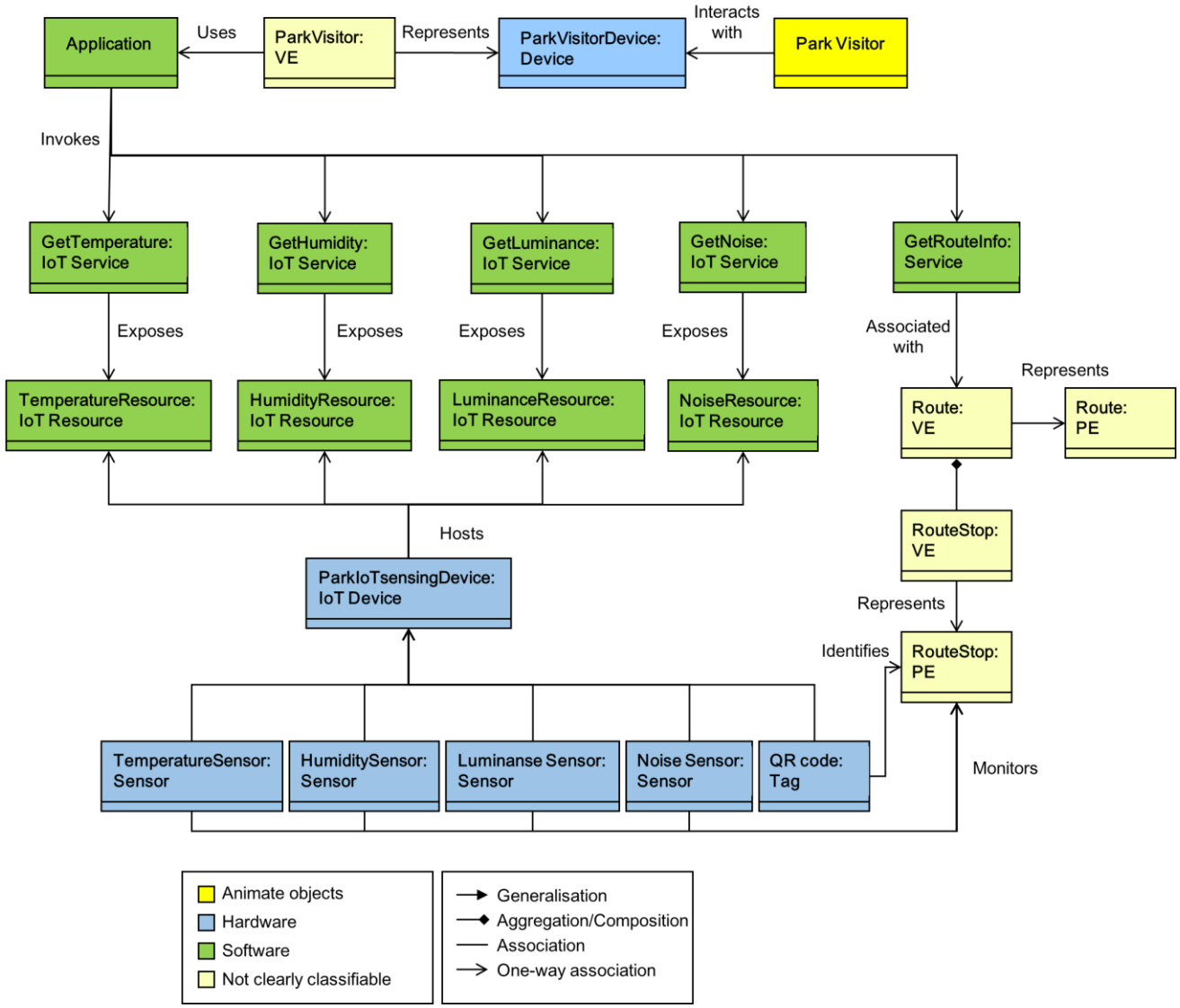

Figure 4—2: M-Sec instantiated IoT Domain Model for Pilot 1

4.3 Information Model

The second model introduced by the IoT ARM, as part of the Reference Model, is the **Information Model**. In this section we describe and explain the IoT Information Model in general; this allows to understand later on how the M-Sec Information View complies to it, as far as the different choices we make in terms of encoding are concerned. The Information Model focuses on the description of the structure of Virtual Entities as a representation in the cyber-space of Physical Entities. The representation of the information (either it is encoded in XML, RDF, binary or any other format) is kept away from the Information Model and left to the architect's choice or to the scope of some Design Choices when dealing with Perspectives.

The central part of the Information Model consists of the structure of the VE which is modelled using a set of attributes and which are associated (via the association relationship) to a Service Description. These associations make the link between an attribute and a corresponding *get<attribute>* function (r-IoT Service) for instance (in this case a readable attribute; *set<attribute>* for an attribute relating to actuation respectively). The Attribute is the aggregation of one to many Value containers. Each of those containers contains one single value and one to many metadata (e.g. time stamp, location, accuracy, etc.). The VEs are described through Services Description where each Service would be characterised e.g. by its interface or any useful information that a lookup service can exploit. As IoT Services are exposing Resources which are themselves hosted by Devices, the Information Model authorises Service Description to contain 0 to many Resource Description(s) and Resource Description to contain 0 to many Device Description(s). The structure of Descriptions is not constrained by the Information Model and therefore is left to the architect's own choice. The IoT Information Model is shown in Figure 4—3 below.

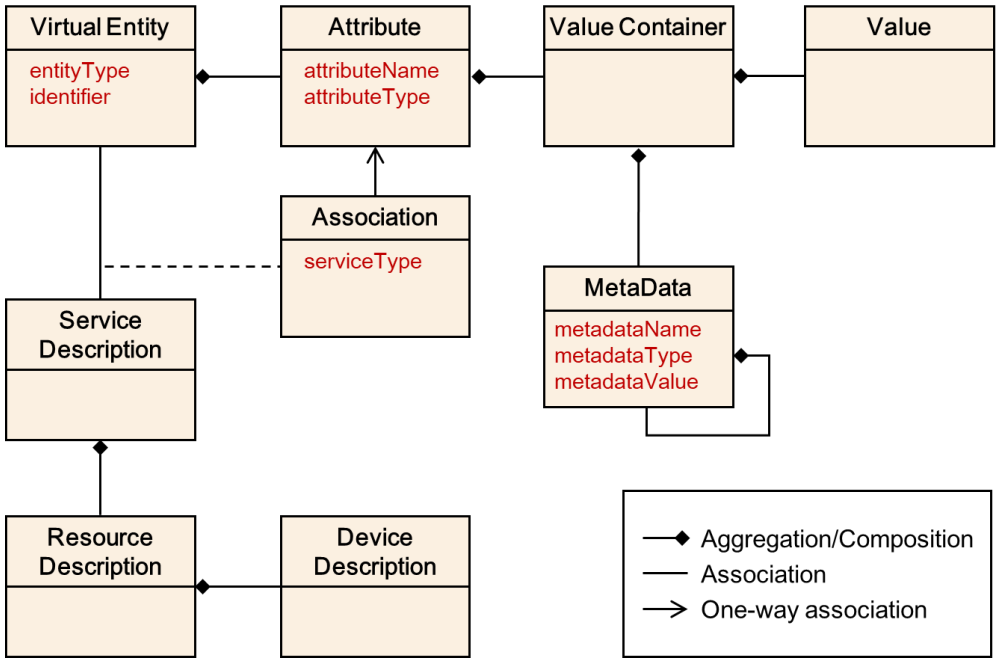

Figure 4—3: IoT Information Model as defined in the IoT ARM

5. Conclusions

This final version of the M-Sec architecture highlighted the key architectural principles for the design of the platform. In addition, the methodology that has been followed for the analysis and design (in order to improve the efficiency of the WP3 work and quality of the results) was extensively described. The M-Sec architecture has been successfully aligned to the IoT-A ARM and has covered all aspects of the 5W1H principle. M-Sec components were grouped in relevant functional groups, taking under consideration the User Requirements and Use Cases and were separated in layers. Also, the links between all components were identified.

Moreover, for all Use Cases, flow diagrams between the several functional components have been identified, following the methodology that was developed in earlier sections.

This report provides useful input for the whole project and especially for the development of the technical Work Package (WP4). In the description of the components and their interactions, the role of each one in the overall architecture was clarified.

Figure 3—1 is the main outcome of this document, summarising all the main results that have to be considered for the further development of and coordination between technical tasks.