Deliverable D44.1
Mobile Business platform specifications and architecture
M12 issue

Document Owner:	Giovanni Casella – SOFTECO
Contributors:	Enrico Morten – SOFTECO, Simona Stringa - SOFTECO, Davide Storelli ENG, Martino Maggio ENG, Charalampos Vassiliou SL

Dissemination:	Public
Contributing to:	WP 44
Date:	28/09/2012
Revision:	1.0

VERSION HISTORY

	DATE	NOTES AND COMMENTS
0.1	**03/07/2012**	**DELIVERABLE STRUCTURE ADDED**
0.2	**05/07/2012**	**SECTION 1 AND 2 ADDED**
0.3	**11/07/2012**	**SECTION 3 AND 4 ADDED**
0.4	**19/07/2012**	**SECTION 5 ADDED**
0.5	**28/08/2012**	**SECTION 6 AND 7 ADDED**
0.6	**31/08/2012**	**SECTION 8 ADDED**
0.7	**18/09/2012**	**SECTIONS 6.1 AND 6.2 REVISED**
0.8	**20/09/2012**	**SOFTECO INTERNAL REVIEW – DOCUMENT REVISION**
0.9	**27/09/2012**	**REVISION ACCORDING MSEE PEER REVIEW COMMENTS**
1.0	**28/09/2012**	**FINAL VERSION**

DELIVERABLE PEER REVIEW SUMMARY

ID	**Comments**	**Addressed (✔) Answered (A)**
1	Too many and deep technical details in some sections.	Some technical details have been removed.
2	Section 6.1 is confusing (too many sw development technologies and approaches) and should be split in more sub-sections.	Section 6.1 has been revised and re-structured.
3	The Mobile Collaboration Module could also extend the Operational/Ecosystem-governance Management System.	✔
4	The Multimodal Module should prioritize visual, touch and speech interaction instead than text.	This is the goal of the module. It has been highlighted in the text.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Executive Summary

This document describes the initial specification and architecture of the MSEE Mobile Business Platform (Mobile Platform) resulting from the work done, in the first six months, in WP44 “MSEE Generic Mobile Business Platform”.

The task T44.1 of WP44 “Support to Mobile and Ubiquitous business” focuses on the exploitation of the ubiquitous computing environment provided by mobile devices in order to support business scenarios with personalised services utilized by users in mobility, and in order to support industrial model for Innovation Ecosystems enhancing mobile collaboration.

To achieve the goal of T44.1 the Mobile Platform provides mobile extension to other MSEE platforms namely the MSEE Generic Development Platform (WP42), the MSEE Generic Service Delivery Platform (WP43) and the Innovation Ecosystems Platform (WP26).

In addition to extending other platforms, the Mobile Platform addresses two relevant topics for mobile devices namely Multimodal Interaction and Ambient Intelligence.

The task T44.2 focuses “on the study and the implementation of new multi-modal user-interaction models, based on adaptive interfaces, which may use visual and speech communication, natural-language recognition of free text, speech or gestures” [MSEE DoW[1]].

The Mobile Platform addresses this topic applying a multimodal adaptive interface to a concrete mobile application to be used in a manufacturing context. One of the main goals of the multimodal adaptive interface is to assess the advantages of a multimodal interface when used by professionals involved in multiple tasks at the same time and working in a changing environment with particular characteristics (i.e. noisy, brightly, dangerous, etc.).

Finally the task T44.3 addresses the innovative scenarios enabled by the Ambient Intelligence issue in the context of MSEE use cases and end-users. The Mobile Platform exploits the engaging features of smartphones and tablets in order to discover and identify the available devices (machines, sensors, etc.) in the user environment providing additional functionalities for these devices. Moreover the platform provides a simple extension mechanism enabling developers to extend the general features of the system in order to address domain specific requirements.

The deliverable is organized as follows:

Section 1 – Introduction: clarifies the objective of the deliverable, its internal structure and the relations with other MSEE deliverables relevant for this document.

Section 2 – Mobile Devices and Operating Systems: provides some background information on mobile devices (cell phones, PDAs, smartphones, tablets) and their operating systems.

Section 3 – Relevant Technologies: presents, very briefly, the technologies that will be used to develop the Mobile Platform in order to simplify the reading of the subsequent sections.

Section 4 – The role of the Mobile Business Platform in MSEE: specifies the role of the Mobile Platform in the context of the Service System Life Cycle composed by the “servitization process” and by the “governance and innovation process” and highlights the position of the Mobile Platform in the MSEE IT System[2].

Section 5 – Mobile Business Platform: presents the general architecture of the platform and introduces its modules.

Section 6 – Mobile Business Platform Modules Specification: depicts the internal architecture of each module and presents its specification.

[1] MSEE DOW: Manufacturing SErvice Ecosystem Description of Work
[2] MSEE_D41.1: MSEE Service-System Functional and Modular Architecture

Section 7 – Conclusions: summarizes the results presented in this document and describes the work to be done in order to implement the initial platform prototype and to define the final version of the specification.

The specification presented in this document (D44.1) will be implemented in the initial prototype of the Mobile Platform (D44.3). Next, the document will be refined and extended to obtain the final specification and architecture of the Mobile Platform (D44.2) that will be the base for the final prototype (D44.4).

1. Introduction

1.1. Objective of the Deliverable

The main objective of this deliverable is to introduce the MSEE Mobile Business Platform:

- Providing a short background on mobile devices and related technologies and providing some links to get further information.
- Highlighting the role of the platform in the MSEE project and describing the relations with the others platforms developed in the project.
- Presenting the Mobile Platform initial architecture and specification.

The document presents the results obtained in the WP44 of MSEE namely "MSEE Generic Mobile Business Platform" and in particular in the three workpackage tasks T4.4.1 "Support to Mobile and Ubiquitous business", T4.4.2 "Multimodal Interaction" and T4.4.3 "Ambient Intelligence".

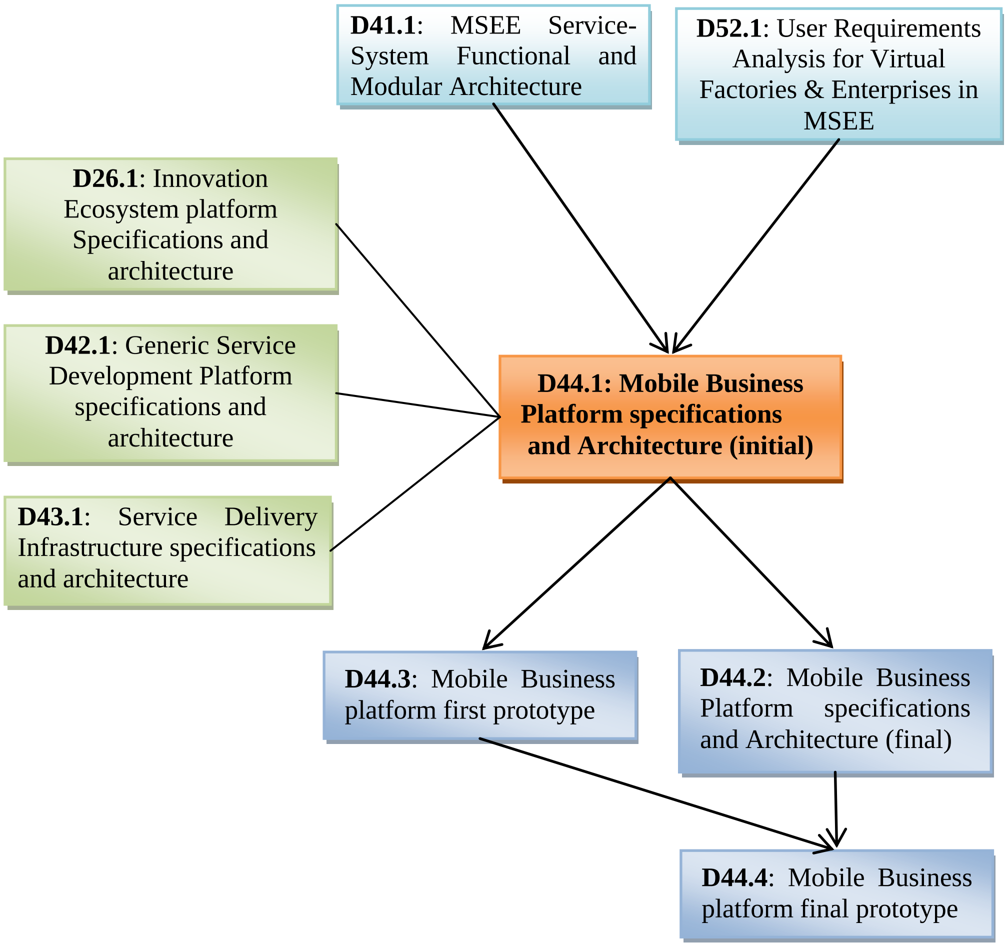


Figure 1: Relevant MSEE deliverables for this document

Figure 1 presents the MSEE deliverables with a particular relevance for this document. The Mobile Platform is part of the MSEE IT System presented in *D41.1* and address the first set of user requirements collected in *D52.1*.
The Mobile Platform focuses on mobile devices extending the other MSEE platforms composing the MSEE IT System, whose initial specification is provided in the deliverables *D26.1*, *D42.1* and *D43.1*.

The architecture and specification presented in this document will be used to develop the Mobile Platform first prototype (*D44.3*). This document will be refined and extended to define the final version of the Mobile Platform architecture and specification (*D44.2*).
Finally the final prototype of the platform (*D44.4*) will be developed as described in the final specification of the platform and starting from the initial prototype.

1.2. Structure of the Deliverable

To improve readability, the document presents the information regarding the Mobile Platform following a top-down approach both for the overall document and inside each section. Firstly general information is presented, and afterwards specific information and/or technical details are provided.

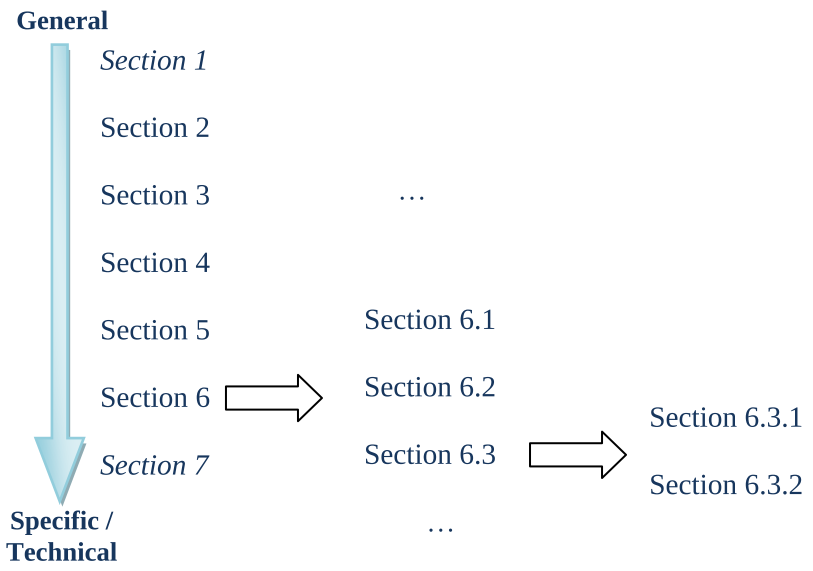


Figure 2: Top-down approach used to structure the deliverable

Figure 2 shows a graphical representation of this approach. While Section 3 and 4 includes some general background information on mobile devices and mobile relevant technologies Section 5 depicts the role of the Mobile Platform in MSEE and Section 6 and 7 present the platform specification and architecture.
The top-down approach is used also in each Section.
For example Section 6 presents the specification of the Mobile Platform modules, Section 6.3 presents the details of a particular module and Section 6.3.1 includes some technical information on a specific aspect of the module.
Having simpler contents, Section 1 “Introduction” and Section 7 “Conclusions” do not follow the top-down approach.

2. Mobile Devices and Operating Systems

2.1. Mobile Devices

In the last years the mobile device market has shown an exponential grow and new technologies have generated a great excitement. In addition to the original cell phones today several devices are available such as PDAs, smartphones, tablets, notebooks, laptops, handheld game console, portable media player, personal navigation device and so on.
Some of these mobile devices (i.e. game console, portable media player, personal navigation) are out of the scope of the Mobile Platform. Also laptops and notebooks will not be addressed by the Mobile Platform because they are able to execute applications designed for Personal Computer so they can use functionalities provided by the other MSEE platforms.
In the rest of the Section will be presented some candidate device categories to be addressed by the Mobile Platform while the precise set of addressed devices is defined in the last part.

Cell Phones
The first models of cell phones (Figure 3) where developed during the first half of the 20th century. They were very expensive devices, weighting more than 30 kilos, and their use was limited to demonstrations and to military uses during the Second World War. The unique purpose of these phones was to send and receive voice communications.
Modern cell phones were developed around 1990. They were quite expensive small devices equipped with an alpha numeric display where the main feature was again to send and receive voice communications. However the rapid expansion of the mobile phone market promoted the addition of new features.
Today cell phones can be small or medium sized and are equipped with a graphical color display and a numeric physical keyboard (sometime a qwerty physical keyboard). New functionalities have been added like the possibility to send and receive text and multimedia messages (SMS and MMS) and to capture low resolution photo. A predefined set of applications (i.e. phone book, calculator, convertor, games, alarm, etc.) is pre-installed on these devices. To find and install new applications is quite complex.

Figure 3: Cell phones

Personal Digital Assistant (PDA)
PDAs (Figure 4) are medium sized devices usually equipped with a medium/large graphical display. Their main purpose of PDAs is to help the user in the management of personal information like addresses, contacts, task lists, appointments, meetings, reminders, etc.
The main PDA input method is usually a resistive touch screen to be used with a stylus. While some PDAs are equipped with a physical QWERTY keyboard the major part are equipped with a virtual qwerty keyboard.
Traditionally PDAs did not provide communication features like telephone/fax, Internet and networking. While cell phones are still on the market due to their price (today they are quite

cheap) and features (they are usually small and simple to use) PDAs have been largely replaced by smartphones.

Figure 4: PDAs

PDAs are equipped with a predefined set of applications that it is difficult to expand while it is easy to synchronize the PDA contents (calendar, address book, etc.) with the contents stored on a Personal Computer.

Smartphone

Smartphones (Figure 5) can be considered a combination of cell phones and PDAs. In fact they are able to manage personal information and to provide GSM (Global System for Mobile Communications) communication capabilities.
Today the main focus is on smartphones (and on tablet presented later) and everyday new models (with new features) appear on the market.
Smartphones are pocket sized devices bigger than a cell phone and similar to PDAs. Their display is usually a high resolution graphical display supporting multi touch (even if in some cases they are still equipped with a qwerty physical keyboard). The display is not resistive (as in PDA requiring a stylus) but capacitive enabling a comfortable and precise use using only fingers.
Smartphones are equipped with powerful CPUs (over 1Ghz of speed, sometime dual-core) and have a large storage space (some gigabytes). They are able to detect their precise location using the Global Position System and to sense the ambient through several sensors (luminosity sensor, accelerometer, digital compass). Finally smartphones are able to guarantee high speed internet access and other communication channels like Wi-Fi and Bluetooth.

Figure 5: Smartphones

Smartphones are sold with a large pre-defined set of applications easy to expand. Finding new applications is fast and simple thanks to the availability of dedicated virtual marketplaces of mobile applications.

Tablet

Tablets (Figure 6) are portable devices quite big (they are usually equipped with a 7'-10' display) in comparison to smartphones and cell phones but they are usually lighter and smaller than a notebook.

Figure 6: Tablets

They have a large multi-touch capacitive touch screen and are able to execute both applications implemented for smartphones and applications specifically developed for tablets able to fully exploit tablet's features (in particular their large screen) and ensuring an improved user experience. As smartphones, tablets have many communication capabilities (GSM, Bluetooth, Wi-Fi), support GPS, are equipped with several sensors and are sold including many applications. It is possible to add new applications easily exploiting available mobile application marketplaces.

Devices addressed by the Mobile Platform

Due to their engaging features and diffusion the main categories of mobile devices addressed by the MSEE mobile platform are smartphones and tablets. They are high technology devices, sold at an accessible price, including state of the art technologies in the mobile device field. Suitable operating systems and powerful programming languages exist for such devices and users are able to easy find new applications and to install them in a user-friendly way.

Finally they are equipped with powerful CPU (less powerful than CPU equipping notebooks but really powerful respect to CPU embedded into cell phones and PDAs) and have a large storage space.

2.2. Mobile Operating Systems

Today the general feeling is that the mobile operating systems provided by Google (leading the Open Handset Alliance) and Apple are dominating the market and this feeling is confirmed, among others, by a recent press release published by comScore[3] in April 2012 (see Table 1) highlighting the most spread mobile operating systems.

[3] http://www.comscore.com/About_comScore

Top Smartphone Platforms Source: comScore MobiLens			
	Share (%) of Smartphone Subscribers		
	Jan-12	Apr-12	Point Change
Google	48.6%	50.8%	2.2
Apple	29.5%	31.4%	1.9
RIM	15.2%	11.6%	-3.6
Microsoft	4.4%	4.0%	-0.4
Symbian	1.5%	1.3%	-0.2

Table 1: Report on mobile operating systems diffusion[4]

The Open Handset Alliance provides the Android platform, a Linux-based operating system for mobile devices (i.e. smartphones and tablet computers) of different manufacturers.
Android applications are written in Java (a customized version) and several tools and libraries are available. Android makes easy to write engaging applications through the provision of powerful APIs with different goals like handset layouts management, internal storage use, connectivity management (bluetooth, Wi-Fi, etc.), access to phone messaging and calls, media support (audio/video), multi-touch support, etc.
Apple provides the IOS platform equipping Apple devices like iPhone and iPad.
iOS applications are written in Objective-C using the Cocoa framework. Even for iOS a powerful development environment is available, namely xcode, as well a complete API and a set of additional libraries.
RIM provides the operating system (a proprietary multitasking environment) equipping worldwide known BlackBerry devices.
Microsoft provides the Windows Phone operating system (the successor of the Windows Mobile platform) and is trying to conquer the mobile market through an alliance with Nokia. However Windows Phone operating system, launched in the second half of 2010, has still a small diffusion (4%).
Finally Symbian, that today is not very diffused, was the most advanced and used operating system before the comparison of smartphones and equipped million of Nokia cell phones.
Android and iOS have defeated the competitors thanks to their unique features and in particular to the features dedicated to the installation of new applications.
In fact users are able to access to online markets (i.e. App Store and Android market shown in Figure 7) directly by their phones and they can easily choose an application (from the thousands of free and not-free applications available) or a game to install on it. The download and installation of the applications is really simple while developers are fully supported in terms of tools and documentation.

[4] http://www.comscore.com/Press_Events/Press_Releases/2012/6/comScore_Reports_April_2012_U.S._Mobile_Subscriber_Market_Share

Figure 7: App Store and Android Market

The main difference between iOS and Android is that iOS is installed only on Apple devices, while Android is installed on the devices of many brands like HTC, Samsung, LG, Sony etc. It is clear that the iOS diffusion is related to the diffusion of Apple products while the diffusion of Android is not related to the products of a specific brand. Moreover while Apple products are quite expensive other brands sell expensive devices but also cheap ones.
Another key difference between Android and iOS is that while Android is open source (supported by the Open handset Alliance) and it is possible to develop an Android application using a generic personal computer, iOS is an Apple proprietary system and a MAC computer is required to develop an iOS application.
In most cases the functionalities of the Mobile Platform will be available to smartphones and tablets regardless of their operating systems while when a strictly interaction between the platform and the mobile device hardware will be required, a native application, specific operating system dependent,will be developed. In this latter case the addressed operating system will be Android mainly due to its diffusion but also to its open source nature.

3. Relevant Technologies

This Section provides some general information on the relevant technologies for the Mobile Platform. While the descriptions are short and concise a web site is indicated for each technology where the reader can find additional details if needed.

Ajax[5]

AJAX is the acronym of Asynchronous JavaScript and XML and indicates a particular technique to exchange data with a server enabling to update parts of a (mobile) web page without reloading the whole page. In practice Javascript is used, on the browser, to interact with the server and to update the web page shown to the user while XML is used as data format to exchange data between the server and the browser.

Android[6]

Android is an open source operating system for mobile devices installed on many mobile devices of almost all manufacturers (the Android presence on the market is increasingly constantly). The Android features, the presence of a vibrant community around the technology, the availability of documentation (books, developer manuals, tutorials, videos, etc.) and the availability of the Android application market enables developers to produce engaging applications that users can easily find and download.
The Android ADT is a powerful Eclipse plug-in supporting developers to implement mobile applications.

Apache HttpComponents[7]

The Apache HttpComponents library is provided by Apache with the goal of simplifying Java based developmentsexploiting the HTTP protocol.
Designed for extension while providing robust support for the base HTTP protocol, the HttpComponents library is of interest for building HTTP-aware client and server applications such as web browsers, web spiders, HTTP proxies, web service transport libraries, or systems that leverage or extend the HTTP protocol for distributed communication.

Apache Jersey[8]

Apache Jersey is an open source framework supporting developers in the provision of REST web services and representing the reference implementation of the JAX-RS (JSR 311) specification. The specification defines a set of Java APIs for the development of Web services built according to the Representational State Transfer architectural style.

Eclipse IDE[9]

Eclipse is a powerful and extensible integrated development environment, including runtimes and application frameworks, for building, deploying and managing software across the entire software lifecycle.

HTML5[10]

[5] http://www.w3schools.com/ajax/default.asp
[6] www.android.com
[7] http://hc.apache.org/
[8] http://jersey.java.net/
[9] http://www.eclipse.org

HTML is a markup language for structuring and presenting content for the World Wide Web and HTML5 represents the fifth revision of the HTML standard (created in 1990 and standardized as HTML4 as of 1997). HTML5 is still under development (as August 2012) and its goal is to improve HTML4 supporting the latest multimedia formats as well as location services while keeping it easily readable by humans and consistently understood by computers and devices.

Javascript [11]

JavaScript is a lightweight scripting language supported by all major browsers. Usually Javascript is embedded in HTML pages and using it is possible to ask the browser to perform some actions like to check if a required form field has been left empty.

JQuery Mobile [12]

The JQuery Mobile framework is designed to support developers in the implementation of web sites, designed to be used by mobile devices regardless by device features (screen dimension and density, available input modes, etc.) and device operating systems.
JQuery Mobile provides a unified user interface system that works seamlessly across all popular mobile device platforms.
The framework includes an Ajax navigation system that brings animated page transitions and a core set of User Interface widgets.

JSON[13]

JSON (JavaScript Object Notation) is a lightweight textual data-interchange format that is easy to read and write for humans and machines. Although JSON is based on a subset of the JavaScript Programming Language it is completely independent by any programming language or operating system. JSON is widely used to represent resources exchanged by REST web services.

REST[14]

REST is the acronym of REpresentational State Transfer and denotes a particular type of software architecture for distributed systems (client/server) adopted by the World Wide Web. In a REST architecture clients send requests to servers that return an appropriate resource. In the World Wide Web the client role is played by the browser (IE, Firefox, etc.) and the requested resource is usually a web page.
In general the client role can be played by a desktop application, a web application or a mobile application and a resource is a representation of a business object of interest in a given domain.
A RESTful web service (also called a RESTful web API) is a web service implemented using HTTP and the principles of REST.

[10] http://www.whatwg.org/specs/web-apps/current-work/
[11] http://www.w3schools.com/js/js_intro.asp
[12] http://jquerymobile.com
[13] http://www.json.org/
[14] http://en.wikipedia.org/wiki/Representational_state_transfer

XML[15]

XML is the acronym of Extensible Markup Language (XML) and denotes a textual markup language defining a set of rules for encoding documents in a format that is both human-readable and machine-readable. The design goals of XML emphasize simplicity, generality, and usability over the Internet.
Although the design of XML focuses on documents, it is widely used for the representation of arbitrary data structures, for example in web services.

XMI[16]

The XML Metadata Interchange (XMI) is an Object Management Group standard for exchanging metadata information via Extensible Markup Language. It can be used for any metadata whose meta-model can be expressed in the Meta-Object Facility language.
The most common use of XMI is as an interchange format for UML models, although it can also be used for the serialization of models of other languages.

[15] http://www.omg.org/spec/XML/ - http://en.wikipedia.org/wiki/XML
[16] http://www.omg.org/spec/XMI/ - http://en.wikipedia.org/wiki/XMI

4. The role of the Mobile Business Platform in MSEE

MSEE supports manufacturing enterprises to go from product-centric offerings to a value-proposition based on product-service bundles. This is a complex process that MSEE supports proposing a structured approach, in order to mitigate the risks and maximize the opportunities, and providing suitable models, methods and tools.
The MSEE IT System[17] is an integrated IT system which will support the whole life cycle of a manufacturing service ecosystem, including the servitization process, and specifically aiming at developing, operating and governing the target service ecosystem.
The scope of the MSEE IT System must be considered in the context of the overall Service System Life Cycle (see Figure 8) that MSEE aims at supporting and guiding in the manufacturing industry.
The Service System Life Cycle is composed by two main processes:

- The servitization process is responsible for the design of the Virtual Manufacturing Enterprise (VME) in order to switch the value offered from physical products to product-service “bundles”.
- The governance, operations and innovation process, on one side allows the Manufacturing Service Ecosystem governance and on the other side delivers the new value proposition to the customer through collaborative processes (VME Operations).

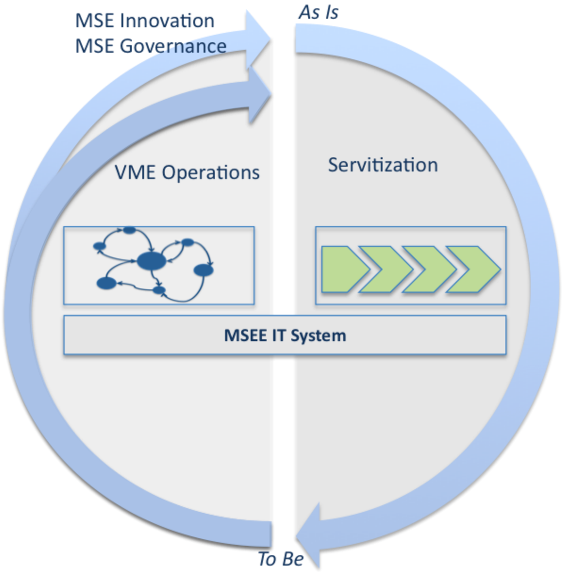


Figure 8: The MSEE Service System Life Cycle

The MSEE IT System, while taking into account the whole Service System Life Cycle, has a specific focus on the servitization process that will be supported by an integrated system composed by three main platforms: the Development Platform[18], the Delivery Platform[19] and the Mobile Platform.
The servitization process[20] starts when an enterprise decides to re-purpose its value offer, shifting from its current servitization-level to a higher one (see Figure 9).

[17] MSEE_D41.1: MSEE Service-System Functional and Modular Architecture.
[18] MSEE_D42.1: Generic Service Development Platform specifications and architecture.
[19] MSEE_D43.1: Service Delivery Infrastructure specifications and architecture.
[20] MSEE_D11.1: Service concepts, models and method: Model Driven Service Engineering.

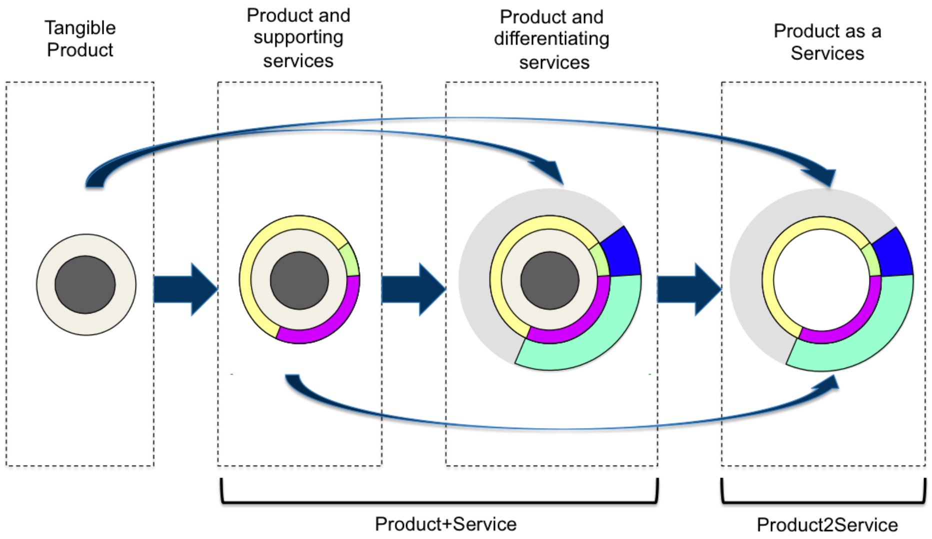


Figure 9: Transitions among the servitization levels

The more an enterprise increases its servitization-level, the more its value proposition is decoupled from the physical product.
Figure 10 shows a high level overview of the MSEE IT system from the point of view of its main components.

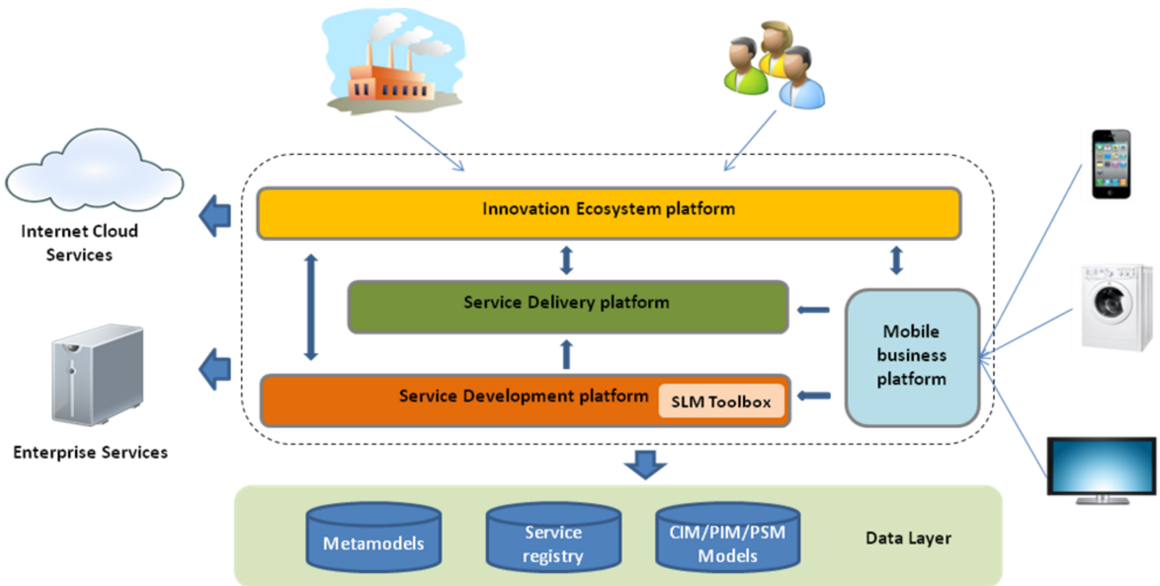


Figure 10: MSEE IT System

The Service Development Platform provides tools for business modeling and service lifecycle management, through the SLM Toolbox, and for the technical development of IT services and applications.
The Service Delivery Platform is in charge of managing service registration and delivery as well as semantic search and invocation.
The Innovation Ecosystem Platform[21] (IEP) supports both the ecosystem governance and the service innovation process interacting with all the other platforms. Moreover it is also the access point to MSEE IT System for the final users (people and enterprises in the ecosystem).

[21] MSEE_D26.1: Innovation Ecosystem Platform specifications and architecture.

In this context the role of the Mobile Platform is to exploit the exciting and continuously increasing capabilities of mobile devices to provide a full set of functionalities supporting Manufacturing Innovation Ecosystems and enabling mobile users to achieve the benefits provided by MSEE.
The Mobile Platform extends other platforms (Development Platform, Delivery Platform, Innovation Ecosystem Platform) functionalities addressing mobile devices in different contexts.
In particular the goals of the mobile platform are:

- to enhance the mobile collaboration inside ecosystems (IEP extension)
- to support developers in the implementation of mobile applications based on back-end services (Development Platform extension)
- to enable business users, involved in the definition of new services, to find already delivered services and applications providing a mobile access to the Delivery Platform (Delivery Platform extension)
- to enable service providers to monitor their services from mobile devices (IEP and Delivery Platform extension)

The Mobile Platform also considers two additional relevant topics for mobile devices namely Ambient Intelligence and Multimodal Interaction, as described in the MSEE Description of Work. These aspects are relevant for mobile devices for different reasons:
Ambient Intelligence refers to (electronic) environments that are sensitive and responsive to the presence of people and are able to interact with them being aware of their personality and needs, desires. Humans almost always carry on with them their mobile devices and the innovative features of these devices make them the ideal candidates to play the role of “universal human interface” in the Ambient Intelligence scenarios.
Multimodal interfaces can improve the user experience when using mobile applications. In fact applications developed for smartphones have some constraints mainly due to the small size of the devices limiting the available space for screens (output) and virtual keyboards (input). The impact of these constraints increases if the users are involved at the same time in multiple activities and works in changing environments like the manufactory ones.
Multimodal interfaces exploits nontraditional adaptive input/output modes (such as speech, touch, gestures, body movements ..), some time in combination, to enhance human-computer interactions and in particular “human-mobile device” interactions.
The MSEE project reuses the generic entity/system life cycle phases defined in the ISO 15704 standard to identify a reference and generic Service System Life Cycle. Figure 11 gives a graphical representation of such cycle.

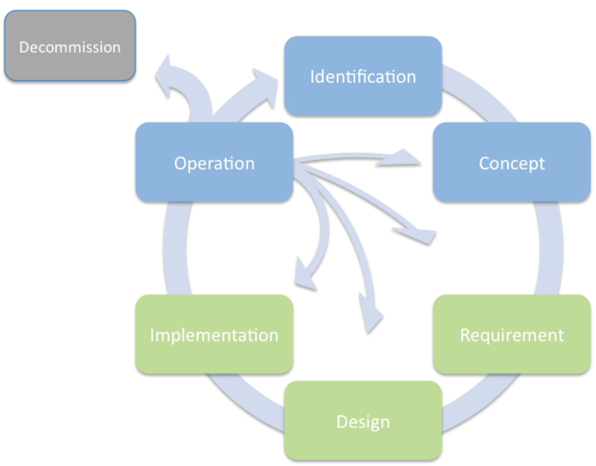


Figure 11: The Service System Life Cycle

The phases of the Service System Life Cycle are the following:

- **Service System Identification**: identify domain and existing components, objectives, challenges for a transition from product to service (or product + service);
- **Service System Concept**: identify and define the main concepts (models, functions, values, etc.) to create services around a product.
- **Service System Requirement**: identify, describe and model end-users requirements for the service system.
- **Service System Design**: design, specify and simulate the system that will provide the service.
- **Service System Implementation**: describe how the designed service system will be realized, deliver and implement physically all the needed components.
- **Service System Operation**: the service system is operational for use by customers and by all the stakeholders. This includes service consumption and interaction with customers, monitoring, evaluation and maintenance.
- **Service System Decommission**: end of the service system. The service system is removed and dismissed, whereas its components are reused.

Table 2 shows a high-level mapping of the MSEE Platforms with the phases of the Service System Life Cycle.

	Identification	Concept	Requirement	Design	Implementation	Operation	Decommission
Generic Service Development Platform (includes SLM ToolBox)							
Generic Service Delivery Platform							
Generic Business Mobile Business Platform							
Innovation Ecosystem Platform							

Table 2: Mapping of the MSEE IT Platforms with the phases of the Service System Life Cycle

The Mobile Platform mainly addresses the implementation and operation phases where mobile devices can play an important role. Secondly the Mobile Platform supports the Identification, Concept and Requirement phases through the extension of the IEP platform.

5. Mobile Business Platform

The main goal of the Mobile Platform (as described in the MSEE DoW) is to provide mobile access to other MSEE platforms making available MSEE functionalities to mobile users, taking advantage of the capabilities of smartphones/tablets and addressing some relevant topics for mobile devices.
The following list summarizes the main objectives of the Mobile Platform:

- Support manufacturing enterprises to provide mobile applications to their customers and to their internal staff.
- Enable mobile users to search/monitor services and applications delivered by MSEE ecosystems.
- Improve the mobile collaboration inside ecosystems.
- Support Ambient Intelligence scenarios.
- Exploit multimodal user interfaces to improve the user experience when using mobile applications.

To achieve these objectives the Mobile Platform is composed by several modules. These modules are implemented by some components executed in the cloud (deployed as web services or as web sites optimized for mobile devices) and by some components executed on the mobile devices (deployed as native mobile applications).
A special case is represented by the Mobile Development module that is implemented as an extension of an Integrated Development Environment and is deployed in a development machine.

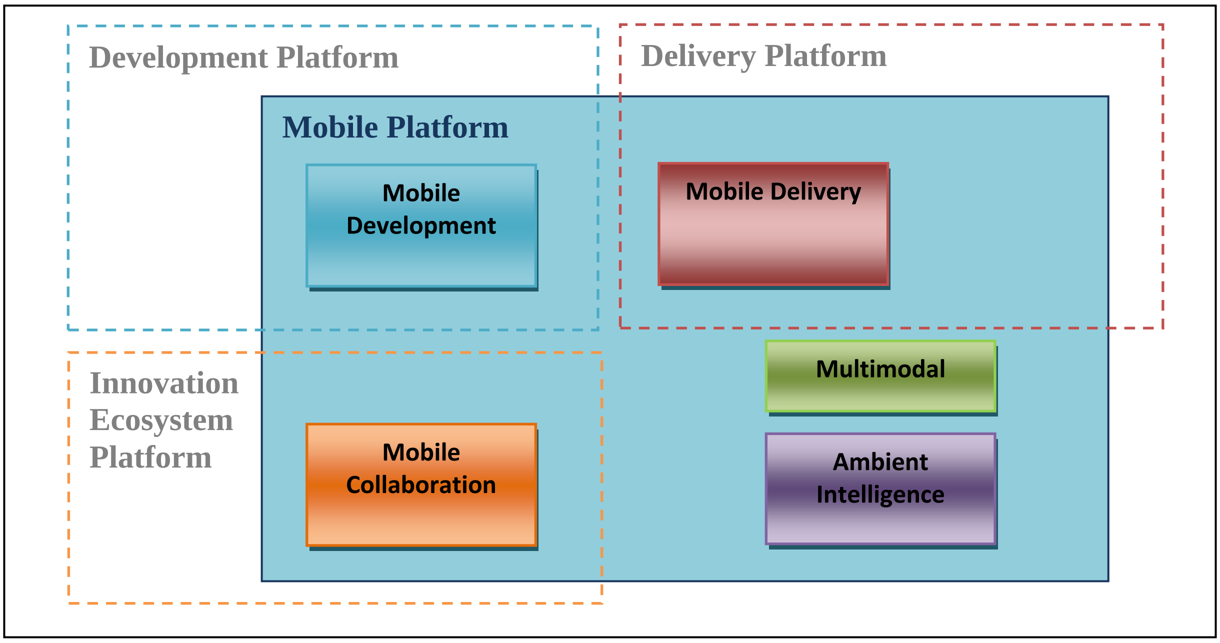


Figure 12: Mobile Platform Modules

As far as possible the modules are independent from particular operating systems, languages or technologies and are based on state-of-the-art technologies possibly open-source.

The following list summarizes the Mobile Platform modules shown in Figure 12:

- Mobile Development: end-users access software services provided by MSEE ecosystems through a set of applications (desktop, web, mobile). The Mobile Development module extends the Development Platform focusing on the development of mobile applications.

- Mobile Delivery: the module represents the mobile channel of the Delivery Platform enabling business users to discover and rank the services they need, exploiting their mobile devices. The module also enables mobile users to monitor the services they provide.
- Mobile Collaboration: the module extends the collaboration features of the Innovation Ecosystem Platform.
- Ambient Intelligence: the module enables users to exploit their mobile devices as their personal universal mobile interface managing the interaction between the owner and the available devices in the ambient.
- Multimodal: the challenge of multi-modal interactivity is to provide scripts, sounds, visual and gesture communication overcoming existing limits of mobile devices displays and (virtual) keyboard. The scope of this module is to provide a software reusable component simplifying the implementation of mobile multimodal interfaces and to show the advantage of multimodality in specific situations, such as when users are involved in multiple activities and/or are in a changing environment.

Section 6 presents an overview of each module. Some modules will be addressed in the first development cycle, dedicated to the implementation of the Mobile Platform initial prototype, while other modules will be addressed in the second development cycle dedicated to the implementation of the final prototype of the Mobile Platform. The modules developed in the first cycle are refined and extended in the second cycle thanks to MSEE end-user feedback.

Innovation Ecosystem Platform Modules	First Cycle	Second Cycle
Mobile Development	**X**	
Mobile Delivery	**X**	
Ambient Intelligence	**X**	
Multimodal		**X**
Mobile Collaboration		**X**

The complete specification of modules to be developed in the second cycle will be provided in the final version of this document.
The following table reports the list of the Mobile Platform modules and highlights if they will be developed in the first or in the second development cycle.

6. Mobile Business Platform Modules Specification

6.1. Mobile Development Module

Today there is a great interest in mobile applications and many enterprises offer to their customers mobile applications to add value to the enterprise products and to support internal staff to accomplish specific tasks when out of the office (i.e. to place orders, to access the enterprise information system, to collaborate with colleagues, etc.).
MSEE end users plan to provide some mobile applications to their customers, to their personnel or to other members of their ecosystems (partners, suppliers, etc.). For example Indesit wants to offer to its customers a mobile application to monitor and control their smart washing machines, Bivolino is going to develop some mobile applications in order to access a larger group of consumers (young mobile and tablet users) while Philips already offers some mobile applications to interact with smart TVs and will add new applications to its portfolio.
To design, develop, test and distribute a mobile application is a very complex tasks involving a lot of experts, therefore the support of the complete development cycle is a goal beyond the scope of the Mobile Development module. There are already some platforms supporting the development of mobile applications for the most popular mobile operating systems like the Android Development Tool[22] (an Eclipse plug-in for creating Android applications) and the Xcode application (an integrated development environment for creating iPhone/iPad applications).
The goal of the Mobile Platform is to complement the available development platforms helping MSEE users to develop the IT parts of their services addressing mobile devices. In particular, as described in the MSEE description of work document, the Mobile Platform will support the development of front-end mobile applications using one or more back-end web services.

6..1.1. Development Process supported by the Module

MSEE users design their services exploiting the concepts and the methodologies provided by the MSEE sub project 1, namely "Service Orientation in Virtual Factory and Enterprises", and using the MSEE Service Lifecycle Toolbox described in the deliverable D15.2[23].
A service is represented by a set of models (Business Service Models, Technology Independent Models, Technology Specific Models) describing its organizational-human, manufacturing-physical and information-IT aspects as defined in the deliverable D11.1[24].
The information regarding the informational-IT aspects include the high level specification of the software, in particular web services and applications, and of the hardware (IT infrastructure) needed to provide the IT parts of the service.
The MSEE Development Platform helps the developers to extend and refine these models in order to specify all the details needed to implement the software components and produces a set of XMI (XML Metadata Interchange) documents including the specification of these IT components.
Then XMI documents are used by the MSEE Development Platform and by the Mobile Development module to generate the programming code, needed to implement software components and addressing a specific set of technologies, automatically.
The MSEE Development Platform and the Mobile Development module will be integrated in a unique integrated development environment (including also the SLM toolbox) and the functionalities provided by the Mobile Development module, enhancing mobile development,

[22] http://developer.android.com/tools/sdk/eclipse-adt.html
[23] MSEE_D15.2: "Specification and Design of SLM Platform"
[24] MSEE_D11.1: "Service concepts, models and method: Model Driven Service Engineering"

will extend the functionalities of the MSEE Development Platform supporting the development of services and desktop/web applications.
In particular the Mobile Development module will take in input the representation of a business object, included in the model of the IT part of the service, and will support the user in the development of a web service able to handle the business object and in the development of a mobile application providing the user interface to exploit the service.

6..1.2. REST Services and Mobile Applications

The Mobile Development module supports a well-established mobile design pattern for the development of front-end mobile applications exploiting back-end services.
This design pattern suggests the use of REST services to enable mobile applications to handle business objects (resources in the REST terminology) stored on the cloud.

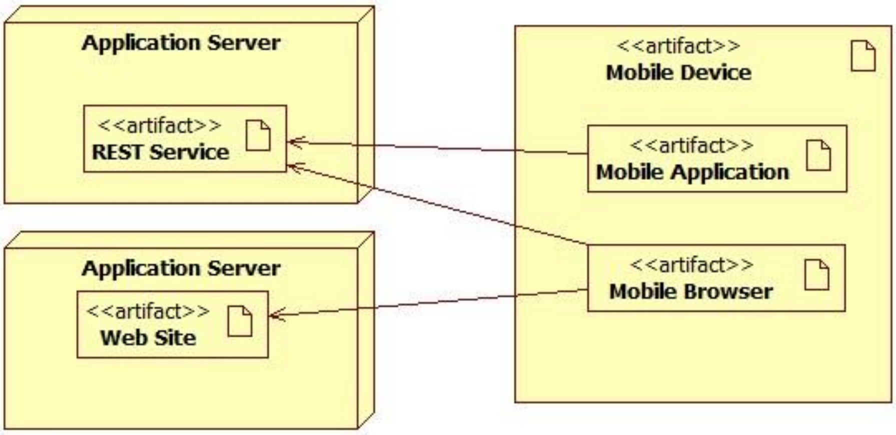


Figure 13: REST Services and Mobile Applications

For example a REST service could enable clients (web applications, mobile applications, etc.) to add, delete, update some appointments in a shared calendar and to get the list of appointments. Clients are in charge of managing the user interface and of accessing the service.
Figure 13 shows a deployment diagram where the REST service is deployed on an Application Server and is accessed through the HTTP protocol from a mobile application deployed on a mobile device. Moreover the figure shows a Mobile Browser deployed on the mobile device accessing a web site providing HTML/Javascript code and accessing the REST service (the Javascript code of the web site handles service calls).
The Mobile Development module will support developers in the implementation of REST services and clients using the Apache Jersey and the JQuery Mobile frameworks.
Apache Jersey represents the standard implementation of the Java API for RESTful Web Services, an API supporting the creation of web services according to the REST architectural style.
JQuery mobile is a javascript library providing specific functions to interact with REST services and suitable stylesheets to develop web applications designed for mobile phones. The module will be extensible to use other frameworks or language (i.e. Android) and possible extensions will be considered during the project.
The Mobile Development module will take as an input the REST resource to manage (i.e. a business a object representing a machine or a customer offer or a tv program, etc.) and will

generate all the needed code on the client and the server sides according the developers settings and preferences (see Figure 14).
The model of the REST resource to manage will be extracted by the XMI documents provided by the MSEE Development Platform or by other representations of these resources like Java Beans or JSON documents.

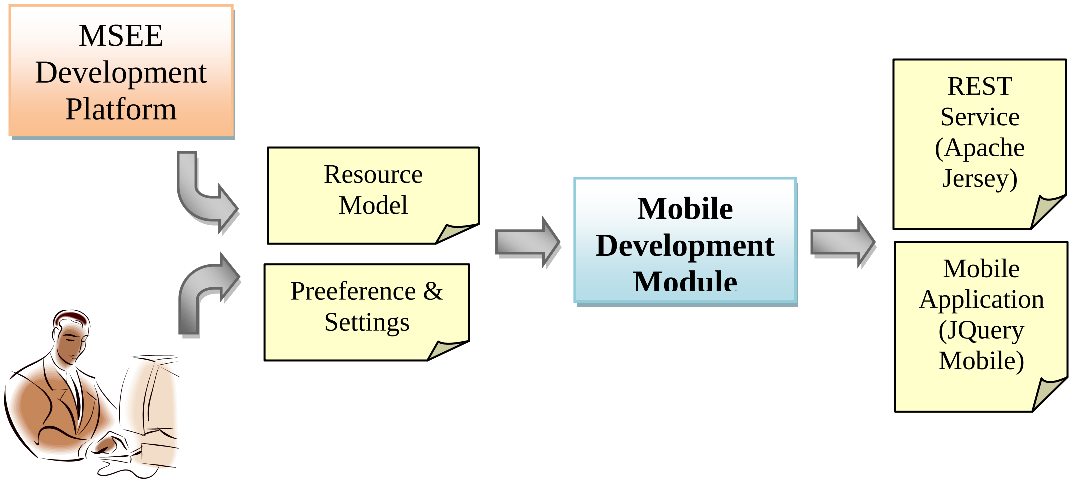


Figure 14: Mobile Development code generation process

REST services provide access to one or more resources (or collection of resource) through the support of the HTTP methods GET, POST, PUT, DELETE.

The following table presents an example describing an hypothetical REST service providing access to the list of products and to a single product delivered by an on-line shop:

Resource	GET	PUT	POST	DELETE
`http://myshop.com/products/`	**List** the URIs and perhaps other details of the products.	**Replace** the entire collection with another collection.	**Create** a new product in the collection. The new entry's URL is assigned automatically and is usually returned by the operation.	**Delete** the entire collection of products.
`http://myshop.com/products/item21`	**Retrieve** the details of a product.	**Replace** the product or if it doesn't exist, **create** it.	Treat the addressed member as a collection in its own right and **create** a new entry in it. For example item21 could represent a collection of shirts of the same model but of different colours and item21/red can represent the red shirt.	**Delete** the product.

Given the above REST Service it is possible to create a mobile application enabling customers to browse the catalog of the online shop and a web application enabling the shop staff to manage the catalogue. Today many well-known platforms (Facebook, SalesForce, Flickr, Yahoo, Delicious, etc.) expose their APIs as rest services.
The main steps to implement a REST service and a client, supported by the Mobile Development module, are the following:

1. Identify the resource to manage

2. Name the resources with URIs: the URI must be designed in such a way that each resource is addressable uniquely (i.e. */products identifies a list of products uniquely while */products/item33 identify a unique product).
3. Design the representation(s) of the resource exchanged between the client and the server: usually a resource as an XML document or a JSON document. In mobile application the lightness of the JSON language is usually preferred.
4. Implement the uniform interface: provide an implementation for the HTTP operations namely GET, POST, PUT, DELETE.
5. Develop the service client.

This time-consuming and error-prone process is simplified and accelerated by the module.

6..1.3. Specification

The Mobile Development module is provided as a plug-in of the Eclipse integrated development environment.
Eclipse (see Section 3) includes a well-defined, supported and documented extension mechanism. In fact Eclipse is composed by a small kernel containing a plug-in loader surrounded by hundreds of plugins executed into the environment managed by the kernel. Each plug-in contributes to the whole in a structured manner, may rely on services provided by other plug-ins, and each may in turn provide services on which other plug-ins may rely. Eclipse is distributed with a large set of pre-installed plugins and new plug-in can be easily installed to extend the basic features at any time.

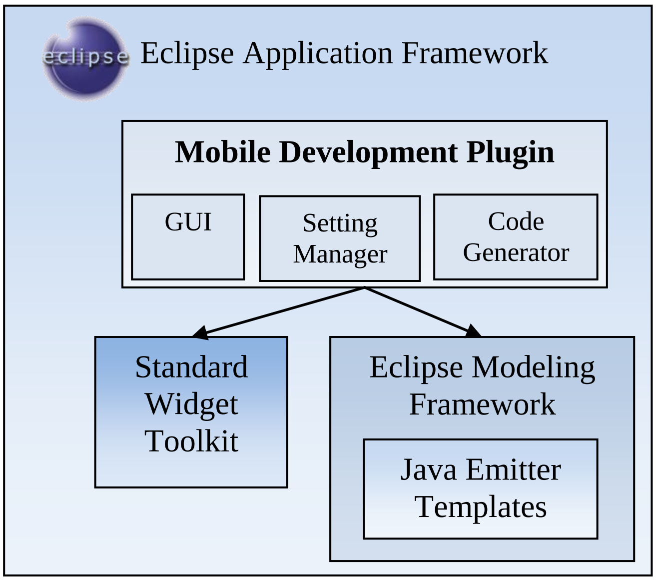


Figure 15: Mobile Development Module

A plug-in extends Eclipse features through the provision of particular editors, wizards, views, code generation features, and so on. The Mobile Development module offers a wizard enabling the developer to provide the representation of the resource to manage, to set all the preferences and settings (i.e. frameworks to use, paths, etc.) and to generate the REST code automatically.
Figure 15 shows the main components of the Mobile Development module. The module is developed as an Eclipse plug-in, living inside the Eclipse Application Framework, and is composed by three functional components namely GUI, Setting Manager and Code Generator.

The GUI component is in charge of building the graphical user interface (GUI) and of managing the user interactions. The GUI is built exploiting the graphical components provided by the Eclipse Standard Widget Toolkit.
The Setting Manager component is in charge of configuring the Eclipse projects (library inclusion, classpath setting, etc.) where the code is generated in order to avoid design-time errors (i.e. compilation errors due to the lack of one or more libraries).
Finally the Code Generator component is in charge of generating the REST code exploiting the Java Emitter Templates (JET) tool provided by the Eclipse Modeling Framework. JET enables to write programs able to generate code (code that generates code) using a JSP-like syntax making easy to write templates that express the code to generate. In practice JET is a generic template engine that can be used to generate SQL, XML, Java source code and other output from templates.

The following tables report some details on each component.

GUI	
Description	This component is in charge of building the user interface and managing the interactions with the user.
Functionalities / services	• GUI graphical building. • GUI event management and update in response of user interactions. • GUI update in response to notification of other components.
Dependencies	Eclipse Standard Widget Toolkit (SWT): the SWT provides a rich set of widgets (buttons, labels, links, menu etc.) and other components (events, commands, key bindings, etc.) that can be used to create either standalone Java applications or Eclipse plug-ins.
User	The user (developer) and all the other components.

Setting Manager	
Description	This component is in charge of modifying the target project where the code is generated in order to add the generated code without errors.
Functionalities / services	• Java Class Path setting • External library Import • Setting of export options
Dependencies	Eclipse Application Framework: the module modifies the project properties through the API provided by the platform.

User	This module is used by the GUI to carry on the code generation process.

Code Generator	
Description	This component is in charge of generating the code needed to implement a REST service able to manage the resource received as input and a REST client enabling the user to exploit the service through a suitable graphical interface.
Functionalities / services	• Server side code generation • Client side code generation
Dependencies	Eclipse Modelling Framework/Java Emitter Template
User	This module is used by the GUI to carry on the code generation process.

6.2. Mobile Delivery Module

The Mobile Delivery module acts as the mobile channel for the MSEE Service Delivery Platform that is responsible for the registration, discovery, ranking, selection, grounding and invocation, and low level monitoring of the services and applications that are used in manufacturing service ecosystems. The Delivery Platform itself is provided as a set of decoupled infrastructure services: Registry, Discovery, Ranking, Selection, Invocation and Monitoring. The Mobile Platform will mainly use the Discovery, Ranking, Selection, and Monitoring services enabling mobile users to consume them.
In addition to guarantee mobile access to the MSEE Delivery Platform, the Delivery module of the Mobile Platform will be designed to interact with any software component playing the Service Broker role introduced by the Future Internet core platform[25] (FI-WARE) project.
FI-WARE is an FP7 European Project (05/2011-04/2014) having the main goal of providing innovative infrastructure for cost-effective creation and delivery of services, providing high QoS and security guarantees. The Reference Architecture of the FI-WARE platform is structured along a number of technical chapters and the Applications/Services Ecosystem and Delivery Framework [26](SDF) chapter introduces several key business roles (see Figure 16) including the Service Broker one.

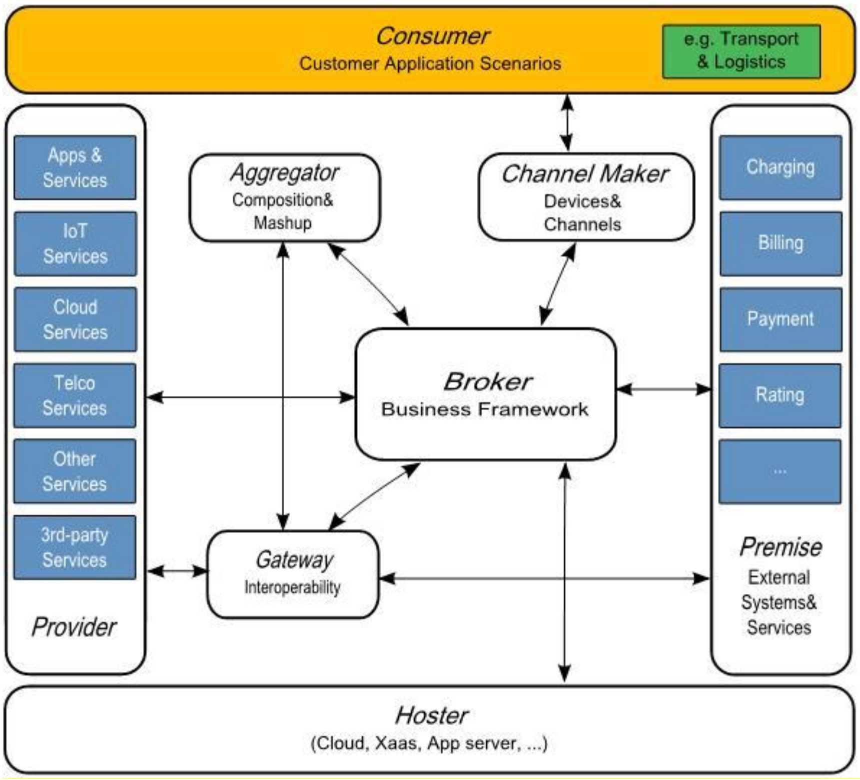


Figure 16: FI-WARE SDF high level architecture

The Service Broker role supports exposing services from diverse providers into new markets, matching consumer requests with capabilities of services advertised through it. In MSEE the Delivery Platform supports the Broker role by providing functionalities to discover services and applications based on their business/technical description. It is expected that in the near future, the Service Broker role will be played by other components developed by research projects and/or by industrial efforts.

[25] http://www.fi-ware.eu/
[26] http://forge.fi-ware.eu/plugins/mediawiki/wiki/fiware/index.php/Applications/Services_Ecosystem_and_Delivery_Framework

As said before the Mobile Delivery module is designed to interact with multiple instances of the MSEE Delivery platform and in general to interact with several components playing the Service Broker role. In this way the mobile user will be able to search for services and applications exploiting multiple sources.
Figure 17 shows a deployment diagram in which the Mobile Delivery module interacts with two instances of the MSEE Delivery Platform and with other four Service Brokers.

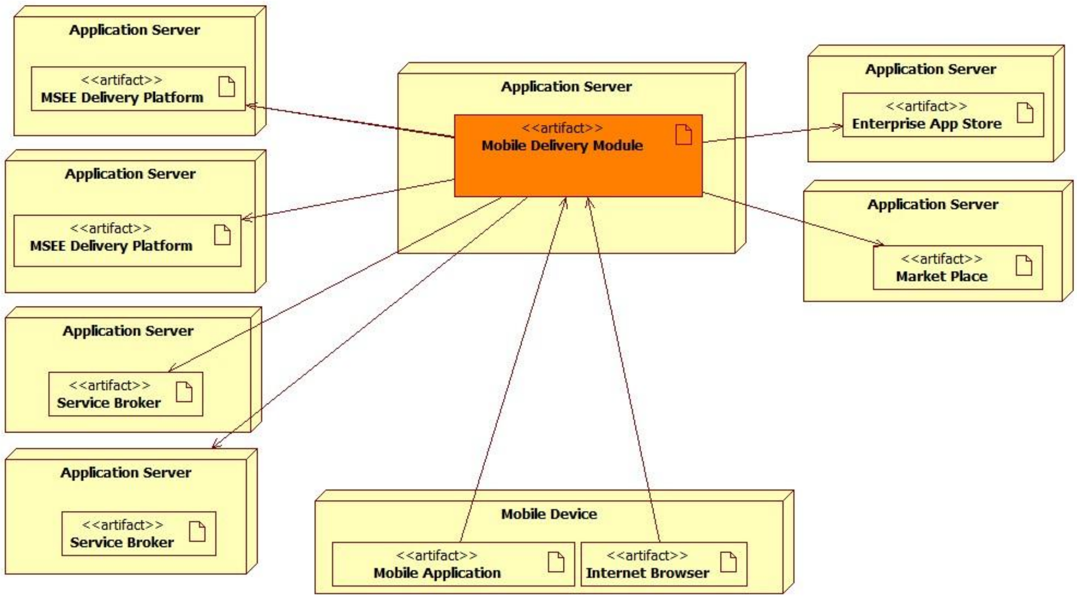


Figure 17: Mobile Delivery Deployment Diagram

The Mobile Delivery Module offers its functionalities through a web site designed for mobile devices and through a REST service (see Figure 18). The user will be able to access the Mobile Delivery Module using a web browser or using a mobile application able to interact with the REST service provided by the module. For example the Ambient Intelligence App described in Section 6.3 enables the user to find services available in discovered devices using the Mobile Delivery module.

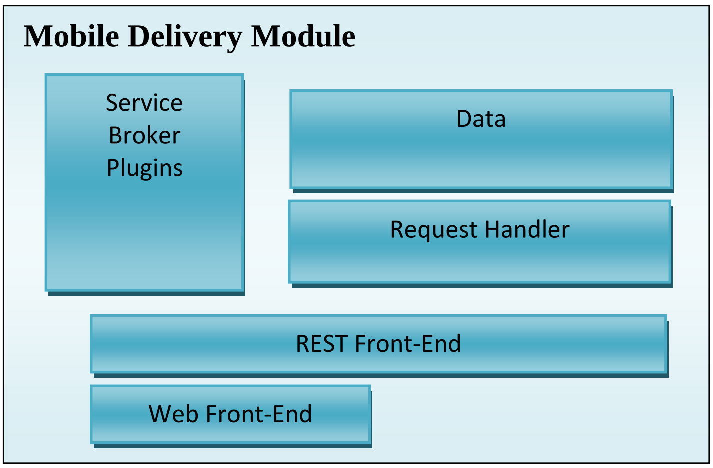
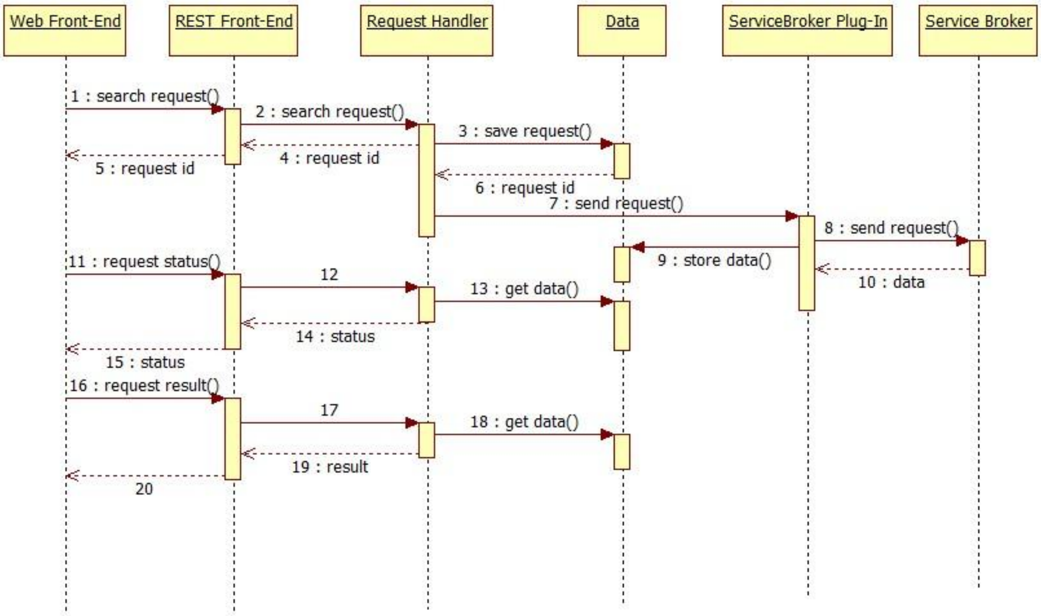


Figure 18: Mobile Delivery Module

The Web Front-End component enables users to access the Mobile Delivery module with mobile browsers and will be implemented as a HTML5 / Javascript (including some Ajax components) web site exploiting internally the REST Front-End component of the module.
The Request Handler will be in charge of managing the searches for services issued by users through the Web Front-End and/or the REST Front-End and of managing the response received by the Delivery Platform as well as by other Service Brokers.
The Service Broker Plugins will include a specific plug-in for each Service Broker to contact since different Service Brokers could be accessible through different interfaces and/or technologies.
Finally the Mobile Delivery module stores (for a certain amount of time) the response received by of the Service Brokers allowing clients (browsers and mobile applications) to access these responses asynchronously (i.e. with Ajax).
The following diagram depicts the main interactions between the various components.



The user exploits the Web Front End to start a search request. The Request Handler saves the request through the Data component that assigns a unique id to the request. The assigned id is returned to the client that from this moment, using the id, can request information about the search status or the search result.
A search can take some time since different Service Brokers are contacted and some of them can be slow or a network delay can occur. With the asynchronous approach the client can avoid to freeze the user interface while it can exploit the "request status" operation and the "request results" operation to update the UI, highlighting the search status or returning the search result.
After saving the request the Request Handler alerts the registered Service Brokers Plug-Ins. Each plug-in manages the interaction with a specific Service Broker saving the search results using the Data component.

Web Front End	
Description	This component is implemented as a web site designed specifically for mobile devices. The web site makes use of Ajax to interact with the REST Front-End component asynchronously ensuring an engaging user experience.
Functionalities /	• Manage the User Interface

services	
Dependencies	
User	This module is used by business users in mobility searching for services.

REST Front End	
Description	This component is implemented as a REST service providing all the functionalities to add new searches and to request update on the search status and on the search results.
Functionalities / services	• Add a search • Cancel a search • Get search status • Get search results
Dependencies	
User	This module is used by the Web front end or by external applications.

Request Handler	
Description	This component handles requests accepted by the REST front end.
Functionalities / services	• Handle Request
Dependencies	• Data component: the Request Handler depends by the Data component to interact with the internal database. • Service Broker Plugins component: the Request Handler depends by plugins to handle search requests.
User	This module is used by the REST Front end.

Data	
Description	This component manages the internal data of the Mobile Delivery module.
Functionalities / services	• Save search • Cancel search

	• Get search result • Get search status • Clean Data
Dependencies	
User	This module is used by the Request Handler component to save searches and to retrieve search status and results. The component is used by plug-ins to store search results and update search status.

Service Broker Plugins	
Description	Each Service Broker is accessed through a dedicated plug-in in charge of issuing the searches and getting back the results.
Functionalities / services	• Send the search to a Service Broker • Receive Service Broker results
Dependencies	• Data component: to store search results
User	This module is used by the Request Handler to handle search requests.

6.3. Ambient Intelligence Module

The Ambient Intelligence vision, advanced in the 1999 by EU IST Advisory Group, included a "human surrounded by computing and advanced networking technology which is aware of his presence, his personality, his needs and is capable of responding intelligently to spoken or gestured indications of desire, and even in engaging in intelligent dialogue".
Even if this vision has not been achieved completely, many steps have been done in the right direction, and today we are surrounded of smart products able to interact with humans and other products.
The main goal of the AMI module is to discover and interact with the devices disseminated in the environment in order to support Ambient Intelligence and Internet of Things (IoT) scenarios and, at the same time, address the general objectives of MSEE and the concrete needs of MSEE end-users.
Figure 19 shows two heterogeneous environments where several devices are available.

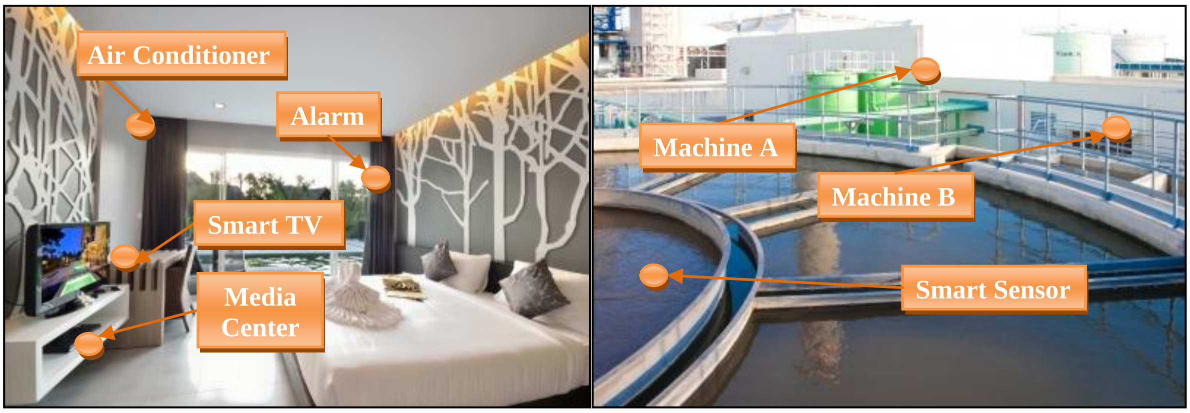


Figure 19: Home and Manufacturing Environments[27]

The following table report the main requirements considered during the definition of AMI functionalities. These requirements have been collected from the MSEE Description of Work document and from the deliverable D52.1 including the initial set of MSEE end-user requirements.

Requirement	Feature
Ambient Intelligence The original Ambient Intelligence vision included a "human surrounded by computing and advanced networking technology which is aware of his presence, his personality, his needs and is capable of responding intelligently to spoken or gestured indications of desire, and even in engaging in intelligent dialogue".	The AMI module helps a product to be aware of other products present in the same environment and to interact with them in order to build a coherent, updated and shared user profile usable by each of them to be aware of user preferences, needs, personality, etc.

[27] Images by arztsamui and John Kasawa (FreeDigitalPhotos.net)

AMI Space The 2002 ISTAG Report "Strategic Orientations & Priorities for IST in FP6" introduced the notion of 'AmI Space' in which there will be seamless interoperation between different environments – home, vehicle, public space, etc.	The AMI module enables the user to save and manage different environments, to group the devices that he uses respect to these environments and to carry on his profile from an environment to another. Moreover the AMI module is able to automatic recognize the user environments.
Internet of Things In the Internet of Things (IoT) the "things" are expected to become active participants in business, information and social processes where they are enabled to interact and communicate among themselves and with the environment by exchanging data and information "sensed" about the environment, while reacting autonomously to the "real/physical world" events and influencing it by running processes that trigger actions and create services with or without direct human intervention.	The AMI module addresses IoT topics helping things to become active participants in business, information and social processes through the access to proper services facilitating interactions with other "smart things" and making things in the same environment aware of each other.
MSEE Servitization Process One of the main objectives of MSEE is to support manufacturing enterprises to go through all servification levels enriching and differentiating their products with new services that could appear and disappear during the products life.	The AMI module enables customers to find services and applications for the products around them and to link the physical available product with their virtual representation in the cloud.
MSEE End Users - Philips Philips pilot application focuses on the extension of the Philips Net TV services accessible from Philips Smart TVs.	The AMI module is able to recognize Philips Net TVs and can suggest the available services and mobile applications. Moreover the module can enable the user to carry, in his mobile devices, all the settings of his TV in order to quickly set up other Net TVs (i.e. TV in a hotel) according his preferences.
MSEE End Users - Indesit The Indesit use case focuses on the "Carefree Washing" service. The capabilities of the Indesit Washing Machines will be extended by a set of services that helping the customer to avoid caring about maintenance, machine control, soap recharge, spare parts, etc. During the product lifecycle (some years) some services will appear while other could be discontinued.	The AMI module can help the user to discover new services for his particular WM (model, features, production year, available firmware, etc.) and can support interactions between the WM, the user and the available services. The module can also helps the Indesit WM to be aware of other machines in the same environment in order to enhance machine-to-machine interactions.

MSEE End Users – Ibarmia	
Ibarmia use case focuses on the “Intelligent Maintenance Services Ecosystem” in which, for foreign countries, the machine maintenance operations will be carried out by external authorized technical assistance service providers.	The AMI module can help external technicians entering in unknown industrial facilities to identify Ibarmia machines (model, serial code, etc.) and can check if new services/information (i.e. technical procedures, advices, etc.) are available for the machines.

Some assets of the AMI module internal architecture and some of its functionalities have been inspired by the results achieved in past research projects (in particular PERSIST, MIMOSA [28]and MINAami[29]). These projects focused specifically on Ambient Intelligence issues applied to mobile devices. The AMI module addresses several Ambient Intelligence issues but must be also considered in the context of MSEE project where plays the innovative role of a virtual bridge between the products, the customers and the services provided by the smart manufacture enterprises, supporting these manufactory enterprises to achieve the *product+services* and *products2services* servitization phases.
The Ambient Intelligence module handles two main concepts namely devices and environments. The term *device* is used here as synonym of product (washing machine, TV, sensor, etc.) or thing or entity of interest (IoT terminology) while the term *mobile device* identifies the user mobile phone, smartphone or tablet.

Figure 20: User, Environments and Devices

Figure 20 shows three environments (home, office, hotel room) where there are some devices (TV, satellite receiver, air conditioner, phone, etc.). The user moves between these environments carrying on his mobile device where the AMI Ambient Intelligence App is installed.

[28] MIMOSA website, www.mimosa-fp6.com, accessed 22 May 2012
[29] MINAmI website, www.fp6-minami.org, accessed 22 May 2012

The AMI module is implemented as an Android application (Figure 21 shows the AMI app internal architecture) exploiting the Android framework building blocks and the mobile phone hardware (i.e. camera, bluetooth, network connections, etc.).
The main components of the module are the Device (dedicated to manage devices) and the Environment (dedicated to manage environments) ones.
In addition there are components devoted to handle the user interface (UI), the internal data (Data) and the discovery of applications and service on the mobile phone and on the cloud (Application/Services).

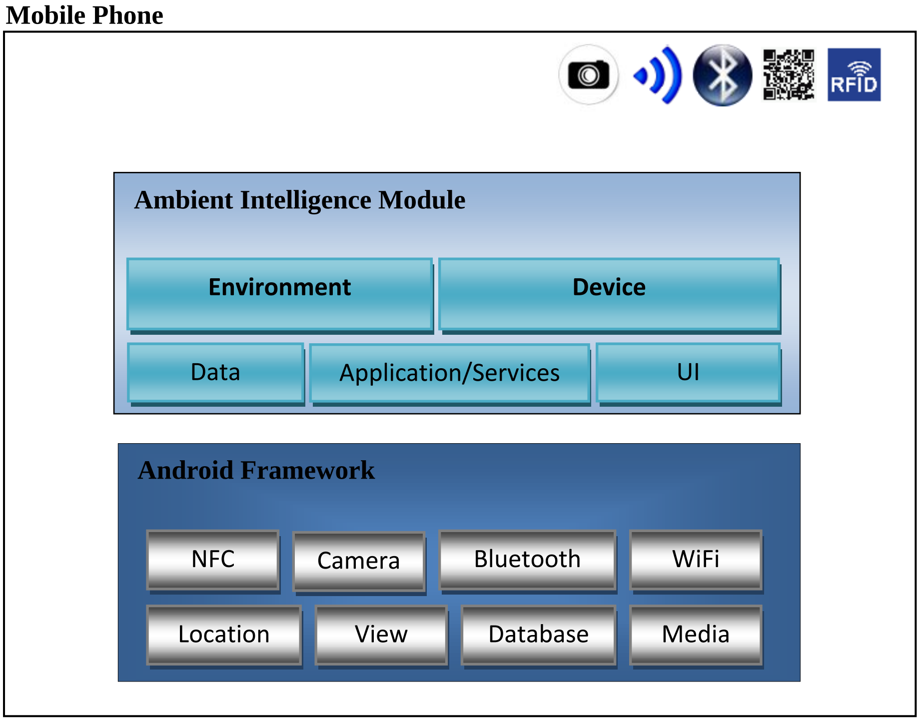


Figure 21: AMI Module

A physical device in the ambient is represented within the AMI module as an object with the following attributes:

- Name: a descriptive name assigned by the user (i.e. Home WM, Hotel Bristol TV, etc.)
- Thumbnail: a thumbnail of the device taken using the mobile device camera
- Identity Information: a generic list of key-value pairs that can include several attributes like the product model, the manufacturer name, the serial code, etc. and that is defined by the device manufacturer.

Once a device has been identified the user can save it and associate it to an environment.
Some devices are able to communicate through one or more communication channels like the Bluetooth or the WiFi technologies while other devices are not able to communicate but can provide some information about themselves through QR codes, 1D bar codes, RFID tag, etc.
In order to discover and identify a device the AMI module exploits several technologies. In some case the discovery and identification is user driven (i.e. the user see a legacy manufactory machine and input its product code using the mobile device keyboard) while in other case is driven by the module (the AMI module discovers a temperature sensor automatically using the Bluetooth technology).
The following sub-sections provide some details regarding relevant topics related to device identification, environment recognition and regarding the functionalities provided by the AMI

module once a device has been identified. Finally the Section 6..3.5 presents the internal components of the module.

6..3.1. Device Id

When a device has been identified it is associated with an internal identifier together with a list of attributes provided by the product manufacturer. The attribute list can include the product model, manufacturer, firmware and so on, and it is used by the AMI module to provide some features such as to find available services for a certain product model. The list of attributes can include information belonging to one or more product identification standards.
Even if today several product identification standards exist, there is not a universal standard and some standards can be applied only to particular products (i.e. books). In the following it is provided a short summary of existing product identification standards.

Universal Product Code
The Universal Product Code (UPC) is a 12 numerical digits barcode, uniquely assigned to each trade item, widely used in North America, and in countries including the UK, Australia, and New Zealand for tracking trade items in stores.

International Article Number
An EAN-13 barcode (originally European Article Number) is a 13 digit barcode standard which is a superset of the original Universal Product Code system.

Global Trade Item Number [30] is an identifier for trade items used to look up product information in a database (often by inputting the number through a bar code scanner pointed at an actual product) which may belong to a retailer, manufacturer, collector, researcher, or other entity. The uniqueness and universality of the identifier is useful in establishing which product in one database corresponds to which product in another database, especially across organizational boundaries.

International Standard Book Number
The International Standard Book Number (ISBN) is a unique numeric commercial book identifier barcode composed by 13 digits and compatible with the International Article Number Bookland. An ISBN code is assigned to each edition and variation (except re printings) of a book.

6..3.2. Device Discovery and Identification

According to its physical and technological features a device can be discovered by the user directly (for example the user see a big machine) and/or by the AMI module (for example the module discovers an hidden temperature sensor) through WiFi, RFID, Bluetooth, NFC, etc.
Once a device has been discovered its attribute list must be retrieved and stored into the AMI module. If the device is able to communicate it can send its identification attributes to the AMI module that will store them. Otherwise the user is in charge of providing these attributes using one of the available input methods provided the AMI module (i.e. using the mobile device keyboard or scanning the product QR code).
The rest of the section reports some details on the available discovery and identification options.

Bluetooth Discovery and Identification

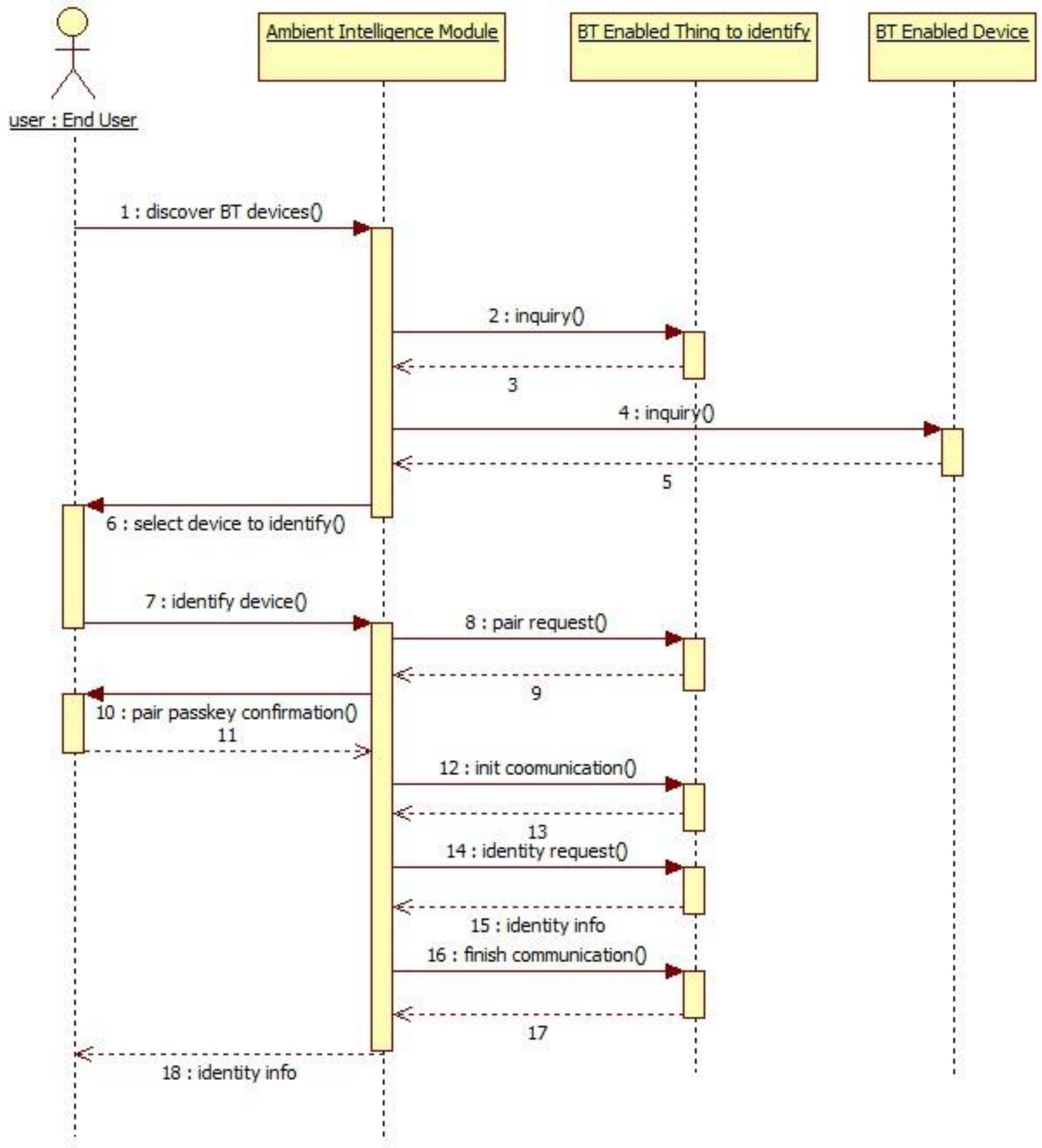
[30] http://www.gtin.info/

Bluetooth (BT) is a proprietary technology standard for exchanging data over short distances (the range depends by the power class of devices and is usually included between 5 and 100 meters) and it is today available on many devices like mobile devices, car navigation systems, sensors, printers, personal computers, game consoles, watches, health devices, etc.
BT technology is based on a packet-based protocol with a master-slave structure. The interoperability of different application exploiting the protocol is accomplished by BT profiles[31].
BT profiles define the interactions, including application behaviours and exchanged data formats, between the layers of the bluetooth technology stack inside a device as well as the peer-to-peer interactions of specific layers between different devices.
The Bluetooth system defines a base profile (GAP), which all BT devices implement, focusing on device discovery, link establishment and security procedures.
The GAP profile enables the development of applications, deployed on a BT device, enabling the user to discover other BT devices, to acquire their name and address and to establish a connection following a security procedure named “pairing process”.
The Serial Port Profile (SDP) enables two devices to establish a communication channel like they were connected by a physical cable through a virtual serial port abstraction. Once the connection has been established the two devices can exploit a communication channel to exchange data.
The GAP and the SPP profiles are used by the AMI module to discover and identify things. In practice the user, exploiting the AMI module deployed on his mobile device, performs the following actions:

1. Search BT devices
2. Select a device of interest and follow the pairing process
3. Retrieve the device identification attributes
4. Save the device in the AMI module

The following diagram summarizes the interactions happening during the identification of a device.

[31] Bluetooth Core Specification V.4 https://www.bluetooth.org/Technical/Specifications/adopted.htm



The user starts asking the AMI module to look for available BT devices in the environment. The AMI module exploits the bluetooth capabilities of the user device to accomplish the request and to find all BT devices in the user ambient showing to the user their descriptive names. Examples of names can be “Motorola Headphone X23” or “Indesit Aqualtis WM” etc. At this point the user is requested to select the device to be identified (i.e. Indesit Aqualtis WM). The device to be identified must support the SPP profile playing the Acceptor role. After the pairing process the AMI module (playing the role of Initiator) establishes a communication channel with the device and sends over the channel an identity request (request message). The device must reply to this request with a JSON string including a set of identification attributes expressed as key-value pairs (i.e. model: aqualtis aq8l-09, manufacturer: indesit, ean: xoo78554). When identity information has been received the AMI module sends a “bye” message so the two parties can release the communication channel correctly and the device can go on waiting for other identity request.

Wi-Fi Discovery and Identification

Wi-Fi is the acronym of "*Wireless-Fidelity*", a trademark of the Wi-Fi Alliance. Wi-Fi[32] networks use radio technologies to provide secure, reliable, fast wireless connectivity. A Wi-Fi Local Area Network can be used to connect electronic devices to each other, to the Internet, and to wired networks which use Ethernet technology.
Today many mobile devices, personal computers and other apparatus are equipped with a wireless network interface controller and in many area (public buildings, industrial plants, private houses, etc.) it is available a Wi-Fi LAN.
Respect to bluetooth the Wi-Fi technology has a greater range and a major power request so, even if some devices support both Wi-Fi and BT, many devices support only one of these technologies according to their features and goals.
The AMI module, exploiting the mobile phones and the Android framework functionalities, is able to discover and to connect to available LAN networks and, if needed, to authenticate to these networks. Once the AMI module is connected to a LAN networks it uses the [33]TCP-IP (Internet protocol suite) to communicate with other devices on the same LAN.
Wi-Fi LAN do not support the discovery of other devices connected for security reasons. The approach used by the AMI module is to send a multi-cast message including an identity request and to handle the responses received by other devices.

Barcode and QRCodes Identification

A barcode (see Figure 22) is an optical machine-readable representation of data, which shows data about the object to which it attaches. Originally barcodes represented data by varying the widths and spacing of parallel lines, and may be referred to as linear or one-dimensional (1D). Later they evolved into rectangles, dots, hexagons and other geometric patterns in two dimensions (2D). Barcodes originally were scanned by special optical scanners called barcode readers; later, scanners and interpretive software became available on devices including desktop printers and smartphones.[34]

Figure 22: Barcode

QR Code[35] (abbreviated from Quick Response Code) is the trademark for a type of matrix barcode (or two-dimensional code) first designed for the automotive industry. More recently, the system has become popular outside the industry due to its fast readability and large storage capacity compared to standard UPC barcodes. Due to their graphical features to scan a bar code with a mobile device is easy and fast and QR code can store a lot of text data (up to 4.296 characters).
The QR code shown in Figure 23 stores the model, the manufacturer and the serial code of a product (JSON encoded): *{ "model": "aspire", "manufacturer": "acer", "serial_code": "882IIOP12"}*.

[32] http://www.wi-fi.org/discover-and-learn
[33] http://en.wikipedia.org/wiki/Internet_protocol_suite
[34] http://en.wikipedia.org/wiki/Barcode
[35] http://en.wikipedia.org/wiki/QR_code

Figure 23: QRCode

The AMI module is able to scan Bar codes and QR codes exploiting the Zxing [36]Android application. Even if the code of Zxing is licensed under the Apache License 2.0 it is not integrated in the module in order to take advantage of Zxing updates provided by the Android Automatic Update mechanism. Zxing supports the scanning of the several following bar code formats.

Keyboard Identification

Even if a device is not able to provide identity information actively or through a Bar/QR code the user can provide some information on the device using the mobile device keyboard.

In the near future new identification and discovery methods will be considered (i.e. Near Field Communication, Radio Frequency Identification) considering the availability of these technology on mobile devices (today only few devices supports these technologies).

6..3.3. Environment Recognition

An environment is a *virtual location* where some devices are available. An environment can represent a place characterized by a geographic location (home), a *class* of places (Hilton hotels) or even an environment able to change location like the user car or a train wagon.
The AMI module enables the user to save his environments assigning them a name and a thumbnail and defining the environment characteristics.
The AMI module can recognize an environment, previously saved by the user, using:

- Geographical attributes (detected using GPS, cell phone location, WiFi location, etc.)
- Presence of a Wi-Fi network
- Presence of a connected bluetooth device (i.e. car wireless earphone)

In some case it is not possible to recognize an environment and the user can select the current environment manually. For example it is not possible to distinguish between two environments representing two adjacent rooms where GPS is not available (GPS does not work inside buildings), where the same WiFi network is available and where there are not particular BT devices connected.
When the user saves a new environment he can define on which basis the AMI module must recognize the environment. The recognition process is transparent for the user and is based on an environment recognition algorithm exploiting several technologies.
The idea behind the algorithm is that when more than an environment is eligible as *current environment* it is selected the environment associated to the available technology with the lowest range. The following ranges are taken in account:

- GSM Cell phone: some kilometres
- Phone GPS Accuracy: 3-5 meters
- Bluetooth range (class 2 and 3): 5-10 meters
- WiFi range: about 32 meters indoors and 95 m outdoor

[36] code.google.com/p/zxing/

To retrieve information on the current user environment (location, available Wi-Fi networks and connected devices) is an expensive operation in terms of battery consumption. The AMI module implements a cache mechanism able to minimize the power request.

6..3.4. Functionalities available for identified devices

Once a device has been identified (automatically or manually) it can be associated to an environment. At this point the AMI module is able to provide the following functionalities:

- find already installed applications to interact with the device
- find new services and applications to interact with the device
- store on the mobile device the information provided by the device (i.e. user profile, settings, preferences) and export them to devices in other environments enabling the user to bring his user context in different environments.
- receive textual and multimedia information from devices not equipped with proper displays to show them (air conditioners, washing machines, temperature sensors, humidity sensors, etc.)
- send a multicast message to all the devices in the same physical environment making them aware of each other (enabling IoT and Ambient Intelligence scenarios).

6..3.5. AMI functional components

The following tables report some details on the functional components shown in Figure 21:

Environment	
Description	This component manages the saving, the modification and the recognition of environments.
Functionalities / services	• Save Environment • Modify Environment • Delete Environment • Recognize Environment
Dependencies	• Android Framework • Data component: used to save/retrieve data about environments
User	This module is used by the UI when the user handles his environments and when the module suggests to the user the current environment.

Device	
Description	This component is dedicated to manage the various operations on devices (discovery, identification, saving, etc.)

Functionalities / services	• Discover Device • Identify Device • Save Device • Delete Device • Interact with Device: settings save/export, request device messages to be shown to the user, inform the device about other devices in the same environment, etc.
Dependencies	• Android Framework • Data component: used to save/retrieve data about devices • Application/Services: used to find application and services for the devices
User	This module is used by the UI when the user handles his devices.

Data	
Description	This component manages the internal data that are stored into a local database.
Functionalities / services	• Save/Update/Delete device • Get list of devices • Save/Update/Delete environment • Get list of environments • Save/Retrieve user preferences
Dependencies	• Android Framework
User	This module is used by the UI, Device and Environment components when they need to store/retrieve data.

Application/Services	
Description	This component is in charge of finding application and services for a device.
Functionalities / services	• Find Application • Find Services
Dependencies	• Android Framework

User	This module is used by the Device component to find applications and services for a device.

GUI	
Description	This component is in charge of building the User Interface (UI) and manage the interactions with the user.
Functionalities / services	• UI graphical building • UI event management and UI update in response of user interactions • UI update in response to notification of other components
Dependencies	• Android Framework
User	The user.

6.4. Mobile Collaboration Module

The role of the Innovation Ecosystem Platform (IEP), described in the deliverable D26.1 "Innovation Ecosystem platform specification platform", is to enable the execution and the enactment of the business processes that characterize a Manufacturing Service Ecosystem. The functional architecture of the platform emphasizes six main modules:

- Ecosystem Front-End: graphical environment integrating all the other components and the point of access to the Ecosystem for ecosystem members
- Change Management Wizard: tool to guide the enterprises through change management processes and toward a higher level of maturity with respect to their servitization level and their involvement in the ecosystem.
- Ideation Process Management System: system for fostering and managing the creation of new and innovative ideas through cross-organizational and user-inclusive initiatives.
- Collaboration Services: set of communication and collaboration services that allow members of the ecosystem to share contents and collaborate on their evolution, discuss and interact through web2.0 tools.
- Marketplace and Tangible/Intangible Assets Repository: component providing the means to ecosystem members to manage virtualized tangible and intangible assets and to offer their capabilities as services to the market.
- Operational/Ecosystem-governance Management System: components representing the front-end applications for the enactment and monitoring of operational and ecosystem-governance processes.

The Mobile Collaboration module will extend the IEC platform making some functionalities accessible from mobile devices and usable by users in mobility. The candidate functionalities for the module will belong to the "Collaboration Services" component and/or to the "Ideation Process Management System" and/or to the Operational/Ecosystem-governance Management System.

IEC Collaboration Services will allow companies and customer to communicate, collaborate and make business. Example of services are messaging services, forum, wikis, knowledge sharing environments, tools for collaborative tasks execution, social network integration and so on.

The Ideation Process Management System is a component devoted to identify, select and improve, in an iterative manner, the ideas that stimulate the servitization cycle where an idea can be a solution, request or proposal about a product, service or process innovation.

Finally the Operational/Ecosystem-governance Management System is being thought as a workflow-like system that will let stakeholders visualize in a simple way the current status of some ecosystem level as well as act/be part of decision-making processes.

The Mobile Collaboration Module will be part of the final prototype of the Mobile Platform and the specification of the module will be included in the final version of this document (D44.2).

6.5. Multimodal Module

Several studies on Human Computer Interaction focus on innovative user interfaces going beyond the traditional keyboard/mouse input and display based output. Oviatt[37] defines multimodal interfaces as follows: "*Multimodal interfaces process two or more combined user input modes (such as speech, pen, touch, manual gesture, gaze, and head and body movements) in a coordinated manner with multimedia system output. They are a new class of interfaces that aim to recognize naturally occurring forms of human language and behavior, and which incorporate one or more recognition-based technologies (e.g. speech, pen, vision)*".
In other words multimodal interfaces are based on additional and alternative input/output modes or combined input/output modes (i.e. display + text-to-speech, etc.).
The goal of multimodal interface is to offer easier and more expressively powerful and more intuitive ways to use computers, enhancing human/computer interaction in a number of ways[38]:

- Enhancing robustness due to combining different partial information input/output sources.
- Ensuring flexible UI personalization based on user and context.
- Providing new functionality involving multi-user and mobile interaction.

Mobile devices, in particular smartphones, have limited interaction capabilities mainly due to their small size reducing the available space for screens and (virtual) keyboards. They are often used in highly changing environment conditions (light, noise, etc.) and while the user is involved in other activities.
For example let's consider a car navigation application for smartphones. The user exploits it while driving and the car light and noise conditions often change (i.e. the car enters in a gallery or the user increases/decreases the volume of the audio system). In this case a user interface adapting to the environment condition (i.e. changing the GUI colors according the brightness, changing the audio settings according the noise), allowing the user to use alternative input modes (simplified keyboard, vocal input, gesture based input) and output modes (vocal driving instructions, graphical instruction presented on the display, etc.) and suggesting the best modes according to the user context (i.e. disabling vocal input in noisy environments) can guarantee an improved user experience when using the application.
Applications designed to be used in manufacture environments, supporting the users in performing specific tasks, could gain great benefits from the use of multi-modal user interfaces. In this context visual, touch and speech interaction can be better than text and keyboard. Let's consider an application enabling the user to follow a maintenance procedure for a certain industrial machine. With a traditional interface the technician should read step-by-step instructions on the smartphone display and apply these instructions performing specific actions on the machine. To read the instructions could be difficult due to the display size and to the environment lightness and the user would be requested to look alternatively the mobile device screen and the machine (attentional shift). After the completion of each step the user should use the mobile device keyboard to input some notes and he should press the "next" button to go on to the next step.
With a multimodal user interface the technician can read instructions on the screen and/or listen them from the device speakers (or using a wireless earphone), then he could dictate notes instead of writing them and finally he could go on to the next step tracing a particular gesture on the device screen (without be forced to focus his attention on the mobile device display to find and press a small button).

[37] Oviatt, S.L.,Advances in Robust Multimodal Interface Design. In: IEEE Computer Graphics and Applications, vol. 23, september 2003 (2003).
[38] Dumas, Lalanne, Oviatt: Multimodal Interfaces: A Survey of Principles, Models and Frameworks

The multimodal module has two main goals:

- Provide a reusable library simplifying the interactions between mobile applications, the mobile device sensors (used to sense the user context) and the mobile device alternative input/output modes (speech recognition, text to speech, etc.). The user context can include environment specific information (i.e. light, noise, etc.) as well as information on the user actions (i.e. the user is standing still, is moving, is shaking the phone, etc.).
- Demonstrate the advantages of a multi-modal user interface in a concrete application in manufacture environments.

The following table reports some relevant features of mobile devices for the development of multimodal user interfaces.

Feature	**Usage**
Capacitive Touch Screen	Capacitive touch screens enable to acquire user inputs easily. On these screen it is possible to present traditional QWERTY keyboards, but also keyboards customized to accomplish specific tasks and numeric keyboards. Moreover touch screens can be used to recognize user gestures on the screen.
Microphone	The microphone can be used to detect the ambient noise or to perform speech recognition.
Built-in speaker	The built-in speaker can be used to drive user actions by vocal commands (when the user cannot look at the device) or to play meaningful sounds.
Camera	The camera can be used to "look around" applying image recognition algorithms to recognize some items (i.e. things and people) in the user environment.
Ambient light sensor	The ambient light sensor can be used to adapt the behavior and look & feel of the application to the ambient brightness.
Accelerator	Using this sensor it is possible to sense some user actions like if the user shakes the phone or rotate it. Moreover it is possible to understand if the user is moving of he is standing still.

Despite the usefulness of sensors to exploit sensors is a power-wasting activity so it is not possible to use them in a continuous way to get real time information. For example to get information on the environment noise it is possible to activate the microphone for some seconds but not to use it continuously.
On the other side to retrieve information on the user context it is needed a certain time and the accuracy of information depends by this period. In the example before to activate the microphone for 10 seconds could enable the component to understand that the user is in a quiet environment where sometime there is some noise, while to activate the microphone for only 1 second when there is some noise could get to a misleading result.
To summarize, the quality of achieved results and the required battery power are influenced by two factors:

- Sense Frequency: how often sensors are used to perceive the environment and the user actions.
- Sense Duration: how long sensors are used each time the user context must be sensed.

Another thing to consider is that different applications require different accuracies and the same application can require different accuracies at different times according the user task.
The context sense component will enable applications to modify the component configuration at runt-time optimizing the trade-off between result accuracy and power consumption.

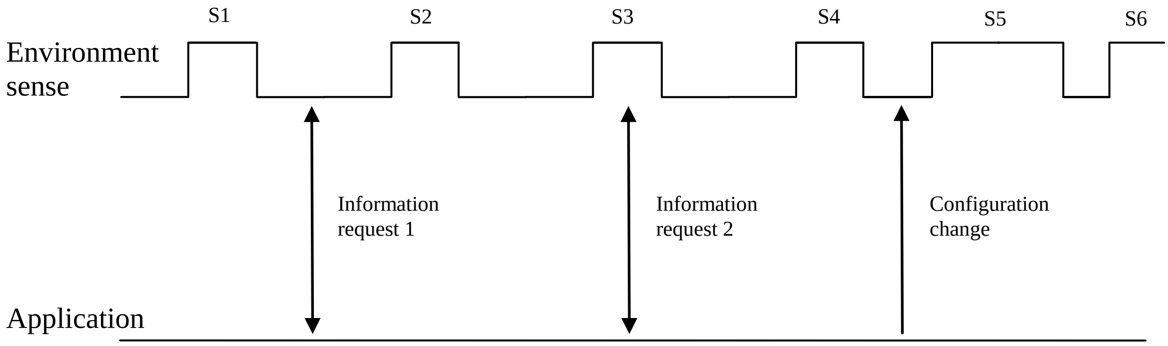


Figure 24: Application and environment sensing

Figure 24 shows an example of interactions between an application and the Multimodal module. The module senses the environment with a given frequency and for a certain period (s1, s2, s3, s4).
The application requires the information on the environment each time it needs them (for example in response to certain user actions). The module always returns the most updated available information (i.e. at the information request 2 are returned the information collected during s2 since the information that the module is collecting in s3 are not still available). Finally the application can change the module configuration. In the figure the application improves the accuracy of information increasing the frequency and the sensing period (S5 is longer than S4 and the time between S6 and S5 is lower than the time between S4 and S3) .
As said before the module will also implement a proof-of-concept prototype showing the advantages of a multi-modal user interface for applications to be used in manufacture environments. The prototype will be inspired by MSEE end users requirements.
The multimodal module will be part of the final Mobile Platform final prototype and its specification will be included in the final version of this document (D44.2).

7. Conclusions

This deliverable consists in an overview of the Mobile Platform architecture and presents the initial specification of the platform and of its modules.

As specified in the MSEE description of work and to achieve its goals, the Mobile Platform addresses several heterogeneous topics starting from mobile development and delivery to ambient intelligence, multi-modal interfaces and mobile collaboration. Due to the heterogeneity of the topics and goals addressed, the platform has been designed as a set of modules. Each module has a particular internal architecture and exploits some state-of-art technologies in order to achieve its own goal and enhance the communication and interoperability with other modules and other MSEE software systems.

When possible, modules are able to interact through web services, independently from the technology of relying parties, and/or they address mobile devices regardless of their manufacturer or operating system.

In the case of the Ambient Intelligence module, in order to strictly interact with the mobile device hardware while ensuring an engaging user experience, it has been necessary to design and develop the module addressing a specific operating system. The Android operating system has been chosen as the main target, considering firstly its diffusion and its continuous market expansion and secondly its open source nature as well as the heterogeneity of mobile devices (in terms of manufacturer, price, features, etc.) based on Android.

The mobile applications included in the Mobile Platform will be implemented as light-weight applications very easy to use and when possible they will be usable even in the absence of Internet connection.

The Mobile Platform addresses smartphones and tablets already available on the market and exploits their appealing features and sensors. However the platform is designed and implemented to be extensible as soon as new devices will appear on the market introducing new features. For example the Ambient Intelligence module is designed to exploit the Near Field Communication technology, which is expected to significantly grow in the coming months and years.

The specification of the modules will be used to develop the first prototype of the Mobile Platform. The mobile platform initial prototype will be presented in several occasions (project meetings, EC review meetings, dissemination events, etc.) in order to collect meaningful feedback driving its extension and refinement for the final prototype.

Finally the Mobile Platform will be exploited in the context of the MSEE pilot applications to be developed under workpackages WP61, WP62, WP63 and WP64.