



Document title: Arrowhead-Tools Deliverable D2.2 “Revised procedure model”

Version
1.0

Author
Gianvito Urgese
Jan van Deventer

Status
Final

Date
2021-05-26

Contact
gianvito.urgese@polito.it
Jan.van.Deventer@ltu.se

Deliverable D2.2 “Revised procedure model”

Work package leader: Gianvito Urgese
gianvito.urgese@polito.it

Jan van Deventer
Jan.van.Deventer@ltu.se

Abstract

This document constitutes deliverable D2.2 of the Arrowhead-Tools project.

“Revised system engineering procedure model”



ECSEL EU project 826452 - Arrowhead Tools
Project Coordinator: Professor Jerker Delsing | Luleå University of Technology

Table of contents

1.	Introduction	3
2.	WP2 activity	6
2.1.	Work Done before D2.1	6
2.2.	Work Done after D2.1	7
3.	The Arrowhead Tools Engineering Process Model.....	7
3.1.	Engineering Process gap analysis	7
3.2.	Definition of technologies and WP2 objectives	9
3.2.1.	Definition of technologies	9
3.2.2.	Definition of WP2 objectives.....	11
3.3.	The AHT-EP ontology for satisfying the WP2 objectives	14
3.4.	The Engineering Process and the eight Phases	19
3.4.1.	Requirements (EPP1) < SubTask 1 >	22
3.4.1.1.	Phase Description.....	23
3.4.1.2.	Advantages of using the Eclipse Arrowhead Framework in this AHT-EP Phase.....	25
3.4.1.3.	Use Case tasks and activities associated to the phase	25
3.4.2.	Functional design (EPP2) < SubTask 2 >	30
3.4.2.1.	Phase Description.....	31
3.4.2.2.	Advantages of using the Eclipse Arrowhead framework in this AHT-EP Phase.....	33
3.4.2.3.	Use Case tasks and activities associated to the phase	33
3.4.3.	Procurement & Engineering (EPP3) < SubTask 3 >	38
3.4.3.1.	Phase Description.....	38
3.4.3.2.	Advantages of using the Eclipse Arrowhead framework in this AHT-EP Phase.....	42
3.4.3.3.	Use Case tasks and activities associated to the phase	44
3.4.4.	Deployment & Commissioning (EPP4) < SubTask 4 >	53
3.4.4.1.	Phase Description.....	53
3.4.4.2.	Advantages of using the Eclipse Arrowhead framework in this AHT-EP Phase.....	55
3.4.4.3.	Use Case tasks and activities associated to the phase	55
3.4.5.	Operations & Management (EPP5) < SubTask 5 >	58
3.4.5.1.	Phase Description.....	59
3.4.5.2.	Advantages of using the Eclipse Arrowhead framework in this AHT-EP Phase.....	64
3.4.5.3.	Use Case tasks and activities associated to the phase	64
3.4.6.	Maintenance, Decommissioning & Recycling (EPP6) < SubTask 6 >	69
3.4.6.1.	Phase Description.....	70
3.4.6.2.	Advantages of using the Eclipse Arrowhead framework in this AHT-EP Phase.....	72

3.4.6.3.	Use Case tasks and activities associated to the phase	72
3.4.7.	Evolution (EPP7) < SubTask 7>.....	77
3.4.7.1.	Phase Description.....	77
3.4.7.2.	Advantages of using the Eclipse Arrowhead framework in this AHT-EP Phase.....	79
3.4.7.3.	Use Case tasks and activities associated to the phase	80
3.4.8.	Training & Education (EPP8) < SubTask 8>	85
3.4.8.1.	Phase Description.....	85
3.4.8.2.	Advantages of using the Eclipse Arrowhead framework in this AHT-EP Phase.....	86
3.4.8.3.	Use Case tasks and activities associated to the phase	87
4.	The Arrowhead Tools Use Cases and the AHT-EP support	95
3.1.	Updated Use Cases summary and AHT-EP analysis	96
3.2.	Updated Use Cases mapping on the AHT-EPPs	97
5.	Conclusions.....	104
6.	Appendixes.....	105
7.	References.....	106
8.	List of abbreviations	110
9.	Revision history	111
	Contributing and reviewing partners	111
9.1.	Amendments.....	112
9.2.	Quality assurance	112

1. Introduction

Work Package 2 (WP2), the Digitalization of the Engineering Process, aims to develop a consolidated engineering process¹ model for supporting the life cycle management of System of Systems (SoS) products that relies on a service oriented architecture (SOA), which can be implemented using an integration platform based on WP3 (Digitalization framework: Integration & Interoperability), WP4 (Tools chain architecture) and WP5 (Tool technology) results.

The proposed engineering process model is broken up into eight engineering phases² where each of them group the main engineering activities performed during the life cycle of a manufacturing system of systems. The goal is to elaborate and consolidate an Eclipse Arrowhead framework [1] compliant engineering process model. The model is designed as a

¹ The process and procedure are being used. The former means: a series of actions or steps taken in order to achieve a particular end; while the latter means: an established or official way of doing something.

² Phases are stages in the process model and not tools. Tools might have the same name and should not be confused with the phases.

flexible system that supports the life cycle of all the use cases of the Arrowhead-Tools (AHT) project.

WP2's main outcome is an engineering process model that has been used to describe the life cycle management of the use cases discussed and implemented in WP3, 4, 5, 6, 7, 8, 9.

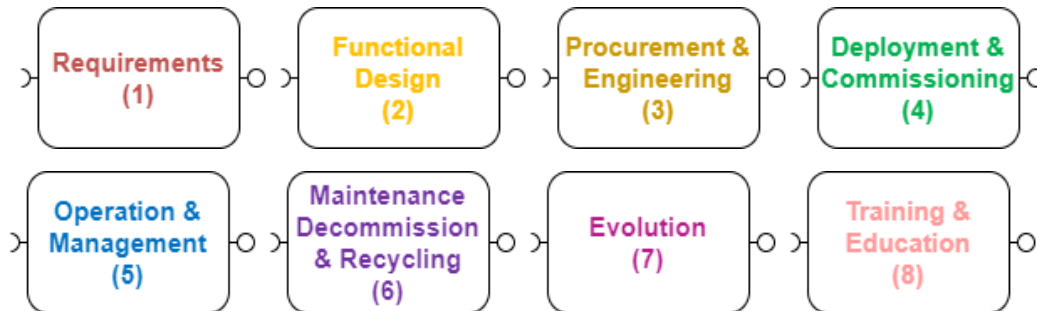


Figure 1 The Arrowhead-Tools Engineering Process (AHT-EP) Phases

The eight phases represented in Figure 1 cover the life cycle of engineered artefacts. The production line, as well as the product, are engineered artefacts such that the engineering process model can apply to both. Traditionally, the production and the product were separate entities, and with the digitalization of the engineering processes, there is an overlap. At times, it might be difficult to distinguish between the engineering phases of the production and the product. This ambiguity becomes even more pronounced with smart products that interact with their production system [2]. With a clear goal, the Arrowhead-Tools' project uses its objective to steer the research.

Throughout the development period, the six Arrowhead-Tools objectives are guiding beacons. They are:

1. Reduction of solution engineering costs by 20-50%.
2. Interoperability for IoT and SoS engineering tools.
3. Interoperability and integration of data from legacy automation engineering tools to the Arrowhead-Framework integration platform.
4. Integration platform interoperability with emerging digitalization and automation framework.
5. Flexible, interoperable and manageable security for digitalization and automation solutions.
6. Training material (HW and SW) for professional engineers.

The process model, the digitalization framework and the tool chain architecture have to be aligned with these six objectives.

Based on the Arrowhead-Tools project's requirement (WP1), tools for each engineering phase, which are based on a SOA, are in the development phase (WP4-WP5) and will ensure interoperability via a framework (WP3). This concept is tied back to reality through the use cases of the project.

In the deliverable D2.1 [3] we introduced a preliminary version of the Arrowhead Tools Engineering Process (AHT-EP) that have been updated and consolidated during the WP2 activity. Now, in month 24 of the AHT project, an updated version is reported in the current document D2.2. D2.1 contains an initial reflection to understand where the Arrowhead-Tools project starts from, and what its use cases have in common in terms of the eight EP phases. Whereas, the current document D2.2 contains the updated version of the Arrowhead Tools Engineering Process.

Yet, WP2 is interdependent with WP1 and WP4 as stated above. Its deliverable had to be coordinated with the other work packages and deliverables (as described in Figure 2).

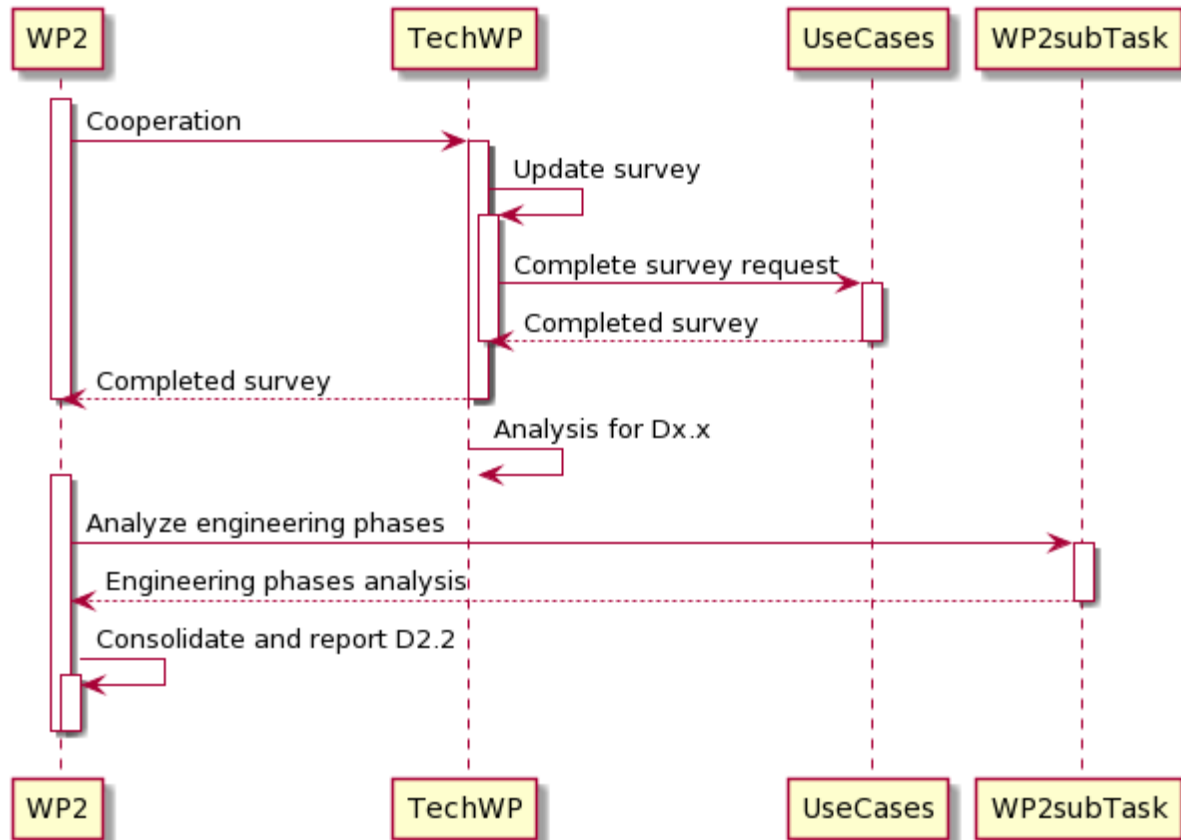


Figure 2 WP2 delivery D2.1 and D2.2 creation process

The cooperation between the interdependent work packages led to the creation of the *WP124_survey* [3] that after D2.1 have been evolved in a second enhanced version named *WP12410_survey* [4] to collect more precise information about the Engineering Process activities implemented in each use case, during the implementation phase, which each use case had to fill out as well as possible. The *WP12410_survey* were then analyzed and information was extracted for each engineering process phase.

The aim of the present document is to describe the consolidated AHT-EP model, the ontology, and how the AHT Use Cases (AHT-UCs) have used the model for defining the life cycle management schema of the system developed in their use case. Thus highlighting also how the adoption of the AHT-EP model help the use cases in matching the four WP2 objectives listed below:

1. The change from design time to run time engineering.
2. The move from single to integrated multi stakeholder automation and digitalization.
3. Handling of substantially increased number of I/O's due to much more fine grained automation.
4. Digital learning and training activities as an integral part of the engineering cycle.

The deliverable starts with a short summary of the activity performed in WP2 for elaborating the final AHT-EP model. Then, we report a gap analysis that summarize the main gaps of alternative engineering process models and reference 3D architectures that are documented in the state-of-the-art.

Following this introduction, we provide a definition of each one of the four WP2 objectives.

The deliverable continues with the introduction of the ontology, consolidated in this second part of the AHT project, for the definition of all the components and concepts used for building the AHT-EP of each use case.

Then, we provide a description of each Engineering Process phase, with focus on the essence of what these phases mean in the context of the AHT project, i.e. service oriented architecture paradigms. We discuss also how the AHT-EP and the Eclipse Arrowhead Framework help the use case leaders in matching the six project and four WP2 objectives. Moreover, for each phase we reported some practical example of activity performed in the phase by some of the use cases supported in the project.

In the last part of the document we reports an updated alignment of the use cases on the eight EP phases by highlighting the WP2 and the project's overall objectives that each use case can potentially match in each engineering phase.

2. WP2 activity

WP2 activity is composed by the Task 2.1 with eight SubTasks.

The task 2.1 investigates existing engineering procedures for automation and digitalization in a production environment. The outcome of this task is an updated and flexible Engineering Process that addresses:

- The change from design time to run time engineering and non-stop evolution
- The move from single stakeholder to integrated multi-stakeholder automation and digitalization
- Scaling to substantially increased number of I/O's due to much more fine grained automation

The task 2.1 is in charge to integrate inputs from SubTasks 2.1.1 to 2.1.8 that describe the eight phases of the proposed AHT Engineering Process (AHT-EP) shown in Figure 1.

Each SubTask integrates the information collected from the Use Cases (UCs) to identify the activities, methodologies, and tools that the AHT-EP should support in each phase for enabling users to easily describe the life cycle of a system by exploiting a SOA paradigm.

In order to support the overall analysis, we interviewed the use case leaders with a new version of survey [4] designed by the leaders of WP1, WP2, WP4 and WP10 for acquiring fundamental aspects of the use cases that should be evaluated for matching the project and WP2 objectives.

2.1. Work Done before D2.1

In the first six months of the project we have produced a detailed description of the AHT Engineering Process Phases (AHT-EPPs) that has been used as reference from UC leaders for mapping the Use Case Engineering Process (UC-EP) on the AHT-EP.

We created an ontology for representing, with a graphical and a tabular notation, the direct graph that link the AHT-EPPs used for the management of the life cycle of each cyber physical systems and services produced in the use cases.

Moreover, we collected information about the Engineering Process phases currently adopted in each of the use cases domain or field. For this purpose, all use case leaders have been requested to fill out a survey that we have created in collaboration with the WP1 and WP4 leaders.

In points 'A, B, C, and G' of the *WP124 survey* [5], each use case leader has described the details of the Engineering Process phases adopted in its field for implementing the use case.

Information regarding the Use Case Engineering Processes has been collected and summarized by Urgese [6] for producing the analysis of the UC mapping on the AHT-EP that is proposed as contribution of the deliverable D2.1 [3].

2.2. Work Done after D2.1

We have created an enhanced version of the survey, named *WP12410_survey* [4] also involving WP10 with questions about the standards adopted in the use cases. Together with WP1, WP4 and WP10 leaders, we have improved the survey for identify aspects of the Engineering Process that can be revealed during the implementation of the technology WPs in charge to implement the use cases with the Eclipse Arrowhead framework technology. Then, we asked partners to update the use case survey with more specific information and to provide us the description of their engineering process with the AHT-EP ontology defined in the WP2.

Once surveys have been released by partners, each SubTask leader have analyzed the points 'A, B, C, and F' of all the *WP12410_surveys* to isolate and characterize the Engineering Process phase which the SubTask addresses specifically. Then, the SubTask leaders have integrated the information so as to define a flexible phase that can have the potential to be adapted and used in the life cycle description of all the use cases.

We have elaborated and defined the four WP2 objectives for provide a common reference baseline. Partners have analyzed the information collect form UCs for evaluating if the six AHT and the four WP2 objectives predicted to be potentially reachable for each Engineering Process phase of the UC have been correctly matched because the adoption of the AHT-EP or because the usage of the Eclipse Arrowhead framework.

The task leader together with the WP2 leaders have analyzed and integrated the Engineering Process phase descriptions, produced by SubTask leaders, and finalized the definition of the final version of the AHT-EP model that can drive the implementation of tools and methodologies by WP3, WP4, and WP5 to support each use case during the life cycle management.

3. The Arrowhead Tools Engineering Process Model

The purpose of WP2 is to provide a consolidated Engineering Process, relying on SOA, which can be implemented using the integration platform based on WP3 and WP4 results. In recent years, industry consortia have defined several standards for integrating the new industry paradigms (Industry 4.0) into Engineering Processes used to regulate the product/system life cycle. These new standards are required for helping the industries in facing new challenges related to the more complexity of the systems that must be reconfigurable and able to collaborate with other systems developed by different stakeholders.

3.1. Engineering Process gap analysis

In the deliverable D10.1 [7] WP10 leaders listed and investigated the most representative standards to be considered during the definition of the AHT-EP architecture and the functionalities to be supported by the AHT-EP. In the following, we briefly list the most significant standards with relative references:

- The evergreen *V-model* [8] represents a development process that may be considered an extension of the waterfall model. Instead of moving down linearly, the process steps are curved upwards after the coding phase, to form the typical V shape.
- *Reference Architecture Model Industrie 4.0 (RAMI 4.0)* [9] is a three-dimensional map showing the most important aspects of Industrie 4.0. Its adoption ensures that all participants involved share a common perspective and develop a common understanding.
- *Smart Manufacturing ecosystem developed by NIST* [10] based on the collaboration manufacturing management model of ARC Advisory Group and the hierarchical model of ISO/IEC 62264. NIST describes the SME that encompasses manufacturing pyramid with three dimensions – product, production, and enterprise (business). The product life cycle in the context of the smart manufacturing ecosystem is described as an Engineering Process with the following six phases: Design, Process Planning, Production Engineering, Manufacturing, Use and Service, and End-of-Life and Recycling.
- *The Industrial Internet Reference Architecture (IIRA)* is a standardized open architecture based on industrial production systems. The IIRA abstracts the common characteristics, features and patterns from diverse uses cases associated with the domain of communication, energy, healthcare, manufacturing, security, transporting and logistics [11]. The previous concerns identified by the Industrial Internet Consortium (IIC) are classified and grouped into four viewpoints (Business, Usage, Functional, and Implementation). The IIRA standard support a flexible strategy for the product/service life cycle definition that can be specialized for each industrial sector depending on the use case needs.

We have analyzed the different aspects of the standards mentioned above to produce a gap analysis aimed at identify key features to be supported by the AHT-EP architecture that have been discussed also in D10.3 document [12].

According to our analysis, these new reference architectures lack the ability to address some of the key enabling factors for Industry 4.0, such as service-oriented solutions and SoS that characterize modern and future industrial factories and production systems.

To meet the flexibility and automation levels required by Industry 4.0, IoT-based and SoS-based solutions represent an adequate approach to support the digitalization of modern manufacturing facilities, but technology is not enough because even the engineering process must ensure the same levels of flexibility and automation across all phases of the product life cycle. For this reason, the rigidity of today's approaches to automation, based on standards such as the ISA-95 architecture [13], represents a significant limiting factor. More recent standards, such as RAMI 4.0 and IIRA, attempt to address the issue of engineering process flexibility but do not support it when moving from pure automation to the automation of the digitalization process. RAMI 4.0, for example, proposes managing this complexity by addressing it from three different perspectives (three dimensions of the problem space): (i) business, (ii) systems & components, and (iii) life cycles. The RAMI 4.0 reference model does not elaborate on how the parts within this convoluted solution space are interconnected and interact. It does not consider different stakeholders, who sometimes might have conflicting interests. In other words, it is not clear how the new generation standards

can implement data-driven architecture (Digital Thread) that links together information generated from across the product life cycle [14].

3.2. Definition of technologies and WP2 objectives

In this part of the document we introduce the definition of the most relevant technologies, standards and concepts used for defining the AHT-EP model. Then we provide the definition of the four WP2 objectives.

3.2.1. Definition of technologies

- **System Life Cycle Models:** The term life cycle is one that engineering has borrowed from the natural sciences; it is used to describe both the changes a single organism goes through over its life and how the lives of multiple organisms interact to sustain or evolve a population. We use it in engineering in the same ways to describe the complete life of an instance of a system-of-interest (Sol); and the managed combination of multiple such instances to provide capabilities that deliver stakeholder satisfaction. A life cycle model identifies the major stages that a specific Sol goes through, from its inception to its retirement. Life cycle models are generally implemented in development projects and are strongly aligned with management planning and decision making. The life cycle model is one of the key concepts of systems engineering (SE). A life cycle for a system generally consists of a series of stages regulated by a set of management decisions which confirm that the system is mature enough to leave one stage and enter another [15].
- **System of Systems (SoS):** According to [15] there are several definitions of system(s) of systems (SoS), some of which are dependent on the particularity of an application area. Maier in 1998 [16] postulated five key characteristics (not criteria) of SoS: operational independence of component systems, managerial independence of component systems, geographical distribution, emergent behavior, and evolutionary development processes, and identified operational independence and managerial independence as the two principal distinguishing characteristics for applying the term 'systems-of-systems.' A system that does not exhibit these two characteristics is not considered a system-of-systems regardless of the complexity or geographic distribution of its components. In the Maier characterization, emergence is noted as a common characteristic of SoS particularly in SoS composed of multiple large existing systems, based on the challenge (in time and resources) of subjecting all possible logical threads across the myriad functions, capabilities, and data of the systems in an SoS. As introduced in the article Emergence [17], there are risks associated with unexpected or unintended behavior resulting from combining systems that have individually complex behavior. These become serious in cases which safety, for example, is threatened through unintended interactions among the functions provided by multiple constituent systems in a SoS.
ISO/IEC/IEEE 21839 (ISO, 2019) [18] provides a definition of SoS and constituent system: i) System of Systems (SoS) — Set of systems or system elements that interact to provide a unique capability that none of the constituent systems can accomplish on its own. Note: Systems elements can be necessary to facilitate the interaction of the constituent systems in the system of systems. ii) Constituent Systems — Constituent systems can be part of one or more SoS. Note: Each constituent is a useful system by itself, having its own development, management goals and resources, but interacts within the SoS to provide the unique capability of the SoS.

- **Service Oriented Architecture (SOA):** Service-oriented architecture (SOA) is an architectural style that supports service orientation [19]. By consequence, it is as well applied in the field of software design where services are provided to the other components by application components, through a communication protocol over a network. A service is a discrete unit of functionality that can be accessed remotely and acted upon and updated independently, such as retrieving a credit card statement online. SOA is also intended to be independent of vendors, products and technologies [20]. Service orientation is a way of thinking in terms of services and service-based development and the outcomes of services.

A service has four properties according to one of many definitions of SOA [21]:

- i) It logically represents a repeatable business activity with a specified outcome.
- ii) It is self-contained.
- iii) It is a black box for its consumers, meaning the consumer does not have to be aware of the service's inner workings.
- iv) It may be composed of other services [22]. Different services can be used in conjunction as a service mesh to provide the functionality of a large software application [23], a principle SOA shares with modular programming. Service-oriented architecture integrates distributed, separately maintained and deployed software components. It is enabled by technologies and standards that facilitate components' communication and cooperation over a network, especially over an IP network.

SOA is related to the idea of an application programming interface (API), an interface or communication protocol between different parts of a computer program intended to simplify the implementation and maintenance of software. An API can be thought of as the service, and the SOA the architecture that allows the service to operate. In SOA, services use protocols that describe how they pass and parse messages using description metadata. This metadata describes both the functional characteristics of the service and quality-of-service characteristics. Service-oriented architecture aims to allow users to combine large chunks of functionality to form applications which are built purely from existing services and combining them in an ad hoc manner. A service presents a simple interface to the requester that abstracts away the underlying complexity acting as a black box. Further users can also access these independent services without any knowledge of their internal implementation [24].

- **Digital Thread:** Digital Thread [25] is a data-driven architecture that links together information generated from across the product life cycle. Though Digital Thread is gaining traction as a digital communication framework to streamline design, manufacturing, and operational processes in order to more efficiently design, build and maintain engineering products. Digital Thread introduces the idea of linking information generated from all stages of the product life cycle (e.g., early concept, design, manufacturing, operation, post-life, and retirement) through a data-driven architecture of shared resources (e.g., sensor output, computational tools, methods, and processes) for real-time and long-term decision making. Furthermore, Digital Thread is envisioned to be the primary or "authoritative" data and communication platform for a company's products at any instance of time. It is important to distinguish the related concept of Digital Twin [26], which is a high-fidelity digital representation to closely mirror the life of a particular product and serial number (e.g., loading history, part replacements, damage, etc.).

The Digital Twin can come in the form of a high-fidelity computational model or a combination of models and tools of sufficient fidelity to simulate the life history of the corresponding product. Digital Thread then can be viewed as containing all the information necessary to generate and provide updates to a Digital Twin.

- **Tool Chain:** A tool chain is a collection of tools and of the definitions of the corresponding interfaces potentially organized in chain-based or parallel structures. Tools in a toolchain can be substituted/replaced with other tools with the same input/output interfaces. It can be design-time, run-time or both. It aims for a certain level of automation in information processing/transfer throughout the engineering process. It can allow iterative use of its parts (tools and toolchains). It can cover only some (not necessarily consecutive) parts of the engineering process, or the whole product lifecycle (typical for general infrastructural tools), even iteratively until the end-of-life phase. Full definition is provided in D4.2 [27].
- **Tool:** According to the definition elaborated during the WP4 activity (D4.2 [27]), a tool is a software or a hardware (with adequate software on-board) entity/artifact that supports cyber physical SoS and SoS engineering activities. The phases of the engineering process in principle can be managed without tools (i.e., with a strong human component), but probably will use some. It could be a design-time or a run-time tool, depending on its place within the process. It can be either service provider, consumer, both or none; in short, it is compliant with the Eclipse Arrowhead framework (the first three cases) or not. We stress that it is not necessary for a tool to support some Arrowhead design phase to implement any services in the strict Eclipse Arrowhead sense; such a tool is called Arrowhead-enabled. In contrast, a Framework-compliant tool is called an Arrowhead Native Tool. The output of a tool should be processable by other tools adopted in the other phases of the engineering process. The output of a tool should be processable by other toolchains. A tool is an atomic part of a toolchain, and cannot be broken down into sub-tools that can work autonomously.
- **Eclipse Arrowhead framework:** The Eclipse Arrowhead idea [28] consists of systems and services that are needed for anyone to design, implement and deploy Arrowhead-compliant System of Systems. The generic concept of the Arrowhead Framework is based on the concept of Service Oriented Architectures, and aims at enabling all of its users to work in a common and unified approach – leading towards high levels of interoperability. The Arrowhead Framework is addressing IoT based automation and digitalization. The approach taken is that the information exchange of elements in the Internet of Things is abstracted to services. This is to enable IoT interoperability in-between almost any IoT elements. The creation of automation is based on the idea of self-contained Local Clouds. Compared to the well-known concept of global clouds, in Arrowhead a local cloud can provide improvements and guarantees regarding: i) Real time data handling, ii) Data and system security, iii) Automation system engineering, and iv) Scalability of automation systems. The WP3 deliverables deeply describe this technology [29].

3.2.2. Definition of WP2 objectives

Work package 2 addresses the digitalization of industrial engineering processes. It is a key undertaking as it revolutionizes the engineering paradigm of how things have been done to how they could be done using a service oriented architecture that binds at run time. The engineering process has traditionally been sequential, static and limited to a single stakeholder. Not anymore!

WP2 proposes additionally an ontology, which, we claim, reveals a structural skeleton for the reference architecture model for Industry 4.0 (RAMI 4.0). That is, it clearly shows how things (systems or tools) are connected within the solution space of the reference architecture.

WP2 provides a consolidated engineering procedure that relies on Service Oriented Architecture (SOA) and can be implemented using the integration platform based on WP3

(Digitalization framework: Integration & Interoperability) and WP4 (Tools chain architecture) results.

From its conception, WP2 had for objectives to investigate existing engineering processes and divulge how loosely coupled and late binding concepts of the Eclipse Arrowhead framework can move modern industry to be more competitive in the world market. The objectives of the work package has been to propose an updated engineering process that addresses four main objectives. We hereafter look at what is meant by these objectives and how do they relate to the Arrowhead Tools project.

1. **The change from design time to run time engineering:** the paradigm shift that this objective tackles is moving from static set of interconnected engineering tools to a set of engineering tools that bind dynamically at execution time to address the overall project objectives.

To grasp the revolutionary impact, we look at what has been and what is to be. In the past, an industrial engineering process could have been modeled with a V-model and implemented with ISA 95 [30]. Both ideas lead to a static toolchain that is implemented at design time. Tools in the requirement phase feed tools in the functional design phase and in turn in the engineering phase. There is no flexibility nor loop back between the phases. One can easily understand that as information propagation is slow in the previous implementations.

Similarly in the ISA 95 pyramid, an asset on the shop floor (a tool in the operation phase) cannot communicate with a procurement system (a tool in the procurement phase) without going through intermediate levels. This forms a static structure that came at the design of a plant.

The vision promoted by the Arrowhead Tools project has been that the tools within an engineering phase can cyber-securely request services or information from other available tools in any engineer phase within the engineering process of any relevant stakeholder. Conceptually that would mean that a tool in the operation phase can make a request to a new tool in the evolution phase, which then makes a new request to a database in the requirements phase.

The ontology developed in WP2 has been used to show these type of connection and making use of the Eclipse Arrowhead framework promotes a service oriented architecture with late binding such that the tools can discover services from other tools at run time forming a system of system forming a toolchain.

2. **The move from single to integrated multi stakeholder automation and digitalization:** the tools or systems described above cannot only communicate with tools in any engineering phases but with multiple tools across the whole engineering process. Making use of Internet technologies enables cyber tools to communicate with any other tools. One can depict this as a web of tools rather than a simple chain of tools.

In a mass customization context, one can consider a scheduling tool in production requesting information from requirements (unique or customized assembly) with additional input from the engineering CAD tool (assembly instruction) and procurement tool (logistics of the correct parts at the correct time in the right place).

Cybersecurity, in between tools, is an issue addressed within a single stakeholder, and of course in between stakeholders. These concepts are clearer to grasp when one look

at the Eclipse Arrowhead framework where systems (or tools in our case) must authenticate themselves with X.509 certificates associated with the stakeholders' certificate authority (CA) and then be authorized to exchange services with other tools. This promote a fined grained implementation since it is at the service level between two specific tools. Furthermore, all communication is encrypted such that only the two communicating tools are able to decrypt the communication. When multiple stakeholders are involved, the concept of local clouds and inter cloud communication is a natural application of the framework.

3. **Handling of substantially increased number of I/O's due to much more fine grained automation:** there is an emerging behavior or byproduct that comes with the above system of systems with multiple input and outputs. The new side effect has to do with the fact that new inputs and outputs do not have to come or go to tools within the same stakeholder but across stakeholders. For example, the output of a procurement tool from one stakeholder becomes the input to the requirements tool of a supplier stakeholder.

An illustrative example would involve the Engineering & Procurement phase of one stakeholder with the Requirement phase of another stakeholder. An engineering tool of the first stakeholder requests a procurement tool for an external service. The procurement tool looks for service providers externally and can discuss the price and delivery dates with different suppliers before placing the order.

4. **Address digital learning and training activities as an integral part of the engineering cycle:** the shift from a static or stiff infrastructure to a fluid one is difficult and requires a clear pedagogical approach. Not only are we trying to replace the pyramid of ISA95, we are having multiple inputs and outputs to each tool at each stakeholder using the Internet with its suite of protocols as a transport medium. Without an integrated education, it is not possible to reach our manufacturing potential.

To help all persons in the involved companies grasp this paradigm shift, WP2 developed an ontology to illustrate how the tools from the different engineering phases interact with each other both within one stakeholder and across multiple stakeholders. Other reference architectures might show a multi- dimensional complexity with base axis such as a business perspective, a manufacturing hierarchy and life cycle perspectives, but in ends up being a green blob with no details. The Arrowhead Tools ontology shows how components are connected to each other. The ontology can further be enhanced with SysML models.

With the integrated education, there is a clear link between the Evolution phase and the Training & Education phase such that all stakeholder's staff can make a smooth transition from old technologies to a digitalized ones.

WP2 becomes a zoom lens enabling stakeholders to understand their big engineering process and then swoop down into each engineering phase, and then further down to the tools. One can follow the connection between tools at some point in time.

This enables the work package to fulfill its four objectives and those of the whole project. To document this, we made use of all use cases [31] within which we considered each of the engineering phases as they form a process. In section 3.4, as we examine each engineering phase in detail, we highlight the relationships of work package objectives and use cases.

3.3. The AHT-EP ontology for satisfying the WP2 objectives

In this work, we decomposed and remodeled the engineering process with the aim of introducing a SoS-based service-oriented solution intended to efficiently, flexibly, and effectively manage the three assets addressed by RAMI 4.0.

In alignment with partners of WP1, WP4, and WP5, we developed an ontology for calling all the various components of the Engineering Process model.

In general, we call the full **Arrowhead Tools Engineering Process** as AHT-EP. The AHT-EP, for matching the use case specific life cycle management flow, can be built by using the **Engineering Process Units (EPU)** that are classified as:

A. **Engineering Process Phase (EPP)**, in the following list are the full EPP names associated with their relative acronyms:

- EPP1: Requirements
- EPP2: Functional Design
- EPP3: Procurement & Engineering
- EPP4: Deployment & Commissioning
- EPP5: Operation & Management
- EPP6: Maintenance, Decommissioning & Recycling
- EPP7: Evolution
- EPP8: Training & Education

B. **Engineering Process Interface**, that represent both the in/out connections between internal AHT-EPPs and the external links with other Engineering Processes controlled by different stakeholders that need to interact with the AHT-EP of a product/service. In the following we report the full list of acronyms categorized as I/O interfaces supporting both internal and external interactions:

- Input:
 - EP-I1: Input for Requirements
 - EP-I2: Input for Functional Design
 - EP-I3: Input for Procurement & Engineering
 - EP-I4: Input for Deployment & Commissioning
 - EP-I5: Input for Operation & Management
 - EP-I6: Input for Maintenance, Decommissioning & Recycling
 - EP-I7: Input for Evolution
 - EP-I8: Input for Training & Education
- Output:
 - EP-O1: Output of Requirements
 - EP-O2: Output of Functional Design
 - EP-O3: Output of Procurement & Engineering
 - EP-O4: Output of Deployment & Commissioning
 - EP-O5: Output of Operation & Management

- EP-O6: Output of Maintenance, Decommissioning & Recycling
- EP-O7: Output of Evolution
- EP-O8: Output of Training & Education

Moreover, we defined a so-called **Engineering Process Mapping (EPM)**, to identify the link between tools and one or more EPU it covers.

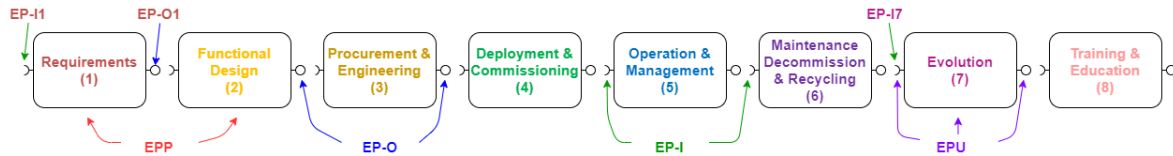


Figure 3 Arrowhead Tools Engineering Process and Engineering Process Units

In order to support the se cases of the different domains, the AHT-EP can be designed by connecting the EPPs in a customized flow that is not necessary the succession of phases proposed in the Figure 3. The EP-I and EP-O interfaces can be more than one for each EPP and can serve for connecting each EPP with external Engineering Processes from other stakeholders that interact in the life cycle of the product/system developed in the UC.

In the following we propose a rule for enumerating the multiple EP-I/EP-O of a single EPP. In case of multiple EP-I we begin to assign a letter to each interface in clockwise order starting from the input on the left-bottom. In case of EP-O we run the enumeration from the output of the EPP placed on the right-up of the block.

Moreover, we introduced the **Engineering Process Connection (EPC)** enumeration to assign a unique numeric identifier to each EP-I/EP-O connection (a pair of interfaces) involving many stakeholders with complex EPs.

In Figure 4 the representation of two EPPs with multiple input/output interface enumerated with the proposed system.

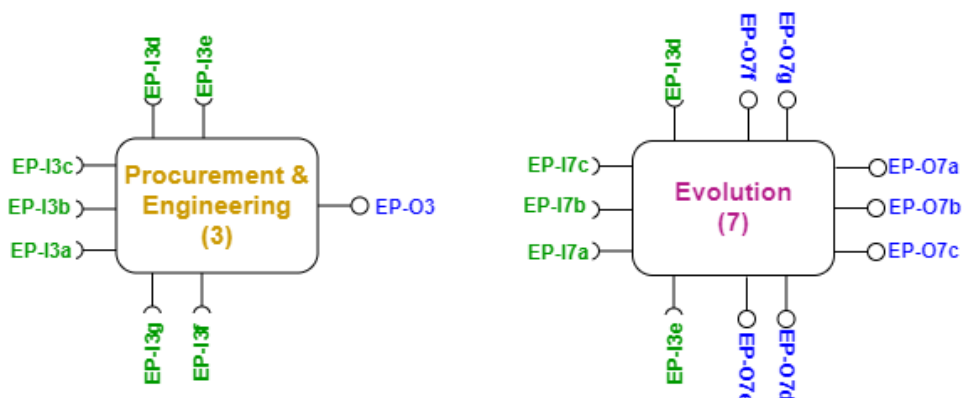


Figure 4 Rule for enumerating the multiple EP-I/EP-O of a single EPP

To minimize the effort in describing the UC EPs in text documents, we proposed a text notation that group several connected EPUs.

In the proposed text representation, the output interface (EP-O) of an EPP connected with the input interface (EP-I) of another EPP can be represented by the arrow symbol “->” so that

Figure 5 can be described as EPP1->EPP2 instead of EPP1, EP-O1, EP-I2, EPP2 that represent the list of all the EPUs.

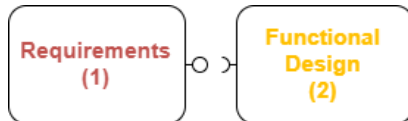


Figure 5 EPP1 connected to EPP2

For representing the connection graph of the AHT-EP we can adopt a standardized tabular format for representing direct graphs (two examples in Table 1 and Table 2).

The structural information in the AHT-EP format is denoted by at least eight rows. The first column contains the EPP name (or relative number). From the second column on, we list the pairing EPP partners of the EPP in the first column. If the EPP on the first column is unpaired, the columns on the right are empty.

In case of interactions with external Engineering Process of third party stakeholders, we can list the external phases from row 9 on by adding a label to the external components. A standard label could be "ex-" appended in front of the external EPP name. A specific file format supported by a parser tool for the automatic management of the EPP structure information could be implemented.

In order to have a concise and understandable idea of the text standard here proposed, we provide in the following two hypothetical configurations represented both graphically and by using a tabular notation that summarizes the life cycle flow architecture.

In the first example, shown in Figure 6, we have the EPP1 connected, by using interfaces EP-O1 and EP-I2 (c1), to the EPP2. EPP2 gives inputs to EPP3 and EPP8 while receive inputs from the evolution phase (EPP7) through the c2 EPC. Procurement & Engineering (EPP3) is connected with EPP4 by using interfaces EP-O3 and EP-I4 (c6) and receives inputs from EPP6 (c5) and EPP7 (c7). EPP4 provides inputs to EPP5 that is connected to EPP6. EPP6 gives feedbacks to EPP3 (EP-O6b, EP-I3b connection) and inputs to EPP7 (EP-O6a, EP-I7a).

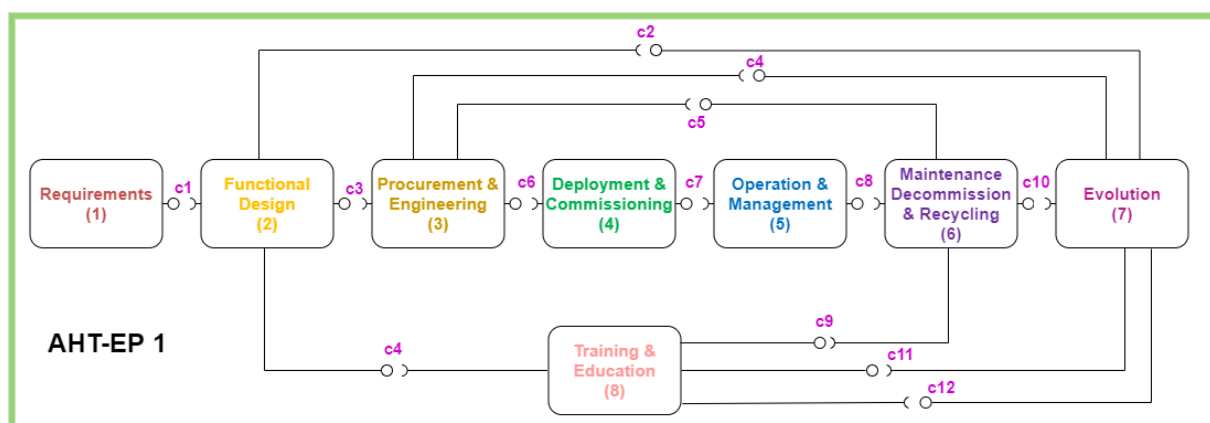


Figure 6 AHT-EP with all the EPPs connected in forward/feedback with multiple EP-Is and EP-Os

The graph of EPP connection is described with the tabular representation (shown in Table 1) where we do not explicitly call the interfaces that can be easily re-extracted when representing the configuration in the table as a direct graph. In this notation, on the first column are listed

the AHT-EPPs that exposes EP-O while in the following columns are reported the EPPs that have input interfaces (EP-I) connected with the EPP of the first column.

Table 1 Tabular representation of the graph of EPP connection of Figure 6

EPP1	EPP2		
EPP2	EPP3	EPP8	
EPP3	EPP4		
EPP4	EPP5		
EPP5	EPP6		
EPP6	EPP3	EPP7	
EPP7	EPP2	EPP3	EPP8
EPP8	EPP6	EPP7	

In Figure 7, we propose a second multi-stakeholder example that uses and connect two AHT-EPs and an unknown engineering process. The dashed lines represent connections external to the main AHT-EP. In this example, the AHT-EP 1 owned by stakeholder 1 (StkH-1) is the main process that uses seven of the eight phases and it is connected with two external engineering processes. The AHT-EP 2 (StkH-2) is composed by three EPPs, where two (EPP1 and EPP4) are connected with the EPP2 (c4) and EPP3 (c5) of the AHT-EP 1, respectively. The external EP receive inputs from the EPP6 (c12) of the AHT-EP1 and provides inputs in the EPP3 (c6).

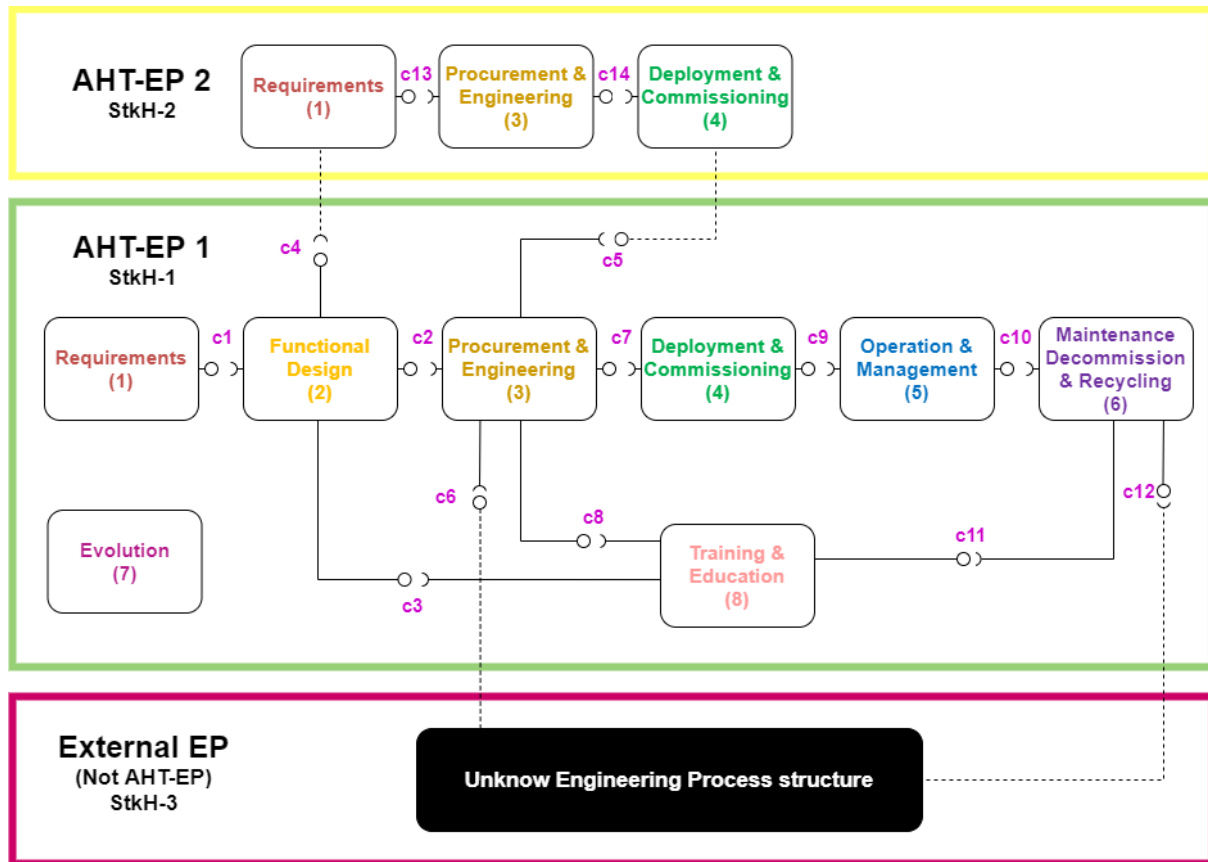


Figure 7 Two AHT-EP from different stakeholders connected with an unknown EP from a third stakeholder

The connection of the AHT-EP 1 representing the main Engineering Process is described with the tabular representation in Table 1.

Table 1 Tabular representation of the graph of EPP connection of Figure 7

EPP1	EPP2		
EPP2	EPP3	EPP8	ex2-EPP1
EPP3	EPP4	EPP8	
EPP4	EPP5		
EPP5	EPP6		
EPP6	EPP3	ex3	
EPP7			
EPP8	EPP6		
ex2-EPP1	ex2-EPP3		
ex2-EPP2			
ex2-EPP3	EPP3		

ex2-EPP4			
ex2-EPP5			
ex2-EPP6			
ex2-EPP7			
ex2-EPP8			
ex3	EPP3		

The ontology here described for producing the AHT-EP can potentially support each UC in the description of an engineering process that match the four WP2 objectives.

1. The change from design time to run time engineering is supported by allowing the developers to include in their AHT-EP the Evolution phase with feedback connections that can give inputs to the other phases.
2. The move from single to integrated multi stakeholder automation and digitalization can be achieved by connecting the AHT-EP of the UC with external engineering process adopted from one or more stakeholders. AHT-EP will support interfaces that can be customized and used for the purpose.
3. The handling of substantially increased number of I/O's due to much more fine grained automation will be guaranteed by the capability of the AHT-EP of handling multiple I/O interfaces for each EPP.
4. The digital learning and training activities as an integral part of the engineering cycle will be supported by the inclusion in the AHT-EP of the Training & Education phase.

3.4. The Engineering Process and the eight Phases

The Engineering Process of the industrial use cases is designed with the AHT-EP model capable of supporting the digitalized life cycle management.

In the Table 2, we provide a short description of the eight phases of the proposed Arrowhead Tools Engineering Process (AHT-EP). The table acknowledges the sub task leaders who dived in the detail of each respective phase and connected it to the use cases.

Table 2 The eight AHT-EPPs

# Phase	Leader	Phase title	Phase description
1	PHC	Requirements	<p><i>Requirements</i> elicitation is the practice of researching and discovering the requirements of a system from users, customers, and other stakeholders. The output of this phase is typically a list of requirements.</p> <p>In this phase the stakeholders cooperate to identify the requirements of the components (HW and SW) composing the IoT and cyber-physical ecosystem.</p>

2	ULMA	Functional design	<p>The <i>functional design</i> phase consists in adopting the "functional design" paradigm to simplify the design of the system/product. A functional design assures that each modular part of the system/product has only one responsibility and performs that responsibility with the minimum of side effects on other parts. Functionally designed modules tend to have low coupling. The output of this phase is typically a model, or an architecture.</p> <p>In this phase, stakeholders develop the cyber-physical models of the components and their functionalities to be subsequently simulated and validated. During the life cycle of the product these models will be continuously used to represent and simulate behaviors of the components in the digital thread eventually supporting the digital twin of the engineering process phases.</p>
3	KAI	Procurement & Engineering	<p>The <i>procurement</i> is the process of finding and agreeing to terms, and acquiring goods, services, or works from an external source required to engineer the system/product, construct, and manufacture it. Procurement is used to ensure the buyer receives goods, services, or works at the best possible price when aspects such as quality, quantity, time, and location are compared. During the selection, it is important that each component of the product and each part of the EP assigned to an external service have a digital interconnection and possibly a digital model for extracting information and for allowing the possibility to create an AHT-EP digital twin.</p> <p>The <i>engineering</i> phase includes the design, development and test of the system/product, generating a prototype of the system/product and, with refinements, bugs corrections, updates, etc. the final version of the system/product (that will be deployed and commissioned). In case of creation of a digital twin of the engineering process, the engineering teams set up the simulation framework that will continuously support the simulation.</p>
4	DAC	Deployment & Commissioning	<p>The <i>deployment</i> phase consists in the installation/integration of the system/product in the final operative environment. The deployment includes also the preliminary verification and validation of the system/product that precede the commissioning. Once installed, a product identifier (e.g. a serial number) is associated to an owner/user id and stored in a database that will be accessed during the monitoring and simulation of the system.</p> <p>The <i>commissioning</i> phase is the process of assuring that the system/product is designed, installed, tested, operated,</p>

			and maintained according to the operational requirements of the owner or final client. A commissioning process may be applied not only to new projects but also to existing units and systems subject to expansion, renovation or revamping. The commissioning usually precedes the operations & management phase.
5	IFAT	Operations & management	<p>These phases consist in <i>operating and managing</i> the system/product according to the operational specification of the system/product and requirements of the owner or final client.</p> <p>In this phase, one of the stakeholders will monitor the data streams coming from the operating system and will be able to explore the status and future behaviors of the product, introducing real data in its digital thread. In case a digital twin is available, the simulated model will be used to evaluate deviations from the normal behavior, thus, the exploration continues without any impact on the real product, until a normal behavior is identified. With this approach, the real system continues to operate according to the operational specification. Moreover, in case the digital version of the product is available, operators will be able to predict warnings or errors, and planning in advance session of predictive maintenance.</p>
6	IKERLAN	Maintenance, Decommissioning & Recycling	<p><i>Maintenance</i> consists in identifying and establish requirements and tasks to be accomplished for achieving, restoring, and maintaining an operational capability for the life of the system/product. For a system/product to be sustained throughout its system life cycle, the maintenance process must be executed concurrently with the operations process. Maintenance addresses bug fixes and minor enhancements, as well as, minor adaptations to standard, new features, etc.. Significant changes in the system/product are considered in the evolution phase.</p> <p>The stakeholders will perform ordinary and predictive maintenance to achieve, restore, and maintain operational capability of the system.</p> <p>Maintenance intervention are reported in details, all the modifications done on the real product are applied also on the digital version in case available, in order to ensure the high fidelity of the digital twin of the product.</p> <p>In this phase, we also consider the decommissioning of the product at end-of-life and the recycling procedure required to reduce the impact on the environment.</p>
7	ABB	Evolution	<p>The <i>evolution</i> phase deals with the inability to predict how user requirements, market and technology trends will evolve a priori. The role of this phase is to monitor these aspects and identify potential significant changes in the</p>

			<p>future version of the system/products. The evolution phase must ensure also a continuous improvement of the system/product, always respecting the user requirements in an efficient, reliable and flexible way. Finally, evolution phase has to take into account various and alternating needs arising when dealing within different periods of lifetime starting from initial phase, following normal and final wear-out phases of the system and product, and send feedback towards engineering process other phases (e.g. requirements, product development, etc.).</p> <p>All the information collected in the "operation & management" and "maintenance" phases are analyzed to identify solutions to faults/bugs, define the necessary updates and identify improvements that could bring to new product releases. In case a digital twin is available, it can be used to simulate and explore the effects of these updates and new releases. This will ensure the continuous evolution of the product.</p>
8	Magillem	Training & Education	<p>This phase includes all the educational and professional training activities required by the engineering process, across the entire system/product life cycle. Source code documentation, how-to, installation manuals and training courses, together with demonstrators and development kits that use the power of the digital twin, will be provided to the stakeholders involved in the AHT-EP.</p>

The AHT-EP model supports also other phases linked to the product life cycle, such as Production, Marketing or Sales, that are not directly related to the EP but that can be represented as black boxes, connected and interacting with the AHT-EP. E.g. linking and including the production phase in the EP enables factory operations to be transformed into data-driven evidence-based practices, offering the capabilities of tracing product fault sources, analyzing production efficient bottlenecks and predicting future resource requirements.

Each of these phases will be contextualized for the service oriented architecture paradigms and described in more details in the following eight sections.

3.4.1. Requirements (EPP1) < SubTask 1 >

Requirements elicitation is the practice of researching and discovering the requirements of a system from users, customers, and other stakeholders. The output of this phase is typically a list of requirements.

In this phase the stakeholders cooperate to identify the requirements of the components (HW and SW) composing the IoT and cyber-physical ecosystem.

The word requirement refers to what is needed or desired. It might be something that is compulsory or a necessary condition. Elicitation of requirements is much more than just asking what a stakeholders their wants or needs. It is to draw out from several stakeholders what is needed and also what is possible in a concrete application. The interesting thing is that often, initial requirements might conflict with each other. Conflicting requirements engender

discussions between stakeholders to resolve the differences. However frustrating the experience might feel, it forms the solution that most likely was only a vague idea at the beginning and now evolves in a real structure.

In order to effectively define, design and create a device, process or SOA, clear design input is essential. Design input capturing is the practice of researching and discovering the requirements of a system from perspective of users, customers, and other stakeholders. The output of this phase is typically a list of requirements.

Design input shall address the intended use of the device, including the needs of the users and other people involved (e.g. patient in case of medical equipment). Typically, several iterations are required to reach a complete set, but in the end the design inputs shall be complete, unambiguous, and not in conflict with each other. Design input is the basis of design verification and design validation activities and shall consist of three main categories:

- User Needs
- Hazard analysis (Risk management assessment)
- Product Requirements

3.4.1.1. Phase Description

The requirement phase is naturally the first phase of the life cycle of an engineered artefact. It defines what the artefact will be, and can be updated during the life of the artefact. The output of this phase is typically a list of requirements. Alternatively, there could be a list of requirements from each of the stakeholders. Requirements come from 'needs', which are described in terms of goals the user or stakeholder (who therefore also has to be identified) wants to achieve. The resulting requirements are described in terms of properties. Needs are validated, typically by executing use cases, and requirements are verified, typically by measurements.

The design input procedure defines the steps for generating each of the three above categories. Note that while the final structure of the design input documents has a clear hierarchy, it does not imply any particular time sequence for the development of these design input documents.

In case of new systems, the process starts with the user needs and requirements. In case of extending and improving already existing systems (e.g. service-oriented architectures (SOAs), also maintenance related aspects need to be taken into account, such as requirements and tasks to be accomplished for achieving, restoring, and maintaining an operational capability for the life of the system/product.

- **Collect User Needs:** User needs are established in terms of goals that need to be achieved within the context of the user. The needs are achieved by completing specific user scenarios in relation to the user context. The following input sources shall be taken into account as appropriate:
 - Predicate product data (including residual anomalies)
 - Product strategy (e.g. marketing requirements, configuration, accessories)
 - Post market data
 - Intended use environment
 - Human factors & usability engineering (e.g. formative studies)
 - Regulations & standards

- Translate inputs into Product & Subsystem Design Requirements:** The user needs, user context, and risk mitigations are the input for the product requirements and the related product technical design. Decomposing the product design to subsystem level results in the corresponding subsystem requirements and subsystem technical design. Also the requirements for the existing SOA needs to be taken into account, if the engineering process using this are starting point. The product requirements shall define:
 - Specifications that specify the capability of the product and process for its manufacture, deployment and services
 - What a product will do. The product technical design will explain how.
 It is important to phrase the requirements as clearly and unambiguously as possible, so S.M.A.R.T.: Specific, Measurable, Achievable, Relevant, and Time-bound.
- Complete Risk Management Activities:** The user needs and the requirements at the different levels are input for the Risk Management activities. When risks of personal harm must be reduced to an acceptable level by means of a risk mitigation, the outputs of Risk Management result in new requirements.

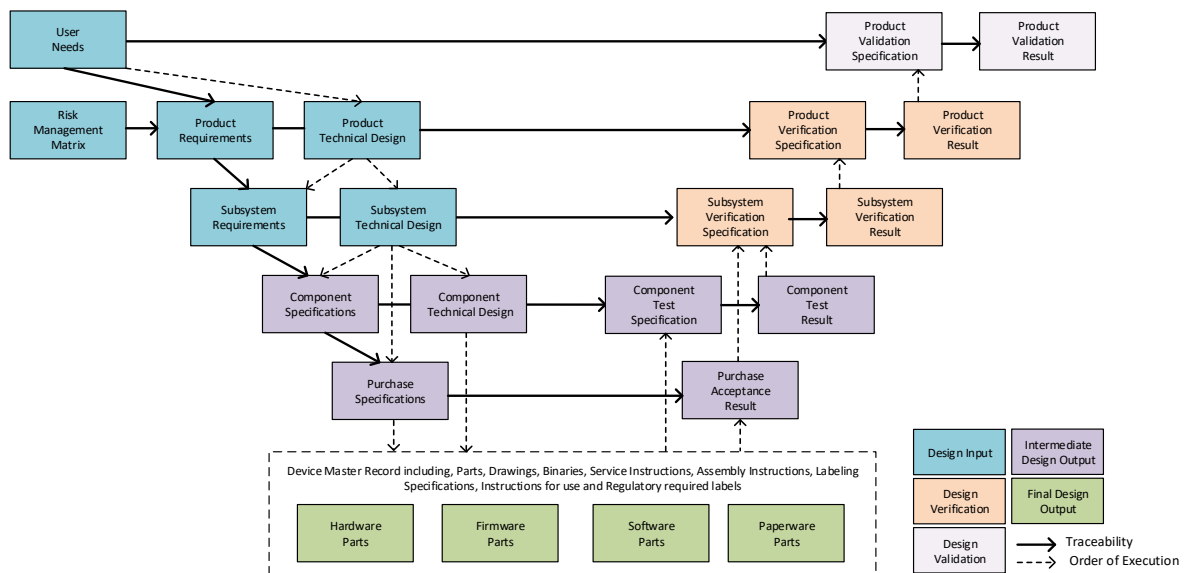


Figure 8 The V-model

The V-model [32], shown in Figure 8, of the Systems Engineering process with various design and relations to verification and validation specifications.

Although the V-model suggests a sequential order of defining the various levels of requirements, practice is much more dynamic and typically requires various iterations to reach mature status. Therefore the requirement phase remains an active part during the whole lifecycle of the engineered artefacts, since it needs to be validated at all times and can be updated. The SOA paradigm supports these processes.

One problem with initial requirements conflicts is the time the feedback loop takes. Another source of difficulty is the update rate of requirement changes that is when a new or updated requirement might take effect. In either case, time delays are the common factor.

A further predicament is meaning of requirements. Stakeholders might not use the same meaning when describing requirements. This might lead to a need to capture knowledge such

that a set of concepts and categories in a subject area or domain that shows their properties and the relations between them. In other words, defined semantics and ontologies are a necessary part of the requirements phase. Also recycling of existing SOA leads to requirements, to be considered in this engineering phase.

3.4.1.2. Advantages of using the Eclipse Arrowhead Framework in this AHT-EP Phase

The requirements for use cases to evaluate the Arrowhead Framework should also explicitly take the six Arrowhead Tools objectives into account. They are, as described in the overall Arrowhead documentation:

1. Reduction of solution engineering costs by 20-50%.
2. Interoperability for IoT and SoS engineering tools.
3. Interoperability and integration of data from legacy automation engineering tools to the Arrowhead framework integration platform.
4. Integration platform interoperability with emerging digitalization and automation framework.
5. Flexible, interoperable and manageable security for digitalization and automation solutions.
6. Training material (HW and SW) for professional engineers.

The process model, the digitalization framework and the tool chain architecture have to be aligned with these six objectives.

In addition, the 4 WP2 objectives should be reflected in the requirements, if applicable for the use case:

1. The change from design time to run time engineering.
2. The move from single to integrated multi stakeholder automation and digitalization.
3. Handling of substantially increased number of I/O's due to much more fine grained automation.
4. Digital learning and training activities as an integral part of the engineering cycle.

The Arrowhead Tools project considers using a service oriented architecture paradigm to achieve its six objectives, listed above. How this could be achieved between phases will be defined in WP4 during the project. Here, we can think solutions based on Eclipse Arrowhead framework. Each stakeholder gets its own database to handle the list of requirements. The databases are assets with an Eclipse Arrowhead framework compliant administrative shell that offer requirements as services. Another asset can consume requirement services to seek for conflicting requirements, or prepare a complete requirements list for the engineering and procurement phase.

The Evolution phase of the previous SOA version can also provide valuable input for the Requirements phase of the updated SOA. This may well lead to a maturity step of the SOA, e.g. to support the transition from design-time to run-time life-cycle management paradigm.

3.4.1.3. Use Case tasks and activities associated to the phase

In fact, all use cases should start with a Requirements phase. For any engineering process it is mandatory to clearly capture the requirements for the solution that is to be provided. Clear understanding of the User Needs guides the organization to make the right solution (to be tested in the validation phase, at the end), and clear definition of technical requirements makes it possible to create the solution right (to be tested in the verification phase), including all relevant safety mitigations.

A standardized procedure for EP requirements also allows effective cooperation between different companies. E.g. The UC-06 (Production preparation tool chain integration) will connect the EPs of the three parties involved in the UC, matching the objective #2 of WP2 concerning the move from single to integrated multi stakeholder automation and digitalization. The three companies can, within the Arrowhead Tools project, collaborate to streamline the process from architectural drawing, via a 3D configurator to created machine files. By utilizing the Eclipse Arrowhead framework, they can implement and verify a more automated yet secure way of transferring data in the information flow.

The following section illustrates the requirements phase by going through a couple of the use cases in the Arrowhead Tools project.

Use Case UC-02

The use case, defined to evaluate the applicability of the Arrowhead Tool Framework to improve engineering efficiency, is the design of complex radio frequent (RF) transmit coils for ultra-high field MRI scanners. In order to create the RF fields in the human body, required for high quality imaging, parallel transmit (pTX) coils are needed: arrays of antennas, each connected to a separate RF amplifier with accurate amplitude and phase control of the transmit signals. The MRI pTX Coil Optimizer, based on the AHT framework, allows seamless data transfer between the various stakeholders (see Figure 10), to optimize the process of designing and engineering of the coil.

In this process the following stakeholders are distinguished:

- A. RF Coil engineers (StkH1) design application specific coils, using their knowledge and experience of analogue RF electronics and antenna design.
- B. RF Coil simulation engineers (StkH2) use models of coils and human

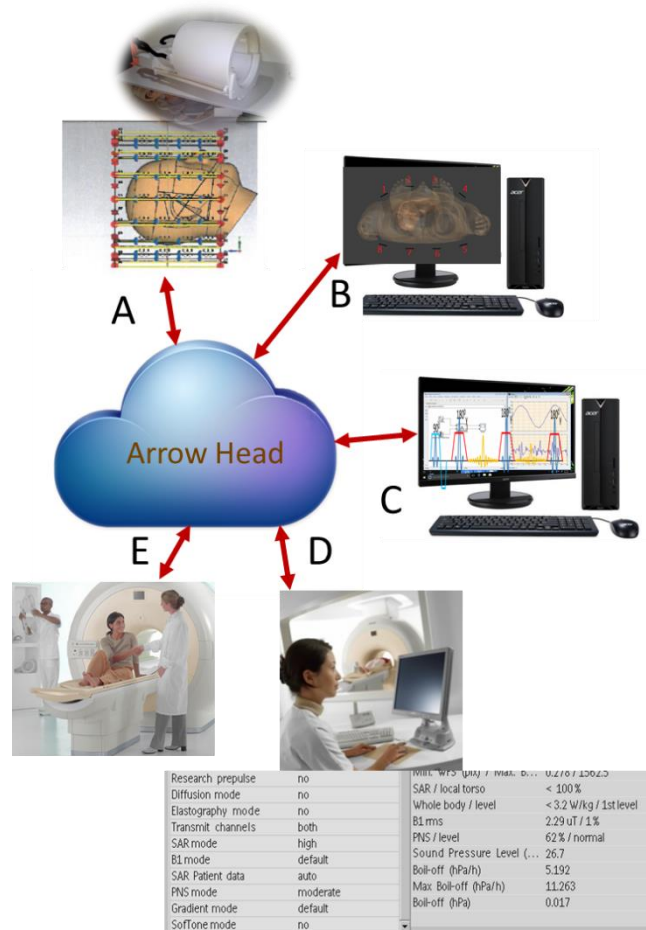


Figure 9 UC-02 Stakeholders

- bodies to determine the relevant coil properties (B1+ field, RF heating), and share the results through a service architecture network (Arrow Head). The RF coil simulation engineers could be employed by the RF coil supplier or by Philips, the MR system manufacturer.
- C. MR Methods engineers (StkH3) create MR scan technique models that optimize performance, based on the selected coil and application, in combination with the simulated RF performance
 - D. MR Application engineers (StkH4) optimize scan protocols and ExamCards, based on the combined knowledge of RF coils and MR scan techniques, to reach optimal Image Quality and in the shortest possible scan time.
 - E. The Clinical MR users (StkH5) use the MRI system routinely for clinical examinations, together with the provided pTX coils.

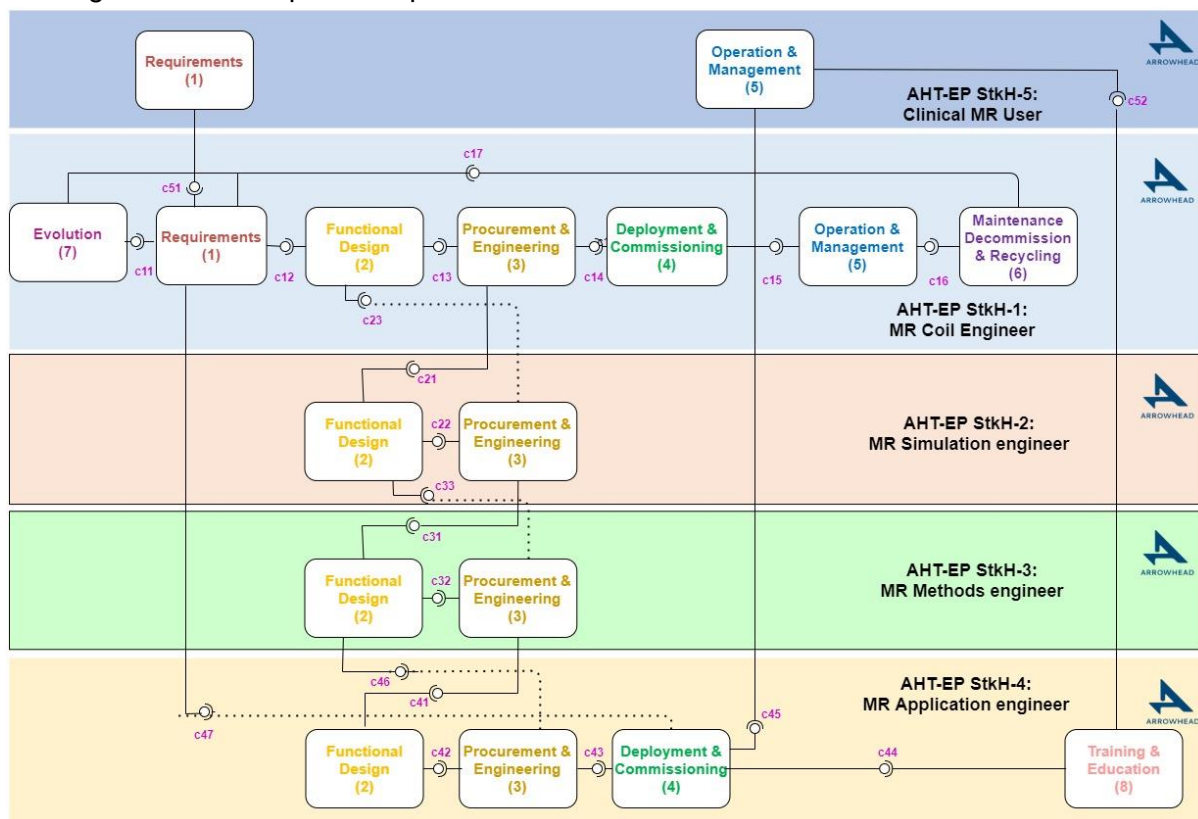


Figure 10 AHT-EP of use case UC-02

The engineering process graph, shown in Figure 10, provides a schematic overview of the relation of the engineering phases of the various stakeholders. Although there may be initial ideas and developments of pre-developed solutions to explore the ideas, the actual engineering process starts with clear definition of the requirements:

- 1) The User Needs are formulated by the Clinical MR User (StkH-5). This defines:
 - a) The clinical needs (e.g. imaging of the heart, liver and prostate in a 7T whole body MRI scanner, for adults between 40 and 150 kg)
 - b) The required image quality (e.g. image uniformity of +/- 10%)
 - c) The required scan techniques (e.g. T1W and T2W-TSE and mDixon-Quant to allow fat quantification)
 - d) Ease of use (incl. weight, size, connections, easy cleaning)

- e) Looks and feel (colors, tactility)
- 2) The product and system requirements for the MR coil, to be development by the MR Coil engineer (StkH-1), are based on the user needs, and are the input for the MR Coil designer, who uses these to create the functional design.
If the functional design is sufficiently clear, also a risk analysis shall be performed. This may result in additional requirements for risk control measures, to prevent human safety related incidents, due to malfunction or foreseeable misuse of the product.
If similar coils have been made before, maintenance data and input from clinical or technical evaluations may be used as input for the requirements.
To give an impression, requirements for the coil may be:
 - a) Size and flexibility of the coil, to cover the required anatomy for the required range of patient sizes
 - b) RF field requirements
 - c) Weight of the coil
 - d) Material choice
 - e) Malfunction detection provisions
 - f) Temperature limitation (since the coil can be touched by the patient)
 - g) etc.
- 3) If the detailed design of the coil is sufficiently clear during the engineering phase, the technical details can be translated into requirements for the functional design for StkH-2, the MR simulation engineer. Together with the requirement based on the user needs, this is used to perform the required simulations.
- 4) The output of these simulations, together with the requirements derived from the user needs, are used by the MR Methods engineer (StkH-3) to optimize the scan techniques for this specific coil.
- 5) The MR application engineer (StkH-4) mainly uses the user needs as input requirements to create the protocols for this clinical scans, based on the output of the MR Methods engineer.

Use Case UC-06

To illustrate some of the Requirement phase concepts, we make use of *Use Case 06 (Production preparation tool chain integration)* where the goal is mass customization of houses. That is the production of individually designed homes assembled along a factory line. There are lots of stakeholders (see Figure 11), each with their list of requirements. Considering only two: the national building standards and the customer, we can easily find a conflict between their requirements. The customer will have to give in to the national building standards but the SOA paradigm allows the feedback loop to be much faster. The paradigm permits all stakeholders, some of which are in other engineering phases, to continually interact with their own requirements throughout the life cycle of the engineered artefact. Referring to Figure 12, tools within EPP1, EPP3, EPP7, and EPP8 interacting with tools in EPP2.

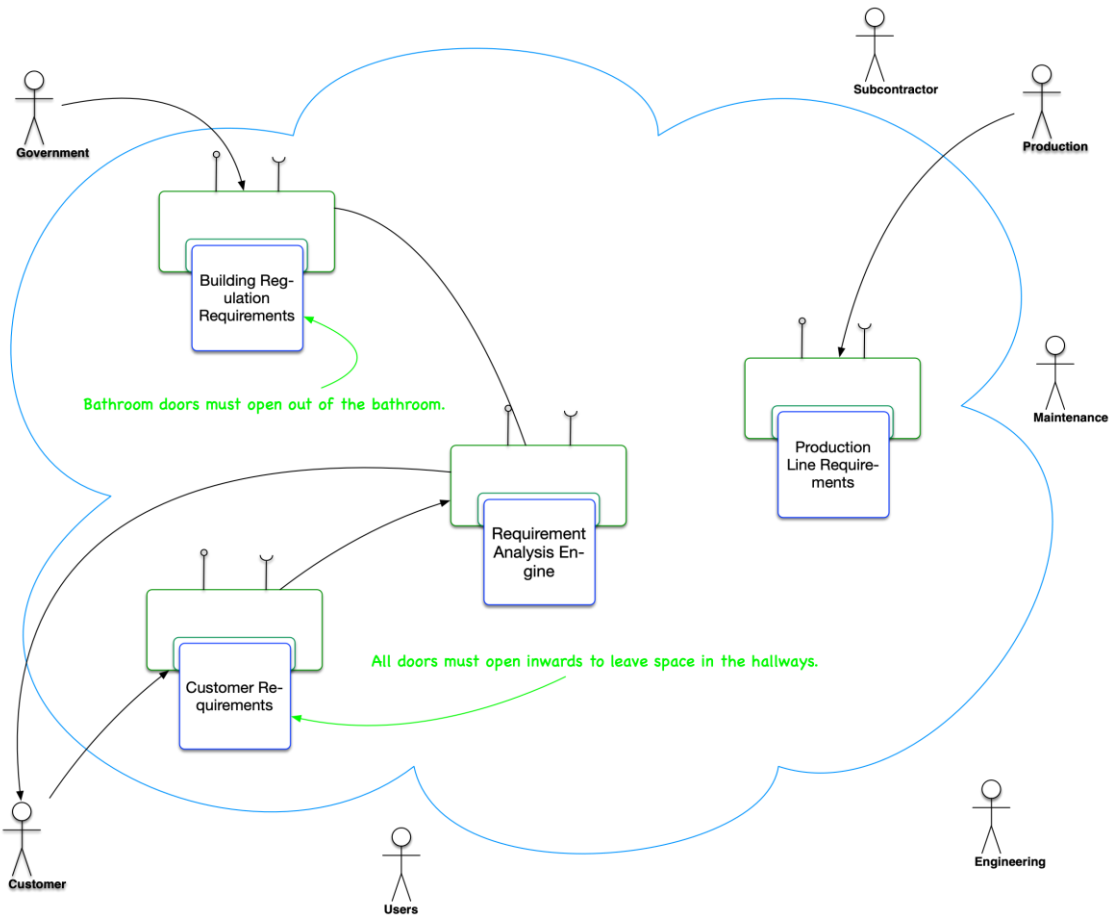


Figure 11 Requirement analysis engine

One could question the service oriented analysis tool that must examine the meaning of the requirements. We see some emergence of that in Task 4.2 with the work on semantic translation within services. The adoption of the SOA paradigm reduces the time necessary to handle requirements. Returning to UC-06, the implementation of the digitalization of engineer processes is currently estimated to move their base from over 1400 minutes per building modules down to 30 minutes.

With so many stakeholders being able to interact with the requirements, one can see that the proposed concept spans the RAMI 4.0 solution space with covering the life cycle on one axis, the business aspect on a second axis and the production on the third one from enterprise resource planning to shop floor requirements. In a later section, the herewith deliverable explains how this specific phase case expands to all phases and use cases.

Requirements do not form a wish list. They shall be fulfilled and validated. The tools developed in the Arrowhead Tools project will have to ensure this validation process with the SOA paradigm in a secure and interoperable fashion. Needs and requirements, and their counterparts validation and verification, can be described according the V-model [33]. Requirements within the requirement set have a level, e.g., system, subsystem, component. At one level, they feed the design process, which in turn feeds the lower level requirements through processes like budgeting and allocation. Requirements therefore typically are described in a hierarchical structure.

In addition, this structure allows tracing of requirements to higher levels and even needs. This

traceability supports impact assessment of a modified requirement, such as when a conflict is detected, a design cannot be made or verification has failed. The developed tools will have to support that. These tools might look like the requirement analysis engine in Figure 11, but in this early stage, that is only concept conjecture.

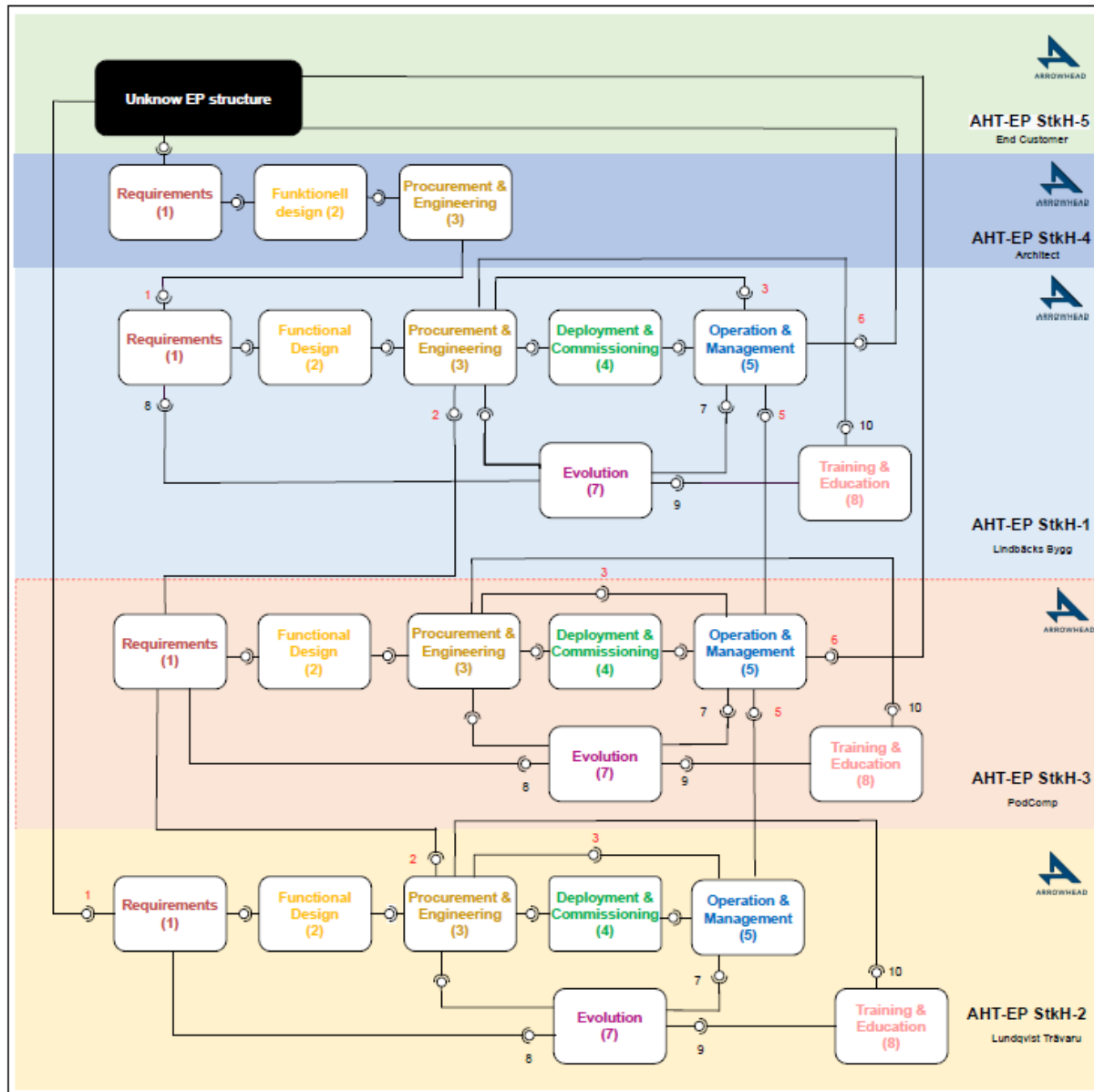


Figure 12 AHT-EP of use case UC-06

3.4.2. Functional design (EPP2) < SubTask 2 >

The functional design phase consists in adopting the "functional design" paradigm to simplify the design of the system/product. A functional design assures that each modular part of the system/product has only one responsibility and performs that responsibility with the minimum of side effects on other parts. Functionally designed modules tend to have low coupling. The output of this phase is typically a model, or an architecture.

In this phase, stakeholders develop the cyber-physical models of the components and their functionalities to be subsequently simulated and validated. During the life cycle of the product these models will be continuously used to represent and simulate behaviors of the components in the digital thread eventually supporting the digital twin of the engineering process phases.

3.4.2.1. Phase Description

Functional analysis and design are key activities in the Systems and Software Engineering process [34] [35] to explore new concepts and define new architectures. The mapping between requirements and functional architectural blocks looks for establishing a set of relationships that are relevant for the new product and/or service and can help to provide a better understanding of the system. In general, the design of a complex system can be divided into three main phases [36]:

- Conceptual modelling
- Architectural modelling
- Detailed design

More specifically, functional analysis is mainly relevant to the first stages of development where many solutions are still feasible. From the first initial set of mission statements or objectives and taking as an input the system requirements specification, a functional analysis is done by creating a functional tree (a kind of functional breakdown structure) [35] or a product tree that serves engineers to have a first distribution of the **system architecture**³. Then the major responsibilities, top-level functions, of the system can be grouped together to determine the functional blocks and dependencies among them. To do so, techniques such as a traceability matrix are used to document the mapping between requirements and architectural blocks. Afterwards, a complete description of the architectural model can be done using as a reference the 4+1 view architectural model [37] and covering both static and dynamic aspects of the system.

The documentation of the functional analysis and design can be done using different diagramming techniques:

- *Functional architecture*: used to provide a top-down definition of system functions (e.g. FBS-Functional Breakdown Structure).
- *Functional flow block diagrams*: used to represent the interactions between components.

³ The purpose of system architecture [15] activities is to define a comprehensive solution based on principles, concepts, and properties logically related to and consistent with each other. The solution architecture has features, properties, and characteristics which satisfy, as far as possible, the problem or opportunity expressed by a set of system requirements (traceable to mission/business and stakeholder requirements) and life cycle concepts (e.g., operational, support) and which are implementable through technologies (e.g., mechanics, electronics, hydraulics, software, services, procedures, human activity).

System Architecture is abstract, conceptualization-oriented, global, and focused to achieve the mission and life cycle concepts of the system. It also focuses on high-level structure in systems and system elements. It addresses the architectural principles, concepts, properties, and characteristics of the system-of-interest. It may also be applied to more than one system, in some cases forming the common structure, pattern, and set of requirements for classes or families of similar or related systems.

- *N-squared diagrams*: used to develop data, function or hardware interfaces.
- *Time-based diagrams*: used to represent time-based interactions between components.

Once the context of functional designed is established, it is important to emphasize the methodologies that can help to deal with this activity. In this frame, recent times have also seen the emergence of Model-based Systems Engineering (MBSE) [38] [39] as a complete methodology to address the challenge of unifying the techniques, methods and tools to support the whole specification process of a system including conceptual design, system requirements, design, analysis, verification or validation.

In the context of the well-known V-model, it means that there is "formalized application of modelling" to support the left-hand side of this system life cycle implying that any process, task or activity will generate different system artefacts but all of them represented as a model. This approach is considered a cornerstone for the improvement of the current practice in the Software and Systems Engineering discipline since it is expected to cover multiple modelling domains, to provide better results in terms of quality and productivity, lower risks and, in general, to support the concept of continuous and collaborative engineering easing the interaction and communication between people (engineers, project managers, quality managers, etc.).

Although MBSE [40] represents a shifting paradigm for the development of safety critical systems, the plethora of engineering methods supported by different tools implies the need of not only easing the communication between people but tools. How could we do requirements management, simulation, diagramming, documenting, and information retrieval or project management without the corresponding tools or IT systems?

The more complex the problems are, the more complex computer tools must be delivered, and the main reason for that is, consequently, because those computer tools are demanded to be "smarter". Up to now, a computer tool is not human independent; it simply "acts" as smart according to its access to relevant data, information and knowledge. In order to enable a collaborative MBSE through IT systems, it is completely necessary to provide the proper implementation of a non-functional requirement to access existing system artefacts (where knowledge is somehow embedded): interoperability. To do so, different initiatives, frameworks, services and languages such as the ISO 10303 (STEP), the SysML [41] or UML languages or the OASIS OSLC (Open Services for Lifecycle Collaboration) initiative can be found. For instance, it is possible to find an OSLC-MBSE working group at OMG. Thus, while MBSE represents an ideal approach to develop complex systems, OSLC can be seen as a key enabler to equip engineering tools with the ability of exchanging data and information under common data and communication protocols.

Since MBSE is focusing on the formalized application of models to cover the whole engineering life cycle, it makes sense to describe the two main types of models that can be found according to the Systems Engineering Body of Knowledge (SEBoK) [42]:

- Descriptive models. *"A descriptive model describes logical relationships, such as the system's whole-part relationship that defines its parts tree, the interconnection between its parts, the functions that its components perform, or the test cases that are used to verify the system requirements. Typical descriptive models may include those that*

describe the functional or physical architecture of a system, or the three dimensional geometric representation of a system."

- Analytical models. In the same manner, "*an analytical model describes mathematical relationships, such as differential equations that support quantifiable analysis about the system parameters. Analytical models can be further classified into dynamic and static models. Dynamic models describe the time-varying state of a system, whereas static models perform computations that do not represent the time-varying state of a system.*"

In summary, the functional analysis and design are key activities for the Software and Systems Engineering process. Furthermore, several methodologies, such as MBSE with SysML and other formal languages, can be used to support the specification process of a complex product and/or service at different levels and views through the creation and transformation of models. Finally, technological support for these methodologies is offered through tools with specific capabilities (e.g. requirements authoring, quality checking, descriptive and analytical modelling, etc.) that are usually exposed as native or standardized APIs (e.g. ReqIF, ISO-STEP, SysML, FMU/FMI, OSLC, etc.) with different access formats (e.g. ReqIF, SysML, RDF, FMU/FMI) and communication protocols (e.g. file, OSLC Services, HTTP Services, etc.). However and due to the necessity of keeping consistency over-time during the engineering life cycle, it is necessary to provide means for:

- Interoperability and connection among tools that can help to build a collaborative engineering environment with capabilities for automatic population (transformation) of models
- Traceability between different types of artefacts
- Integration of models at different description levels
- Execution of analytical models
- Generation of documentation
- Quality checking (consistency)
- Reuse of system elements
- Etc.

3.4.2.2. **Advantages of using the Eclipse Arrowhead framework in this AHT-EP Phase**

In the context of engineering process for this phase, the application of the Eclipse Arrowhead framework helps to decrease the effort and cost of the communication and syntax level (service-oriented architecture implemented through a bus integration pattern), automating the exchange of system artifacts in a safe environment and enabling the consumption of operations that require the collaboration of several tools, but the interpretation of the different meta-models must be implemented in any case. Additionally, the Eclipse Arrowhead framework also provides other non-functional aspects such as security, service discovery, scalability, etc. that are un-valuable assets to define a toolchain supporting the activity of this phase in the industrial use cases.

3.4.2.3. **Use Case tasks and activities associated to the phase**

Use Case UC-03

Integration of electronic design automation tools with product lifecycle

UC-3 aims at providing means for improving the reuse of physical models covering the abstraction, selection, representation and customization of system artifacts for the whole development lifecycle. The reuse of any system artifact goes beyond the mere discovery of a potential reuse and it must focus on evaluating what and how a system artifact can be reused (requirements, analytical models, descriptive models, test cases, etc.). To do so, quality also plays a role since it is assumed that high-quality system artifacts may help to improve the reusability factor of a system artifact. Furthermore, in this use case, there is another major objective focusing on the improvement of traceability to be able to automatically keep traces [43] from the very early stage of development to the final release of a complex product.

In this context, the specific engineering process covers different technical engineering processes and engineering methods (supported by different techniques and tools) creating the next toolchain (see the following sub-sections and Table 3).

In general, there are three tools providers: IBM, The Reuse Company and Altium. Most of them provide standardized ways of accessing (files and services) and consuming work products data and operations. However, the interpretation of standards (such as SysML or ReqIF) may differ from one tool to another and, in most of cases, the tools also manage more relevant information that is not exposed being critical for processes such as traceability or quality management.

Currently, the integration between tools is done point to point creating specific connectors to consume (read/write) the information and functionalities from a tool provider.

Table 3 Summary of engineering processes, methods, techniques and tools for UC-03

Tool	Engineering process	System artifact	Data model	Access/ Communication model
IBM Doors	Requirements	Requirement	Native ReqIF OSLC RM	Native API Native database File
RAT	Requirements	Requirement	Native OSLC RM (Requirements Management) ReqIF	Native API WSDL-based Service OSLC/Rest Service File
IBM Rhapsody	Functional design	Logical models	Native SysML/UML	Native API Native database File
Altium designer	Functional design	Hardware model	Native	Native API WSDL-based Service
Verification Studio	Procurement & engineering, Deployment & Commissioning	Quality metrics	Native OSLC KM	Native API WSDL-based Service OSLC/Rest Service File
Traceability Studio	Procurement & engineering, Training & education	Trace	Native OSLC KM	Native API WSDL-based Service OSLC/Rest Service File
KnowledgeManager	Not available	Ontology, vocabulary, etc.	Native OSLC KM SKOS OWL	Native API WSDL-based Service OSLC/Rest Service File

In order to deliver the two major objectives of this use case (traceability management and reuse), the following integrations must be done (see Table 4) where x: a connector or integration already exists but it is not based in standards in both senses (communication and data model) and R: a new standard-based connector is required to properly implement the use case.

Table 4 Integration for achieve the traceability management and reuse objectives

Source/ Target tool	IBM Doors	IBM Rhapsody	RAT	VS	KM	Altium designer
IBM Doors		x	x	x	R	R
IBM Rhapsody	x		x, R			R
RAT	x	x, R		x	x	
Verification Studio	x	x, R	x		x	R
Traceability Studio	x	R	x	x	x	R
Knowledge Manager	R	R	x	x		R
Altium designer	R	R	R	R	R	

The first step is to establish the architecture and data exchange needs relies on establishing the communication protocols, formats and data models to be used by each tool.

- IBM Doors: file, XML, ReqIF metamodel.
- IBM Rhapsody: file, XML, SysML v1 metamodel.
- RAT: HTTP, JSON, SRL (System Representation Language) metamodel [44] [43].
- VS: HTTP, JSON, SRL metamodel.
- KM: HTTP, JSON, SRL metamodel.
- Altium designer: file, XML, native metamodel.

From a conceptual perspective, the Arrowhead framework provides a complete platform of cross-cutting aspects like security or interoperability [45] to orchestrate services for different purposes under a bus integration pattern e.g. IoT applications [46] [47] comprising sensors and software services.

In this use case, a service-oriented architecture implemented with the Eclipse Arrowhead framework (Figure 13) is used to expose the different tools vendors as services that offer different resources (system artifacts) and operations under a unified communication bus.

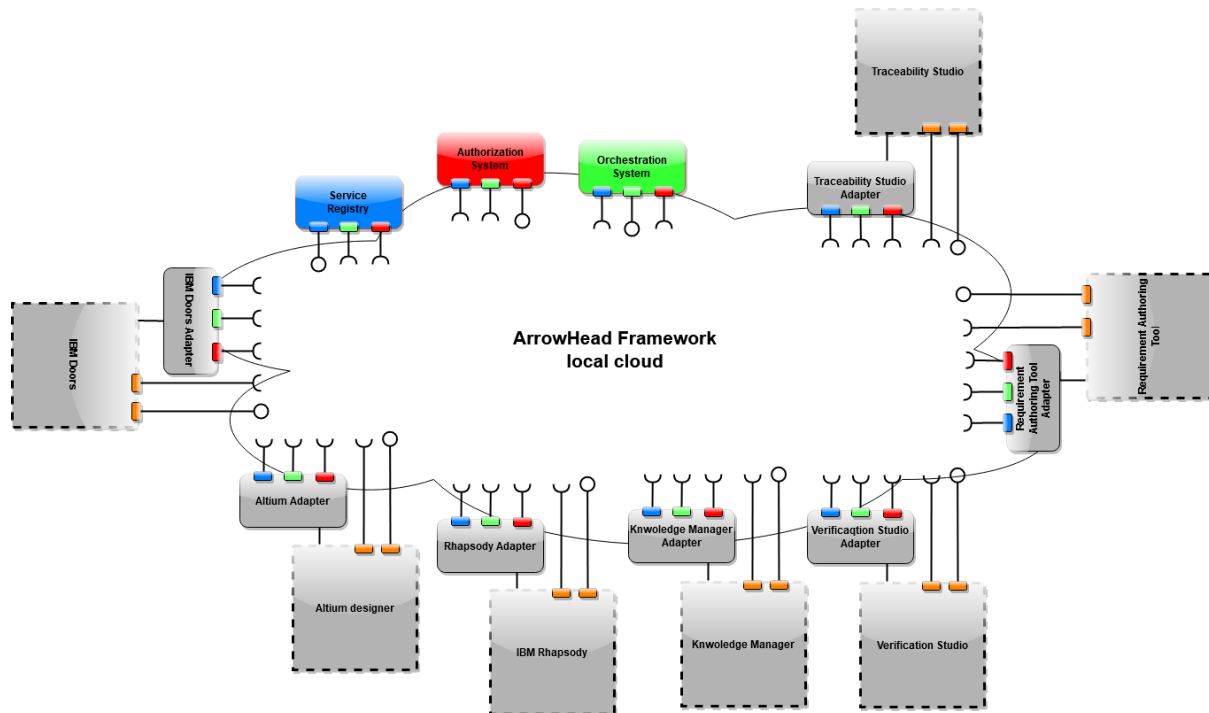


Figure 13 UC-03 - Toolchain architecture

Use Case UC-04

Interoperability between (modelling) tools for cost effective lithography process integration.

ASML is the world's leading provider of complex lithography systems for the semiconductor industry. The design process of these Cyber-Physical Systems of Systems involves multi-disciplinary engineering teams focusing on functional specification and verification of scenarios and mono-disciplinary engineering teams focusing on the realization of these scenarios in a platform composed of mechanical, optical, electrical and software components. These engineering disciplines each use a specific set of engineering methods, tools and technologies that are loosely coupled both on a syntactic and on a semantic level. This has a major impact on engineering efficiency. It hampers verification early in the development process, especially concerning system-wide (performance) aspects (e.g. throughput and accuracy). In addition, it hinders system evolvability and deployment, i.e. introducing new scenarios or adapting existing ones.

To significantly improve engineering efficiency, the goal of this use case is to establish seamless syntactic and semantic interoperability between the multi-disciplinary modelling tools enabling rapid development and deployment of (new) machine calibration, performance and diagnostics test scenarios and effective prediction and trading-off of key system aspects concerning performance and correctness.

The Scenario modelling, Synthesis, Verification and Product line (SSVP) engineering toolchain studied within UC-04 focusses on the design, optimization, validation and maintainability of cyber-physical, software as a service, composed (System of Systems) systems. Development of these systems relies heavily on domain specific tooling which when interconnected provide a more flexible and efficient workflow. A commonality of the various tools employed is their model-based engineering approach. Although from different domains, model elements often

depict the same components in functionality. During design, optimization, verification, and in a context of product lines, realizations made in one domain should easily propagate to the other domains. Commonalities and variabilities of system artefacts and rules for potential combinations of variabilities will be represented in a product line using product line engineering. The Eclipse Arrowhead framework is employed to allow model domain transformations to be made on the fly and during development, and serve as an interconnector between the tools, of both domain model and domain specialist.

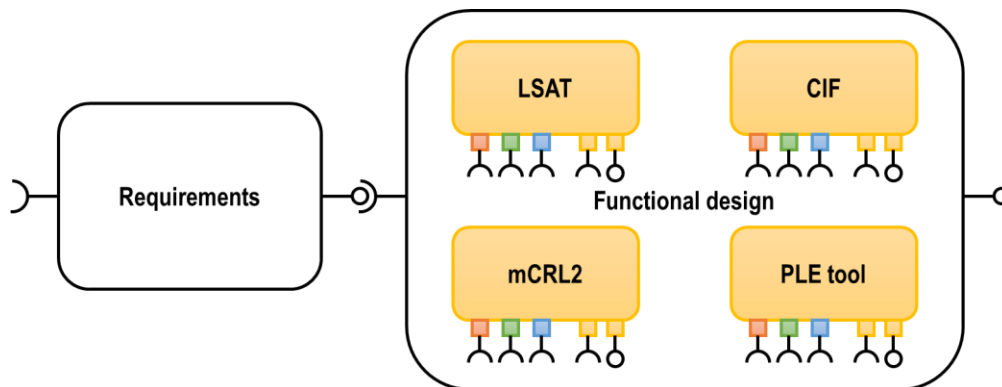


Figure 14 UC-04 Tool chain architecture

In the UC-04 conversion methods have been developed to exchange data between the tools CIF, LSAT, mCRL2, and (to be developed or chosen) a PLE tool which are employed in the *Functional design* phase Figure 14. Currently this is done through file conversions, with some automatic conversions and some manual, and copying files. By hooking up the tools to the Eclipse Arrowhead framework, steps of this process can be automated.

- **CIF service:** General description of the Tool: CIF (Compositional Interchange Format for hybrid systems, developed by TUE) is an automata-based modeling language for the specification of discrete event, timed, and hybrid systems. The CIF tooling supports the entire development process of controllers, including, amongst others specification, supervisory controller synthesis, simulation-based validation and visualization, verification, real-time testing, and code generation.
- **mCRL2 service:** General description of the Tool: mCRL2 (developed by TUE in collaboration with University of Twente) is a formal specification language with an associated toolset. The toolset can be used for modelling, validation and verification of concurrent systems and protocols. The toolset supports a collection of tools for linearization, simulation, state-space exploration and generation, and tools to optimize and analyze specifications. Moreover, state spaces can be manipulated, visualized and analyzed.
- **LSAT service:** General description of the Tool: LSAT (developed by ASML, TNO-ESI and TUE) provides a formal modeling approach for compositional specification of both functionality and timing of manufacturing systems. The performance of the controller can be analyzed and optimized by taking into account the timing characteristics. Since formal semantics are given in terms of a (max, +) state space, various existing performance analysis techniques can be applied.
- **PLETool service:** General description of the Tool: PLETool is a tool for product line variability management. PLETool provides the infrastructure for managing the

variability in a product line. The tool incorporates different components to support variability modeling, analysis and configuration of the variability model, and mechanisms for product derivation.

- **SDF3 service:** General description of the Tool: SDF3 (developed by TUE) is a tool for analysis and synthesis of Synchronous DataFlow Graphs (SDFGs). It includes an extensive library of SDFG analysis and transformation algorithms as well as functionality to visualize them. It also includes analysis algorithms for switching (max,+) models. The tool can also create SDFG benchmarks that mimic DSP or multimedia applications.
- **Model translation service:** General description of the Tool: An Arrowhead compliant service used to translate and convert models between tools in the tool chain of the UC. A new service is made for each model translation.

3.4.3. Procurement & Engineering (EPP3) < SubTask 3 >

The procurement is the process of finding and agreeing to terms, and acquiring goods, services, or works from an external source required to engineer the system/product, construct, and manufacture it. Procurement is used to ensure the buyer receives goods, services, or works at the best possible price when aspects such as quality, quantity, time, and location are compared. During the selection, it is important that each component of the product and each part of the EP assigned to an external service have a digital interconnection and possibly a digital model for extracting information and for allowing the possibility to create an AHT-EP digital twin.

The engineering phase includes the design, development and test of the system/product, generating a prototype of the system/product and, with refinements, bugs corrections, updates, etc. the final version of the system/product (that will be deployed and commissioned). In case of creation of a digital twin of the engineering process, the engineering teams set up the simulation framework that will continuously support the simulation.

3.4.3.1. Phase Description

According to the INCOSE Systems Engineering Handbook [34], there are several processes in which procurement plays a key role:

- In the Integration Process. The acquisition enablers can be done “through different ways such as rental, procurement, development, reuse or subcontracting”. An enabler is a complete system different from the System of Interest.
- In the Verification process, to ensure that all necessary enabling systems for the verification actions are available, procurement will have a relevant role.
- In the Transition process, to ensure that all necessary enabling systems are available. More specifically, it is necessary to identify all requirements and interfaces for the enablers being procurement a method to provide such dependencies.
- In the Operation process, to ensure that the system can enter in a production mode, all enabling systems must be ready using as methods to acquire them the ones presented in the first bullet (including procurement).

- In the Maintenance process, to support trades required for “maintenance and to ensure affordability, feasibility, supportability and sustainability of the system maintenance”.
- In the Disposal process, to again provide the enabling systems to retire the system of interest.

On the other hand, in the context of technical management process, procurement is a key activity in the definition of top-level work packages and tasks. It is also important to remark that for high-risk (time and cost) technical tasks early procurement can help to mitigate risks through a strategy for provisioning in parallel to developments.

As a final remark, Quality Assurance policies may apply and affect procurement (mainly of raw materials) to support quality goals and Logistics engineering will also include procurement as an activity to acquire goods and/or services.

In the context of acquisition process, there are also specific remarks for ground and construction systems. Depending on the country, public procurement may also subject to regulations to boost the notion of Green Public Procurement (GPP).

Other references to situate the procurement and engineering of complex systems is the “NASA systems engineering handbook” [35] where procurement is mainly referred to in the acquisition process.

The engineering phase, described in [34] and [35] as well, (see Figure 15 as an overview of technical engineering processes in the context of the ISO 15288) includes the design, development and test of the system/product, generating a prototype of the system/product and, after some iterations the final version of system/product (that will be deployed and commissioned). In the context of the AHT Engineering Process, *Procurement & Engineering* may refer to the implementation technical engineering process including cross-cutting activities such as V&V. However, depending on the life cycle model and organization or project specific restrictions, this process may change.

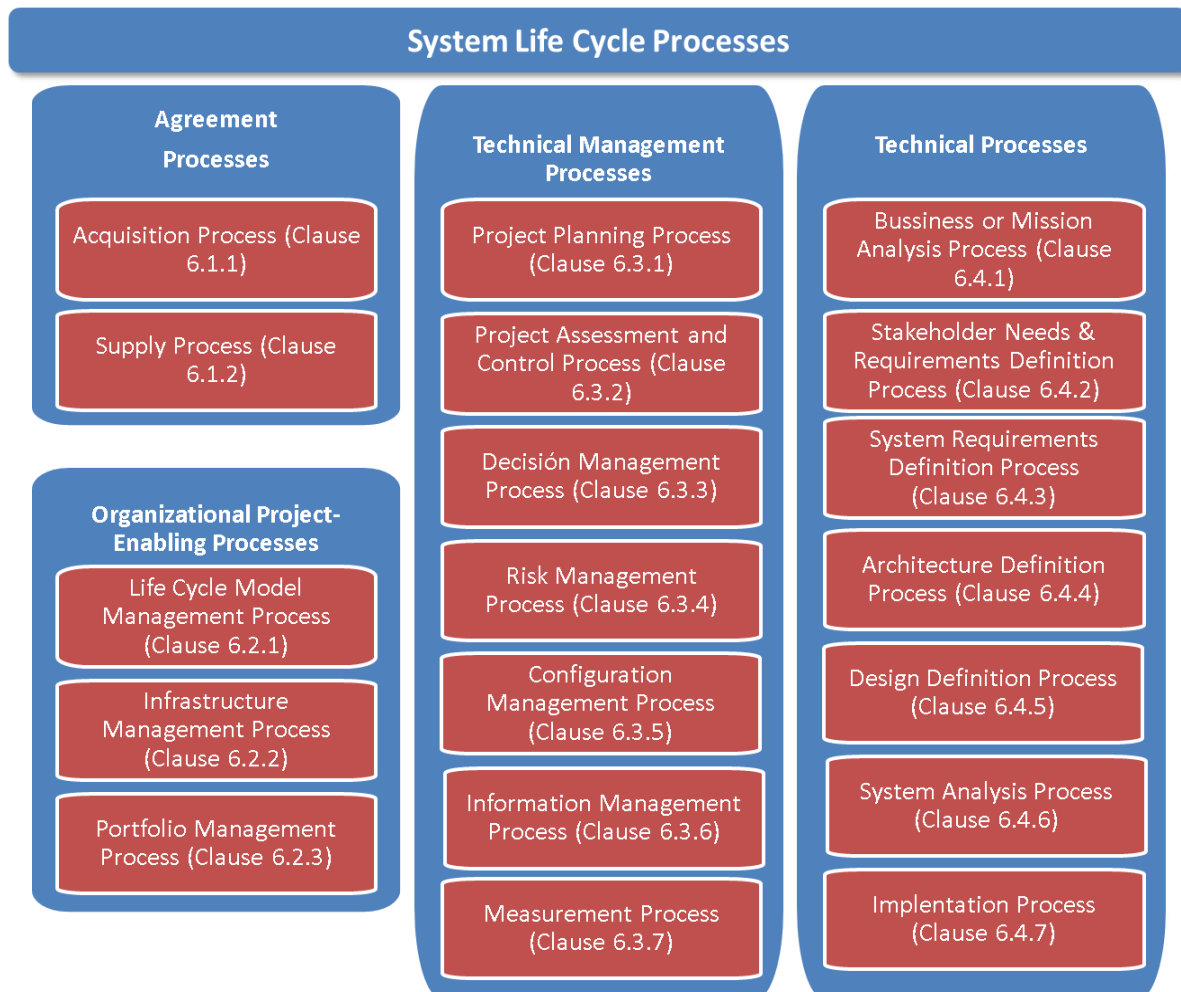


Figure 15 Overview of the Systems and software engineering — System life cycle processes ISO 15288: 2015 [48]

Procurement & Engineering is an essential phase in any service-oriented architecture. Such an agile IT environment enables rapid response to business changes, lowers total cost of ownership by re-using services, increases performance and provides an ideal framework to bring services and products to market much faster.

Procurement

The procurement process involves identifying different needs for suppliers, often based on defined business rules. Therefore, tools and processes must be selected and prepared to communicate with suppliers. In addition, tenders and offers as well as guidelines for the evaluation of offers and thus the supplier himself must be created. For this purpose, the procurement cycle represented in Figure 16 is run through.



Figure 16 Procurement cycle [24]

First of all, dependent on the process to be improved or adapted during the procurement phase, the **need** and corresponding **requirements** (which are passed from the AHT-EPP1 – Requirements) for new or additional products or services have to be identified. Then, a **procurement plan** can be outlined including, among others, identification of suitable suppliers and choosing a tendering process. This also including relevant documentation specifications (e.g. terms and conditions, product specifications, volumes and service agreements) which helps the suppliers quoting accordingly to fulfil the initially defined requirements.

Before the tender evaluation can be done, suitable suppliers need to be identified. In the first approach, a so-called "Request for Information" can be requested from relevant suppliers to get basic information, on e.g. financials and resources. Then, in a next step, a Request for Quotation (**RFQ**) is only send to preferred suppliers, including details on the required product or service. The **tender evaluation** itself consists of assessing the supplier's quality of products and service, overall timescales and financial details, including, e.g., price comparisons and fulfilments of capabilities. Based on this assessment, a final supplier is chosen, with which the **contract** is then drawn up. Afterwards, **supply chain, warehouse and asset management** come into play, being also aware of future trends and business requirements for the product and services provided.

Engineering

After the procurement phase, in which new or additional goods or services are purchased, they must be integrated into the overall system or end-product. In addition, the interconnectivity with the existing environment and the goods and services already available must be examined. Often possible modifications are necessary in order to ensure a perfect interaction. Therefore, different interfaces must be available or created. Within the Arrowhead Tools project different use cases deal with different interfaces:

- Data: Interface between different data sources, e.g., saved in different databases

and/or different data formats. Here for instance, access rights and data merging have to be considered.

- Machine: Interface between different physical systems, where e.g. mechanical and electrical parameters have to be considered.
- Hardware: Interfaces between physical systems in electrical engineering and electronics. The interface equipment of a device is often referred to as connectivity.
- Network: Interface that allows a computer or a network component to access a computer network (also called port or network connection).
- Software: Interface that enables and controls the exchange of commands and data between different processes and components.
- User: Interface between human and machine.

In this phase, the engineering team implement the design of simpler technological products and the **system design**⁴ activities to conceive a set of system elements that answers a specific, intended purpose, using principles and concepts; it includes assessments and decisions to select system elements that compose the system, fit the architecture of the system, and comply with traded-off system requirements. It is the complete set of detailed models, properties, and/or characteristics described into a form suitable for implementation. Prototyping and testing activities are implemented for evaluating the performances and reliability of the designed SoS solutions.

3.4.3.2. Advantages of using the Eclipse Arrowhead framework in this AHT-EP Phase

In the context of engineering process for this phase, the application of the Eclipse Arrowhead framework helps to decrease the effort and cost of the communication and syntax level (service-oriented architecture implemented through a bus integration pattern), automating the exchange of system artifacts in a safe environment and enabling the consumption of operations that require the collaboration of several tools. Additionally, the Eclipse Arrowhead framework also provides other non-functional aspects such as security, service discovery, scalability, etc. that are un-valuable assets to define a toolchain supporting the activity of this phase in the industrial use cases. In the following we report on how the Eclipse Arrowhead framework have given benefits to some AHT use cases:

- **UC-05: Support quick and reliable decision making in the semiconductor**

⁴ The purpose of the System Design [15] is to supplement the system architecture (developed in the AHT-EPP2) by providing information and data useful and necessary for implementation of the system elements. Design definition is the process of developing, expressing, documenting, and communicating the realization of the architecture of the system through a complete set of design characteristics described in a form suitable for implementation.

System design is intended to be the link between the system architecture (at whatever point this milestone is defined in the specific application of the systems engineering process) and the implementation of technological system elements that compose the physical architecture model of the system.

Design definition is driven by specified requirements, the system architecture, and more detailed analysis of performance and feasibility. It addresses the implementation technologies and their assimilation. Design provides the "how-" or "implement-to" level of the definition.

Design concerns every system element composed of implementation technologies, such as mechanics, electronics, software, chemistry, human operations and services for which specific engineering processes are needed. System design provides feedback to the parent system architecture to consolidate or confirm the allocation and partitioning of architectural characteristics and design properties to system elements.

industry: The usefulness and applicability of the Eclipse Arrowhead framework (Eaf) in the context of UC-05 (mainly active in EPP3) has been assessed since the beginning of the project. This includes the installation of a local Eaf environment and many discussions on integration possibilities of TePEX, WHF and DR. During a dedicated and detailed presentation and discussion of the UC-05 activities to AHF experts from the Arrowhead Tools consortium at the Budapest Workshop, it has been commonly decided, that at that stage of development (i.e. within the timespan of the AHT project), an integration of TePEX and WHF is not suitable. Nevertheless, the Eaf concept for TePEX and WHF at a more mature point in time might be beneficial (beyond AHT project). A concept on how this could look like in future is provided in the detailed AHT-EP drawing of UC-05.

- **UC-06: Production preparation tool chain integration:** The three stakeholders Lindbäcks Bygg AB (Lindbäcks), Lundqvist Trävaru AB (Lundqvist), and PodComp AB (PodComp) collaborate to streamline the process from architectural drawing, via a 3D configurator to created machine files (contributing to *Obj. 2 - The move from single to integrated multi stakeholder automation and digitalization*). By utilizing the Eaf they can implement and verify a more automated yet secure way of transferring data in the information flow. Additionally, any last-minute changes in the bathroom design is immediately pushed to PodComp such that at the start of the manufacture of a bathroom pod, the information is correct. Since both Lindbäcks and Lundqvist procure themselves bathroom pods from PodComp, this use case is a good example of procurement negotiations in terms of production availability (e.g., scheduling) and price. (The secure transfer of information with a SOA paradigm must be completed first).
- **UC-08.1: SoS engineering of IoT edge devices: Smart City - Env. Monitoring:** By using the Eaf, stakeholders in this use case were able to completely remove the phase of sensors and services integration which were done via manual installation at the beginning of the project. Using the AHT API services through registration, authentication and orchestration phases it is possible to link different IoT devices and SoS engineering tools and make the communication happen using MQTT. The whole process of service discovery of Vital-IoT (the legacy system) was evolved to integrate with Eaf using HTTP-REST. Hence, it is possible to easily link with services from different stakeholders with minor adaptation operations, which empowers the framework remarkably.
- **UC-08.2: SoS engineering of IoT edge devices: Smart City - AI driven Env. Monitoring:** The IoT integration framework is based on popular solution for IoT and edge computing. The framework will be integrated with the Eaf (contributing to *Obj. 4 - Integration platform interoperability with emerging digitalization and automation framework*). Additionally, the Eaf provides a security mechanism for ensuring that only authorized third-party consumers can access the energy data. Furthermore, the Eaf provides the encryption of communications between consumers and providers (contributing to *Obj. 5 - Flexible, interoperable and manageable security for digitalization and automation solutions*).
- **UC-08.4: SoS engineering of IoT edge devices: Smart Energy - Smart Home:** The IoT integration framework is based on Eclipse Kura and Kapua, a popular solution for

IoT and edge computing. The framework will be integrated with the EAF (contributing to *Obj. 4* - Integration platform interoperability with emerging digitalization and automation framework). The gas smart meter solution will provide an End to End (E2E) security solution. For the LF-NILM the EAF provides a security mechanism for ensuring that only authorized third-party consumers can access the energy data. Furthermore, the EAF provides the encryption of communications between consumers and providers (contributing to *Obj. 5* - Flexible, interoperable and manageable security for digitalization and automation solutions).

- **UC-08.5: SoS engineering of IoT edge devices: Smart Energy – Industrial:** Significant added values from the Target SoS are expected to be enabled by the integration of data originating from heterogeneous platforms through the EAF (e.g. support ML from multidimensional data originating from: sensor networks for structural monitoring, environmental monitoring platforms, weather stations, and possibly other platforms). Specifically, with the EAF the vibration sensors that provides also environmental and ultrasound analysis are integrated (contributing to *Obj. 3* - Interoperability and integration of data from legacy automation engineering tools to the Eclipse Arrowhead framework integration platform). Moreover, a unique integrated toolchain for machine learning life cycle management is offered.

3.4.3.3. Use Case tasks and activities associated to the phase

Use Case UC-05

Support quick and reliable decision making in the semiconductor industry

Within this use case, three “tools” will be further developed and modified, and finally integrated into an existing tool chain. They are:

- DR (Digital Reference): During the engineering phase, integration of added goods and services is a key step for the engineering phase. Thus, interconnectivity and interoperability should be guaranteed. The proposed Semantic Web representation of the Supply Chain, namely Digital Reference is a lingua franca understandable by machines as well as humans. Semantic Web implementation can guarantee interoperability as it creates an abstraction layer that defines concepts and relationships between heterogeneous data sources. Digital Reference allows the interconnectivity between different physical systems, machines, systems and users.
- TePEX (Test pattern extraction): An algorithm which is able to detect test patterns, which are related to malfunctioning testing equipment.
- WHF (Wafer health factor): An algorithm which is able to detect process patterns, which are related to deviations during production.

Tasks related to TePEX and WHF are mainly performed in the engineering phase. Here, special focus lies on data interfaces (Ad TePEX in Figure 17).

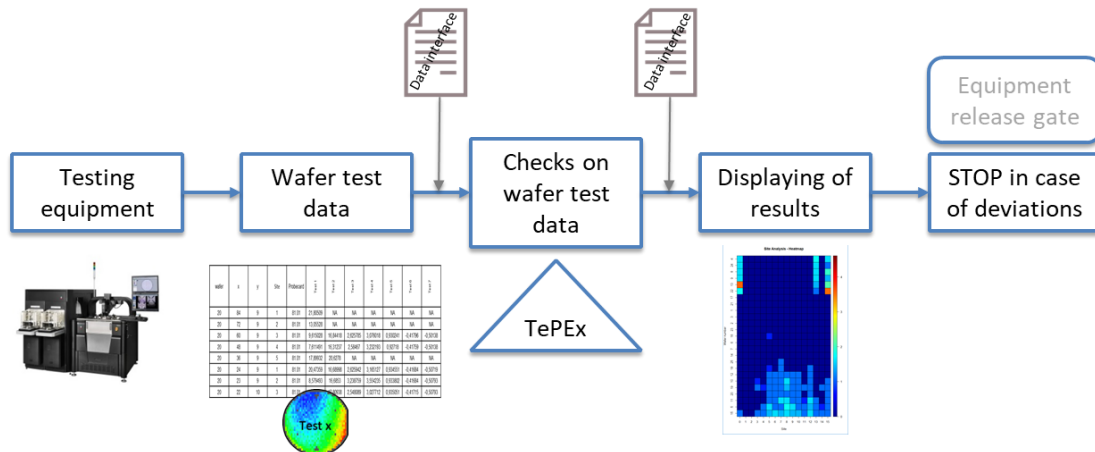


Figure 17 Two data interfaces are needed for TePEX

For the TePEX algorithm, wafer test data are used, which are electrical tests, taken per device. The relation between wafer test data and the testing equipment comes over the probe card, which is the part, connecting the testing equipment with the wafer to take the electrical test as depicted in Figure 18.

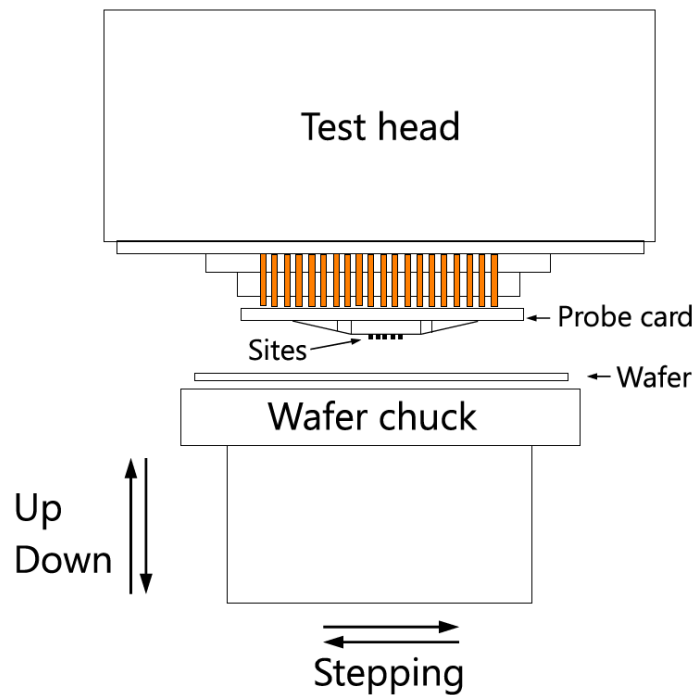


Figure 18 Testing Equipment

More specific, a probe card consists of multiple “sites” in order to contact and test multiple devices in parallel. In case of e.g. degradation of one site, so-called test patterns are visible on the wafermap, which is a representation of one electrical test on the corresponding x-y position at the wafer (see Figure 19).

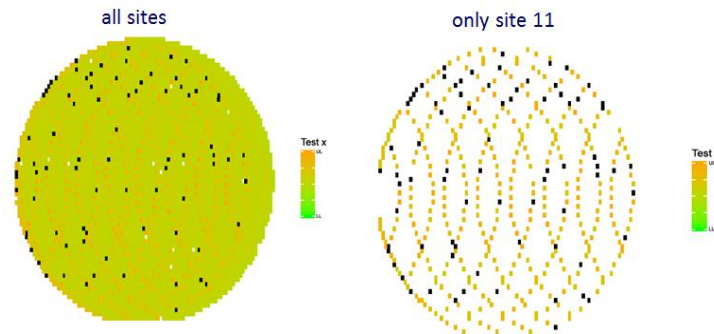


Figure 19 A systematic pattern is visible on the wafermap of the full wafer (left) and can be related to a single site, Site 11 (right).

Since a probe card consists of multiple sites (see Figure 18), each site needs to be investigated on its own. Hence, the first data interface must provide the following information in order to apply the TePEX algorithm: Wafer ID, electrical tests, site number.

With the TePEX algorithm, for each wafer and each site one value per electrical test, i.e. per wafermap, is calculated, indicating whether a test pattern is visible (value > 0) or not (value = 0). Hence, the second interface must provide the output format of the TePEX algorithm, which is one column, containing the calculated TePEX values ≥ 0 , additionally to the information from the first data interface (Wafer ID, electrical tests, site number), visualized in the heat map of Figure 17.



Figure 20 Two interfaces are needed for WHF

For the WHF (Figure 20), also wafer test data provide the input for the WHF calculation. Here, compared to TePEX, the same information is needed for the first data interface, except for the site.

The output of the WHF is one value per wafermap, or can also be aggregated to one value per wafer. The value is between 0% and 100% reflecting the health status, dependent on the presence of detected process patterns. Hence, 0% means that the wafer is "unhealthy", because strong process patterns are visible, whereas 100% means that no critical pattern is present at all and hence, the wafer (or wafermap) is healthy.

Concerning the AHT-EP of the UC-05 shown in Figure 21, StkH-1 is the main stakeholder in this UC, which develops artefacts such as algorithms or ontologies (in case of UC-05, these are TePEX, WHF and DR) and then provides it or delivers it to further stakeholders. Main tasks of this UC are performed in EPP3, because they need further investigation and development, and scalability to various application scenarios in the semiconductor industry.

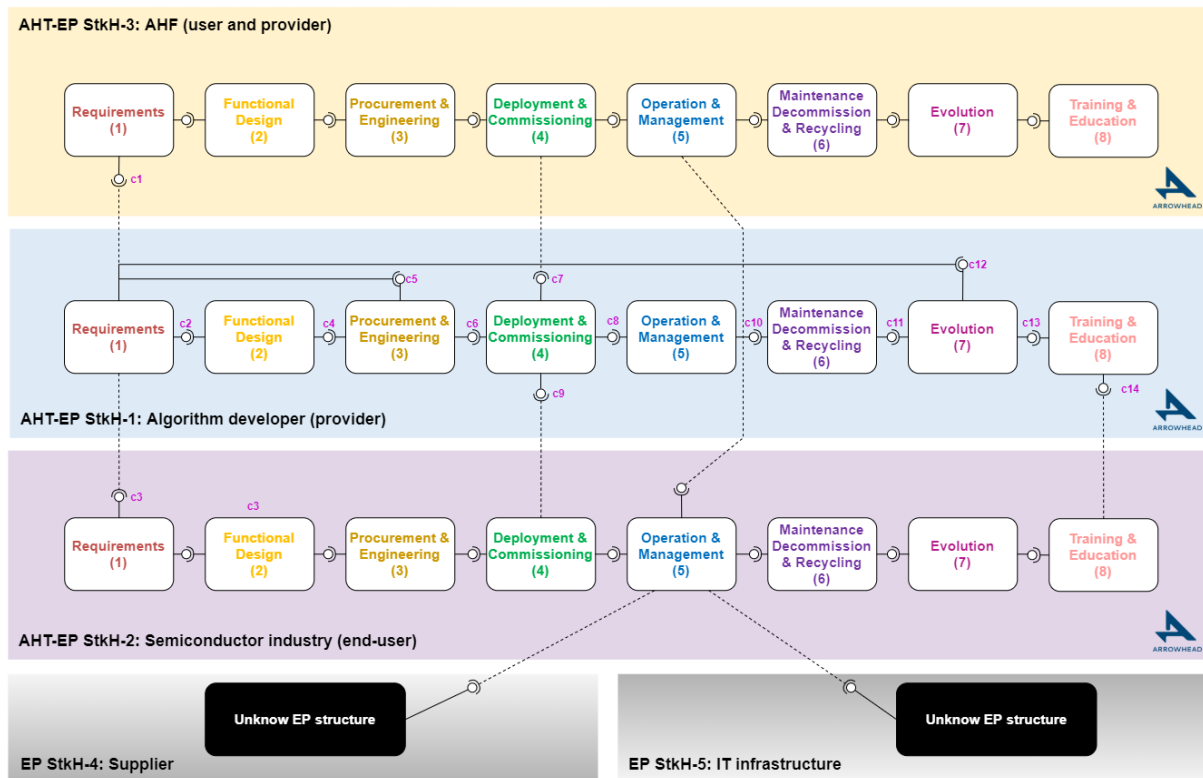


Figure 21 AHT-EP of use case UC-05

Use Case UC-08.1

SoS engineering of IoT edge devices: Smart City - Env. Monitoring

In this UC, EPP3 focuses on 4 parts:

- Measurement services: Purchase of unavailable sensors and development of software code for interfaces to send data. Development Test of implemented functions
- Edge Computing: Purchase of unavailable industrial PC to run Edgex framework and related pieces of code. Development of the edge processing software code.
- Vital/IoT: Development of code to adapt GUI and internal process of discovery to involve Arrowhead Frameworks architecture services.
- Robofuse: Development of code to adapt GUI and internal process of discovery to involve Arrowhead Frameworks architecture services.

Work done in this phase is directly fed to the phase of *Training & Education* (EPP8) whose primary output is to train people of the End Customer.

Use Case UC-08.2

SoS engineering of IoT edge devices: Smart City - AI driven Env. Monitoring

In this UC represented in Figure 22, all stakeholders (StkHs) have activities in EPP3. StkH1 receives from the customer the information about the wireless capability in the deployment environment. StkH1 will develop the system modules with specific Hardware configuration, StkH 2 designs the PCB with Orcad and select the components and technologies for the PCB design. Develop the firmware for AI-Camera sensors using vector processing compiler and GCC toolchain. StkH3 will provide the tailored implementation of the algorithm for the control of accesses.

Concrete Engineering tasks are:

- Image processing training and validation phases will be executed as offline procedure
- Image Classification will eventually run on the AI-drive camera after an offline test.
- New software releases will be assessed at server side and offline, before updating.
- Release of the new software versions

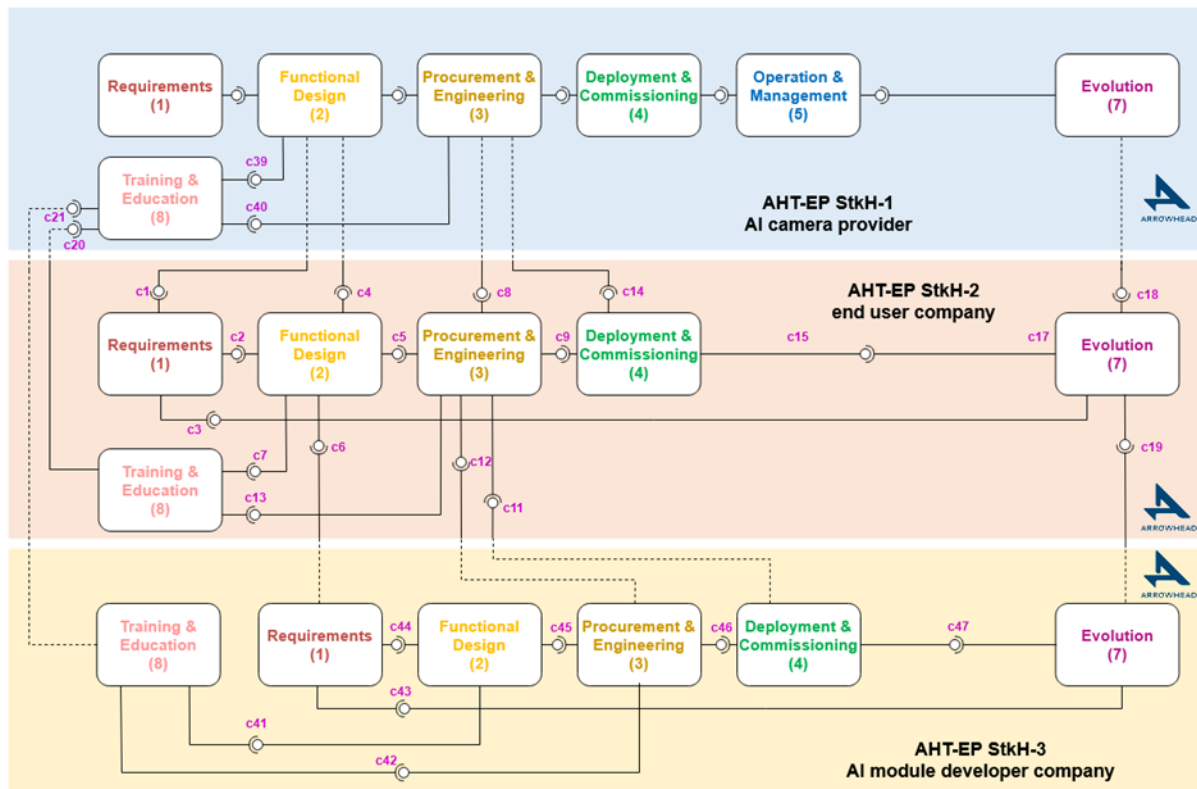


Figure 22 AHT-EP of use case UC-08.2

Use Cases UC-08.3 and UC-08.5

SoS engineering of IoT edge devices: Smart City Condition monitoring

SoS engineering of IoT edge devices: Smart Energy - Industrial

In both UCs, the goal of EPP3 is the production of the Local Cloud Gateway golden sample, where two processes are developed in parallel:

- design, development, test and debug of the Local Cloud Gateway hardware and software; design, development, test and debug of the use case specific business logic;
- interaction with suppliers to acquire electronic and mechanical components, define external services to manufacture hardware prototypes; interaction with suppliers to acquire source code, software libraries and software licenses potentially required for the software part of the gateway.

Use Case UC-08.4

SoS engineering of IoT edge devices: Smart Energy - Smart Home

EPP3 in this UC, represented in Figure 23, comprises 4 parts:

- Smart Gas Meter: StkH2 produce the Smart meter and supply them to the vendor

product lines

- Smart Electric meter (hw existent): The update firmware from evolution is deployed to the different installations of the smart meters (if required) or deployed in a new generation of smart meter
- LF-NILM: StkH4 receives from the owner of the legacy infrastructure the credentials to access the energy data stored in the database. StkH4 develops the system modules (provider and consumer) and the web dashboards for data visualization. StkH4 develops a security mechanism to ensure that only authorized third-party consumers can access energy data.
- IoT integration platform:
 - Supply chain planning based on functional design aiming at an efficient and cost effective procurement (acquire the software and hardware goods needed from suppliers to build the prototype)
 - Development of the IoT integration platform by using a well-defined toolchain;
 - Design and development of the use case specific business logic
 - Test of the IoT integration platform

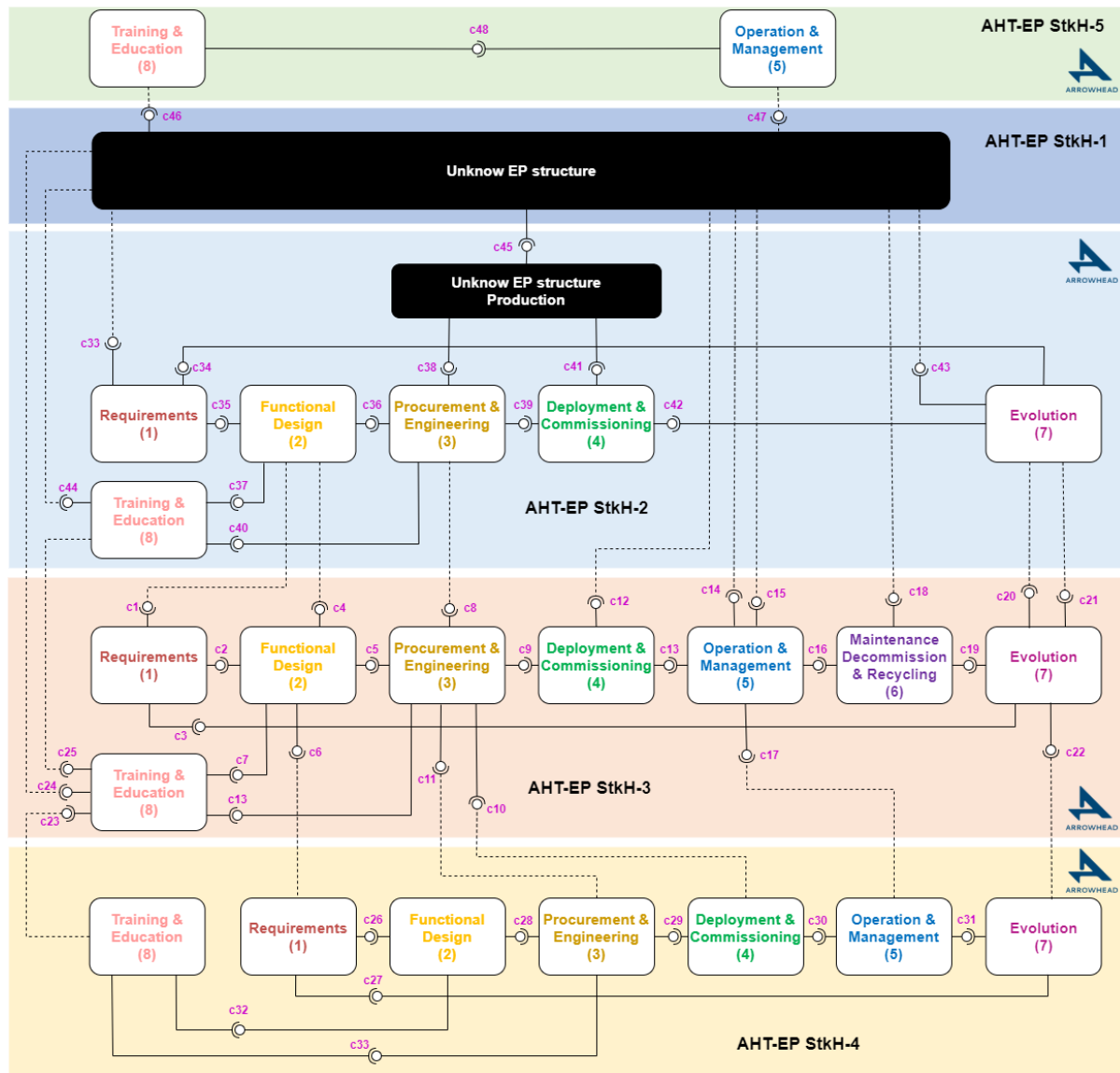


Figure 23 AHT-EP of use case UC-08.4

Use Case UC-07 CNC Machine Automation

The use case presented under this title includes many scenarios. One of them relates with the Engineering phase Procurement & Engineering. There are different actors in the UC-07, taking part mainly in the deployment and commissioning phase, but there is an important, especially from the point of view of added value, scenario that falls under the hood of the procurement phase.

A CNC includes some standard subroutines (canned cycles) to perform some basic operations (Figure 25). These routines can be considered services that take some parameters as inputs (depth of cut, geometry, etc.) and produce ISO code for the CNC. This code can be pure G-code following the standard or specific code for the target CNC.

Besides standard, factory-installed canned cycles, a number of optional cycles exists that can be bought both included in the CNC (ordered from the OEM) or after machine tool installation

(ordered by the final user).

Three different kinds of routines or services are possible:

- **Post-processor:** In this case, a tool (as defined in WP4) performs the translation between the CL-Data provided by a CAM and produces G-code for the CNC, provided with some parameters that define how the operations must be done.
- **Canned cycle:** This is the traditional canned cycle. Usually it must be provided as specific G-Code for the target CNC due to the need of doing math inside it to find the points that the cutting tool must follow.
- **Technology cycle:** This is a different, more advanced cycle where inputs include not only the geometry but also the material to cut and the available tools. Cutting conditions are otherwise determined by the machine operator and input to the canned cycle. This new way of obtaining the ISO code automates another step of the part piece production, very well in the spirit and definition of tool as defined in WP4.

It must be noted that the above definition of the cycles inside the procurement phase allows a pay per use definition of the service. In such a case, the user will upload the "problem" (the material, the operation (ex: pocket milling), the geometry, etc.) and would get the ISO code and even the suggested tools.

This extreme case will not be addressed during this project but shows the direction to be taken in the future for fully connected machines. This connection can be done inside the factory (with a central computer serving the machines) or even to the cloud, where a market for on-line services can be developed.

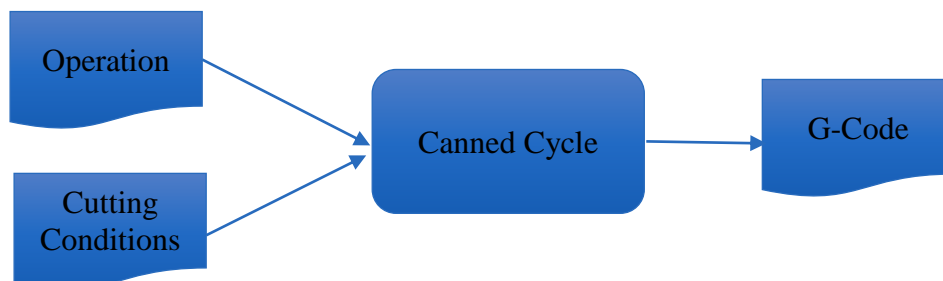


Figure 24 CNC standard subroutines

A simple drawing of the above described process is proposed in Figure 26 to illustrate the description. For technology cycles the process is slightly more complicated inside, but easier for the operator. Instead of calculating himself the better cutting conditions, this is left to the own service:

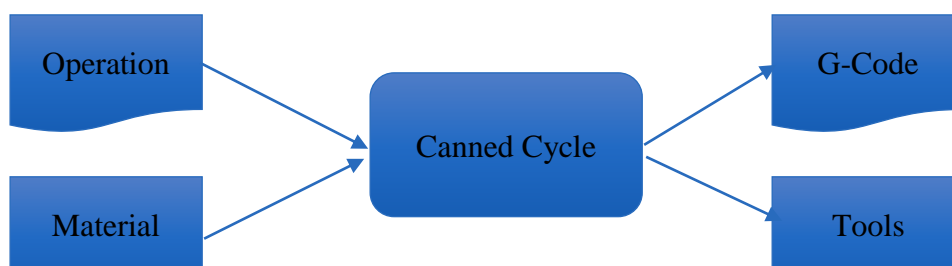


Figure 25 Routines or services

An important point to consider is that this canned cycle can be provided not only by the CNC manufacturer but also by the Machine tool builder or even third parties. As an example, there are situations where a manufacturer of measuring probes sells these subroutines to move and measure a part piece, including all the necessary math to get accurate values for the measured dimensions.

This scenario imposes some restrictions on the use of the cycles. Being ISO code an ascii code, the CNC must provide some means to protect the seller know-how. This problem, as well as the licensing model and handling, will be part of the present project in the WP2.

Use Case UC-09

Machine operation optimisation

The Procurement phase of the UC-EP target mainly two purposes: acquisition of hardware components for the IoT layer (e.g. GPS modules, power components, protections, etc.), and of cloud infrastructure for hosting the upper layers of the platform. The purchase process is subject to the general rules of the company (ACCIONA). Usual practice is trying to identify at least three alternative suppliers, ask quotations from them, and select the best one based on a combination of criteria: price, technical quality of the offer, delivery terms, payment conditions, technical support, etc. If the supplier selected was not previously registered in the supplier database of the company, it would be necessary to carry out a registration process, thus adding more time to the purchase process. Management of procurement tasks within an iteration/sprint of the product is carried out following the same stages as the engineering tasks that are described in the next paragraph.

The Engineering phase within the UC-EP targets the development, testing and documentation of the software and hardware modules of which the digital platform is composed. As it has been described for the Functional Design phase of the UC-EP, the product backlog items to be developed within an iteration/sprint are broken down into tasks, which evolve through the following stages within the sprint: Analysis, Development, Quality Control, Documentation, and Done. In the Analysis stage, an analyst drills down into the requirements of a task, and produces a specification of the work to be done in the next phases of the UC-EP. If the aim of the task is the development/evolution of a software or hardware module, then the analyst produces the functional specification/design of the module. After the Analysis, a developer will develop/evolve the software/hardware module according to the design. Then, a tester would perform the Quality Control of the module. Several iteration loops between Analysis and Development, and Development and Quality Control can take place until the task result is deemed suitable, and then the task goes through the Documentation phase, in which the software/hardware module produced will be documented. Once the outcomes of the task comply with the acceptance criteria that were defined, it reaches the last stage of Done.

Interaction of EPP3 with other AHT-EPs:

As can be seen from previous examples, EPP3 is highly interlinked with other AHT-EPs but also across different stakeholders. The work done in EPP3 by many UCs and corresponding partners are versatile. To show the relation to other AHT-EPs, in the following, the activities performed in EPP3 are globally called "applications":

- First of all, requirements are collected. The defined requirements are, in a first step, needed for EPP2 but have to be cross-checked in EPP3 as well.
- In EPP2 the concept is created, which means that single functional blocks and the system architecture are created, finding then together to a bigger picture in EPP3.
- EPP3, implementing the system design tasks, represents a kind of pre-requisite for EPP4, where e.g. graphical user interfaces are implemented to make the outcome of EPP3 applicable.
- In EPP5 the developed applications are in a kind of "productive" usage, where also feedback to EPP3 is provided e.g. if adaptations are necessary.
- EPP6 covers the maintenance and sustainability of such applications developed in EPP3.
- There is a strong tie between the procurement exercises of one stakeholder which becomes the requirement of the supplying stakeholder (i.e., the supplier).
- In EPP7, the evolution phase, improvements of the exiting application from EPP3 can be made. Usually, the analysis done in the Evolution phase drives an update of the requirements that triggers again all following phases and hence, EPP3 as well.
- In EPP8, information about the applications from EPP3 is documented, like, the used data, meta-data and structure, tips & tricks, the usage, or simply, what it can/what it does not can. For instance, this is provided in form of a user manual or handbook.

3.4.4. Deployment & Commissioning (EPP4) < SubTask 4 >

The *deployment* sub-phase consists in the installation/integration of the system/product in the final operative environment. The deployment also includes the preliminary verification and validation of the system/product that precede the commissioning. Once installed, a product identifier (e.g. a serial number) is associated to an owner/user id and stored in a database that will be accessed during the monitoring and simulation of the system.

The *commissioning* sub-phase is the process of assuring that the system/product is designed, installed, tested, operated, and maintained according to the operational requirements of the owner or final client. A commissioning process may be applied not only to new projects but also to existing units and systems subject to expansion, renovation or revamping. The commissioning usually precedes the operations & management phase.

It should be noted that the two sub-phases, deployment and commissioning, should be considered jointly as it allows to integrate quality assurance with the deployment of the systems, emphasizing the importance of commissioning.

3.4.4.1. Phase Description

Between the design phase and production environment, the deployment comes into play. Each and every system eventually comes to a point, where abstract yet functional and engineered model should become a physical implementation. Then, right after deployment, the system should be tested whether it realizes the required functionality. This is where the deployment and commissioning phases should be considered.

In the context of system design and operation, the phase "Deployment & Commissioning" is the one on the border of design time and run time. The first part, deployment, is devoted to preparation and instantiation of the Arrowhead core systems and the whole local cloud in a secure and reliable way. The commissioning part concerns all the actions that need to be

undertaken to assure that the deployed system is working properly. In the worst-case scenario, the commissioning may lead back to one of the previous phases to revisit requirements, design or engineering parts.

Deployment

The deployment part, in particular, should serve as an interconnection between the engineered functional model of the system and the final configuration of a cloud. All the tools supporting the transition should be included in this phase, as well as the smooth transition to the commissioning sub-phase should be assured.

As an outcome of the deployment part, one should have the first working version of the system, which in the context of Eclipse Arrowhead framework is a local cloud. Since it is intended to implement Service-Oriented Architecture (SOA), not all the systems have to be interconnected and configured at the first run, which results from the loose coupling requirement, allowing services to discover and connect on demand. They should, however, be accordingly authorized and registered in the Service Registry.

Commissioning

The commissioning sub-phase should concern checking the cross integration within the Eclipse Arrowhead framework and proper interconnection of all the services. Within this part a set of tests should be performed, for instance:

- **Unit tests** - usually performed before deployment to verify whether single units of the whole solution (methods, services) are working properly.
- **Integration tests** - performed with an aim of identifying any defects on interfaces between services, and assuring proper interaction between the parts of the system.
- **System tests** - one of the final steps on the way to assuring that the system works as a whole as it is supposed to, and its operation is compared with the requirements specified during the design considerations.
- **Acceptance tests** - final tests made to ensure that the business targets are realized and the customer needs are satisfied.

In terms of Arrowhead, it might be beneficial to include in the above tests of adequate operation and maintenance, depending on the implemented use case and the requirements.

The proposed engineering process model aspires not only to reduce the engineering costs related to the set-up of the SoSs, but also to assure the quality of the deployed systems. This goal is visualized through the numerous supporting core systems and tools (e.g. test tool, sandboxing tool, on-boarding tool) that are developed as a part of the AHT project. It's important to mention that commissioning concerns not only software but also hardware tools and systems.

Examples of tools

As an example of tools being a part of the Deployment & Commissioning phase one can enumerate:

1. **Docker image** of the Arrowhead Framework core services.
2. **System testing tools**, with the aim of analyzing cross-integration and communication between interfaces at the time of the first run.
3. **Configuration tool** that supports the final configuration of a local cloud with the associated components and services.

4. **Code generation tool** for e.g. microcontrollers, Programmable Logic Controllers (PLCs) or sensing modules.

3.4.4.2. Advantages of using the Eclipse Arrowhead framework in this AHT-EP Phase

In the context of engineering process for this phase, the application of the Eclipse Arrowhead framework helps to decrease the effort and cost of the communication and syntax level (service-oriented architecture implemented through a bus integration pattern), automating the exchange of system artifacts in a safe environment and enabling the consumption of operations that require the collaboration of several tools. Additionally, the Eclipse Arrowhead framework also provides other non-functional aspects such as security, service discovery, scalability, etc. that are un-valuable assets to define a toolchain supporting the activity of this phase in the industrial use cases.

Eclipse Arrowhead framework itself, through its core services, enables the deployment of new Systems of Systems with the support for Service Registry, Authorization, and finally - the orchestration of the services. In the latest version, Arrowhead supports not only the intra-cloud assembly of SoSs, but also inter-cloud, which addresses the scalability of the SoSs, maintaining at the same time the autonomy of local clouds and security-related aspects.

Currently, there is ongoing work to develop a solution integrated with Arrowhead that allows for testing and evaluating systems prior to connecting them to a local cloud. Also, solutions allow for rapid set-up of new services in local clouds, reducing the engineering costs required to deploy the solution.

3.4.4.3. Use Case tasks and activities associated to the phase

In this section, a brief overview of some use cases related to the deployment & commissioning phase is provided.

Use Case UC-10

Rapid HW development, prototyping, testing and evaluation (ARCELIK)

The aim of the use case is to develop automated electronics validation test tool and tool chain for power supply circuits. The tool and tool chain supports different white and black goods communication protocols.

In the following the steps of the tool to be developed in the UC:

- Enter setup configuration and measurement set by operator via UI
- Receive test configuration data via AH local cloud
- Configure AC power supply and variable load
- Take measurement via FMC board
- Save measurement data and calculate necessary values
- Prepare test results table and send operator via AH Local Cloud

In the following we discuss the architecture of the use case presented in Figure 26 with the importance of the Deployment & Commissioning phase.

Within this phase manual deployment and automated configuration of system is performed. After deployment system is tested and its performance validated.

There are two parts of the use case that contribute to the reduction of the engineering costs:

- automatic electronic validation of power supply units
- automated data collecting and reporting of test results

The focus is on the following tools:

- Embedded Zynq Ultrascale+ Unit (ZynqU+)
- FMC A/D (A/D)
- Relays(Relay)
- FPGA Control Unit for Power/Load (FPGA)
- Remote reconfiguration of FPGA (RemCtrl)
- Accelerated digital design on multiple PCs (ParDesign)

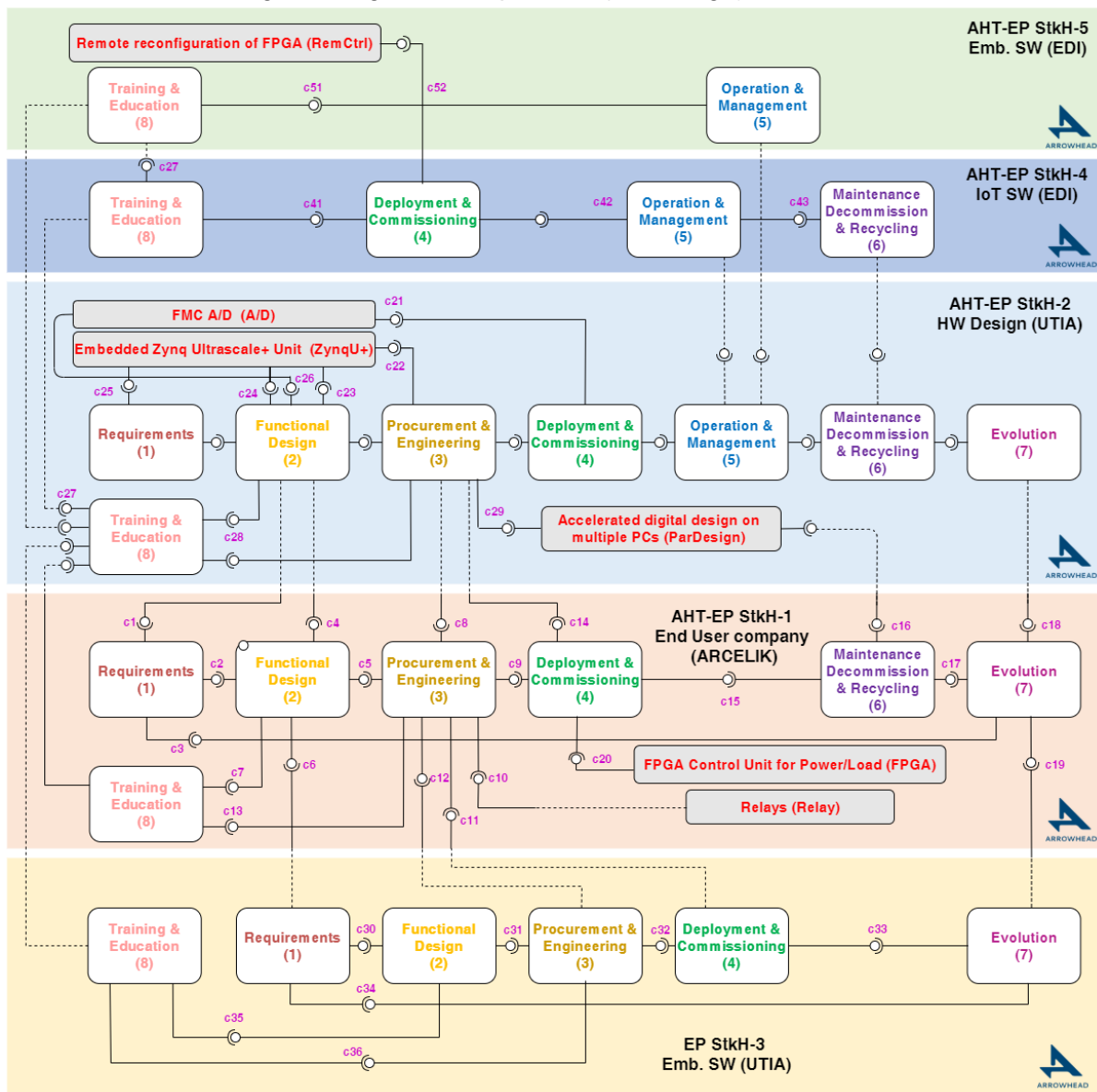


Figure 26 AHT-EP of use case UC-10

Use Case UC-11

Configuration tool for autonomous provisioning of local clouds (DAC)

The aim of the use case is to show the applicability of part of Arrowhead Tools to deploy and

manage local cloud in logistics. The use case will be focused mainly on the Authorization and Authentication system in connection with the on-boarding process of a new device. For the purpose of the use case, we assume that the complete, base model of a system is known and properly engineered.

In the following we discuss the architecture of the use case presented in Figure 27 with the importance of the Deployment & Commissioning phase.

Within this phase, automatic deployment and configuration of the local cloud will be performed. StkH1 and StkH3 cooperate during the design phases to develop the final architecture of the solution that meets the requirements. As a result, StkH1 develops training materials and documentation of the solution, which can be consumed by other stakeholders. StkH2, solution integrator, use onboarding toolchain to deploy the designed solution (EPP-4), which can be further used by other stakeholders through Cloud Management Infrastructure in EPP5. Also, tools supports EPP6 to examine the performance of local clouds, and to diagnose the systems. As a part of EPP7, the designed system might be further extended (which also addresses moving from the design-time to run-time engineering) using the onboarding toolchain.

On the basis of a complete model of a system, the initial configuration will be set-up. After deployment, the system will be tested and its proper work will be validated.

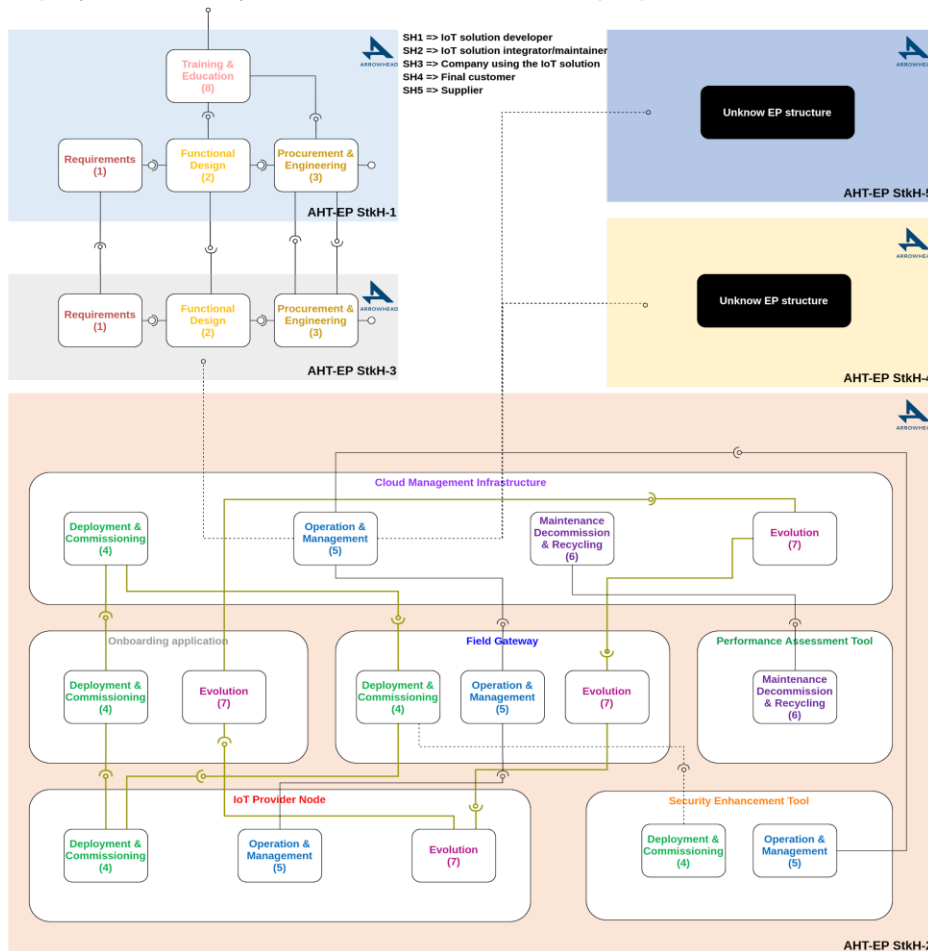


Figure 27 AHT-EP of use case UC-11

Use Case UC-13

Digital twins and structural monitoring (NTNU)

The use case is to develop digital twin of crane using both online and offline sensor data to provide structural monitoring.

Currently, UC partners are working on the functional blocks of this use case:

- Determination of use the Palfinger crane mounted on NTNU research Gunnerus vessel as the testbed of the use case;
- Investigation of sensor types and how to install them on the specific crane;
- Crane model including dynamic model and 3D model are under development;

In the following we discuss the architecture of the use case presented in Figure 28 with the importance of the Deployment & Commissioning phase.

StkH2 implements and StkH1, StkH3 test fulfilment in delivery according to specification.

Whereas, Internal SIB team plans and prepare deployment of the two tools:

- Boliden Integration Box backend (BIBB).
- Boliden Integration Box adapter (BIBA).

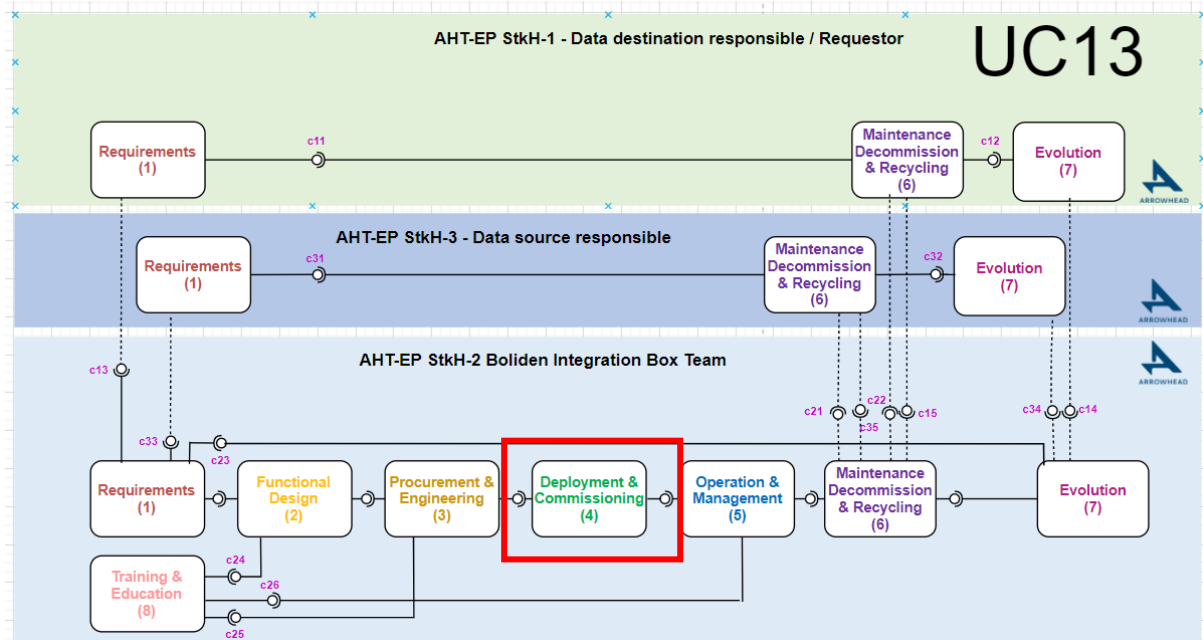


Figure 28 AHT-EP of use case UC-13

3.4.5. Operations & Management (EPP5) < SubTask 5 >

These phases consist in operating and managing the system/product according to the operational specification of the system/product and requirements of the owner or final client.

In this phase, one of the stakeholders will monitor the data streams coming from the operating system and will be able to explore the status and future behaviors of the product, introducing real data in its digital thread. In case a digital twin is available, the simulated model will be used to evaluate deviations from the normal behavior, thus, the exploration continues without any impact on the real product, until a normal behavior is identified. With this approach, the real system continues to operate according to the operational specification. Moreover, in case the digital version of the product is available, operators will be able to predict warnings or errors,

and planning in advance session of predictive maintenance.

Operations is divided into operative controlling and strategic planning of the manufacturing of goods. The main goal of operations is to convert material and labor into products with highest quality concerning the customer requirements. Operations management teams attempt to balance costs with revenue to achieve the highest net operating profit possible.

The operation management is working in a business that is continuously improving the products.

Operations comprises the work of managing the inner workings of the business as efficiently as possible. Whether products or services are provided, every company must oversee the design and management of behind-the-scenes work.

The operations strategy concerns policies and plans of use of the firm productive resources with the aim of supporting long term competitive strategy. Metrics in operations management can be broadly classified into efficiency metrics and effectiveness metrics. Effectiveness metrics involves:

- Price (actually fixed by marketing, but lower bounded by production cost): purchase price, use costs, maintenance costs, upgrade costs, disposal costs
- Quality: specification and compliance
- Time: productive lead time, information lead time, punctuality
- Flexibility: mix, volume
- Stock availability
- Ecological Soundness: biological and environmental impacts of the system

3.4.5.1. Phase Description

The Operation & Management engineering phase is as essential as all of the other phases, yet it plays a central role as it characterizes the manufacturing profile of the stakeholders. The tools and systems in the Operation & Management phase communicate with those found in the other engineering phases (referred here has horizontal interaction). We also find vertical interactions from and to physical devices and tools to enterprise management. This vertical dimension has been modelled as a layered pyramid (ISA 95 [30]) with only interlayer interactions.

The industrial internet of things (IIoT) provides many new technologies and applications, which can be used to improve operations and maintenance in modern factories (see Figure 29 below).

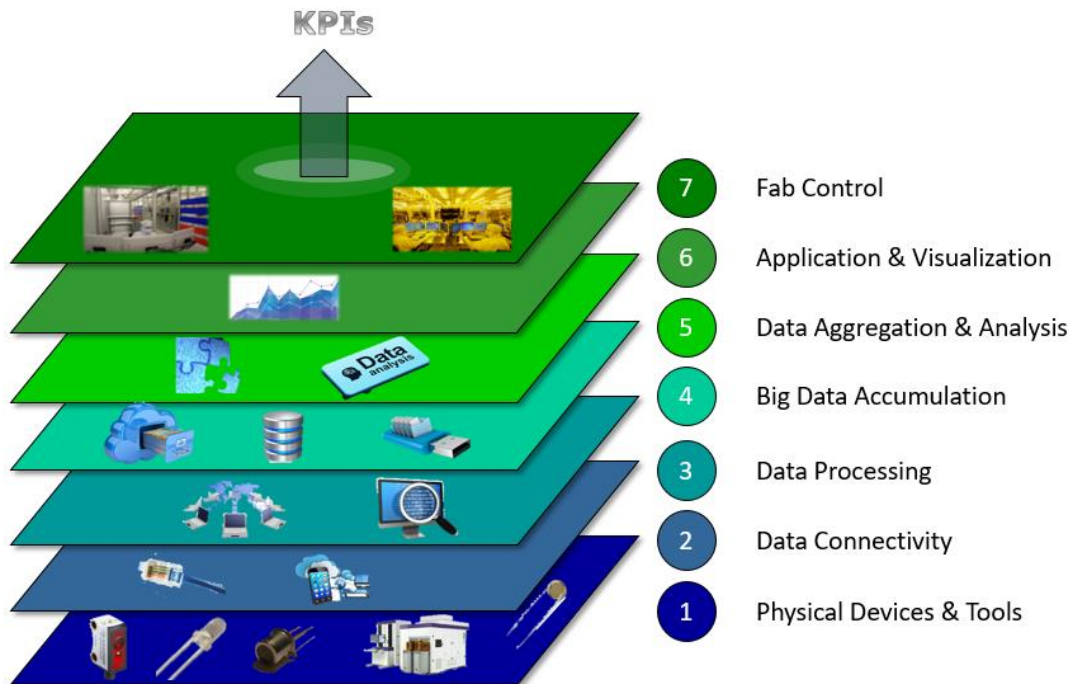


Figure 29 Seven layers for the integration of data from devices to fab control

The base of the seven layers are physical devices, e.g., any sensors or actuators on the shop floor of the manufacturing plant. In the next step the data connectivity must be assured to get out the relevant data from the distributed sensor systems.

Within the Arrowhead Tools project an IoT Gateway has been researched and conceptualized, which enables sensor and data connectivity including levels 1 and 2. Given this concept, all other levels can make use of this capability, which enables the integration of new sensor systems into the entire Smart Manufacturing Framework. This creates novelty insights and opportunities into so far hidden potentials for optimization of the entire production system and corresponding methods. It is important to see that based on this approach new arising key sensor technologies contribute directly to new knowledge discovery and production control improvements.

Data lake applications and the Arrowhead Framework are important enabler for the tracking and orchestration of the data. For the data pre-processing, data accumulation and advanced data analysis are the tools that will be established in the Arrowhead Tools project and which will be the enablers for step 3 to step 5. Data visualization can be performed by different software solutions e.g. dashboards established with commercial software tools, be it "tableau" or any other software application. Based on this pyramid, fab control can be performed in a way that the most relevant key parameters (KPIs) in fab are well balanced and under control. From this general description, we can turn our attention to the activities in work package 9 to be more concrete and then look deeper into some use cases since there are "horizontal" interactions with the other the other engineering phases and their tools and systems.

The industrial use cases within WP9 are more or less use cases with background on manufacturing e.g. UC15 from the automotive company Volvo for "smart kitting", use case 5 & 16 from Infineon, Bosch and KAI for the semiconductor manufacturing, and use case 18 from the company Boliden for the metal industry. The use case 17 is for an engineering building for the semiconductor fab at Infineon Austria. All of those use cases have the goal to set up a

process along the whole engineering phase from the first idea to a stable high volume production. Use case 16 is mainly focusing on semiconductor front end manufacturing. This means that this industry has to develop and manufacture products with short life times. For products in the computer and IT technology the product life times are sometimes only between 1-2 years e.g. a new smart phone which requires development times of sometimes also up to minimum one year. So a perfect planning and a fast engineering phase is absolutely needed. Operative controlling is the active control of the material flow (in a wafer factory, the automated processing of the lots) which is based on different workflows defined in the manufacturing execution system. In a modern semiconductor factory all of the material and data flows should be automated and digitized. The main challenge in this context is to ensure the exact tracking and monitoring of the material and the key performance indicators in real time. Sensor data are absolutely needed for a correct tracking of all data. Therefore, an excellent management of the operation phase with the previous engineering phase are the key factors (/indispensable enablers) for the successful operation in the factories.

The IIoT offers many interesting applications for the industry. For a semiconductor factory, Figure 30 below gives an overview about a lot of fields, which can be used in a semiconductor factory.

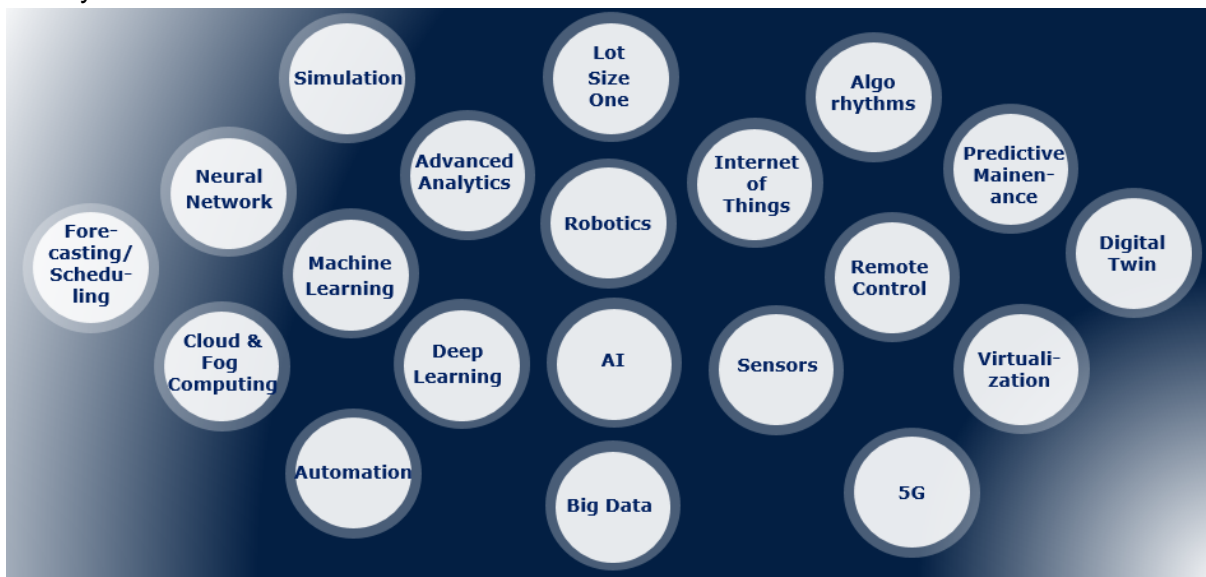


Figure 30 Main elements out of the Industrial Internet of Things (IIoT) used in semiconductor manufacturing

Lot size one is a very important topic in terms of the customer demands meaning that more and more products have to follow very specific customer requirements and, hence, special working instructions and equipment. Therefore, digitization e.g. the definition of AI based algorithms are essential to manage the future demands of manufacturing in the high tech industry. In the Arrowhead project the semiconductor companies have already shown that a shop floor with more than thousand machines delivers more than 1 Billion sensor data a day, which can be used for monitoring and controlling the full manufacturing shop floor using the manufacturing execution system (MES) with all features to control the data and material flow. Figure 31 shows how an innovative MES is established to fulfil these demands.

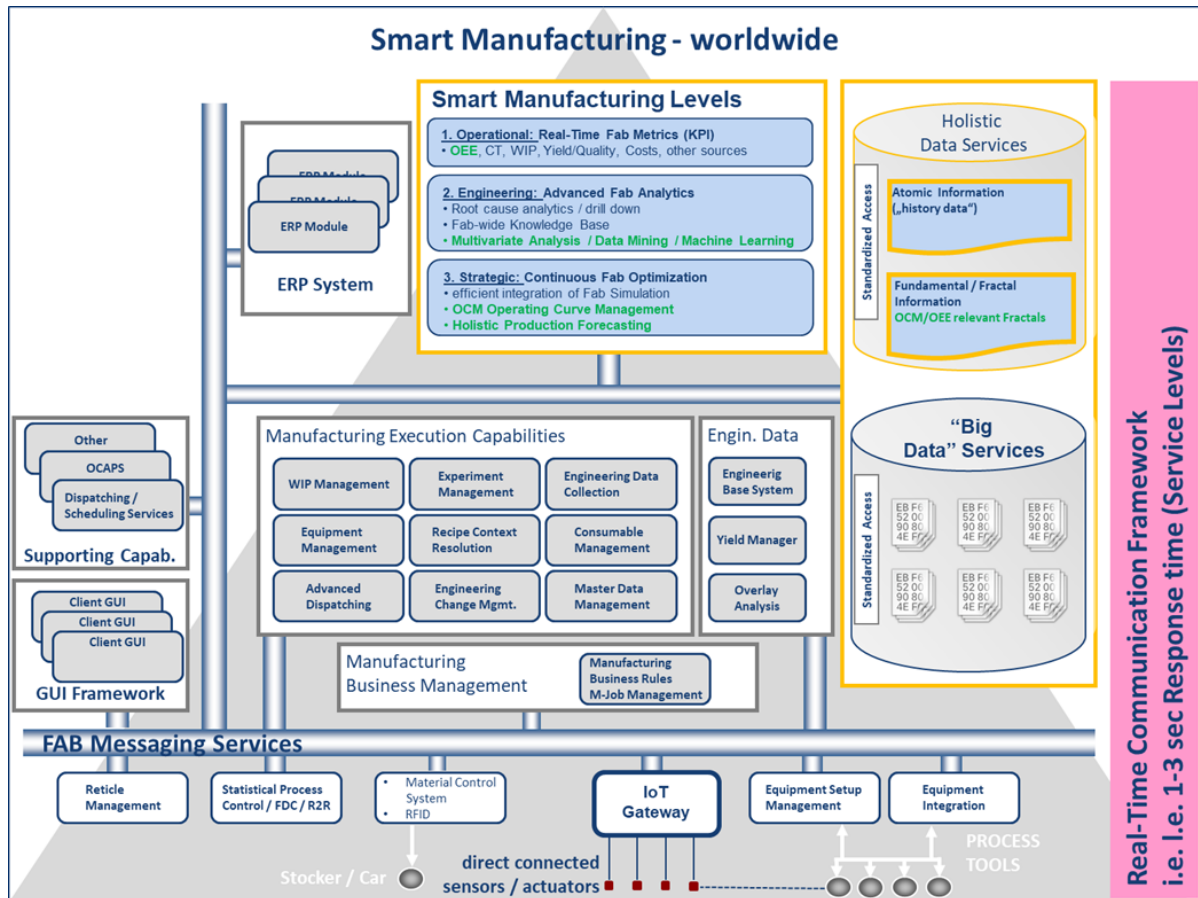


Figure 31 Overview of a modern MES in a semiconductor facility (source: MES @ IFD)

The core of this system is a message service, which is the connection to the IoT Gateway and the sensors and actuators. In the past, the classical semiconductor equipment was already equipped with different sensors which have been specified by the tool owners in respect to the process requirement. In this case, the generated data flow is performed by the equipment coupling. In a 300 mm factory, all equipment has must fulfil SEMI 300 standards for the interfaces to MES and the data interfaces e.g. SECS/GEM communication.

Potentially, not only new IoT devices but also the entire communication between all the components shown in Figure 31 can be integrated and orchestrated via the Eclipse Arrowhead framework (Eaf). Within such a perspective the entire Smart Manufacturing Framework can take fully profit of the benefits of the Eaf, such as: authentication, orchestration, advanced configuration (such as dedication and distribution / connection management).

Figure 32 below shows the main SEMI 300 standards, which are needed as an interface.

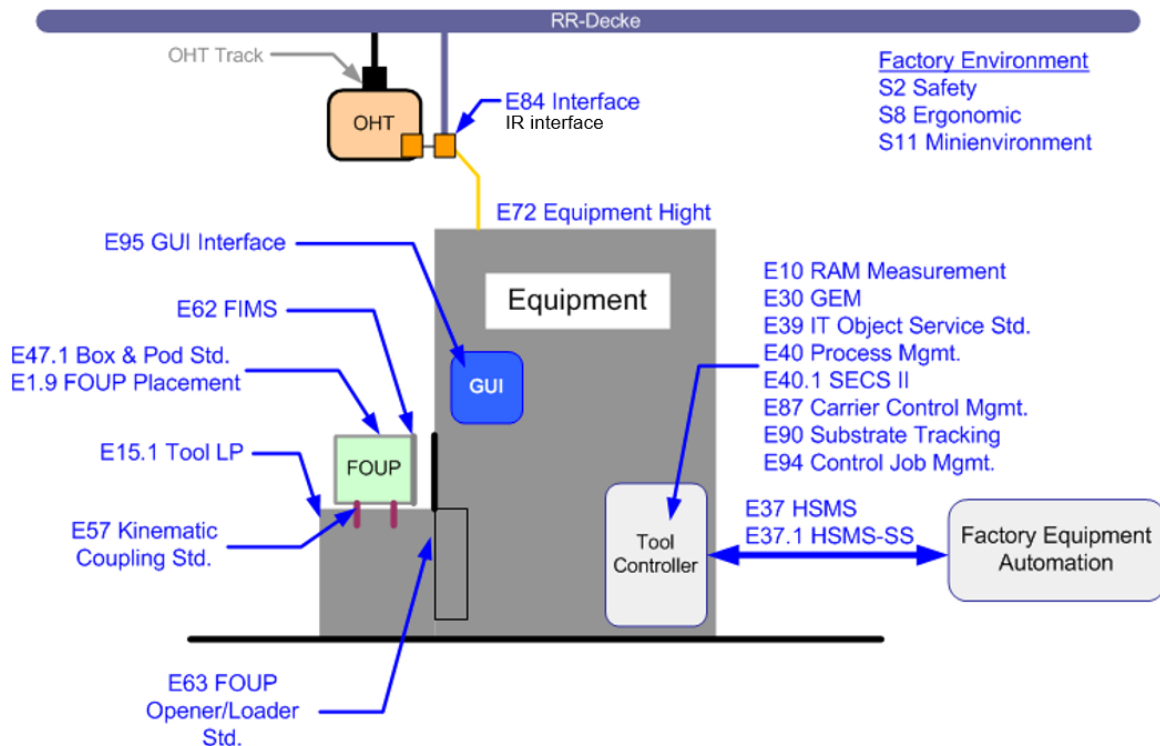


Figure 32 Main data interfaces (including SECS / GEM data interface for internal sensor data) acc. SEMI 300 specifications.

For the internal data flow, e.g., from internal sensor of a tool, E40.1 interface (SECS II) is the main interface to transfer the data from the equipment to advanced process control systems. But, more and more data are also used from sensors, which are not directly installed within the semiconductor equipment but also directly in the clean room, e.g. physical sensors or camera based systems which allows to control process parameter outside the tool. For those sensor systems an external sensor integration concept must be developed. In a wafer front-end factory, e.g. IFD, many sensors outside the equipment are already installed and integrated into the MES, but the effort of this additional integration is still high. Therefore, a new integration scheme based on an IoT framework is essential for plug and play sensor integration in the semiconductor industry. The main work for operations in the semiconductor industry is shown and explored within UC16, especially within the lead sub use case 16.2, where different sensor systems are placed on an I/O-link platform and connected to the IoT framework. Nevertheless, UC17 is about sensors in a modern building for development engineers of the semiconductor plant from Infineon Villach, where the sensor data are used for fully automated control of the building. Another important use case from automotive industry is UC15 from the Volvo Company. The automotive industry also uses complex manufacturing monitoring systems, and in this use case the data flow for smart kitting of the parts used in the factory is important. Another use case dealing with the data flow is from the metal industry: Boliden's UC18 with the scope of secure and stable data sharing in a metal company.

Main security issues for operations & management

For the use cases in operations and management, the following requirements are important regarding security:

- Save data exchange must be assured by either using internal IT infrastructure or by

using the Eclipse Arrowhead framework.

- Ontologies as meta-structure must allow data exchange and interoperability across heterogeneous sources and service clouds.
- Status and Measurement data formats must be defined, encrypted connection for X-site transfer.
- Data formats and interfaces to ISO legacy equipment must be defined.
- Secure authentication mechanisms available.
- Role definitions with gradual access rights defined.

Main Interoperability related aspects for operations and management

- Authentication services needed for the manufacturing equipment.
- No management of different roles.
- Data exchange must be performed through SECS/GEM or similar interfaces in a closed and secure network.
- Data exchange and legacy aspects kept.
- The legacy protocol of data exchange should be kept to support the integrated data architectures of the companies.
- For production related data at Boliden two security aspects are critical:
 - The data must be identifiable and use rights to be documented.
 - Data provisioning has to be based on agreed scope.
- Data are streamed from the field to the cloud must have secure encryption.
- Data injected from the field should be visualized on a legacy system, integrated with EAf.
- User authentication and authorization on dashboards and to use related back-end REST-API methods (use of tokens) must be assured.
- Dummy Instruction Insertion: dummy instructions should be inserted at random intervals into the execution pipeline.
- Enabled secure connection on MQTT broker by using SSL certificates and client authentication possible.

3.4.5.2. Advantages of using the Eclipse Arrowhead framework in this AHT-EP Phase

The main benefits for using the Arrowhead Framework and the tools which are developed in the project is to get simple plug and play solutions for sensor systems which are used in the different industrial applications. Today, the effort for sensor integration is very high (up to 3 months per single sensor integration). Therefore, the project should help to reduce these high efforts to a factor, which makes sensor integration compatible. On the other hand, all developed tools should be designed for multiple purposes in the factories to achieve the goals of interoperability of the tools and applications.

3.4.5.3. Use Case tasks and activities associated to the phase

In the following part the main industrial use cases in terms of operations and management from WP9 out of semiconductor industry (UC5 and UC16), building applications (UC17) and metal industry (UC18) are shown regarding their specific use case architectures.

The following use cases are related to the chapter Operations and Management:

- UC-12 <T8.7> Digital twins and structural monitoring
- UC-15 <T9.3> Smart Kitting to Manage High Diversity
- UC-16 <T9.4> Production Support, Energy Efficiency, Task Management, Data Analytics and Smart Maintenance
- UC-17 <T9.5> Linking Building Simulation to a Physical Building in Real-Time
- UC-18 <T9.6> Secure sharing of IoT generated data with partner ecosystem

Use Case UC-05

The engineering process for UC-05 (represented in Figure 21) shows the most important stakeholders involved in UC-05. UC-05 has 3 stakeholders, following the AHT-EP and 2 further Stakeholders, where the EP is unknown to us.

Main stakeholder is StkH-1, which is the algorithm developer, so, those, who are developing the two algorithms TePEX, WHF and the ontology DR.

The second stakeholder is the semiconductor industry, which is the end-user of the developed algorithms from StkH-1.

As a third stakeholder we mention here the Arrowhead Framework (AHF), which can on the one hand be the user of the product from StkH-1 and at the same time provider of this service for StkH-2. It can be seen as possible connection/service provider between StkH-1 and StkH-2 (c.f. 2.1.3.2 Digitalisation framework – WP3).

StkH-4 and StkH-5, for whom the EP is unknown to us, are for instance the supplier of semiconductor equipment and related IT infrastructure.

Use Case UC-16

UC-16 is a use case about sensor and data integration in front-end semiconductor manufacturing. This use case has the scope on production efficiency, energy data management and predictive maintenance applications. The engineering process for the UC-16 – based on the lead use case 16.2 "I/O-Link sub use case" – is shown in the Figure 33. There are many different stakeholders in this process, which are described the figure.

This use case architecture is very similar in semiconductor factories. And due to the fact that the stakeholder metric is very similar, it can be used for many different sensor and data integration topics.

StkH1 => Engineering and Development Department

StkH2 => IT Network Engineer

StkH3 => IT FI (Factory Integration)

StkH4 => Engineer for APC (Advanced Process Control)

StkH5 => Maintenance technician

StkH6 => Final user (Process Engineer)

StkH7 => Data Engineer

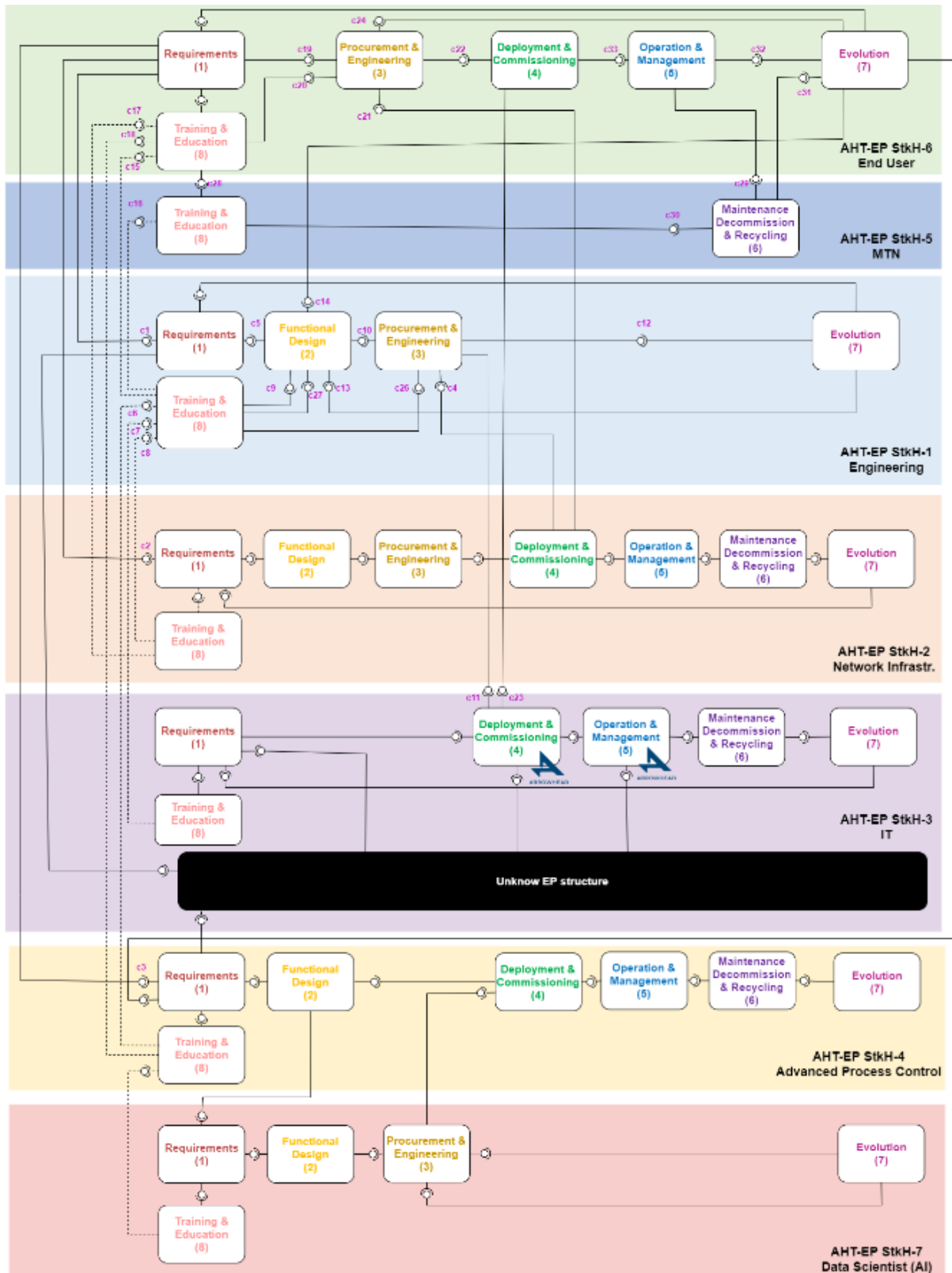


Figure 33 AHT-EP of use case UC-16

Phase 1 - First Solution

- c1 Stkh6 requires new sensor

- c17 Stkh6 get information about LAN Port request from Stkh2
- c18 Stkh6 get information about Key Value request from Stkh4
- c2 Skh6 req. new LAN PORT and IP
- c3 Stkh6 req. new key value extraction from sensor data and limit setup
- c4 Stkh2 provide LAN Port and IP
- c5 Stkh1 concept study from requirements
- c6,c7,c8,c9 get design rules and requirements for functional design
- c10 Stkh1 start engineering for first sensor implementation
- c11 Stkh1 configure the sensor for self registration and self deployment to StkH 3
- c12,c13 Stkh1 Check result and improve design if necessary
- c14 get feedback from Stkh6 and improve design
- c26,c27 create documentation
- c15 Stkh1 provide documentation to Stkh6
- c16 Stkh1 provide documentation to Stkh5

Phase 2 - Roll out solution to same kind of equipment (Toolpark of same supplier and same process). Stkh6 can do roll out by himself without engineering support from Stkh1.

- c2 Skh6 req. new LAN PORT and IP for multiple equipment
- c3 Stkh6 req. new key value extraction from sensor data and limit setup
- c19, c20 Stkh6 start sensor configuration as provided from Stkh1 (c15)
- c21 Stkh2 provide LAN Port's and IP's to Stkh6
- c22 Stkh6 has finished sensor configuration
- c33, c29 Stkh6 radvice installing a new sensor to Stkh5
- c30 Stkh5 get information how to install and maintain the sensor
- c23 sensors send self registration and deploy itself
- c24 Stkh1 check and improve sensor setup (repeat C22 , c23)
- c25 Stkh1 provide new requirements to StkH 4 (e.g. change limit setup, new key value calculation)
- c28 Stkh6 provide documentation and procedures to Stkh5
- c31,c32,Monitoring sensor data and trigger reaction on limit violation or sensor failure
- Not labeled connections are not part of the sensor integration process. They estimate the internal process of the different Stakeholders

Use Case UC-17

The schematic in Figure 34 shows the engineering process for the optimization process for the specific building use case UC-17.

Three stakeholders are involved:

- The developer of the building tracker (StkH1),
- The installer and / or planner of the building services (StkH2)
- The facility manager (StkH3) who is responsible for the operation of the building.

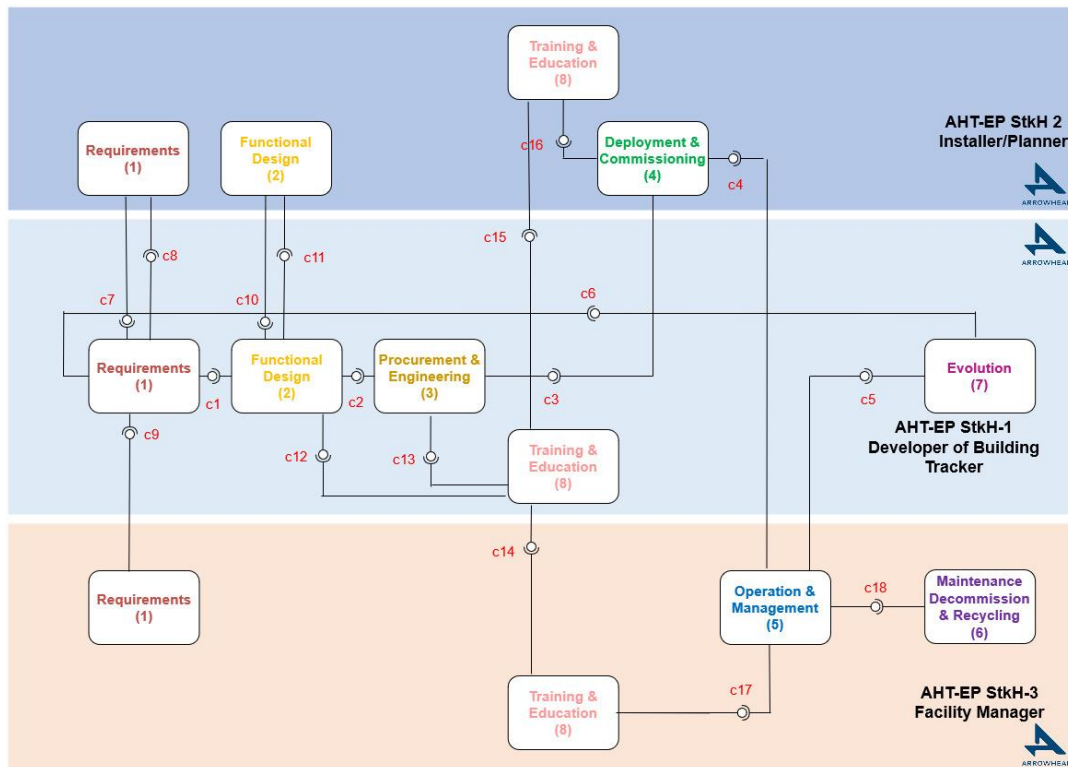


Figure 34 AHT-EP of use case UC-17

Use Case UC-18

For the metal industry UC-18 is defined by the company Boliden.

In general, three components are looked at in this use case with relative engineering process defined in Figure 35:

1. Data integration of a data source into the data platform requested by StkH3 and implemented with StkH1,2,4;
2. Security setup based on StkH1 requirements and details from StkH3 and
3. The data provision to StkH3 based on preparation by StkH2& 4.

The three components can be executed together or timely separated, e.g. if data source are integrated before actual usage.

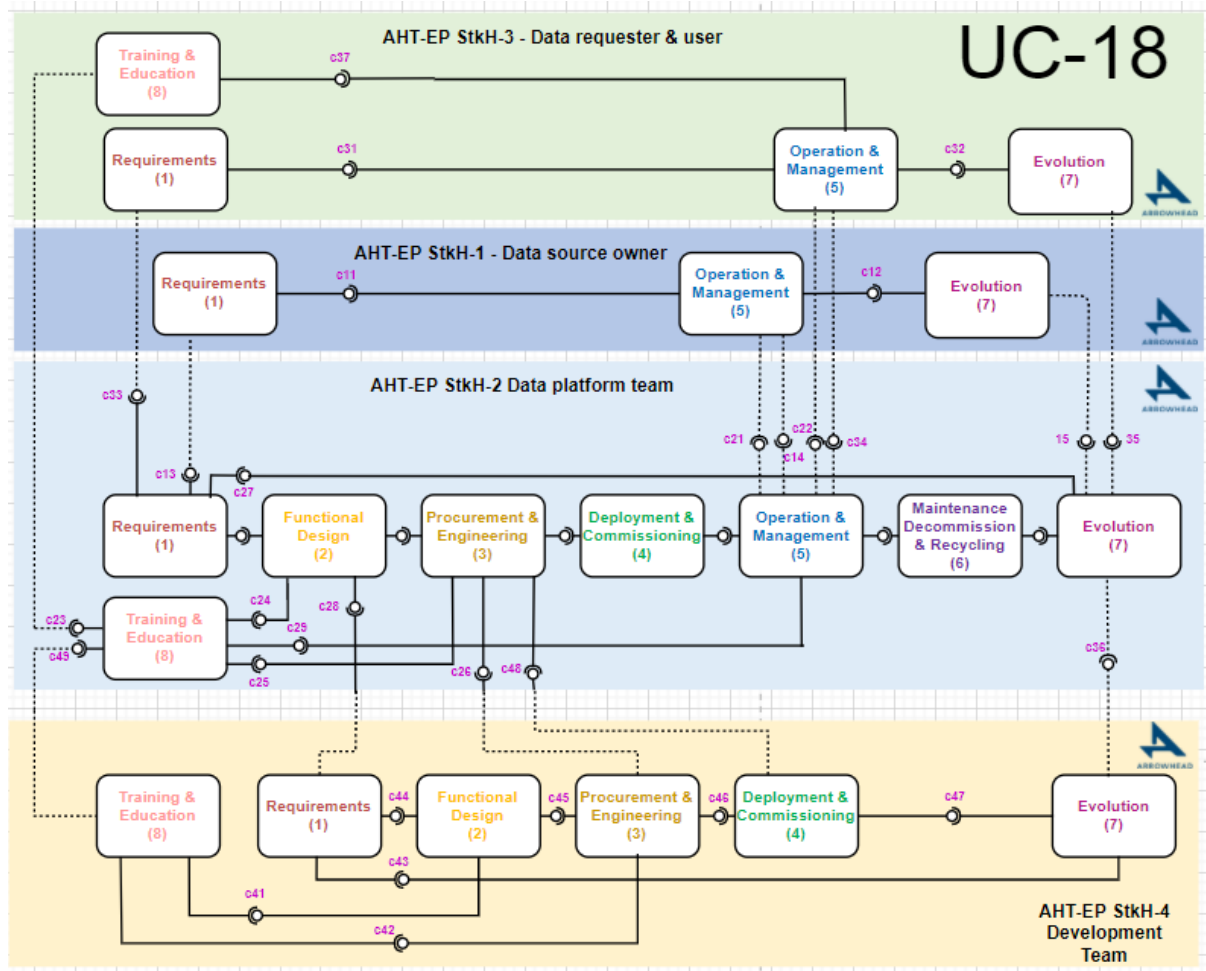


Figure 35 AHT-EP of use case UC-18

3.4.6. Maintenance, Decommissioning & Recycling (EPP6) < SubTask 6 >

Maintenance consists in identifying and establish requirements and tasks to be accomplished for achieving, restoring, and maintaining an operational capability for the life of the system/product. For a system/product to be sustained throughout its system life cycle, the maintenance process must be executed concurrently with the operations process. Maintenance addresses bug fixes and minor enhancements, as well as, minor adaptations to standard, new features, etc.. Significant changes in the system/product are considered in the Evolution phase.

The stakeholders will perform ordinary and predictive maintenance to achieve, restore, and maintain operational capability of the system.

Maintenance intervention are reported in details, all the modifications done on the real product are applied also on the digital version in case available, in order to ensure the high fidelity of the digital twin of the product.

In this engineering phase unit, we can also consider the decommissioning of the system/product at its end-of-life, as well as Recycling procedure of the components of the devices that are going to be retired.

3.4.6.1. Phase Description

A service-oriented architecture is an enterprise system based on existing software functionalities. These functionalities are considered services, which are developed by different organizations which can be useful for different domains.

Inside Industry 4.0, every domain will have one to many SOAs which everyone is composed by different services. Thus, the same service can be used in different SOAs.

By adopting a service-oriented architecture, it is possible to adapt the architecture based on the end-user needs. New services can be introduced, removed, or existing ones can evolve. This causes the architecture to evolve over time. That is why the maintenance of SOAs is an important part of the engineering phase. Due to incorrect maintenance, the services deployed in the architecture may fail, which causes a negative impact (among other money) in the industrial domains. Additionally, note that a service can be developed and updated by other organizations (stakeholders), thus, if a change is made in those services these can fail in the architecture deployed in the industrial domain.

This is why in the maintenance phase of the engineering process within Industry 4.0 there are three important points to consider:

- **Service maintenance**
- **Security maintenance**
- **Visualization maintenance**

Service maintenance

The service maintenance is related to the services which are used by other end-users. First of all, it is necessary to be sure about the update of the service since this one can affect many end-users. Thus, when a new version of a service is deployed before deploying that new service into the architecture, it is necessary to be sure of the implications that this may have.

There are many factors to be considered in order to maintain services:

- Healthiness:** it is necessary to monitor the services deployed in the SOA, in order to verify if the services are working properly.
- Deployed services:** it is necessary to check which services are deployed in the architecture and which is the version deployed.
- Track record:** Besides managing which services are deployed and their state, a service control system is important since thanks to it would be possible to have the traceability of the services deployed and the changes made on the SOA.
- License control:** before deploying the services into the corresponding architecture, it is necessary to verify if the concrete SOA has the licenses to deploy the new service. Consider that many services created for the industry can be associated with a usage license. The developer of the service must protect its service from improper use. Moreover, the user of the architecture itself must be sure that what is deploying contains the appropriate certification to be deployed.
- Service validation:** in addition to verifying if the services to deploy have the correct licenses, it is important to verify if the service deployed is suitable for that type of SOA. Otherwise, deploying an unsuitable service can cause fails in the system. That is why, before deploying is important to simulate the service in order to verify the services.

By verifying which are the healthiness, deployed services, the track record, having a license

control and making the specific verification, it would be possible to maintain the SOAs avoiding issues, which makes to maintain the SOA and the services healthily.

Security maintenance

Inside Industry 4.0 many IoT devices exist, which are connected to each other and are transferring information to different platforms, such as Amazon, Azure, etc. In this process is necessary to maintain a robust level of security when connecting to IoT platforms in the cloud. Thus, in an industrial environment where amount of devices exist, is important to verify who is sending information in order to have the traceability of what is happening. That is why every IoT device needs a certificate in order to have access to different cloud platforms. Thus, in the maintenance phase it is necessary to provide:

- **Check Certificates:** it is necessary to check if devices are using the correct certificates, otherwise it would not be possible to stream data to Cloud platforms.
- **Certificate Maintenance:** if an unsuitable certificate is being used, it is necessary to provide services able to update in a secure way those certificates. In this manner, the communications between the IoT devices and the Cloud Platforms will be established in a correct and secure way.

With security maintenance, it is possible to maintain different IoT devices deployed in different platforms in a secure way. Additionally, if a new device is introduced, it would be possible to enable that device in the system and provide the correct certificates in order to provide a secure way to connect with the cloud platforms.

Visualization maintenance

Another part of Industry 4.0 is the visualization, the information captured via different services then is visualized in different digital platforms. The main problem of these platforms is maintenance. These digital platforms evolve over time and they need to continue being accessible services in order to help the end-user.

Besides the necessity to upload new versions of the systems, it is necessary to prove the security of these digital platforms; i.e., during the different engineering phases of the development and maintenance of a digital platform some security requirements need to be fixed. Thus, it is important to manage the security of digital platforms overtime to ensure that these digital platforms are correctly developed and do not cause any issue to the end-user.

In addition to maintenance, in this engineering process phase there also are considered decommissioning and recycling procedures:

Decommissioning

Decommissioning is a process by which a business application or system is removed from use in an organization. In this phase a system is replaced by a new target system covering the same functionality, or the system is obsolete because it no longer supports the business process.

In Industry 4.0 is very important that OEMs (original equipment manufacturers) plan for decommissioning a device at the design stage. This enables end users and stakeholders to remove a device from the system securely and deploy a new one. It should ensure that a decommissioned product does not expose a vulnerability due to which the system gets exposed to security breaches after removal of an IoT device from the system.

Recycling

Recycling is closely related to the decommissioning procedure. When a hardware or system has to be retired starts the process of recycling the materials or components for reduce cost and take care of natural environment.

In Industry 4.0, a circular economy aims to redefine growth, focusing on positive society-wide benefits. It entails gradually decoupling economic activity from the consumption of finite resources and designing waste out of the system. Underpinned by a transition to renewable energy sources, the circular model builds economic, natural, and social capital.

The evolution phase can provide feedbacks to this phase with useful information for the management of the end-of-life (EOL) of the system or product. A product or system has reached its end-of-life when it can no longer fulfil its function and therefore has lost its functionality. The end-of-life phase varies considerably according to the product or system type under investigation. However, in general, an EOL product is a product that does not receive continuing support, either because existing support, evolution and other processes are terminated; or the product itself is at the end of its useful life. EOL management is much about economics how long it is cost-effective to evolve the product/system, and when it is time to substitute the old one with a new replacement.

3.4.6.2. Advantages of using the Eclipse Arrowhead framework in this AHT-EP Phase

Eclipse Arrowhead framework aims for digitalization and automation for the industry. During the implantation of such digitalization, the first challenge focuses on the automation of the main tasks of the companies, those related to the main business activities. Focusing on the business activities is the first level of digitalization maturity.

When a company advances in the digitalization maturity, the automation of other complementary activities gains importance. One of those supporting activities are related with the maintenance, decommissioning and recycling of the solution. These supporting activities need to be automatized because they should be realized continuously during operation and can lead to a degradation of the system or even worse, to unplanned stops. In such scenario, the framework is not only useful to orchestrate business related process but also is extensible to those related with the maintenance of the business solution. The maintenance activities that can be automatized and supported with Eclipse Arrowhead framework are:

- Creation and updating of the inventory of the business logic components.
- Monitoring of the health of the solution.
- Upgrading / rollback of parts of the solution.
- Logging of the main events of the production.

3.4.6.3. Use Case tasks and activities associated to the phase

In this section there is a brief review on some use cases that have some activities related to the maintenance, decommissioning and recycling phase.

Use Case UC-01

Automated formal verification (CAMEA)

This Use Case concentrates on one of the phases critical in the development of embedded devices and that is verification. More concretely, the use case primarily aims at systems developed for intelligent traffic surveillance in an SME (CAMEA in particular) that is currently not using latest advanced verification techniques at all. The aim is to improve the situation by allowing the company to use advanced verification techniques. In order to derive verification scenarios for intersection of several roads, a scaled-down (1/10) autonomous car provided by CVUT will be used.

The activities of this use case are involved in several engineering phases. Related to maintenance phase, if some change in the system is needed due to some malfunction, the changed system should be verified too.

Use Case UC-05

Support quick and reliable decision making in the semiconductor industry (KAI)

To support quick and reliable decision making in the semiconductor industry, three self-developed existing tools will be improved and implemented in the course the use case. They are TePEX, WHF and DR.

- TePEX (Test pattern extraction): An algorithm which is able to detect test patterns, which are related to malfunctioning testing equipment. With TePEX, malfunctioning wafer testing equipment is detected before the yield is affected.
- WHF (Wafer health factor): An algorithm which is able to detect process patterns, which are related to deviations during production. WHF is used to rate each wafer regarding its health. WHF is based on an ML-pipeline, which automatically detects and classifies each wafermap regarding pre-defined critical process patterns. With WHF, critical process patterns are detected at an early stage, before yield loss occurs.
- DR (Digital Reference): The Digital Reference is a Semantic Web Representation of the Supply Chain to guarantee interoperability as it creates an abstraction layer that defines concepts and relationships between heterogeneous data sources. The Digital Reference is a generic approach, allowing interconnectivity, to enable sharing and integration of information, data bases and tools.

In Figure 21 there is the AHT-EP schema for the use case, where it can see that the maintenance, decommissioning and recycling phase of AHT-EP from StkH-1 is involved in the algorithm developer (provider) stakeholder (c11 connection).

Regarding maintenance of such an application, in this case, an algorithm, one has to think about needed updating concepts in case of any influencing environmental changes. Further, also versioning concepts must be considered here.

Use Case UC-07

CNC machine automation (FAUT)

Customers of Fagor Automation usually spend one or two days parameterizing and tuning the machine axes. Customizing HMI is done with a proprietary tool based on legacy technologies on a computer.

Tuning tools are both located at the CNC or with a computer connected with DCOM protocols. Version tracking and adaptation between CNC and tools is usually a source of problems. The new tools should reduce by 50% the time needed by our customers. The new tools will follow the evergreen approach to cope with the versioning problem.

The current tools are mostly integrated in the CNC and as such are difficult to update (need to update the full code). The new tools will follow a modular approach and rely on open source standards for version managing if necessary. Moreover, for better interoperability, the tools will use semantics from standards, as OPC-UA and, more importantly, related companion standards like the already published by VDW for machine tools. This will improve integration with third party tools.

There is currently a CNC simulator for Fagor Automation's CNC. The system is very good for CNC programming training, but cannot be used easily with customizing and tuning tools. This simulator will be modified up to some extent with control algorithms as a platform where both intended tools of the project will be demonstrated.

In Figure 36 is the AHT-EP diagram of the use case. Focusing on maintenance, decommissioning and recycling phase, in the diagram above it can be seen that in the highlighted area there are some connections related to this phase. Specifically, c43 (CNC applications for machine health assessment), c53 (CNC applications to gather, process and display operational data coming from the peripherals, drives and the own CNC) and c61 (that represents data gathering of the machine status that can be further be related to other operational data and lead to improvements in maintenance and diagnosis).

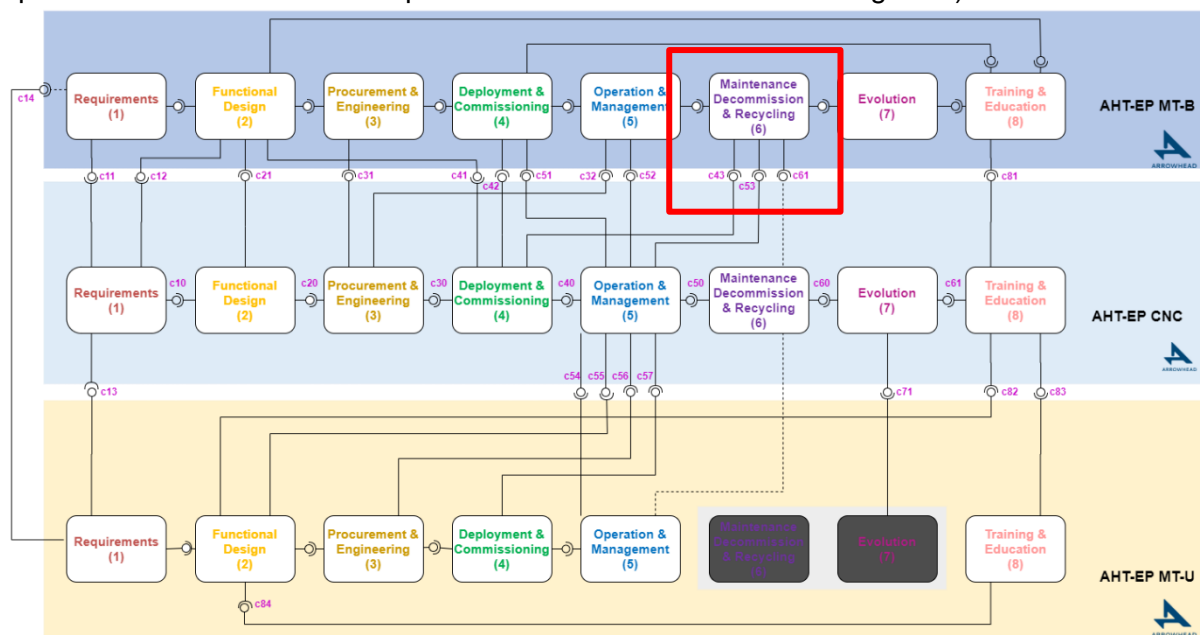


Figure 36 AHT-EP of use case UC-07

Use Case UC-08.1

SoS engineering of IoT edge devices: Smart City - Env. Monitoring (REPLY)

This use case will propose a highly pervasive sensing infrastructure must provide chemical, PM 2.5-10, noise, temperature, and weather data at sampling times of less than 1 minute and potentially with a spatial granularity of less than 100 m.

Such constellation of devices requires an extreme attention on power consumptions, that can be accomplished only by using next generation silicon sensors, energy-aware software applications and low-power, medium-to-long range wireless communications.

An IoT architecture must consider the environmental constraints where the sensors will be deployed.

The constellation of sensors and edge devices capable of collecting, processing and transmit data from the field will exploit wireless connections that preserve bandwidth, battery duration and extend the sensor's life spanning multiple years without maintenance.

Following the AHT-EP diagram for this use case (shown in Figure 37), where it can see that the maintenance, recycling and decommissioning phase is involved in the developer's stakeholder: In this phase measurement services, edge computing, Vital/IoT, Robofuse: Manual tracking of bugs discovered and reported by users are involved. In addition, dedicated resolution of reported bugs also are part of the maintenance phase.

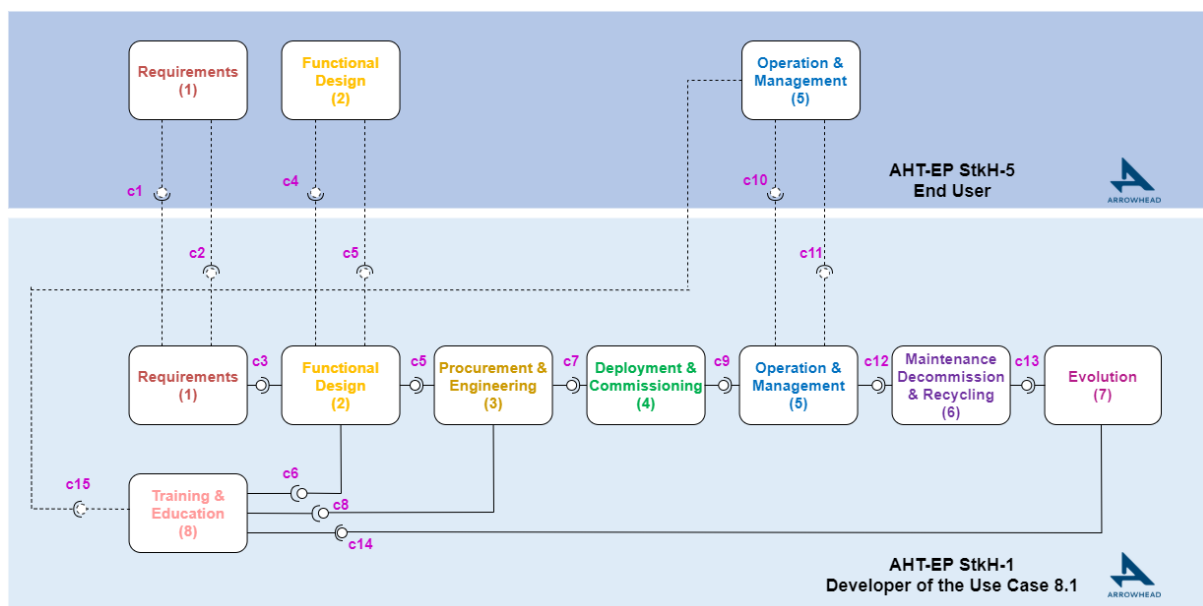


Figure 37 AHT-EP of use case UC-08.1

Use Case UC-20 Elastic Data Acquisition System (FARR)

The aim of the UC20 is to develop an Elastic Data Acquisition System that implements different services that will help in Deployment & Commissioning, Operation and Management and Maintenance, Decommissioning & Recycling engineering processes phases.

The system is composed of different elements:

- PLC: Siemens, Beckhoff
- DAS: .NET based custom software
- BBDD: SQL Server / Redis
- Custom: Custom applications developed by engineers or Data Scientists.
- Format: .NET based software prepared to format the data.
- Dispatch: .NET based software prepared to send the data.
- Cloud: Fagor Arrasate's IoT Platform.

With the DAS is possible to define: 1) the PLC variables that are going to be monitored and 2) with which protocol each variable is going to be captured. Then automatically the systems starts monitoring those variables and these ones are introduced in a database (BBDD).

The platform supports the deployment of Custom applications if needed such as machine learning applications, custom dashboards and so on. The function of the format is to convert the csv file to a json file with the PPMP specification. Finally the data is sent by the dispatcher with the correct format to the Cloud.

According to the Maintenance phase, a toolchain will be implemented related to application updates. This toolchain aims to reduce the deployment time need in order to update or introduce a new software in the industrial domain. To do so, a toolchain will be created, where the status of the already deployed systems is captured and then the toolchain is able to verify if any update is needed. If so, the toolchain will automatically deploy the new version and it will do the corresponding checks to verify that the operation has been done correctly.

The following diagram in Figure 38 indicates the engineering phases, toolchains and tools involved in this use case.

The remote-updates toolchain has activities in the maintenance, decommissioning and recycling EP and is composed by the next tools: NT7, NT8, NT9, NT10, NT11 and NT12. Above all the tools are described:

NT7 - Getting available versions for updating: In case of available software updates, this tool will enumerate them and will include the version of each one.

- Inputs: Installed applications.
- Outputs: Available software versions to be updated.
- Connected to AHF: Yes
- Phase: Maintenance, Decommissioning & Recycling.

NT8 - Selecting applications to be updated: Once this tool knows the installed applications and the available ones, it will indicate those installed applications whose version number is lower than the available version.

- Inputs: Installed applications and available applications to be updated.
- Outputs: Applications which could be updated.
- Connected to AHF: Yes
- Phase: Maintenance, Decommissioning & Recycling.

NT9 - Verifying compatibility: This tool will identify whether there is an incompatibility between the system and the application to be updated.

- Inputs: Applications which could be updated.
- Output: Applications which meet the requirement to be updated.
- Connected to AHF: Yes
- Phase: Maintenance, Decommissioning & Recycling.

NT10 - Verifying system requirement and download applications: It is necessary to check some PC requirements, such as disk space or RAM memory. Once verified that system meets the requirements, the tool will proceed to download the applications.

- Inputs: Applications to be updated.
- Outputs: In case of success, the downloaded installers. In addition, the tool will show a message indicating the founded problems.
- Connected to AHF: Yes
- Phase: Maintenance, Decommissioning & Recycling.

NT11 - Validation of new SW (simulation): Before installing the new deployed applications, it is recommended to carry out a previous simulation in order to verify the correct functioning of the system (for instance, using a digital twin).

- Inputs: The installers of new applications.
- Outputs: Simulation results. In case of success, this tool will indicate it to the tool in charge of installation.
- Connected to AHF: Yes
- Phase: Maintenance, Decommissioning & Recycling.

NT12 - Installation of applications: This tool will install the applications with the updates.

- Inputs: The installers of new applications and simulation results.
- Outputs: Deployed applications.
- Connected to AHF: Yes
- Phase: Maintenance, Decommissioning & Recycling.

Additionally, this process needs to be done in a secure way, i.e., the person who deploys the new service version needs to be authorized and furthermore, all services need to be uploaded and downloaded in a secure way.

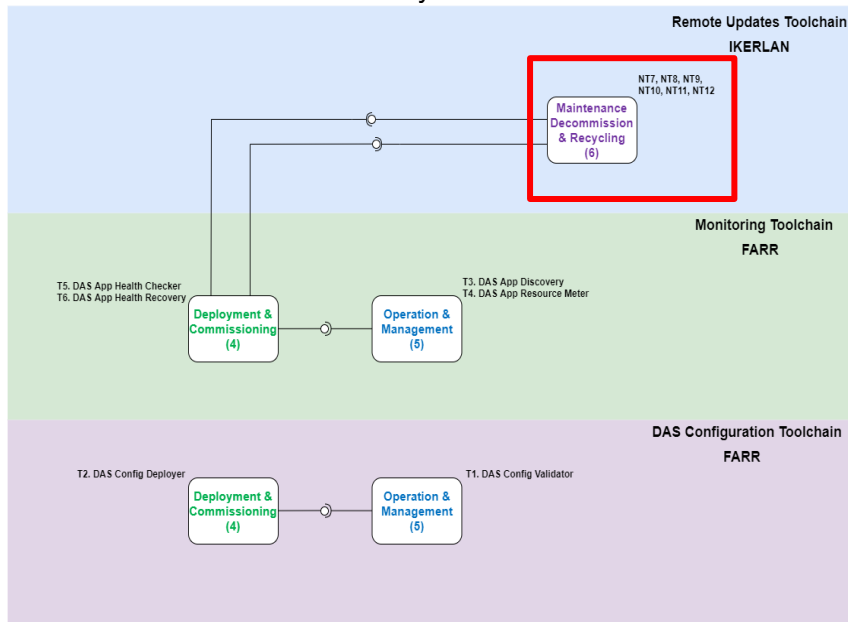


Figure 38 AHT-EP of use case UC-20

3.4.7. Evolution (EPP7) < SubTask 7 >

The evolution phase deals with the inability to predict how user requirements, market and technology trends will evolve a priori. The role of this phase is to monitor these aspects and identify potential significant changes in the future version of the system/products. The evolution phase must also ensure a continuous improvement of the system/product, always respecting the user requirements in an efficient, reliable and flexible way. Finally, evolution phase has to take into account various and alternating needs arising when dealing within different periods of lifetime starting from initial phase, following normal and final wear-out phases of the system and product, and send feedback towards engineering process other phases (e.g. requirements, product development, etc.).

All the information collected in the "operation & management" and "maintenance" phases are analyzed to identify solutions to faults/bugs, define the necessary updates and identify improvements that could bring to new product releases. In case a digital twin is available [49], it can be used to simulate and explore the effects of these updates and new releases. This will ensure the continuous evolution of the product.

3.4.7.1. Phase Description

The Arrowhead Tools project investigates and proposes extensions to the current automation engineering standards like IEC 81346, adding maintenance and evolution there. What is the

difference between maintenance and evolution? For example, maintenance of software focuses on bug fixes and other minor enhancements and software evolution deals with adaptation and migration. Evolution is a process where a system or product is changed during its lifetime in response to continually emerging or changing needs. From another viewpoint, evolution is a permanent condition for service-oriented systems.

The need for product/service evolution is due to the fact that one cannot predict how user requirements, market and technology trends will evolve a priori. That is, the existing systems are never complete and continue to evolve. The main objectives of product evolution are to ensure functional relevance, reliability and flexibility of the system. The goal of evolution is to adapt the system to the evolving operating environment or user requirements.

The ability to evolve enables also longer life cycles. For example, production machinery and the factory buildings life cycle is very much longer than the automation system and its parts in most cases. Hence, there are obvious needs for system maintenance and evolution of the automation to extend the lifetime of the product.

The impacts of this for the end-users are:

- An extended lifetime of production investments.
- Reduced costs for continuous evolution of automation and digitalization solution targeting production, e.g. flexibility, cost, environmental footprints, validation and deployment.

The evolution in engineering procedure will break the border between product development and maintenance.

Evolution in SOA environments

Evolution in Service-Oriented-Architecture (SOA) environments can be divided into different subsections. From fundamentals to service provider and consumer architecture, and tools to make applied software to be reconfigured based on the alternating needs. Furthermore, software evolution can be classified as static and dynamic in nature [50].

- In the static case, the running version is first shut down and then a new version is installed.
- In the dynamic evolution case, the behavior of the software is updated without breaking down the activities and if successfully executed, it is improving the software adaptability.

The service itself can be seen as a piece of routine(s) that provides to its consumer certain capabilities. According to SOA, the service has to be loosely coupled and be able to operate over network, if needed. The service evolution in SOA can be classified as shallow and deep ones. The shallow ones are affecting locally to restricted number of consumers whereas deep changes are larger and more challenging affecting operational level activities [50]. Evolution version compatibility between service consumer and provider as well as forward and backward compatibility concepts need to be considered during coherent evolution process.

Adaptation can also be part of the evolution. However, the principal difference between adaptation and evolution is that adaptation will not change the service itself, but evolution will. In software systems, there is also a description that is called System of Systems (SoS). They are dynamic and evolutionary in nature, and responding to developments in inter-connectivity, ubiquity and agile software [51]. In industrial implementations, different domains, need for interaction of legacy and new systems has to be taken into account, among others. In general,

it is seen that a combination of formal and informal approaches is required to specify SOA-SoS architecture. This should be done both statically and dynamically without compromising functional and non-functional requirements to provide multi-viewpoints for different stakeholders for decision making at various stages of the SoS life cycle [51].

Evolution process

The evolution phase of system or product engineering process consists of several steps (Figure 39). Each of these steps is realized with a modification process, which can, for example, be initiated with a Request for Change document (RFC).

- A. Change assessment of the requested change (as described in RFC) considering its impact on quality, functionality, surroundings etc. and its benefits, risks, urgency, and costs.
- B. Based on information of the first step, complete the change approval and scheduling
- C. Perform the changes, i.e. change development and deployment
- D. Review of the changes

The first step is the most complicated one, including the analysis of all the impacts produced by the change. Before the change assessment can be approved, it is also possible to perform a change prototyping to help the approval process.

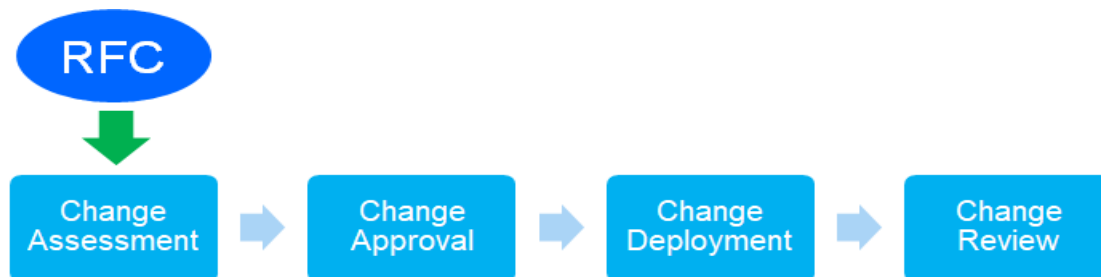


Figure 39 Evolution phase process in product or service engineering procedure

From the service standpoint, the evolution process cycle can be viewed through constantly evolving change detection, impact analysis and reaction evaluation phases [50]. In change management, it is essential to find out what are the required changes for attached services, and to understand the impact between the change and service. Naturally, procedure for how to react is needed. Finally evolution phase should have a feedback to the requirements for any reassessments related to the sequential engineering process phases, e.g. for new performance, quality, availability etc. aspects.

3.4.7.2. Advantages of using the Eclipse Arrowhead framework in this AHT-EP Phase

Typically automation and industrial IoT can face serious interoperability issues on the design level. Here, to support the engineering processes and phases, and for sure their evolution, Eclipse Arrowhead framework (EAF) is used for the service management. EAF relies on the SOA and it will aid and guide the developments of the automation and industrial IoT systems.

There can be multiple service providers and service consumers from multiple stakeholders communicating based on the service interaction definitions (service registry, authorization rules and <dynamic> orchestration rules) according to EAF and its core. It is seen suitable for service evolution: a given communication link may not be established by the hardwiring of certain resource instances together, but rather it relies on the loose coupling and late binding design patterns of the SOA. This is, giving the IoT systems the dynamic re-configurability to be able to interconnect systems on demand in runtime. In addition, with the AF based governing, mandatory core systems and additional supporting core system, the AF further aids the SOA based IoT system of systems by providing a secure and convenient way of interconnecting systems even from different clouds [52].

Linking different components together and making use of data from multilevel sources improve the building up and orchestrating of the new dynamically evolving systems. This empowers development of new services and business opportunities. The ecosystem of similar architectures can be linked both at the local and cloud levels. The created interoperability makes it possible to expand and multiply the existing systems when needed by keeping the common orchestration manageable.

3.4.7.3. Use Case tasks and activities associated to the phase

Next the activities related to EP Evolution phase in four use cases are presented. The use cases (UC) are UC8.3 SoS engineering of IoT edge devices: Smart City, UC10 Rapid HW development, prototyping, testing and evaluation, UC13 Deployment engine for production related sensor data and UC21 Digital twin evolution. The cases are presented in numerical order.

Use Case UC-8.3

SoS engineering of IoT edge devices: Smart City – CM

In UC8.3 engineering process (shown in Figure 40) there will be an evolution step taken during the change from design time to run time: In UC8.3 an IoT Integration Platform will provide feedback on the health monitoring infrastructure status that will allow to identify new functionalities or new releases of the system as a basement for evolution.

The evolution objective is in particular met in the evolution phase used by two stakeholders: Inertial-based WSN Provider (StkH 1) and Edge computing and IoT framework provider (StkH 3). The data produced by both the WoT sensor system and the Local Cloud Gateway is analyzed to identify need for improvement. Actually every year this might result into a new, evolved tool. Here the evolution outcome will provide design guidelines for the PMUT simulator (StkH 5). Detailed procedure descriptions can be seen from the use case UC8.3 document reported in the earlier published *WP12410_survey* and in the D2.2 Deliverable Appendix document [31].

StkHs 1 and 3 also make use of evolution engineering process phase because they are in charge of detection of any anomalies concerning the whole SoS and gathering data about the usage to trigger a new development cycle (i.e., they are the operative stakeholders).

Additionally, a toolset to support the operation and evolution of the sensing infrastructure is targeted. This is done by providing data to context sensitive optimization tools. The tools can possibly be provided by third parties. In particular, it is possible of feeding the configurator with

the best configuration according to some parameters. This can be done in order to maximize the revenue against the targeted objectives. This is done with the Optimizer tool, which takes in the input the parameters of the networks and outputs an optimized configuration.

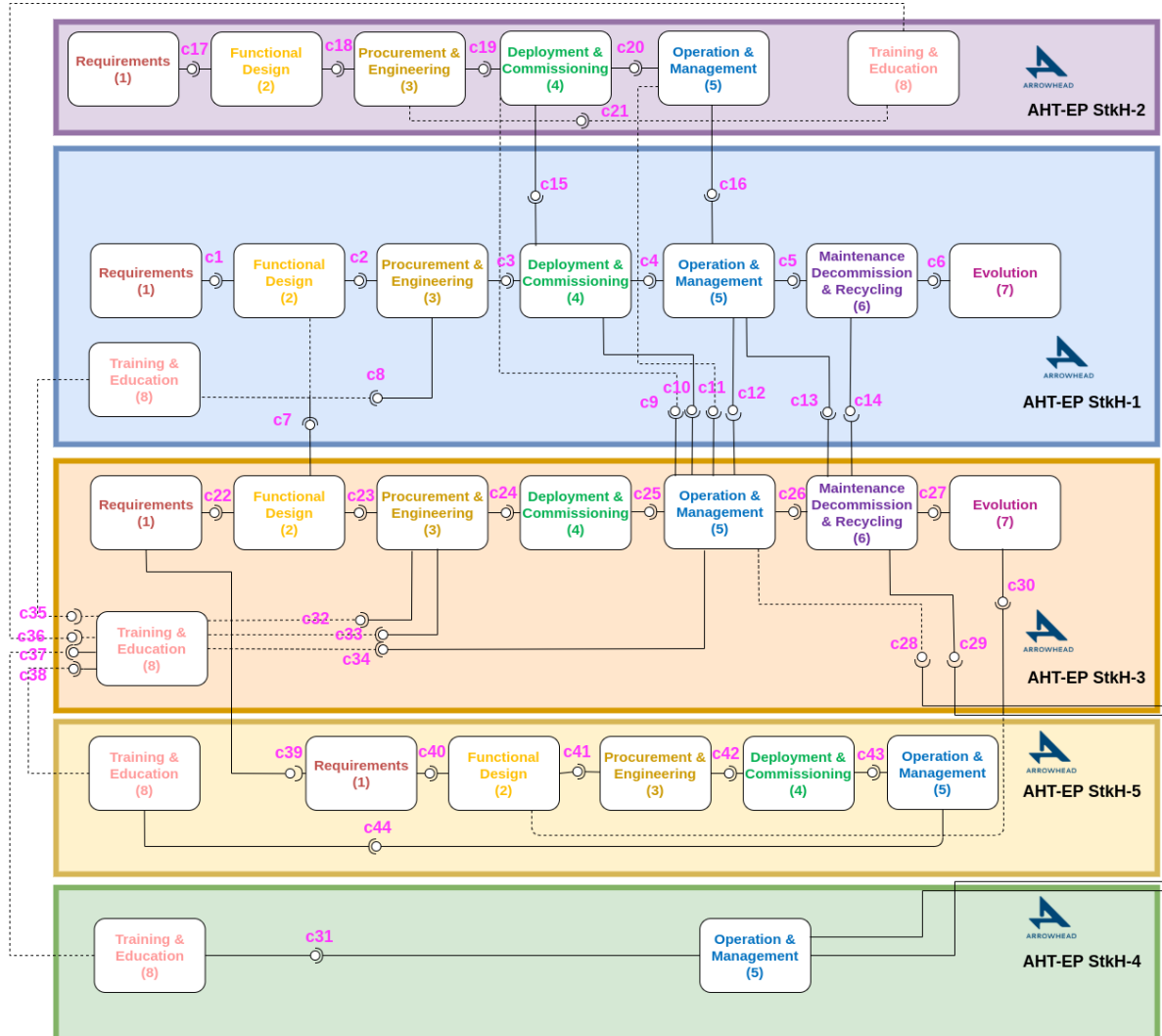


Figure 40 AHT-EP of use case UC-08.3

Use Case UC-10

Rapid HW development, prototyping, testing and evaluation

Also in UC-10 shown in Figure 26, there is an evolution phase employed during the change from design time to run time engineering. Here it means that the solution will go from laboratory environment to the industry environment of ARCELIK. Data will be collected in the field by engineering process operation and management phases by end user Company (ARCELIK, StkH 1). After the collection, the data will be evaluated by stakeholders for further improvement opportunities. In the evolution phase, the project input/output and the aims may be revised and fed into the requirements. Detailed procedure description can be seen from the use case UC10 document reported in the earlier published WP12410_survey.

Use Case UC-13

Deployment engine for production related sensor data

In evolution, the change from design time to run time engineering objective is valid in UC-13 (Figure 28). The defined transformation/transition can be changed in the engineering process during run time. Here the evolution phase resonates with all the incorporated stakeholders: Data destination responsible/requestor (StkH 1), Data source responsible (StkH 3) and Boliden integration box team (StkH 2). Additionally, within the StkH 2 its evolution phase is a provider (feedback) for its own requirements phase. Detailed procedure description can be seen from the use case UC13 document in the earlier published WP12410_survey.

In general, two components are looked at in this use case: 1) Boliden Integration Box (BIB) backend (BIBB) where "local survivability" is needed in case the internet line goes down; and 2) the BIB adaptor (BIBA) which is a specific interoperability implementation between the two systems. The setup of backend and configuration is done by the Boliden Integration Box Team internally. Also operation is with the BIB Team. Interaction is used with the data source and the destination responsible both in the maintenance and evolution phases.

Boliden Integration Box backend (BIBB):

- StkH 2 aggregates Evolution demands
- Input (StkH 2 - Maintenance): Improvement needs from maintenance
- Input (StkHs 1 & 3 - Evolution): Evolution needs by StkH 1 on data integration, e.g. sizing of BIBB
- Output (StkH 2 - Requirements): Improvement requirements for BIBB

Boliden Integration Box adapter (BIBA):

- StkH 2 aggregates evolution demands from StkH 1 and StkH 3
- Input (StkH 2 - Maintenance); Improvement needs from maintenance
- Input (StkH 1 - Evolution): Evolution needs by StkH 1 on data integration, e.g. adaptation for destination system upgrades affecting data end-points
- Input (StkH 3 - Evolution): Evolution needs by StkH 3, e.g. upgrade of source system affecting data end-points
- Output (StkH 2 - Requirements): Improvement requirements for BIBA

Use Case UC-21

Digital twin evolution

A digital twin is a software model acting as a digital replica of a physical product, process, or system. However, the twin cannot be a fixed entity, as it needs to evolve to match the evolution of its real counterpart. The digital twin is updated to match the experimental data from the physical counterpart along the entire life cycle of the physical asset. An over-the-life-cycle updated digital twin can be utilized e.g. for condition monitoring, optimizing the operation and performance of the product, and virtual testing of control or operation.

The objective in UC-21 (represented in Figure 41) is to configure and deploy analytic and light weight machine learning tools (StkHs: Elmer/HPC and ML) and to deploy them as digital twins for electrical machines (StkHs: e-machine and test bench) operation and maintenance monitoring (StkHs: ML and CBM) through IoT -services (StkH IoT). From the engineering procedure standpoint the core stakeholders are the electrical machines and the test bench used during the project for testing the operation and behavior of the machines. Other stakeholders are built around these. Detailed procedure description can be seen from the use case UC-21 document in the earlier published WP12410_survey.

In UC-21 the evolution phase represents natural improvements of the systems and their characteristics, and the following phases (training and education) are in practice represented by the documentation of the actions and lessons learned which targets further development/evolution.

The UC21 platform is build up on the core services of the Eclipse Arrowhead framework. During the SOA service evolution, dynamically orchestrated provider and consumer EAF services are implemented and tested in parallel with Data Acquisition (DA) and IoT Systems at edge, gateway and cloud levels and required data models. Process wise, data from a digital vibration sensor flows and is used within and outside a communication and processing gateway via MQTT/REST protocol, implemented with Python. Part of the evolution phase is detecting the need for changes/updates, and in UC-21 this has been seen mainly in the requirement for Dynamic, Advanced and Complex orchestration support in the EAF core (e.g. dynamic and automatic service pipelining). However, since this is not yet supported in the EAF it will be part of the evolution targeted in following phases of the project, following which it should also be validated.

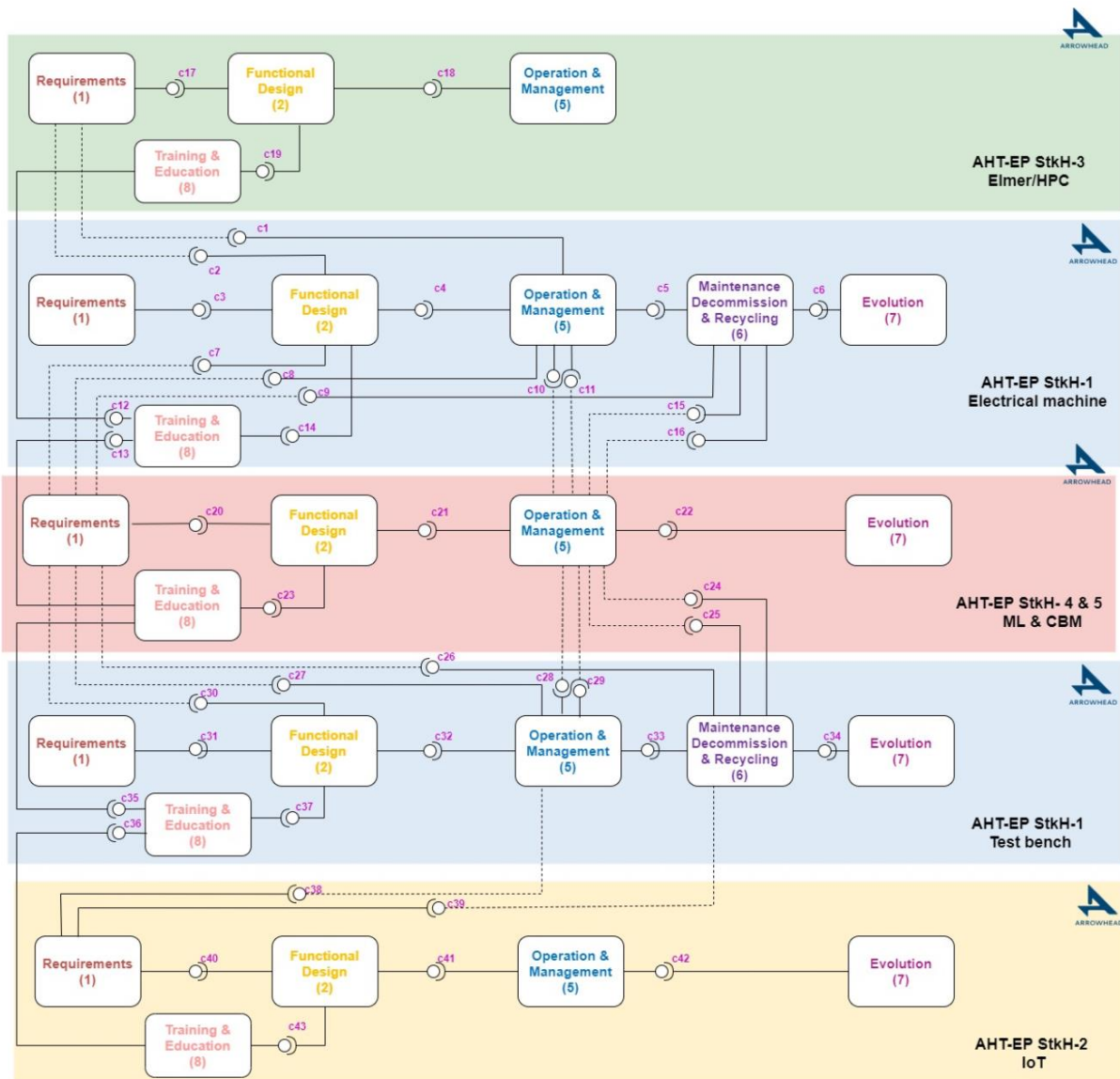


Figure 41 AHT-EP of use case UC-21

Evolution considering different life cycle phases and end-of-life

The product needs to be evolutionarily redesigned during its whole life cycle covering as a whole procurement, operation/use, maintenance, modernization and disposal phases. Thus, the evolutionarily redesigning is kind of a constantly running circular procedure that should be utilizing feedback from each of these different phases. For the optimization of the life-cycle evolution, e.g. the following questions and decisions need to be handled non-stop:

1. What kind of performance is required?
2. What kind of reliability is required?
3. What are the critical parts and how they can be identified?
4. When to maintain and where?
5. Replace or maintain?
6. How much to invest for the maintenance?
7. Modernize or change to a new, like the previous one?

8. When is time for disposal?

3.4.8. Training & Education (EPP8) < SubTask 8 >

This phase includes all the educational and professional training activities required by the engineering process, across the entire system/product life cycle. Source code documentation, how-to, installation manuals and training courses, together with demonstrators and development kits eventually using the power of the digital twin, will be provided to the stakeholders involved in the AHT-EP.

3.4.8.1. Phase Description

This phase occurs during the life cycle of a product through all the phases of the engineering process. It collects all the information relating to the materials available and created by the various technical stakeholders.

We talk about documents in the broad sense because it brings together user guides, presentations, webinars, videos, scientific papers, etc. Any element allowing a user to increase his theoretical and practical knowledge on a part of the Arrowhead Tools framework is considered as a Training & Education document.

In this phase are collected and managed all the documents relating to the technologies and methodologies resulting from all the engineering phases of the life cycle of the system produced. These documents are collected and managed during the AHT project by WP6.

During this phase it is also necessary to organize the documents by categories and levels of expertise in order to facilitate their use, or the production of specific courses according to the needs of a company.

The goal of Training Activity in the Arrowhead Project is to support and proactively guide the users of the Arrowhead technology, by providing the appropriate training material covering the engineering process, the SOA framework and associated tooling. Hence, this goal shall be achieved by structuring the reference material available and identifying the expectations regarding training material. It is considered that the targeted training material shall be made available for project internal and external usage, as well as for users of different levels of expertise.

In this perspective, we target the definition of conformity criteria and verification method to be applied for building consistent quality training material, as well as associated good practices. The definition of these conformity criteria and verification methods will be updated yearly along the course of the Arrowhead Tools project. The deliverable D6.1 [53] consists of the first iteration that have been updated in D6.2 [54].

The Arrowhead platform is integrated (as an open-source project) to the Eclipse IDE (Integrated Development Environment). The Eclipse IDE already offers a way to integrate an interactive tutorial for several of its project (CDT, JDT, GIT, etc.). Moreover, every Eclipse-based tool can take advantages of this integrated way to present the tool to the user and conduct him by following some ordered steps on how the tool should be used to avoid getting him lost from the beginning. In fact, the Eclipse platform main feature (*eclipse.platform.feature*) releases the *Cheatsheet plugin* along with the help, welcome page, documentation and many

other plugins to assist the client using the Eclipse tool. A cheat sheet is a kind of interactive tutorial.

In the following, we provide a list of the State-of-the-Art methods and technology that can be considered in terms of training material. The list is not necessarily exhaustive and will be completed if required along the course of the project in the WP6 activity.

- Blended Learning
- Teaching Packages
- Tutorial
- Training Videos
- Virtual Reality (VR) and Augmented Reality (AR)
- Workshops

AHT is proposing to constitute an entry point for accessing training material in the scope of the project, for project-internal dissemination as well as for external dissemination. This requires to characterize and index training material. The resulting cartography aims to be used for positioning the available training material, as well as identifying needs for training material not yet available.

The quality criteria required for trainings to be integrated into the Arrowhead Tools context in a consistent manner will have to be defined and standardized, as well as the corresponding verification methods:

- Common criteria
- Criteria for SOA Framework training and support material
- Criteria for tool chain architecture training and support material
- Criteria for tools usage training and support material
- Verification methods

3.4.8.2. Advantages of using the Eclipse Arrowhead framework in this AHT-EP Phase

The Eclipse Arrowhead framework (EAf) targets digitalization and automation for industry by offering a whole environment of innovative tools. The propagation and deployment of such tools within companies constitutes a major challenge at the technical level, but also at the human level. It is necessary to undertake training, upgrading, or specialization initiatives with the various groups of employees concerned.

The Arrowhead tools project took into account the notion of training by devoting a large part of the effort to this subject, either through a complete WP (WP6) or in a disseminated and integrated way with the technical developments of other WPs.

The advantage of using the EAf in this AHT-EP phase is that the very technology of AHT makes it possible to build evolutionary trainings linked to all the technological bricks or methodologies implemented in the project.

It is thus possible for those responsible for industrial training to draw from the entire repository of training material (documents, videos, sessions, presentations, etc.) in order either to follow a predefined training program or to build a program specific to their needs.

The modularity of the EAF bricks allows for easy evolution and maintenance of training programs. We can also imagine setting up traceability mechanisms, using AHT connectors, to establish intelligent links with the other phases of the engineering procedure.

A specific use case (UC22) has also been specially developed to demonstrate the usability of the EAF on a textbook case. It will either be an easy point of entry into technology, or train future students of AHT technology, or represent a demonstrator accessible to all types of public concerned.

3.4.8.3. Use Case tasks and activities associated to the phase

In this section there is a brief review on some use cases that have some activities related to the training and education phase.

Use Case UC-02

Engineering processes and tool chains for digitalized and networked diagnostic imaging (PHC)

EPP Training & Education: The pTX chain optimizer on the Eclipse Arrowhead framework should be, if this all works as expected, be a good starting point for further improvements of the design process of other parts and chains of the MRI system. Therefore training how to use and extend the framework for other applications is important to inspire and allow others to adopt this. The MR application engineers are responsible for the training and education of the end users, to use the coil and provided scan techniques most effectively to satisfy the customer needs.

Training & Education in the engineering process: In the AHT-EP schema, in Figure 10, it can be seen that the training phase is on output (c44) of Deployment and Commissioning phase, as an input (c52) to Operation and Management.

Obj. 4 - Address digital learning and training activities as an integral part of the engineering cycle: Each AHT-EP considers the development of documentation and reference manuals. The StkH 4, which will develop the user application, is in charge of producing tutorials and interactive tutorials for final users (StkH 5). These are also very useful for the internal stakeholders to get a better understanding of the applicability of their output.

Use Case UC-03

Integration of electronic design automation tools with product lifecycle tools (ULMA)

EPP Training & Education: Tool associated with phase Traceability Studio

Training & Education in the engineering process: In this UC, the training and education phase is an input at several level of the process, as shown in Figure 42.

Obj. 4 - Address digital learning and training activities as an integral part of the engineering cycle: Each AHT-EP and each stakeholder considers the development of documentation and reference manuals.

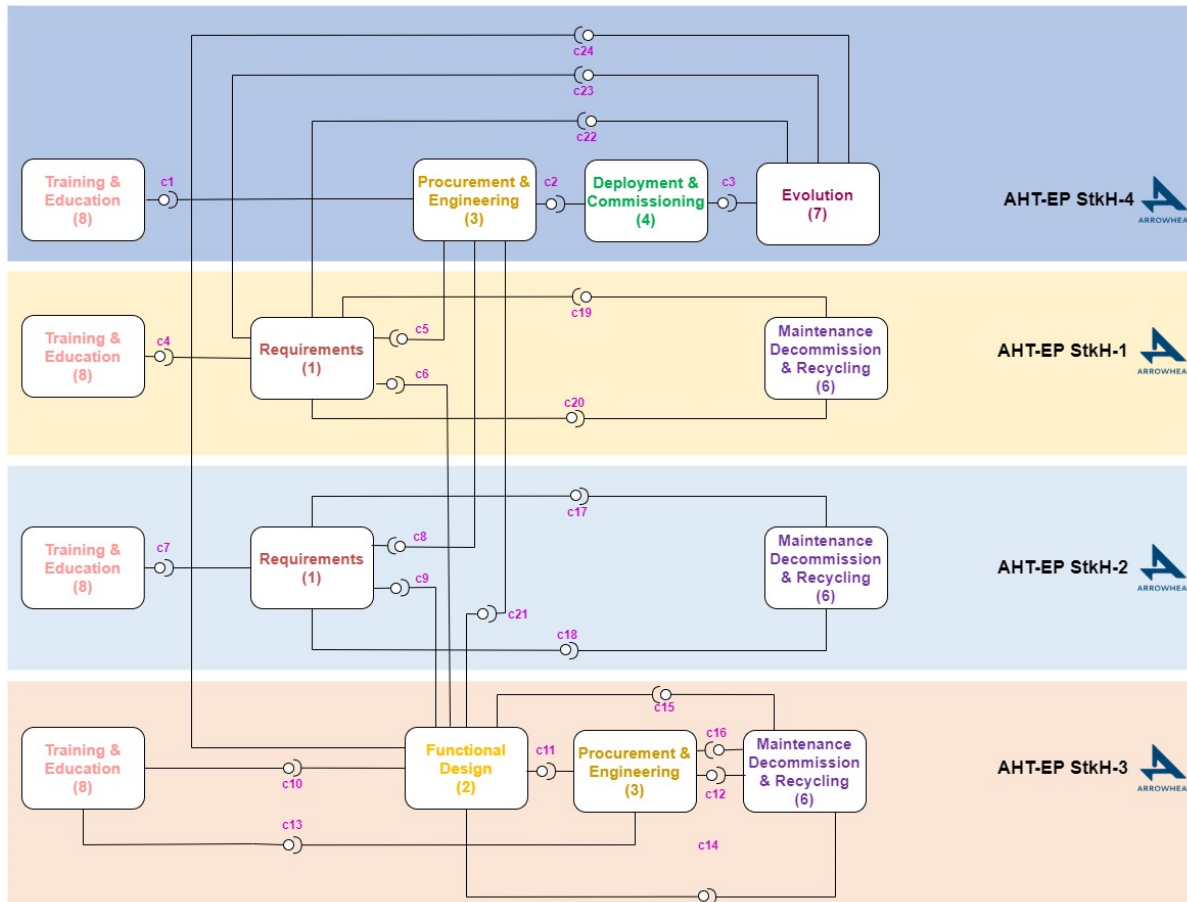


Figure 42 AHT-EP of use case UC-03

Use Case UC-05

Support quick and reliable decision making in the semiconductor industry (KAI)

Training & Education in the engineering process: EPP Connections c14 in Figure 21 represent the connection through which information on the developed algorithm is documented by StkH-1. This means, the purpose, the used data, meta-data and structure, tips&tricks, the usage, or simply, what it can/what it doesn't can is documented, for instance in form of a user manual or handbook, which provides important information to StkH-2, the end user of the developed algorithms.

Training & Education in the engineering process: In this UC, the training and education phase is an output, as the last element of the process chain, as depicted in in Figure 21 diagram.

Obj. 4 - Address digital learning and training activities as an integral part of the engineering cycle: Each EP from every AHT Stakeholder includes phase 8, the Training & Education phase. For StkH-1 for instance, this means to document the functionalities of the developed algorithms and to provide a user manual, including recommendations on the usage and highlighting the purpose of the algorithms, for instance, the application area, the reason what it is made for, but also mentioning possible limitations. This is especially important to guarantee a satisfying and reliable outcome of the analysis. This user manual is handed over to StkH-2. There, it might has to be adapted or extended dependent on additional information which might be important for the end-user, or other software, which might be related to the one provided by StkH-1.

Use Case UC-06

Production preparation tool chain integration (LIND)

Training & Education in the engineering process: In this UC, the training and education phase is an output of the evolution phase accordingly to the AHT-EP proposed in Figure 12.

Obj. 4 - Address digital learning and training activities as an integral part of the engineering cycle: This use case address digital learning and training activities as an integral part of the engineering cycle: to generate a paradigm shift of an interconnected set of service-oriented tools across multiple stakeholders as well as the introduction of new tools has required learning and training activities. This ranged from Vertex BD to the Eclipse Arrowhead framework. They have not been all well documented to make them general activities across the project and beyond.

Use Case UC-07

CNC machine automation (FAUT)

AHT_Obj. 6 - Training material (HW and SW) for professional engineers: Compatibility paths for data import/export to proprietary standard engineering tools (eg: commands for Matlab import or export). Data visualization tool (oscilloscope, etc) tied to free available engineering tools (eg: octave). CNC Simulation tool available for training in CNC programming or tuning.

Training & Education in the engineering process: In the AHT-EP represented in Figure 36, the c81, c82 and c83 (along some others not numbered) try to express the cooperation needed to configure a simulation environment (digital twin for specific processes) with information provided by the MT-builder (machine drawings, kinematics...) and the MT-user (tools, parts) where the application is provided by the CNC manufacturer. The resulting application can be used for training & education (c83) but also as the design tool of the MT-user (c84).

The MTBld can use the CNC Simulator incorporating the kinematics and drawings of its own machine. This produces, in fact, a digital twin useful for the programming of the machine and, much important, to detect collision between machine and part, etc. The simulator is in this case very helpful. Many licenses are today sold for training and education in CNC programming. Regarding this, the basic version is freely downloadable, and is used at professional schools. The look and feel is just as that of the CNC, what improves familiarity for students.

Obj. 4 - Address digital learning and training activities as an integral part of the engineering cycle: The toolchains developed will be used also in education and training. The Tuning tool can be a great education tool. Real data from machines will be read from the tool and the junior engineer will try to guess the best control parameters. The CNC simulator is already a very good tool for education. During the project, the new editors will include an operations editor that includes technology help.

Use Case UC-08.1

SoS engineering of IoT edge devices: Smart City - Env. Monitoring (REPLY)

AHT_Obj. 6 - Training material (HW and SW) for professional engineers: Measurements services: A few training materials are available. The sensors are provided with a short instruction showing how to install and configure the device and some examples to get and send data. Edge: Documentation of EdgeX Foundry and K3S are the main source of the training material for this part. Edgex Foundry training material is available on the Web. Vital-IoT: Documentation is available from previous projects about Vital-IoT architecture, configuration and integration patterns. Online documentation is available about the integration

patterns of Arrowhead. Robofuse: Documentation is available from previous projects about Robofuse architecture, configuration and integration patterns.

Training & Education in the engineering process: In the Operation & Management phase represented in Figure 37, the end user takes part in the phase of Operation & Management, in the sense that, after a phase of Training & Education on the documents and material provided by the Development Team, the end user starts owning and managing the deployed platform. The phase of Training & Education is a mostly manual phase, fed by inputs coming from the "Functional Design", "Procurement & Engineering" and "Evolution phase", whose primary output is to train people of the End Customer who have a role in the "Operation & Management" phase on his side, but whose additional output is our Activity of Education and Dissemination.

Obj. 4 - Address digital learning and training activities as an integral part of the engineering cycle: This goal is achieved through the documentation provided during the project at the end of every phase, usable in eventual training activities.

Use Case UC-08.2

SoS engineering of IoT edge devices: Smart City - AI driven Env. Monitoring (IUNET)

AHT_Obj. 6 - Training material (HW and SW) for professional engineers: The full documentation of the source code will be available for supporting the developers who are in charge of integrating new legacy infrastructures. The end users will be provided with a short description for deploying the consumer application in their working stations, in addition to the API documentation of the provider application.

Training & Education in the engineering process: StkH1 of the AHT-EP in Figure 22 create the datasheet of Ai-Camera PCB and reference manual of low level API. StkH3 will document the code to be shared with the StkH1.

Obj. 4 - Address digital learning and training activities as an integral part of the engineering cycle: StkH1 and StkH3 will improve the automation level of the tools adopted for the editing of the documentation of the IoT integration platform and, partially, also of the use case (including: source code documentation, technical documentation, user manuals, application manuals, etc.). All the three StkHs will provide user guides and manuals. Moreover, interactive tutorials for installer technicians (StkH1) and final users (StkH2).

Use Case UC-08.3

SoS engineering of IoT edge devices: Smart City Condition monitoring (IUNET)

AHT_Obj. 6 - Training material (HW and SW) for professional engineers: At the use case level, the contribution will be mainly derived from the proposed tool chain. For StkH3, the following training material will be provided:

- demo showing the functionalities of the infrastructure health monitoring
- code documentation as part of the Eclipse Kura and Kapua projects
- user manuals for the developer
- documentation of the adapters required to integrate legacy tools with the AF

Training & Education in the engineering process: Regarding Training of AHT-EP shown in Figure 40, we can observe how in most cases training comes from EPP3 of the respective AHT-EP, which represents the actual instruction manual produced while implementing/assembling the artifact(s). Also, it is notable how all the training activities are interacting among each other (seminars and tutorials) between stakeholders and they are all sinking into stkh3 training which feeds into EPP3 of its AHT-EP. This happens because the Local Cloud Gateway is the final aggregator of the data and inputs from the other artifacts.

Local Cloud Gateway: Generation of design and source code documentation. Editing of technical and user manuals.

Obj. 4 - Address digital learning and training activities as an integral part of the engineering cycle: The software part of the Local Cloud Gateway is partially already automated: the source code documentation is generated automatically. SDK and Demo Kit of the IoT Integration Platform are not automatically generated, but the release of the platform in the kit follows the main development stream, therefore is always up to date. In general, many of the EEP3 phases output a user manual for their artifacts. Moreover, the Local Cloud Gateway is expected to produce training material in the form of tutorials to the final user who should be able to navigate the interface.

Use Case UC-10

Rapid HW development, prototyping, testing and evaluation (ARCELIK)

Obj. 6 - Training material (HW and SW) for professional engineers: We will consider the internal application notes used by UTIA, EDI and ARCELIK as candidates for public training material.

Training & Education in the engineering process: Embedded Zynq Ultrascale+ Unit (ZynqU+): StkH2 and StkH3 in Figure 26 will create the datasheet of ZynqU+ system and reference manual of low level SW API for A53 CPU running Debian embedded Linux OS. FMC A/D (A/D): StkH2 create the datasheet of A/D integration in ZynqU+ and reference manual of low level SW API for A53 CPU running Debian embedded Linux OS. Relays (Relay): StkH-1 create the datasheet of Relay and reference manual of low level SW API for STM32 MPU. FPGA Control Unit for Power/Load (FPGA): StkH4 and StkH5 will create the datasheet of FPGA system and reference manual of low level SW API for MicroBlaze soft core running in FPGA.

Remote reconfiguration of FPGA (RemCtrl): StkH4 and StkH-5 will create the datasheet of RemCtrl system and reference manual of low level communication protocol description. Accelerated digital design on multiple PCs (ParDesign): StkH-3 will create the datasheet of Design acceleration flow and reference manual describing configuration of AH framework for ParDesign acceleration on multiple PCs in a local cloud for StkH-1.

Obj. 4 - Address digital learning and training activities as an integral part of the engineering cycle: Each AHT-EP considers the development of documentation and reference manuals. The StkH-3, which will develop the user application, is in charge of producing application notes and user manuals for operators in the industry environment (StkH-1). Also, the operator (StkH-1) is in charge of the operation manual that describes the operation and the maintenance of the end product.

Use Case UC-13

Deployment engine for production related sensor data (BOLIDEN)

Training & Education in the engineering process: In the AHT-EP represented in Figure 28, related to training and education, we have:

- EPP8: StkH-2 documents backends information for EPP-5 in Wiki, Output <c26> Relevant documentation and training for SH2 in operations & management all stored in Wiki
- EPP5: StkH-2 operated BIBA: Input: <c26> training information needed to operate BIBA stored in wiki
- EPP8: StkH-2 documents adapter information for EPP-5 in Wiki: Input <c24>:Design documentation and information

StkH-2 educated data users on data usage possibility, limitation and duties. StkH-2 creates needed documentation for instance deployment.

Obj. 4 - Address digital learning and training activities as an integral part of the engineering cycle: The objective is supported as consolidated material is placed in a wiki available for all team members and part of all engineering phases.

Use Case UC-16.4 Smart Maintenance (IFD-TUD)

AHT_Obj. 6 - Training material (HW and SW) for professional engineers: Training activities for maintenance experts and end users regarding the correct usage of the smart maintenance platform, e.g., how to correctly input kinematic data, will be planned and executed. Guidelines regarding the correct installation of measurement equipment and sensors to retrofit additional machines will be compiled and provisioned. The further automation of the diagnosis process including automated maintenance suggestions will reduce the required training & education effort.

Training & Education in the engineering process: The AHT-EP is described in Figure 43. StkH-1: TU Dresden; responsible for the design and implementation of the measurement FOUF toolchain as well as for the provision of the required services, additional software and hardware. Once an iteration of development is complete, StkH-2 and StkH-3 are provided with manuals and training how to configure the tools, execute measurements and interpret the data (StkH-3) and what is required to enable the integration of the toolchain within the fabs IT infrastructure (StkH-2), e.g., configure WLAN Access, preparation of external tools and data sources.

StkH-2: Infineon Dresden (IFD) IT integrates (deploys) the finished toolchain within the fabs IT landscape based on the training.

StkH-3: IFD Maintenance and Automation department; uses the toolchain for measurement and maintains the HW systems (e.g., battery charge) based on the training, collects feedback and enhancement suggestions (evolution) which is the basis for the refinement of the requirements as input for StkH-1.

Different stakeholders have to cope with the software, reaching from operation personal over diagnosis experts to managers. This requires training material for each kind of user.

Obj. 4 - Address digital learning and training activities as an integral part of the engineering cycle: By improving the automated machine diagnosis methods, much less expert knowledge is required to interpret the results. Smart displays on sensor nodes and –hubs could also provide digital and interactive guidelines for the installation and positioning, which also reduces the effort within the Deployment & Commissioning phase. By adopting the AH-EP, a policy is in place to enforce an efficient and exhaustive user training and education for the new, automated products

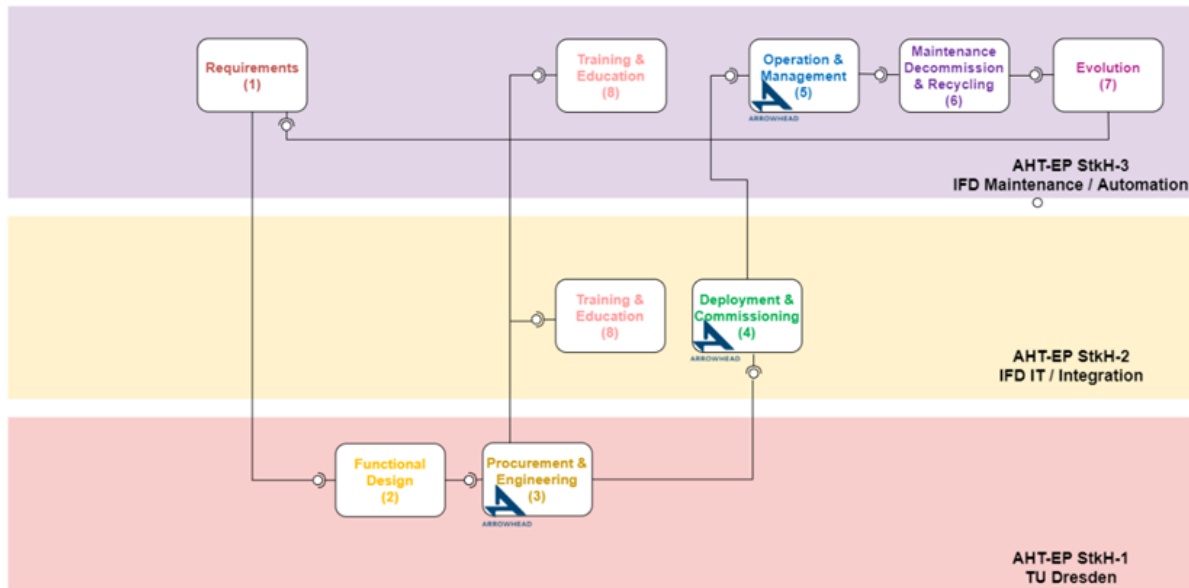


Figure 43 AHT-EP of use case UC-16.5

Use Case UC-16.5 Tester Integration (IFD-TUD)

AHT_Obj. 6 - Training material (HW and SW) for professional engineers: Training activities for maintenance experts and end users regarding the correct usage of the smart maintenance platform, e.g., how to correctly input kinematic data, will be planned and executed. Guidelines regarding the correct installation of measurement equipment and sensors to retrofit additional machines will be compiled and provisioned. The further automation of the diagnosis process including automated maintenance suggestions will reduce the required training & education effort.

Training & Education in the engineering process: In the AHT-EP shown in Figure 44: C5: StkH1 generates material for training and installation guides for reproducibility of the engineering process. C7: StkH2 uses the training material.

Create training material in form of a presentation or a Wiki page. Inform relevant groups i.e. Maintenance and process engineers of the production department. Users that have to do with sensor integration, Databases or Equipment integration. Train 24*7 service. Train maintenance technicians in shift.

Obj. 4 - Address digital learning and training activities as an integral part of the engineering cycle: In this use case will be produced some hand books and user guides including simple examples to describe the usage of EAF and how to integrate it into a running system.

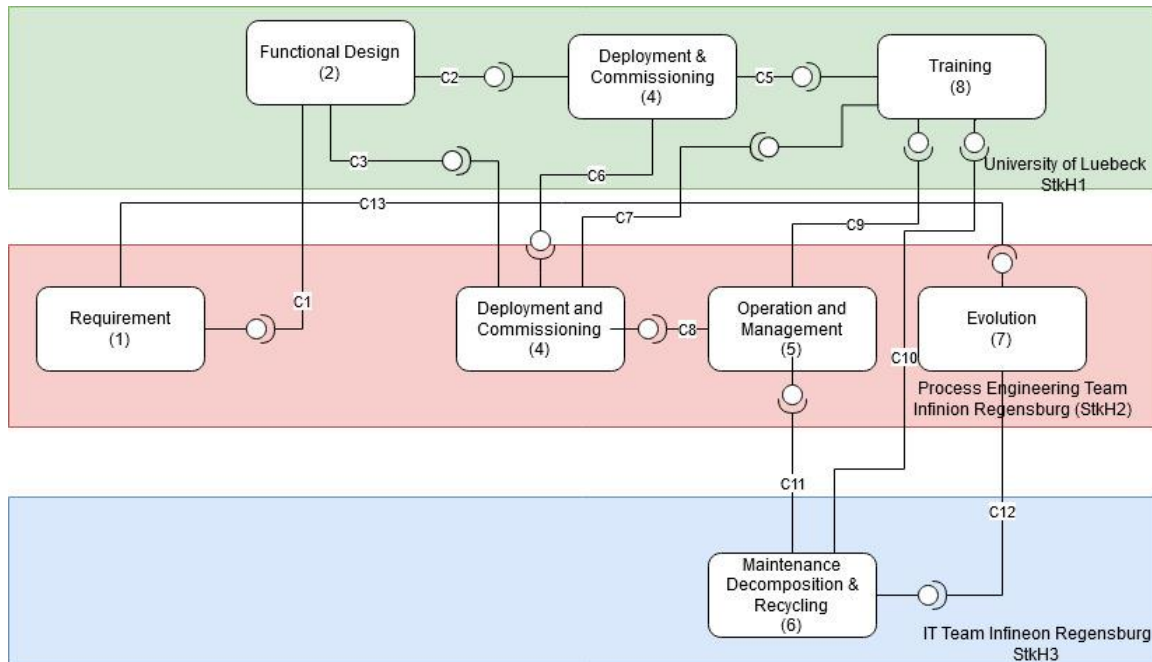


Figure 44 AHT-EP of use case UC-16.6

Use Case UC-17

Linking a Building Simulator to a Physical Building in Real-Time (AEE INTEC)

AHT_Obj. 6 - Training material (HW and SW) for professional engineers: Use case will prepare training materials for the developed building tracking system.

Training & Education in the engineering process: The operation of the system is the task of StkH3 (Facility Management) shown in Figure 34. The preparation of training material and conduction of training courses is done by STkH1 who trains the other stakeholders (c14 and c15). The development of training materials is based on information from the functional design (c12) and the procurement and engineering (c13) phases.

Training material to be developed by StkH1 to train StkH2 and StkH3.

Obj. 4 - Address digital learning and training activities as an integral part of the engineering cycle: Training activities are in integral part of the use case because it is important to transfer the know-how of the building tracking system from the developer of the building tracker to the other stakeholders. In the future (after a number of buildings has been successfully been equipped with a building tracking system), some of the engineering phases may even be transferred to planners or installers of BAS. This may make sense for the functional design phase and the procurement and procurement and engineering phase. The building tracker developer could than only be the vendor of the building tracking software as well as of various algorithm options for different building configurations.

Use Case UC-18

Secure sharing of IoT generated data with partner ecosystem (Boliden)

Training & Education in the engineering process: In the AHT-EP shown in Figure 35, the following links are related to training and education:

- Output: <c25> Training information StkH-2, StkH-1 for integrated data source
- Output <StkH-2-EPP5>: DSI in operation and handed over using created training material and documentation

- Input <41,42>: in case StkH-4 is used for development, this input aggregates needed documentation and training for SH2
- Output <c29> Relevant documentation and training for StkH-2 in operations & management
- Output: <c25> Training information StkH-2, StkH-1 on security access processes and usage
- Output <StkH-2-EPP5>: SEC in operation and handed over using created training material and documentation
- Input <41,42>: in case StkH-4 is used for development, this input aggregates needed documentation and training for StkH-2
- Output <c29> Relevant documentation and training for StkH-2 in operations & management
- Output: <SH2 - EPP2> Data source available for usage. This triggers as well needed trainings
- Output: <c25> Training information StkH-2, StkH-3 on data set usage
- Output <StkH-2-EPP5>: DP in operation and handed over using created training material and documentation
- Input <41,42>: in case StkH-4 is used for development, this input aggregates needed documentation and training for StkH-2
- Output <c29> Relevant documentation and training for StkH-2 in operations & management
- Output <c37>: training and documentation for data users including security rules

Data source integration into data platform: StkH-2 created needed documentation for data sources, meta-data and interpretations if needed. Education for StkH-1 is done data source by data source.

Security setup for data source: Handled as part of data source integration into data platform.

Data provisioning: StkH-2 educated data users on data usage possibility, limitation and duties.

Obj. 4 - Address digital learning and training activities as an integral part of the engineering cycle: The objective fits as the platform development and improvement process is orchestrated by the platform team and new developments such as the security concept validated. This Team also handles documentation, training and learning needs.

Use Case UC-21

Data-based digital twin for electrical machine condition monitoring (ABB)

AHT_Obj. 6 - Training material (HW and SW) for professional engineers: The documentation of the code, of its functionalities, of the related services are handled following the traditional process: new manuals will be provided for the administrator of the IoT integration framework, for the system operator, for the maintenance operator, and other relevant stakeholders.

Training & Education in the engineering process: In the procedure described in the AHT-EP shown in Figure 41 Electrical machine and Test bench provide Training and Education.

Obj. 4 - Address digital learning and training activities as an integral part of the engineering cycle: Use case training activities will focus mainly use case (UC-21) stakeholders. The training materials will be documentations and recommendations on separate matters.

4. The Arrowhead Tools Use Cases and the AHT-EP support

In this section, we update the analysis done in the D2.1 with the most updated summary of the use cases provided by the UC leaders in the *WP12410_survey* [4]. For each use case we use info provided in sections **A-B-C-F** of the survey for aligning the UC on the AHT-EP, and check if the AHT-EP can support all the UC in the definition of the product/system life cycle. We provide an updated picture describing what are the EPPs that each UC can exploit for structuring its own AHT-EP. In doing this analysis we highlight what are the general AHT objectives and the WP1-WP2 objectives that each UC can potentially match by using each EPP. Raw information of each use case are reported in the D2.2 appendix [31].

3.1. Updated Use Cases summary and AHT-EP analysis

In the document [31] we collect all the updated information provided by use case leaders that have been analyzed during the definition of the AHT-EP feature, structure and components. We introduced the AHT-EP schema of all the use cases and discuss how the technologies developed in this project (AHT-EP and EAf) can help the use case for matching the six project and four WP2 objectives. Thus, improving the life cycle management of the system under development. Moreover, for each use case we provide a short summary describing the field of application and the main product/system developed. A more detailed explanation can be found in WP1 deliverable D1.2 [55], WP4 deliverable D4.2 [56], WP7 D7.3 [57], WP8 D8.3 [58], and WP9 D9.3 [59].

In the same document, that integrates the present deliverable, we reported all the information of the UCs describing the specific aspects related to the Engineering Process. Using this information, we have generated an updated version of the short summary (in a tabular format) for each Use Case showing the AHT-EPP used by the UC and what are the potential WP2-WP1 and AHT objectives that the UC can potentially reach in each single EP phase.

The first version have been presented in D2.1 (in Figure 46 for your commodity), whereas, an updated version of these tables is summarized in the UC-EPP picture of the AHT project proposed in Figure 49.

Analyzing the inputs collected in the document [31] we confirm that the life cycle of all the use cases can be well described by using the AHT-EP.

In contrast to the initial expectation, the AHT-EP of many UCs have been designed by using all the EPPs proposed in the AHT-EP model. The majority of the UC EPs have been designed as multi-stakeholder interconnected EP with feedbacks between EPPs thus for representing the data-flow nature of these Industry 4.0 UCs which need information to be passed from phase-to-phase and from one stakeholder to another. Thus improving the information transmission through partners involved in the value chain.

An example of such a graph-based EPs is the Engineering process currently adopted in the UC-07 that is designed by implementing almost any development process as an iterative process; ergo the engineering process contains loops.

A second example is the EP of UC-09 that implement several iteration loops between Analysis and Development, and Development and Quality Control until the task result is deemed suitable, and then the task goes through the Documentation phase, in which the software/hardware module produced will be documented.

The UC-06, as many other UCs, connects the EPs of the five stakeholders involved in the UC matching the objective #2 of WP2 concerning the move from single to integrated multi stakeholder automation and digitalization. The stakeholders, within the Arrowhead Tools project, are collaborating to streamline the process from architectural drawing, via a 3D configurator to created machine files. By utilizing the Eclipse Arrowhead framework, they can implement and verify a more automated yet secure way of transferring data in the information flow.

Usually, the Evolution phase is connected with a feedback on the requirements such that can be used for triggering the update of some requirements and the following procedure aimed at developing a new version of the system. Thus matching the WP2 objective number one "Change from design time to run time engineering".

3.2. Updated Use Cases mapping on the AHT-EPPs

In this section, we provide the matching analysis that evaluate the level of support that the AHT-EP can offer to the use cases for designing the life cycle plan of products/system developed in the project.

The analysis is based on the information provided by use case leaders reported in the document [31].

In the DoA [60], it was proposed an alignment between use cases and the eight phases of the AHT-EP. This alignment was summarized in the picture in Figure 45 where each use case, represented by one of the circular colored icons, was assigned to the EPP in which the UC is expected to have the major contribution.

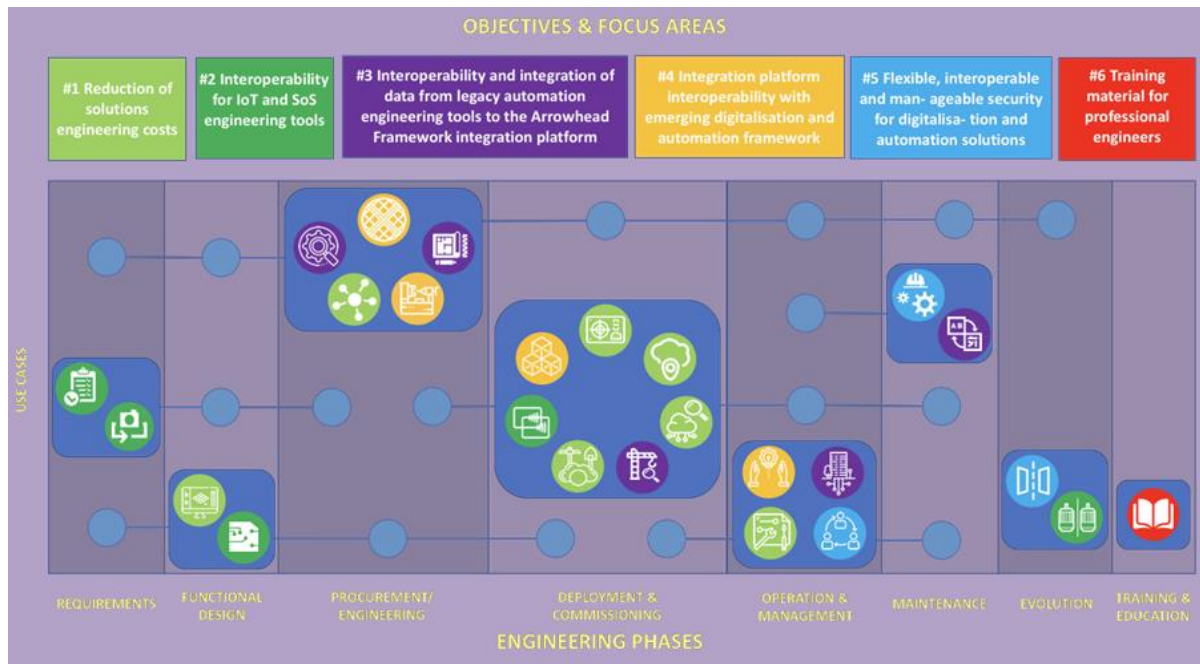


Figure 45 Old mapping of Use Cases on the AHT-EPPs hypothesized in the DoA with focus on potential AHT objectives that the UC can match

In this first attempt of UC-Objective matching, the use cases were grouped with others that have the main contribution in the same EPP. Then, blue circles have been used for connecting the grouped UCs with the other EPPs used by some of the UCs of the group. Each Use Case was represented by an icon embedded in a circle that has the color of the main AHT objective that the UC can potentially satisfy.

In the D2.1 we have analyzed the information provided by the UC leaders [3] for creating a second and more detailed version of this picture (Figure 46) that shows the first prediction on how each single UC can exploit the AHT-EPPs and which AHT-WP1-WP2 objectives the UC can potentially satisfy in each used EPP during the project. An updated list of the use cases with the relative representative icon was provided in Figure 47.

On top of the Figure 46, we can find the six objectives of the AHT project, the matching with the objective of WP1 and the four objectives that the UCs should match by using the EP described in WP2.

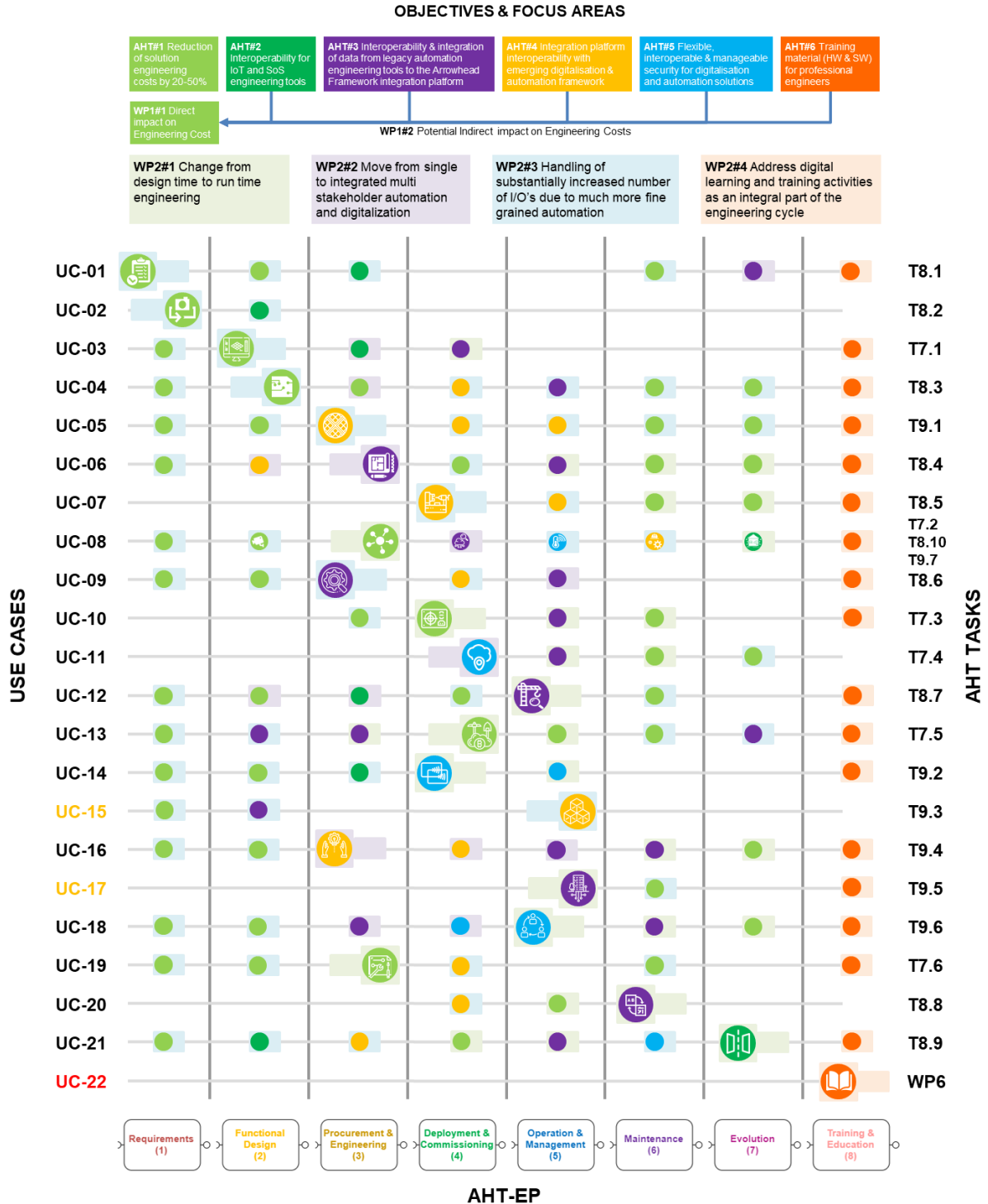


Figure 46 First prediction of the mapping of Use Cases on the AHT-EPPs hypothesized in the DoA with focus on potential AHT and WP2 objectives that the UC can match in each AHT-EPP




























Requirements	
	UC-01 Automated Formal Verification
	UC-02 Engineering processes and tool chains for digitalized and networked diagnostic imaging
Functional Design	
	UC-03 Integration of electronic design automation tools with product lifecycle tools
	UC-04 Interoperability between (modelling) tools for cost effective lithography process integration
Procurement & Engineering	
	UC-05 Support quick and reliable decision making in the semiconductor industry
	UC-06 Production preparation tool chain integration
	UC-08 SoS engineering of IoT edge devices
	UC-08.1 Smart City - Environmental Monitoring
	UC-08.2 Smart City - AI-driven Environmental Monitoring
	UC-08.3 Smart City - Condition Monitoring
	UC-08.4 Smart Energy - Smart Home
	UC-08.5 Smart Energy - Industrial
	UC-09 Machine operation optimization
	UC-16 Production Support, Energy Efficiency, Task Management, Data Analytics and Smart Maintenance
	UC-19 Deployment and configuration
Deployment & Commissioning	
	UC-07 CNC machine automation
	UC-10 Rapid HW development, prototyping, testing and evaluation
	UC-11 Configuration tool for autonomous provisioning of local clouds
	UC-13 Deployment engine for production related sensor data
	UC-14 Smart Diagnostic Environment for Contactless Module Testers
Operation & Management	
	UC-12 Digital twins and structural monitoring
	UC-15 Smart Kitting to Manage High Diversity
	UC-17 Linking Building Simulation to a Physical Building in Real-Time
	UC-18 Secure sharing of IoT generated data with partner ecosystem
Maintenance	
	UC-20 Elastic Data Acquisition System
Evolution	
	UC-21 Data-based digital twin for electrical machine condition monitoring
Training & Education	
	UC-22 Arrowhead Framework training tool

Figure 47 Old list of UCs associated at a graphical icon and grouped in the phase where we expect the major contribution

Each use case is described in a row with the UC id number and the relative task to which it will be developed. Colored UC id number means that the UC information is incomplete (red) or partial (yellow) for defining the UC matching on the EPPs.

The engineering phases are represented in the bottom part of the figure and have been used to create a tabular structure for forming the UC-EPP table. In each cell of the table we reported a small rectangle, colored with one of the four colors associated with one of the WP2 objectives, and a circle colored with the AHT objectives colors. The color assignment has been done by using information collected in the UC analysis [3] and represent the objectives that the UC can potentially match in the specific AHT-EPP. The bigger labels with the UC-specific icons are placed in the Engineering Phase that is expected to be the main phase for the use case.

Comparing the first prediction of mapping (Figure 46) with the old mapping (Figure 45) we can note that the main focus of some UC has been moved when the project has started and partners began to work on their UCs. In particular, the main focus of UC-07, that in the project conceptualization was placed in the Procurement & Engineering phase, has been declared at the begin of the project to be in the Deployment & Commissioning phase. In addition, UC-12, UC-16, and UC-19 have been shifted the focus on another EPP when the operative part of the project has started after six months.

In Figure 48, we aggregated the mapping reported in Figure 46 for generating the per-EPP clustered version of the UC mapping comparable with the mapping done in the DoA represented in the Figure 45.

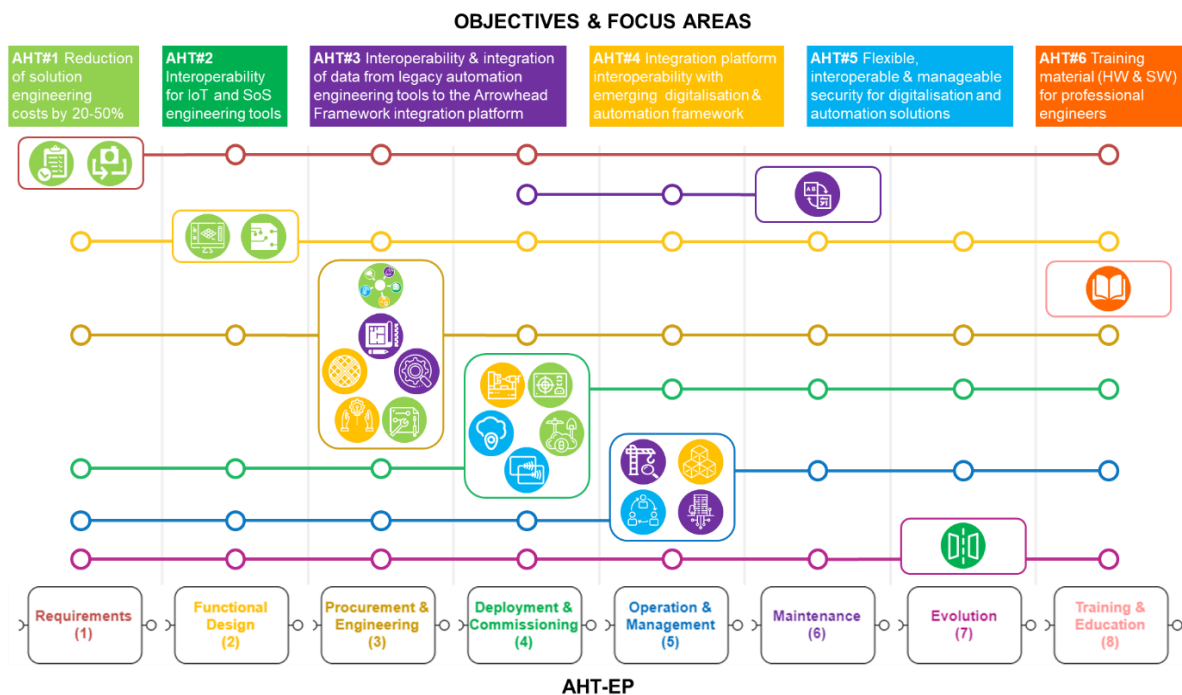


Figure 48 First prediction of the mapping of Use Cases on the AHT-EPPs based on the information collected from UC leaders, with focus on potential AHT objectives that the UC can match

In the present document we re-evaluate this mapping with new updated information collected during the project for providing a more accurate mapping of Use Cases on the AHT-EPPs.

In Figure 49 the final mapping of Use Cases on the AHT-EPPs with focus on AHT and WP2 objectives that the UC can match in each AHT-EPP. This new mapping has been produced accordingly to the information provided by the UC leaders collected in the Appendix 2 document [61]. In Figure 50 we reported the most updated list of use cases associated to the

relative graphical icon, whereas in Figure 51 we show the new aggregated per-EPP clustered version of the UC mapping.

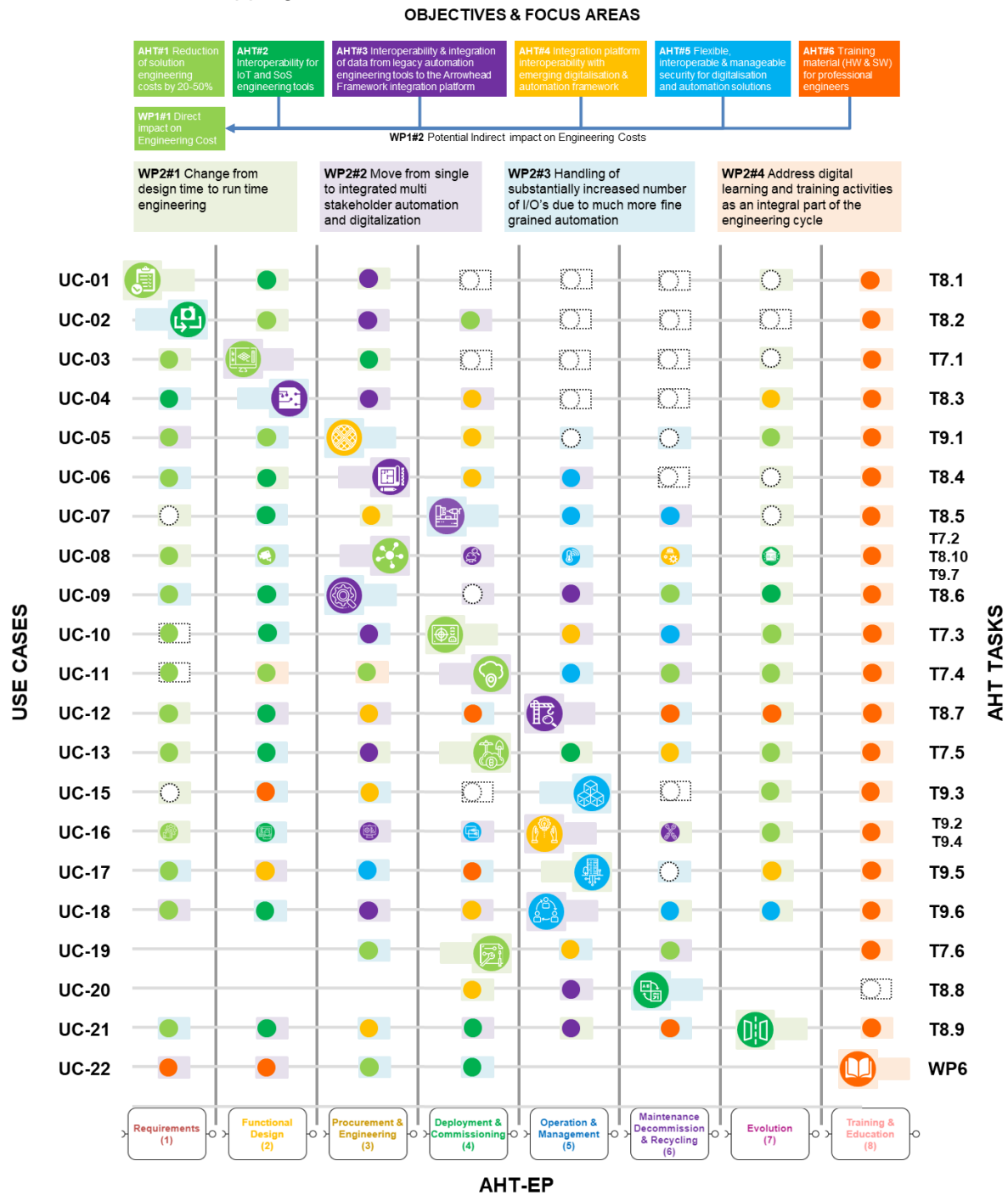


Figure 49 Final mapping of Use Cases on the AHT-EPPs with focus on AHT and WP2 objectives that the UC can match in each AHT-EPP

In this new updated version of the mapping we can note that several use cases (UC-01, UC-02, UC-03, UC-06, UC-15, and UC-22) have used in their engineering process even phases not considered in the early phase of the project.

In this updated mapping, some UC have used phases which, accordingly to the provided information, do not explicitly target a specific objective (these are represented by the dashed

shapes in the figure). However, the usage of the AHT-EP model and the definition of the activities done in each phase help UC leaders in clarifying the flow of actions, tools to be adopted and data to be transmitted intra and inter stakeholders in such a way that the adoption of phases not directly matched to a specific objective can anyhow potentially have an indirect impact on the engineering costs.

If we compare the new mapping (Figure 49) with the old mapping (Figure 46) we can note that several use cases have used more EPPs in their engineering process, suggesting that the activity of dissemination of the AHT-EP model performed by the WP2 leaders during the AHT project conferences and WP2 meetings have augmented the confidence of partners in using this new technology for clarifying roles, tools, and activities of partners in each phase of the engineering process of the system under development. UC-02, UC-03, UC-07, and UC-22 are examples the most evident examples where, within the project period, UC leaders have refined the definition of the AHT-EPs by adding new phases in their engineering processes.

A second main difference is the aggregation of the UC-14 in the UC-16, so that the UC-16, which have the main focus on EPP5, now have been reshaped in a form with 5 sub use cases with secondary main focuses distributed on the other EPPs.

With this update, we have 19 stand-alone use cases and two use cases composed by five use cases each (UC-08 and UC-16).

In our analysis we highlighted how the AHT-EP model and ontology can be helpful for enabling the digitalised management of the life cycle of complex industrial multi-stakeholder use cases. In the following bullet list we summarize some of the most notable advantages offered by the adoption of the AHT-EP model in the Industry 4.0 domain.

- The AHT-EP model allows the usage of feedback connections that provide inputs to previous phases and an Evolution phase intended to provide feedback for future enhancements of the product. This ensures continuous engineering and a wanted shift of paradigm from design-time to run-time.
Also the structure of the AHT-EP can change during the time, for this motivation our SOA based model allows adding/removing/modify interaction between phases and stockholders during the time.
- The shift from single to integrated multi-stakeholder automation and digitalisation can be achieved by connecting the AHT-EP of one stakeholder with external EPs defining the life cycle of sub-systems developed by other stakeholders involved in the value chain.
- The handling of an increased number of I/Os due to much more fine-grained automation will be guaranteed by the capability of the AHT-EP to handle multiple I/O interfaces (i.e. services) for each EPP.
- The digital learning and training activities as an integral part of the EP cycle is supported by the inclusion in the AHT-EP of the Training & Education phase designed for training with the most updated techniques all the figures involved in the EP: analysts, engineers, technicians, specialised operators, maintenance operators and even the final user.
With the AHT-EP model, digital learning and training activities are integral part of the AH-EP, which includes a specific phase devoted to "Training & Education".
- Reduction of engineering costs for the full life-cycle management comes when stakeholders make use of AHT-EP, an engineering process model capable of providing a more efficient automation and digitalisation engineering procedure because designed to exploit modern IoT/SoS integration platforms, new tools and integrated tool chains. AHT-EP model is capable to address this particular aspects since conceived to reduce the manual overhead because supporting a more automatised management of the life-cycle of the product. In this direction, the AHT-EP model combined with the Eclipse Arrowhead framework will make possible an early usage and data from companies

Requirements	
	UC-01 Automated Formal Verification
	UC-02 The MRI pTX Optimizer: Design tool framework for MRI RF transmit chains
Functional Design	
	UC-03 Integration of electronic design automation tools with product lifecycle tools
	UC-04 Interoperability between (modelling) tools for cost effective lithography process integration
Procurement & Engineering	
	UC-05 Support quick and reliable decision making in the semiconductor industry
	UC-06 Production preparation tool chain integration
	UC-08 SoS engineering of IoT edge devices
	UC-08.1 SoS engineering of IoT edge devices (Environmental Monitoring)
	UC-08.2 SoS engineering of IoT edge devices (AI-Driven Environmental Monitoring)
	UC-08.3 SoS engineering of IoT edge devices (Condition Monitoring)
	UC-08.4 SoS engineering of IoT edge devices (Smart Home)
	UC-08.5 SoS engineering of IoT edge devices (Industrial Energy Monitoring)
	UC-09 Machine operation optimization
Deployment & Commissioning	
	UC-07 CNC machine automation
	UC-10 Rapid HW development, prototyping, testing and evaluation
	UC-11 Configuration tool for autonomous provisioning of local clouds
	UC-13 Deployment engine for production related sensor data
	UC-19 Deployment and configuration
	UC-14 Smart Diagnostic Environment for Contactless Module Testers (moved in UC-16.5)
Operation & Management	
	UC-12 Digital twins and structural monitoring
	UC-15 Smart Kitting to Manage High Diversity
	UC-16 Production Support, Energy Efficiency, Task Management, Data Analytics and Smart Maintenance
	UC-16.1 Equipment and Energy Data Management
	UC-16.2 I/O Link SC sensors
	UC-16.3 SC Data Management
	UC-16.4 Smart Maintenance
	UC-16.5 Tester Integration
	UC-17 Linking Building Simulation to a Physical Building in Real-Time
	UC-18 Secure sharing of IoT generated data with partner ecosystem
Maintenance, Decommissioning & Recycling	
	UC-20 Elastic Data Acquisition System
Evolution	
	UC-21 Data-based digital twin for electrical machine condition monitoring
Training & Education	
	UC-22 Arrowhead Framework training tool

Figure 50 List of UCs associated at a graphical icon and grouped in the phase of major contribution

involved in the value-chain. Thus, impacting on the engineering cost savings in the range of 65-80%. Such savings on engineering costs indicate a strong innovation potential on all automation/digitalisation solutions that can be engineered using in this manner.

- Whereas, the tool and framework availability will be addressed by developing automatic systems in the Eclipse Arrowhead Framework , that provides an architecture, appropriate tools and services required for the efficient digitalisation and automation of a complex multi-stakeholder EP.
- %end{enumerate}

All these combined advantages makes the AHT-EP model widely flexible and adaptable to support a large spectrum of heterogeneous vertical UCs that ranging from semiconductor engineering, to mining, construction industry, finance etc.

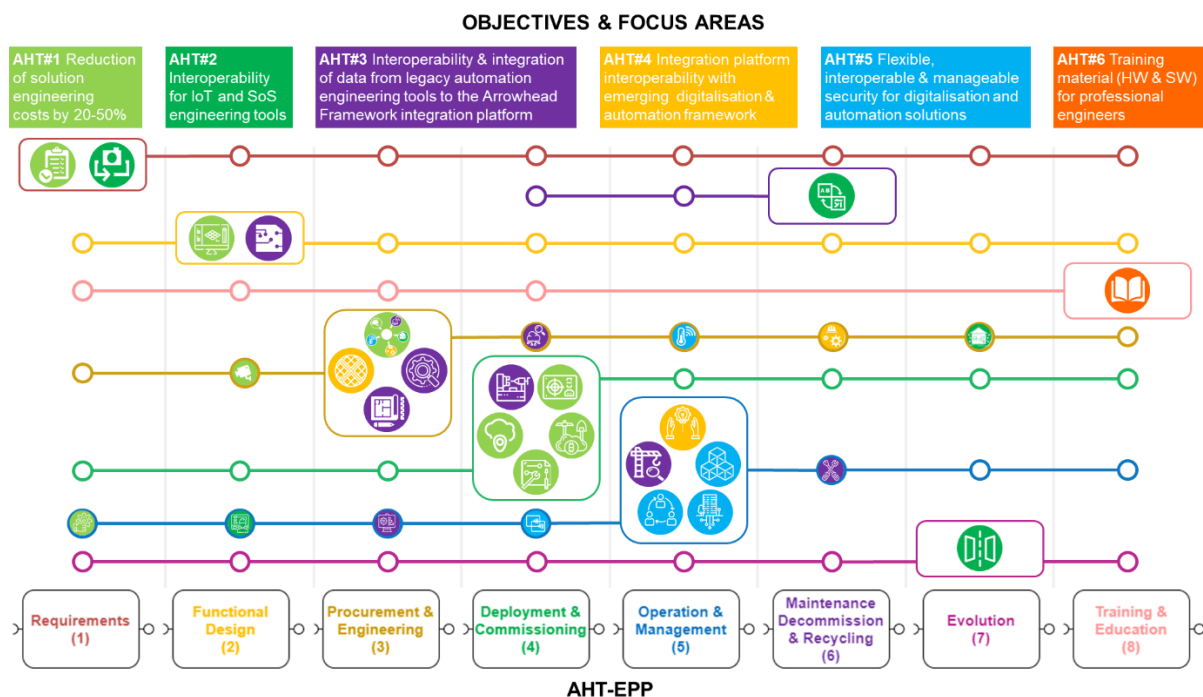


Figure 51 Updated mapping of Use Cases on the AHT-EPPs based on the information collected from UC leaders, with focus on AHT objectives that the UC can match because using the AHT technology

5. Conclusions

In the deliverable D2.2 we have updated the detailed description of the AHT Engineering Process Phases and their interactions that have been used as a reference by the *Use Case (UC)* leaders to map the *Use Case Engineering Process (UC-EP)* phases on the *Arrowhead Tools Engineering Process (AHT-EP)*. The Eight SubTask leaders have been called to produce an updated description of their phase of the engineering process, highlighting the essence of what the Phase means in the context of service-oriented architecture paradigms. Moreover, we complemented the description of EPPs with an analysis of the activities implemented by each UC in each EPP and with a focus on what are the main advantages for the stakeholders involved in each EPP in using the AHT-EP model and the EAf technology. These Engineering Process Phases descriptions have been conceived for being used to guide the partners in the classification of the HW/SW tools used in the various fields of the Use Cases.

We created an ontology to represent, with a graphical and a tabular notation, the direct interplay that links the AHT-EP Phases used for the management of the life cycle of each product/service of the use cases.

During the reporting period, WP2, WP1, and WP4 leaders conceived and created a shared template (the *WP124_Survey* [5]) which have been updated in *WP12410_Survey* [4] for collecting more specific information. These surveys have been intended to collect, from the use case leaders, all the information required for the analysis of the adopted Engineering Process, and the definition of the use case's baselines at two different times: at month 6 (which have been used for D2.1) and at month 24 (which have been used for the current document D2.2). The WP leaders have established joint multi-WP collaboration, with interaction with all the other WPs, to ensure a coherent and coordinated development of the concepts that will drive the Arrowhead Tools project.

The *WP124_Survey* and the *WP12410_Survey* have followed a process that guides the use case leaders in the analysis of the use cases, trying to simplify the analysis and unify the results obtained from it.

The material collected with the *WP124_Survey* has been used to prepare the D2.1 deliverable (D1.1 and D4.1 in WP1 and WP4 respectively). Whereas, the material collected with the *WP12410_Survey* is used to prepare the D2.2 deliverable (D1.2, D4.2, D5.3, D10.2 in WP1, WP4, WP5 and WP10 respectively).

The material collected with the *WP124_Survey* describing the Engineering Process aspects has been analyzed to produce a preliminary mapping of the use cases on the AHT-EP and identify the WP2 and AHT objectives that each UC can potentially match in each AHT-EP phase. Whereas, the material collected with the *WP12410_Survey* have been used to generate an updated version of this map, which better reflect the status end potentiality offered by the AHT-EP and EAf technologies in supporting the use cases.

An initial mapping of the UCs phases onto the AHT-EP, highlighting the WP2 and AHT objectives, have shown in D2.1 how each UC can potentially match in each Engineering Process Phase. This mapping have been updated in the present D2.2 document with the new information available. In the *WP124_Survey* and *WP12410_survey*, we collected information about the Engineering Process phases currently used in each of the use cases domain or field. For this purpose, all use case leaders have been called twice (at M6 and at M24) to fill out the surveys that we have created in collaboration with the WP1, WP4 and WP10 leaders. In several points of the surveys, each use case leader have described the details of the Engineering Process phases adopted in its field for implementing the use case. Information regarding the Use Case Engineering Processes have been collected and summarised for producing the final analysis of the UC mapping on the AHT-EP that is proposed as a contribution of the D2.2 deliverable.

This UC map represent the updated, corrected, extended, and consolidated picture of the UC Engineering Processes mapped on the six project and four WP2 objectives.

6. Appendixes

List of appendixes, i.e. other files that are bundled together with this document and as such are part of this deliverable.

1. Appendix1: D2.2 Deliverable Appendix - WP1 WP2 WP4 Use Cases survey -> *DA1_2_WP12410_survey.docx* + *DA1_WP12410_survey.xlsx*
2. Appendix2: D2.2 Deliverable Appendix - Use Cases analysis for AHT-EP definition -> *DA2_2_WP2_UC_analysis.pdf*

7. References

- [1] J. Delsing, *Iot automation: Arrowhead framework*, CRC Press, 2017.
- [2] J. Garcia Represa e J. Delsing, «Autonomous Production Workstation Operation, Reconfiguration and Synchronization,» in *25th International Conference on Production Research, Chicago, August 9-15, 2019.*, Chicago, 2019.
- [3] G. Urgese, J. V. Deventer e A. Andrea, «Deliverable D2.1 "Procedure model",» 2019.
- [4] G. Urgese, P. Azzoni e F. Montori, «WP12410_survey,» 2020.
- [5] P. Azzoni, G. Urgese e F. Montori, «WP124 Use Cases survey,» 2019.
- [6] G. Urgese, «Use Cases analysis for AHT-EP definition,» 2019.
- [7] M. Iñigo e C. Schmittner, «Deliverable D10.1: Standardisation base line,» Arrowhead Tools Project, 2019.
- [8] F. Kevin e M. Harold, «The Relationship of System Engineering to the Project Cycle,» in *Proceedings of the First Annual Symposium of National Council on System Engineering*, 1991.
- [9] M. Hankel e R. Bosch, «The reference architectural model industrie 4.0 (rami 4.0),» ZVEI, 2015.
- [10] Y. Lu, K. C. Morris e S. Frechette, «Current standards landscape for smart manufacturing systems,» *National Institute of Standards and Technology, NISTIR 8107*, vol. 39, 2016.
- [11] S. Lin, B. Miller, J. Durand, G. Bleakley, A. Chigani, R. Martin, B. Murphy e M. Crawford, «The industrial internet of things volume G1: reference architecture,» *Industrial Internet Consortium*, pp. 10-46, 2017.
- [12] I. Michel e S. Christoph, «Deliverable D10.3: Standardisation report year 2,» Arrowhead Tools Project, 2021.
- [13] B. Scholten, «The road to integration: A guide to applying the ISA-95standard in manufacturing,» Isa, 2007.
- [14] S. V e W. K E, «Engineering design with digital thread,» *AIAA Journal* , vol. 11, pp. 4515-4528, 2018.
- [15] R. Cloutier e N. Hoboken, *The Guide to the Systems Engineering Body of Knowledge (SEBoK) v2.3*, vol. 2.3, SEBoK Editorial Board, 2020.
- [16] M. Maier, «Architecting Principles for Systems-of-Systems,» *Systems Engineering*, vol. 1, n. 4, pp. 267-284, 1998.
- [17] R. Adcock, «Emergence,» [Online]. Available: <https://www.sebokwiki.org/wiki/Emergence>. [Consultato il giorno 20 April 2021].
- [18] «ISO/IEC/IEEE 21839 – System of Systems (SoS) Considerations in Life Cycle Stages of a System,» ISO/IEC/IEEE, 2019.
- [19] «SOA Source Book - What Is SOA?,» collaboration.opengroup.org, [Online]. Available: <https://collaboration.opengroup.org/projects/soa-book/pages.php?action=show&ggid=1314>. [Consultato il giorno 31 March 2021].

- [20] "Chapter 1: Service Oriented Architecture (SOA)," msdn.microsoft.com, [Online]. Available: <https://web.archive.org/web/20170707052149/https://msdn.microsoft.com/en-us/library/bb833022.aspx>. [Accessed 05 April 2021].
- [21] «Service-Oriented Architecture Ontology, Version 2.0,» www.opengroup.org, [Online]. Available: <https://publications.opengroup.org/standards/soa>. [Consultato il giorno 05 April 2021].
- [22] «Service Oriented Architecture : What Is SOA?,» www.opengroup.org, [Online]. Available: <https://web.archive.org/web/20160819141303/http://opengroup.org/soa/source-book/soa/soa.htm>.
- [23] A. T. Velte, Cloud Computing: A Practical Approach, McGraw Hill, 2010.
- [24] «Service-oriented architecture,» wikipedia.org, [Online]. Available: https://en.wikipedia.org/wiki/Service-oriented_architecture. [Consultato il giorno 05 April 2021].
- [25] V. Singh e K. E. Willcox, «Engineering Design with Digital Thread,» *American Institute of Aeronautics and Astronautics, AIAA Journal*, vol. 56, n. 11, pp. 4515-4528, 2018.
- [26] E. Kraft, «HPCMP CREATE-AV and the Air Force Digital Thread,» in *AIAA SciTech 2015, 53th AIAA Aerospace*, Kissimmee, Florida, 2015.
- [27] M. Tatara, F. Montori, G. Kulcsár and S. Bocchio, "Deliverable D4.2: Arrowhead Tools toolchain design," Arrowhead Tools Project, 2020.
- [28] J. Delsing e P. Varga, «Eclipse Arrowhead,» 2021. [Online]. Available: <https://projects.eclipse.org/projects/iot.arrowhead>.
- [29] J. Kristan, J. Schuster e P. Varga, «"Consolidated SOA framework platform architecture, design and sw implementations of core systems – Y2",» Arrowhead Toos Project, 2021.
- [30] «ISA-95,» [Online]. Available: isa-95.com. [Consultato il giorno 25 November 2019].
- [31] G. Urgese e e. al., «D2.2 Deliverable Appendix – Updated Use Cases analysis for AHT-EP definition,» Attowhead Tools Project, 2021.
- [32] S. Balaji e M. Murugaiyan, «Waterfall vs. V-Model vs. Agile: A comparative study on SDLC,» *International Journal of Information Technology and Business Management*, pp. 26-30, 2012.
- [33] S. Mathur e S. Malik, «Advancements in the V-Model,» *International Journal of Computer Applications*, pp. 29-34, 2010.
- [34] D. D. Walden, G. J. Roedler, K. Forsberg, R. D. Hamelin e T. M. Shortell, Systems engineering handbook: a guide for system life cycle processes and activities, Fourth edition a cura di, Hoboken, A cura di, New Jersey: Wiley, 2015.
- [35] NASA, NASA SYSTEMS ENGINEERING HANDBOOK, nasa/sp-2016 -6105 rev2, 2017.
- [36] N. Viola, S. Corpino, M. Fioriti e F. Stesina, «Functional analysis in systems engineering: Methodology and applications,» *Systems engineering-practice and theory*, 2012.

- [37] P. B. Kruchten, «The 4+1 View Model of architecture,» *IEEE Software*, vol. 12, n. 6, p. 42–50, 1995.
- [38] P. Micouin, *Model-based systems engineering: fundamentals and methods*, N. U. Hoboken, A cura di, London: Wiley, 2014.
- [39] A. W. Wymore, *Model-based systems engineering*, Boca Raton: Chapman and Hall/CRC, 2018.
- [40] J. M. Borcky e T. H. Bradley, *Effective model-based systems engineering*, New York: Springer Science+Business Media, 2018.
- [41] L. Delligatti, *SysML distilled: a brief guide to the systems modeling language*, Upper Saddle River, NJ: Addison-Wesley, 2014.
- [42] I. C. Society, P. Bourque e R. E. Fairley, *Guide to the Software Engineering Body of Knowledge (SWEBOK(R)): Version 3.0, 3rd ed a cura di*, Los Alamitos, CA, USA: IEEE Computer Society Press, 2014.
- [43] J. Alvarez-Rodríguez, R. Mendieta, V. Moreno, M. Sánchez-Puebla e L. J., «Semantic recovery of traceability links between system artifacts,» *Int. J. Softw. Eng. Knowl. Eng. IJSEKE*, vol. 30, n. 11, pp. 1-28, 2020.
- [44] J. M. Alvarez-Rodríguez, R. Mendieta, J. De la Vara, A. Fraga e J. Llorens, «Enabling system artefact exchange and selection through a Linked Data layer,» *J UCS JUCS*, p. 1–24, 2018.
- [45] P. Varga e e. al, «Making system of systems interoperable – The core components of the arrowhead framework,» *J. Netw. Comput. Appl.*, vol. 81, p. 85–95, 2017.
- [46] H. Derhamy, J. Eliasson e J. Delsing, «System of System Composition Based on Decentralized Service-Oriented Architecture,» *IEEE Syst. J.*, vol. 13, n. 4, p. 3675–3686, 2019.
- [47] C. Paniagua, J. Eliasson e J. Delsing, «Efficient Device-to-Device Service Invocation Using Arrowhead Orchestration,» *IEEE Internet Things J.*, vol. 7, n. 1, p. 429–439, 2020.
- [48] S. Amira, V. Perelman e D. Dori, «A Project-Product Lifecycle Management approach for improved systems engineering practices,» in *INCOSE International Symposium*, 2008.
- [49] Y. L. C. Lu, I. Kevin and K. Wang, “Digital Twin-driven smart manufacturing: Connotation, reference model, applications and research issues; Huang, Huiyue; Xu, Xun,» *Robotics and Computer-Integrated Manufacturing*, vol. 61, p. 101837, 2020.
- [50] H. T. Tran, T. Nguyen Van e P. Vinh, «Towards Service Co-evolution in SOA Environments: A Survey,» in *ICTCC 2020: Context-Aware Systems and Applications, and Nature of Computation and Communication*, 2021.
- [51] A. J. N. Mohsin, «A review and future directions of SOA-based software architecture modelling approaches for System of Systems,» in *2018, SOCA 12*.
- [52] J. Halme e e. al, «Monitoring of Production Processes and the Condition of the Production Equipment through the Internet,» in *6th International Conference on Control, Decision and Information Technologies (CoDIT)*, Paris, 2019.
- [53] E. Vaumorin, «Deliverable D6.1 Training and support material for project internal and external usage,» AHT Project, 2019.

- [54] E. e. a. Vaumorin, «Deliverable D6.2: Usage and training report for Y1 plan for Y2,» Arrowhead Tools Project, 2020.
- [55] P. Azzoni e Ø. Haugen, «Deliverable D1.2 - Requirements, state of the art and project base line after Year 1,» Arrowhead Tools Project, 2021.
- [56] M. Tataro, F. Montori, G. Kulcsár e S. Bocchio, «Deliverable D4.2 - Arrowhead Tools toolchain design,» Arrowhead Tools Project, 2021.
- [57] S. Bocchio e e. al., «Deliverable D7.3 - Detailed specification of use cases for reduction of solution engineering cost. For Year 2.,» 2021.
- [58] F. Rosbak e S. Roubtsov, «Deliverable D8.3 - Detailed specification of use cases for data exchange between IoT/SoS and legacy engineering tools. For Year 2.,» Arrowhead Tools Project, 2021.
- [59] G. Schneider e G. Pfitzner, «Deliverable D9.3 WP9: Detailed specification of use cases for data and security interoperability across integration platforms – for year 2,» Arrowhead Tools Project, 2021.
- [60] J. Delsing, «AHT DoA Annex 1 (part B) - Arrowhead Tools for Engineering of Digitalisation Solutions,» AHT Project, 2019.
- [61] G. Urgese e e. al., «D2.2 Deliverable Appendix - Updated Use Cases analysis for AHT-EP definition,» Arrowhead Tools Project, 2021.
- [62] «AutomationML,» [Online]. Available: <https://en.wikipedia.org/wiki/AutomationML>. [Consultato il giorno 25 November 2019].
- [63] F. Zezulka, P. Marcon, I. Vesely e O. Sajdl, «Industry 4.0—An Introduction in the phenomenon,» *IFAC-PapersOnLine*, vol. 49, n. 25, pp. 8-12, 2016.
- [64] J. Virta, S. Ilkka, T. Antti e K. Kari, «SOA-Based integration for batch process management with OPC UA and ISA-88/95,» in *IEEE 15th Conference on Emerging Technologies & Factory Automation (ETFA 2010)*, 2010.
- [65] M. Tataro, F. Montori e B. Sara, «O3: AHT use case analysis - Use cases analysis,» AHT-Project, 2019.
- [66] B. Scholten, *The road to integration: A guide to applying the ISA-95 standard in manufacturing*, Isa, 2007.
- [67] G. Peeren e S. Roubtsov, «Deliverable D8.1 Detailed specification of use cases for data exchange between IoT/SoS and legacy engineering tools. For Year 1.,» AHT Project, 2019.
- [68] T. Holm, L. Christiansen, M. Göring, T. Jäger e A. Fay, «ISO 15926 vs. IEC 62424—Comparison of plant structure modeling concepts.,» in *IEEE 17th International Conference on Emerging Technologies & Factory Automation (ETFA 2012)*, 2012.
- [69] D. Evans, «The internet of things: How the next evolution of the internet is changing everything,» *CISCO white paper*, pp. 1-11, 2011.
- [70] H. E. J. Derhamy, J. Delsing e P. Priller, «A survey of commercial frameworks for the internet of things,» in *IEEE 20th Conference on Emerging Technologies & Factory Automation (ETFA)*, 2015.
- [71] C. De Luca e G. Schneider, «Deliverable D9.1 Use case spec for Y1,» AHT Project, 2019.
- [72] O. Carlsson, J. Delsing, F. Arrigucci, A. W. Colombo, T. Bangemann e P. Nappey, «Migration of industrial process control systems to service-oriented architectures,»

- International Journal of Computer Integrated Manufacturing*, vol. 31, n. 2, pp. 175-198, 2018.
- [73] S. Bocchio, «D7.1 : WP7 Use case spec for Y1. Detailed specification of use cases for reduction of solution engineering cost. For Year 1.,» AHT Project, 2019.
- [74] A. Bicaku, C. Schmittner, M. Tauber e J. Delsing, «Monitoring Industry 4.0 applications for security and safety standard compliance,» in *IEEE Industrial Cyber-Physical Systems (ICPS)*, 2018.
- [75] A. Bicaku, S. Maksuti, C. Hegedűs, M. Tauber, J. Delsing e J. Eliasson, «Interacting with the arrowhead local cloud: On-boarding procedure.,» in *IEEE Industrial Cyber-Physical Systems (ICPS)*, 2018.
- [76] P. Azzoni, G. Urgese e F. Montori, «WP124 Use Cases survey,» 2019.
- [77] P. Azzoni e Ø. Haugen, «Deliverable D1.1- Requirements, state of the art,» AHT Project, 2019.
- [78] E. Allouis, R. Blake, S. Gunes-Lasnet, T. Jorden, B. Maddison, H. Schroeven-Deceuninck, M. Stuttard, P. W. K. Truss, R. Ward e M. Woods, «A FACILITY FOR THE VERIFICATION & VALIDATION OF ROBOTICS & AUTONOMY FOR PLANETARY EXPLORATION,» in *Advanced Space Technologies in Robotics and Automation (ASTRA)*, Noordwijk, The Netherlands., 2013.
- [79] «Procurement Process Flow – A Guide to Procurement in Business,» 11 03 2019. [Online]. Available: <https://www.codelessplatforms.com/blog/procurement-process-flow/>. [Consultato il giorno 20 10 2019].
- [80] «CAEX web site of the Chair of Process Control Engineering,» RWTH Aachen, Germany, [Online]. Available: http://www.plt.rwth-aachen.de/cms/PLT/Forschung/Projekte2/~ejwy/CAEX_IEC_62424/?lidx=1. [Consultato il giorno 25 November 2019].
- [81] G. Urgese, J. v. Deventer e A. Acquaviva, «D2.1 Procedure model,» 2019.

8. List of abbreviations

Abbreviation	Meaning
AHT	ArrowHead-Tools
SOA	Service Oriented Architecture
DoA	Declaration of Agreement
UC	Use Case

UC-EP	Use Case Engineering Process
AHT-EP	ArrowHead-Tools Engineerign Process
EOL	end-of-life

9. Revision history

Contributing and reviewing partners

Contributions	Reviews	Participants	Representing partner
X	X	Gianvito Urgese	POLITO
X	X	Jan Van Deverter	LTU
X	X	Paolo Azzoni	Eurotech
X	X	Sara Bocchio	ST-I
X	X	Enrico Macii	POLITO
X		Frans Rosbak	PHC
X		Peter van der Meulen	PHC
X		Oskar Berreteaga	UMLA
X		Odei Ayastuy Lizarralde	UMLA
X		Jose María Alvarez Rodríguez	UC3M
X		Anja Zernig	KAI
X		Marek Tatara	DAC
X		Germar Schneider	IFD

X		Gabor Pfitzner	SYSTEMA
X		Gerhard Luhn	IFD
X		Arellano Cristobal	IKERLAND
X		Muñoz Unai	IKERLAND
X		Fanjul Jacobo	IKERLAND
X		Jan Westerlund	ABB
X		Janne Keränen	VTT
X		Jari Halme	VTT
X		Emmanuel Vaumorin	Magillem
X		Asma Smaoui	Magillem
X		Saadia Dhouib	CEA

9.1. Amendments

No.	Date	Version	Subject of Amendments	Author
1	2021-03-30	0.1	First draft	Urgese Gianvito
2	2021-04-25	0.2	Second draft	Urgese Gianvito, Jan Van Deventer
3	2021-05-24	0.3	Third draft	Urgese Gianvito
4	2021-05-26	1.0	Finalised	Urgese Gianvito

9.2. Quality assurance

No	Date	Version	Approved by
1	2021-05-26	1.0	Jerker Delsing