Document title: ArrowHead-Tools Deliverable D2.1 “Procedure model”

Version	**Status**	**Date**
1.0	final	2019-12-03

Author
Gianvito Urgese
Jan Van Deventer

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

Deliverable D2.1
“Procedure model”

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

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

Andrea Acquaviva
andrea.acquaviva@unibo.it

Abstract

This document constitutes deliverable D2.1 of the ArrowHead-Tools project.

“Consolidated and elaborated system engineering procedure model”

Table of contents

1. Introduction

Work Package 2 (WP2), the Digitalization of the Engineering Process, aims to develop a consolidated engineering process[1] model that relies on a service oriented architecture (SOA), which can be implemented using an integration platform based on WP3 (Digitalization framework: Integration & Interoperability) and WP4 (Tools chain architecture) results.

The proposed engineering process model is broken up into eight engineering phases[2], and activities are provided for each of the phases. The goal is to elaborate and consolidate an ArrowHead Framework [1] compliant engineering process model. The model is designed as a 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 is used in WP3, 4, 5, 6, 7, 8, 9.

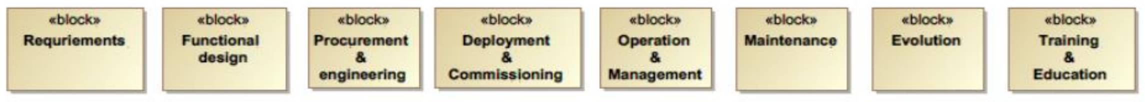


Figure 1 ArrowHead-Tools Engineering Process (AHT-EP)

The eight phases cover the life cycle of an engineered artefact. 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.

[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.

Based on the ArrowHead-Tools project’s requirement (WP1), tools for each engineering phase that are based on a SOA will be developed (WP4-WP5) and will ensure interoperability via a framework (WP3). This concept is tied back to reality through the 22 use cases of the project.

This deliverable, D2.1, describes a preliminary process model that from now on will be defined as Engineering Process (EP). In month 24 of this project, an updated version will be reported in D2.2. D2.1 is an initial reflection to understand where the ArrowHead-Tools project starts from, and what its twenty-two use cases have in common in terms of the eight EP phases. Yet, WP2 is interdependent with WP1 and WP4 as stated above. Its first deliverable had to be coordinated with the other work packages and deliverables (as described in Figure 2).

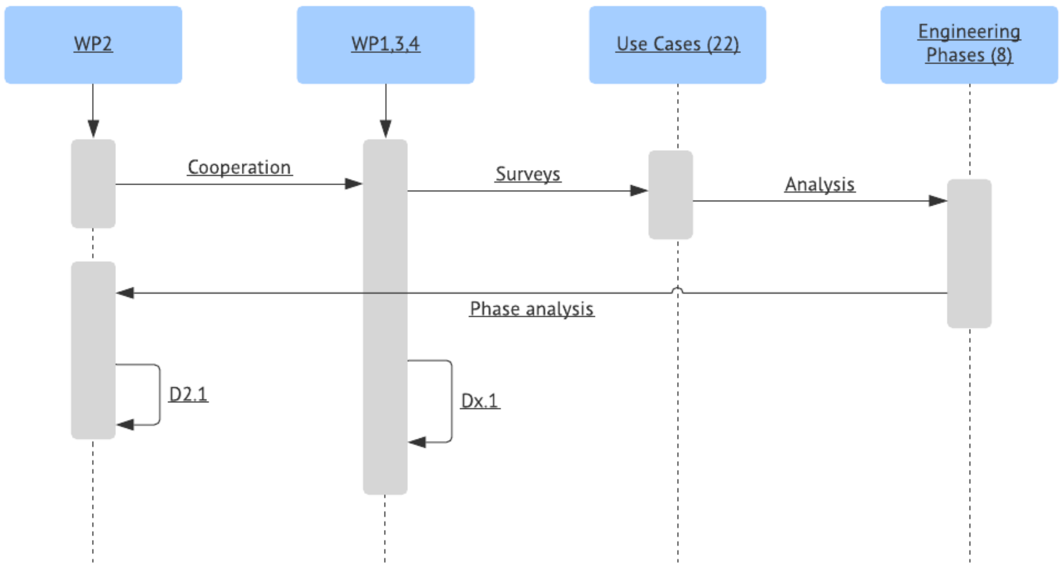


Figure 2 WP2 delivery creation process

The cooperation between the interdependent work packages led to the creation of the *WP124 survey* [3], which each use case had to fill out as well as possible. The surveys were then analysed and information was extracted for each engineering phase.

The aim of the present document is to describe the initial overall plan for the design of a flexible Engineering Process usable by the 22 AHT Use Cases (AHT-UCs) for 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.

Following this introduction, we discuss the process implemented through this initial phase of the project that has been to uncover the requirements from the stakeholders as well as their current engineering processes. This picture of their current

engineering processes forms a baseline on which we will assess the progress towards the AHT-WP2 objectives.
The deliverable continues with the introduction of the ontology, designed in this first part of the project, for the definition of all the components and concepts to be used for building the Engineering Process of each use case.
In Chapter 2, 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.
Then, in the last part of the document we reports an alignment of the 22 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. Execution plan

WP2 activity is composed by a task 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 SubTask 2.1.1-2.1.8 which will 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 methodologies and tools that the AHT-EP should support in each phase for enabling users to easily describe the product/service life cycle.
In order to support the overall analysis, we interviewed the use uase leaders with a survey [3] designed by leaders of WP1, WP2, and WP4 for acquiring fundamental aspects of the use cases that should be evaluated for matching the project and WP2 objectives.

2.1. Work Done

In the first six months of the project we have produced a detailed description of the AHT Engineering Process Phases (AHT-EPPs) that must be 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 product/service of 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* [3], 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 summarised by Urgese [4] for producing the analysis of the UC mapping on the AHT-EP that is proposed as contribution of the present deliverable in section 4.

2.2. Work To Be Done

During the next 12 months, we will ask partners to update our survey with more specific information. If required, we will improve 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 ArrowHead framework.

In the last 6 months of WP2, each SubTask leader shall analyse the points ‘A, B, C, and G’ of all the *WP124 surveys* [3] to isolate and characterize the Engineering Process phase which the SubTask addresses specifically. Then, the SubTask leaders are requested to integrate the information and conceptualise a flexible phase that can have the potential to be adapted and used in the life cycle description of all the use cases.
Standard EPs used in the various use cases will be evaluated and eventually integrated in the AHT-EP.
Here, we need to come out with eight flexible phases that can be customised and combined easily to match the specific requirements of the various use cases.

SubTask leaders will analyse the information collect form UCs after the first 18 months 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.

The task leader together with the WP2 leaders will analyse and integrate the hypothesized Engineering Process phases, produced by SubTask leaders, and create a harmonized procedure that can drive the implementation of tools and methodologies by WP3, WP4, and WP5 to support each use case. The procedure will be refined and finalised for matching the project and WP2 objectives.

Each use case leader will use the AHT-EP ontology (described in Section 3.1 of this document) for describing the architecture of each UC-EP and discuss how the AHT-EP is useful for matching the AHT and the WP2 objectives.

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/service 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 reconfigurable and able to collaborate with other systems developed by different stakeholders.
In the deliverable D10.1 [5] 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* [6] 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 (Figure 3). The V-Model demonstrates the relationships between each phase of the development life cycle and its associated phase of testing. The horizontal and vertical axes represent time or project completeness (left-to-right) and level of abstraction (coarsest-grain abstraction uppermost), respectively.

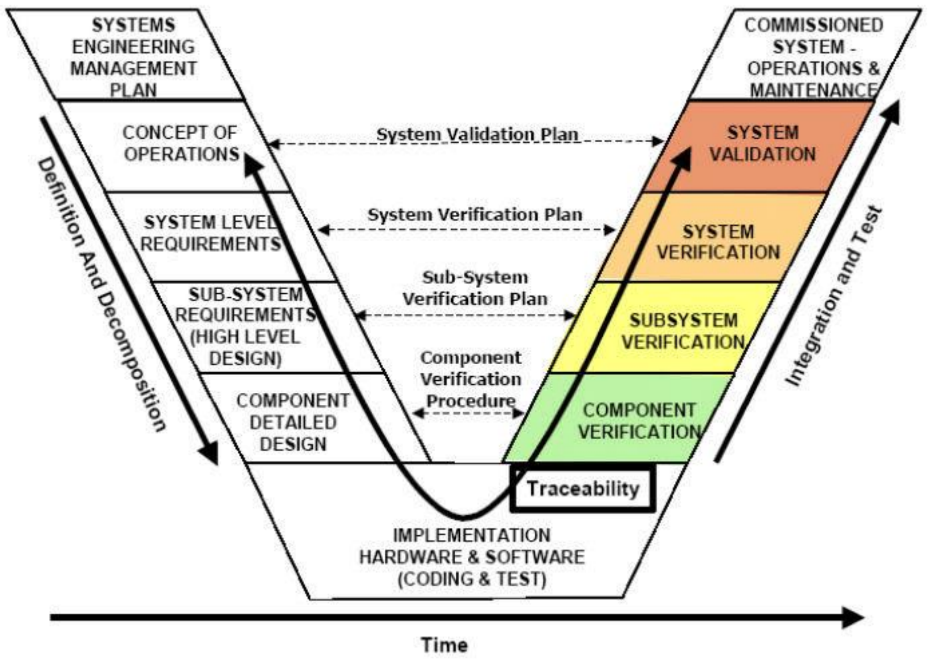

SYSTEMS ENGINEERING MANAGEMENT PLAN
CONCEPT OF OPERATIONS
SYSTEM LEVEL REQUIREMENTS
SUB-SYSTEM REQUIREMENTS (HIGH LEVEL DESIGN)
COMPONENT DETAILED DESIGN
IMPLEMENTATION HARDWARE & SOFTWARE (CODING & TEST)
COMMISSIONED SYSTEM - OPERATIONS & MAINTENANCE
SYSTEM VALIDATION
SYSTEM VERIFICATION
SUBSYSTEM VERIFICATION
COMPONENT VERIFICATION
System Validation Plan
System Verification Plan
Sub-System Verification Plan
Component Verification Procedure
Traceability
Definition And Decomposition
Integration and Test
Time

Figure 3 The V-model of the Systems Engineering Process [7]

- *Reference Architecture Model Industrie 4.0 (RAMI 4.0)* [8] is a three-dimensional map showing the most important aspects of Industrie 4.0 (Figure 4). Its adoption ensures that all participants involved share a common perspective and develop a common understanding. The model is described on three axes [9]:
  - *Hierarchy Levels*: horizontal axis based on the IEC 62264 Enterprise control system integration.
  - *Life Cycle and Value Stream*: axis representing the life cycle of product/service, which is based on IEC 62890 Life Cycle Management. The product life cycle in the context of the RAMI 4.0 is described as an Engineering Process with two stages; Type and Instance. A type become an instance when design and prototype have been complete and the actual product is being manufactured. In the two stages we have two phases for each:

        ▪ Stage *Type* is implemented in two phases; Development followed by the Maintenance/Usage phase.
        ▪ Stage *Instance* is implemented with the Production phase and the Maintenance/Usage phase.
    o *Layers*: vertical axis describing the decomposition of product/service in a way to enable its virtual mapping (Business, Functional, Information, Communication, Integration, and Asset).

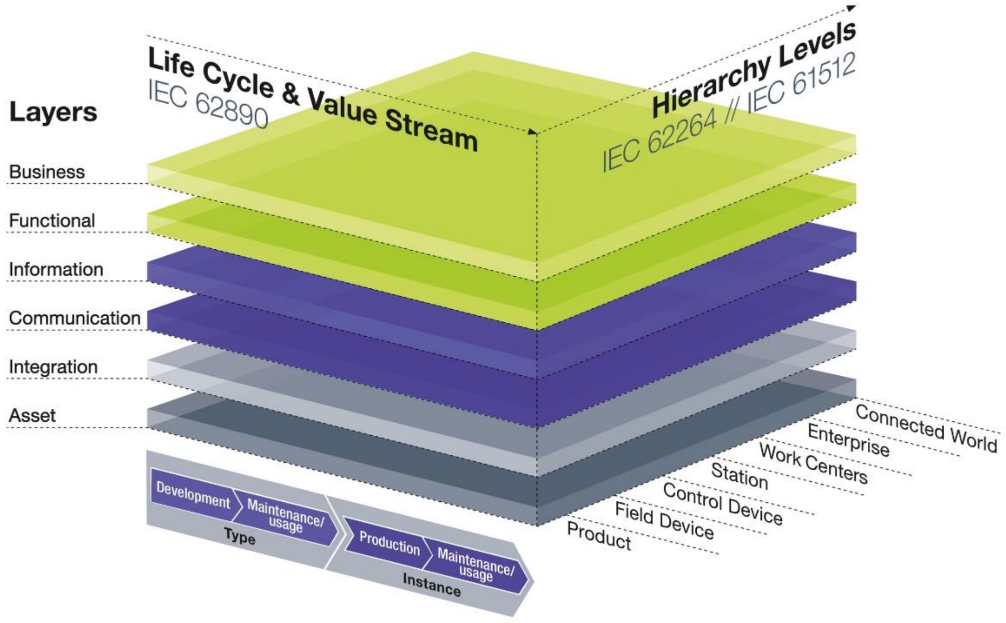


Figure 4 Reference Architecture Model Industrie 4.0 (RAMI 4.0).

- *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 (shown in Figure 5) 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.

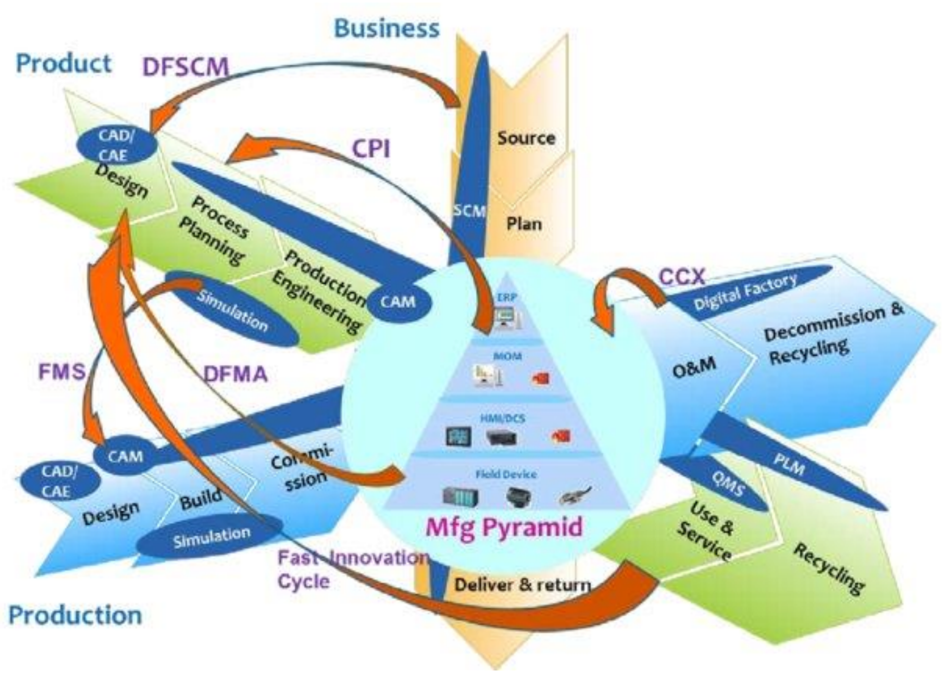


Figure 5 Smart Manufacturing Ecosystem [10].

- *The Industrial Internet Reference Architecture (IIRA)* is a standardised 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, Implementation). The IIRA standard support a flexible strategy for the product/service life cycle definition that can be specialised for each industrial sector depending on the use case needs (Figure 6).

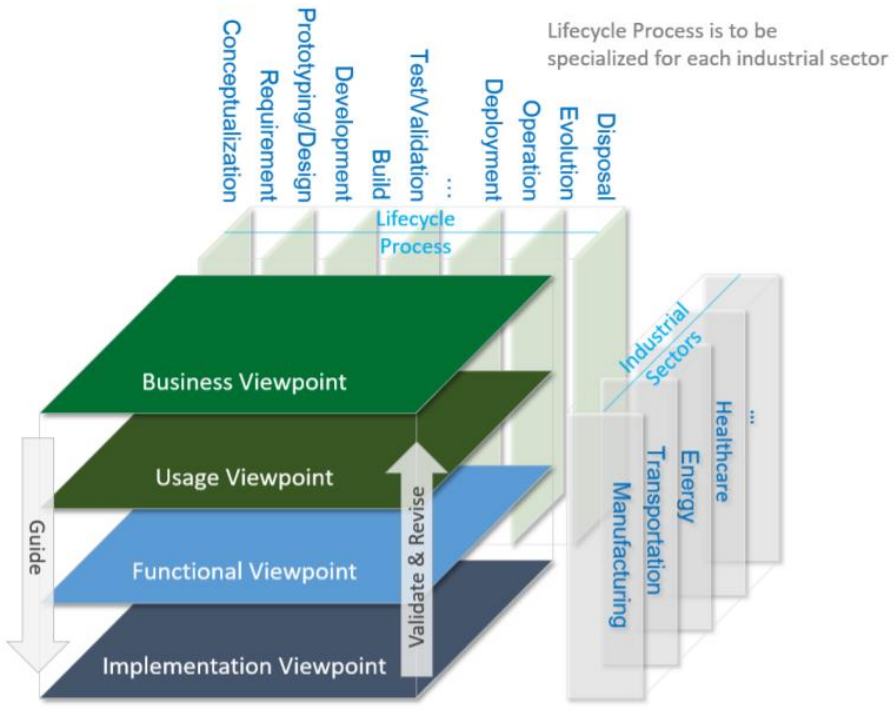


Figure 6 Relationship among IIRA Viewpoints, Application Scope and System Lifecycle Process [11].

Standards mentioned above will be deeply investigated during the project to identify key features to be supported by the AHT-EP architecture.

3.1. The ontology for satisfying the WP2 objectives

In alignment with partners of WP1 and WP4 an ontology has been created for calling all the various components of the Engineering Process that will be implemented for matching the WP2 objectives.
In general, we call the full **ArrowHead-Tools Engineering Process** as AHT-EP. The AHT-EP, for matching the use case specific life-cycle flow, can be built by using the **Engineering Process Units** (EPU) that are classified as:

A. **Engineering Process Phase (EPP)** In the following the full list of EPP names associated with the relative acronyms:
   - EPP1: Requirements
   - EPP2: Functional Design
   - EPP3: Procurement & Engineering
   - EPP4: Deployment & Commissioning
   - EPP5: Operation & Management
   - EPP6: Maintenance
   - EPP7: Evolution
   - EPP8: Training

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
     - EP-I7: Input for Evolution
     - EP-I8: Input for Training
   - 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
     - EP-O7: Output of Evolution
     - EP-O8: Output of Training

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

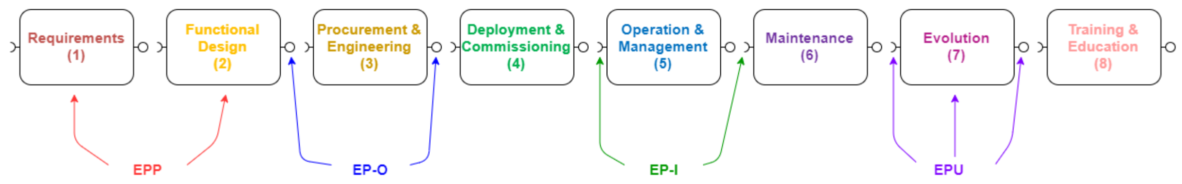


Figure 7 ArrowHead-Tools Engineering Process and Engineering Process Units

In order to support the 22 use cases of the different domains, the AHT-EP can be designed by connecting the EPPs in a customised flow that is not necessary the succession of phases proposed in the Figure 7. Moreover, 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/service 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.
In the following the representation of two EPPs with multiple input/output interface enumerated with the proposed system.

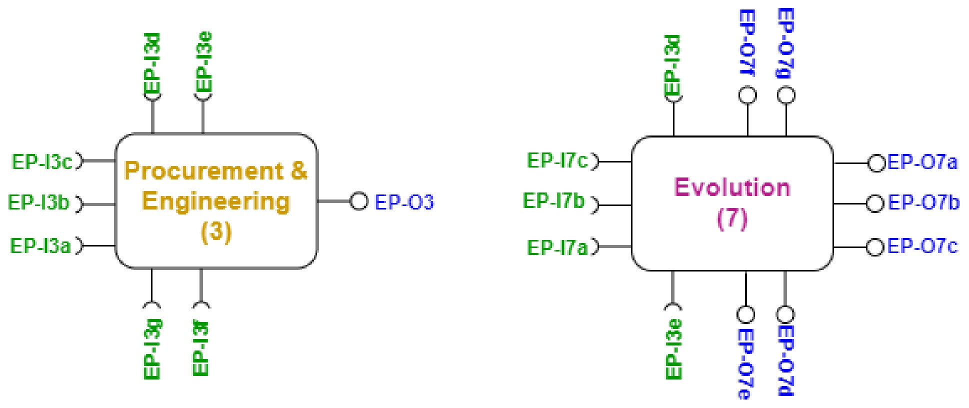


Figure 8 Rule for enumerating the multiple EP-I/EP-O of a single EPP

To minimise 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 9 can be described as EPP1->EPP2 instead of EPP1, EP-O1, EP-I2, EPP2 that represent the list of all the EPUs.

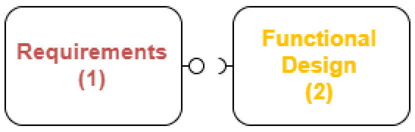


Figure 9 EPP1 connected to EPP2

For representing the connection graph of the AHT-EP we can adopt a standardised 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.

In the project, we will evaluate if it is the case to generate a specific file format supported by a parser tool for the automatic management of the EPP structure information.

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 summarises the life-cycle flow architecture.

In the first example, shown in Figure 10, we have the EPP1 connected, by using interfaces EP-O1 and EP-I2, to the EPP2. EPP2 gives inputs to EPP3 and EPP8 while receive inputs from the evolution phase (EPP7). Procurement & Engineering (EPP3) is connected with EPP4 by using interfaces EP-O3 and EP-I4 and receives inputs from EPP6 and EPP7. 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).

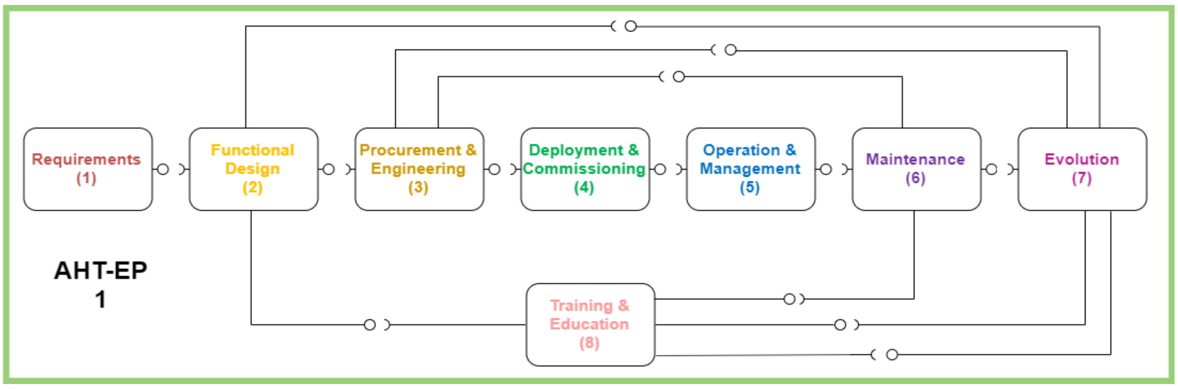


Figure 10 AHT-EP with all the EPPs connected in forward/feedback with multiple EP-Is and EP-Os

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 10

EPP1	EPP2		
EPP2	EPP3	EPP8	
EPP3	EPP4		
EPP4	EPP5		
EPP5	EPP6		

EPP6	EPP3	EPP7	
EPP7	EPP2	EPP3	EPP8
EPP8	EPP6	EPP7	

In Figure 11, we propose a second 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 is the main process that uses seven of the eight phases and it is connected with two external engineering processes. The AHT-EP 2 is composed by three EPPs, where two (EPP1 and EPP4) are connected with the EPP2 and EPP3 of the AHT-EP 1, respectively.
The external EP receive inputs from the EPP6 of the AHT-EP1 and provides inputs in the EPP3.

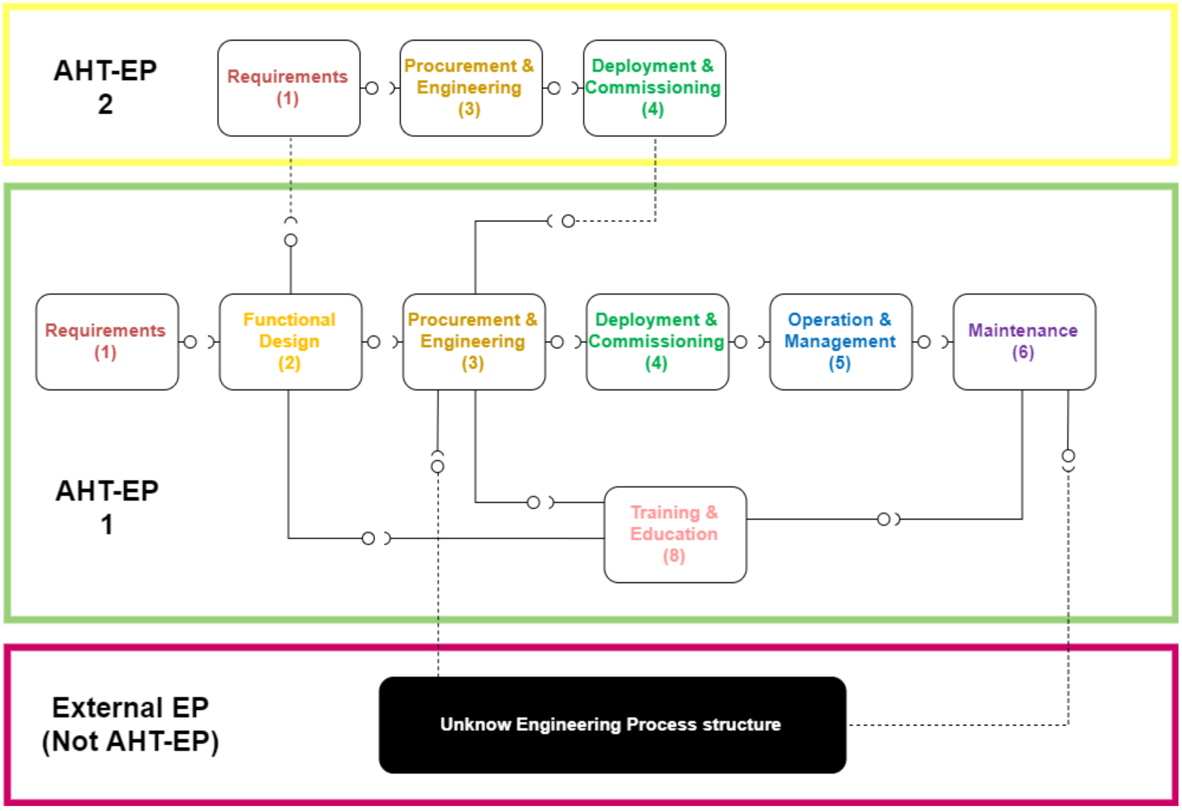

AHT-EP 2
Requirements (1)
Procurement & Engineering (3)
Deployment & Commissioning (4)
Requirements (1)
Functional Design (2)
Procurement & Engineering (3)
Deployment & Commissioning (4)
Operation & Management (5)
Maintenance (6)
AHT-EP 1
Training & Education (8)
External EP (Not AHT-EP)
Unknow Engineering Process structure

Figure 11 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 2.

Table 2 Tabular representation of the graph of EPP connection of Figure 11

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 customised 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 and Education phase.

3.2. The Engineering Process and the eight Phases

The Engineering Process will be designed as a flexible system for supporting the Life cycle of all the use cases of the AHT project.
In the Table 3, we provide a short description of the eight phases of the proposed ArrowHead-Tools Engineering Process (AHT-EP).

# Phase	Leader	Phase title	Phase description
1	PHC	Requirements	*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.
2	ULMA	Functional design	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.
3	KAI	Procurement & Engineering	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 and construct/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. The *engineering* phase 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).
4	DAC	Deployment & Commissioning	The *deployment* 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. The *commissioning* 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 precedes the operations & management phase.

5	IFAT	Operations & management	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.
6	IKERLAN	Maintenance	*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 has to 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. In the maintenance phase, we can also consider the de-commissioning of the system/product at its end-of-life.
7	ABB	Evolution	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 ensure also a continuous improvement of the system/product, always respecting the user requirements in an efficient, reliable and flexible way. Finally, this phase has to deal with the management of the end-of-life of the system/product.
8	Magillem	Training & Education	This phase includes all the educational and professional training activities required by the engineering process, across the entire system/product lifecycle.

Table 3 The eight AHT-EPPs

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

3.2.1. Requirements (EPP1) < SubTask 1 >

The word requirement refers to what is needed or wanted. It might be something that is compulsory or a necessary condition. Elicitation of requirement 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 which most likely was only a vague idea at the beginning and now evolves in a real structure. 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 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.

The ArrowHead-Tools project considers using a service oriented architecture paradigm to achieve its six objectives. How this could be achieved between phases will be defined in WP4 during the project. Here, we can think solutions based on ArrowHead Framework. Each stakeholder gets its own database to handle the list of requirements. Similar to the RAMI 4.0 [8], the databases are assets with an 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.

To illustrate this early concept, 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, 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 10, tools within EPP1, EPP3, EPP7, and EPP8 interacting with tools in EPP2.

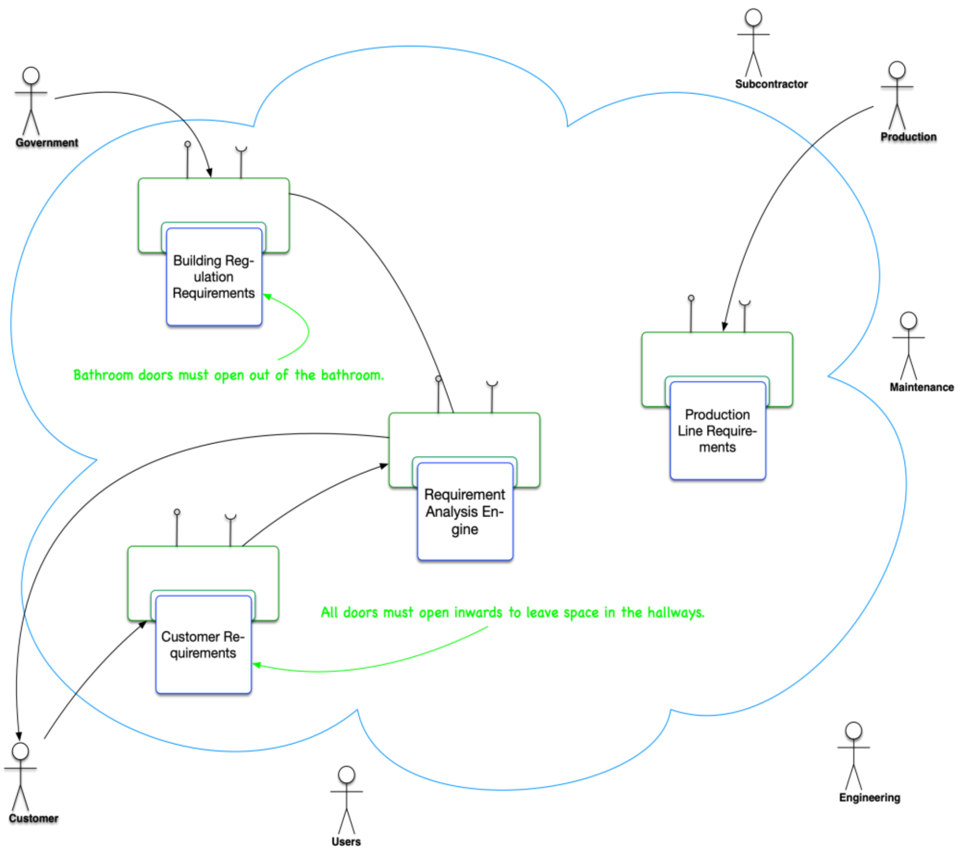

Government
Subcontractor
Production
Building Regulation Requirements
Bathroom doors must open out of the bathroom.
Production Line Requirements
Maintenance
Requirement Analysis Engine
Customer Requirements
All doors must open inwards to leave space in the hallways.
Customer
Users
Engineering

Figure 12 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 [8] 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 [12] [13]. 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 12, but in this early stage, that is only concept conjecture.

3.2.2. Functional design (EPP2) < SubTask 2 >

Functional analysis and design are key activities in the Systems and Software Engineering process [14] [15] 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 [16]:

- 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) [15] or a product tree that serves engineers to have a first distribution of the system architecture. 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 [17] 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.
- *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) [18] [19] 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 lifecycle model, it means that there is “formalized application of modelling” to support the left-hand side of this system lifecycle 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 [20] 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, 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 [21] 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 lifecycle, it makes sense to describe the two main types of models that can be found according to the Systems Engineering Body of Knowledge (SEBoK) [22]:

- 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 lifecycle, 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.

That is why, in the context of AHT project, the focus will be on the application of an MBSE methodology to cover the different phases of the engineering process and the views of model (MIL) and software in the loop (SIL), focusing on reuse capabilities of functional design. More specifically, in the UC-03 “*Integration of electronic design automation tools with product lifecycle tools*” a tool chain is being defined (see WP1 requirements and Figure 13) and will be implemented through different tools to support the technical engineering processes listed in Table 4 that have impact in the functional design of a new product and/or service.

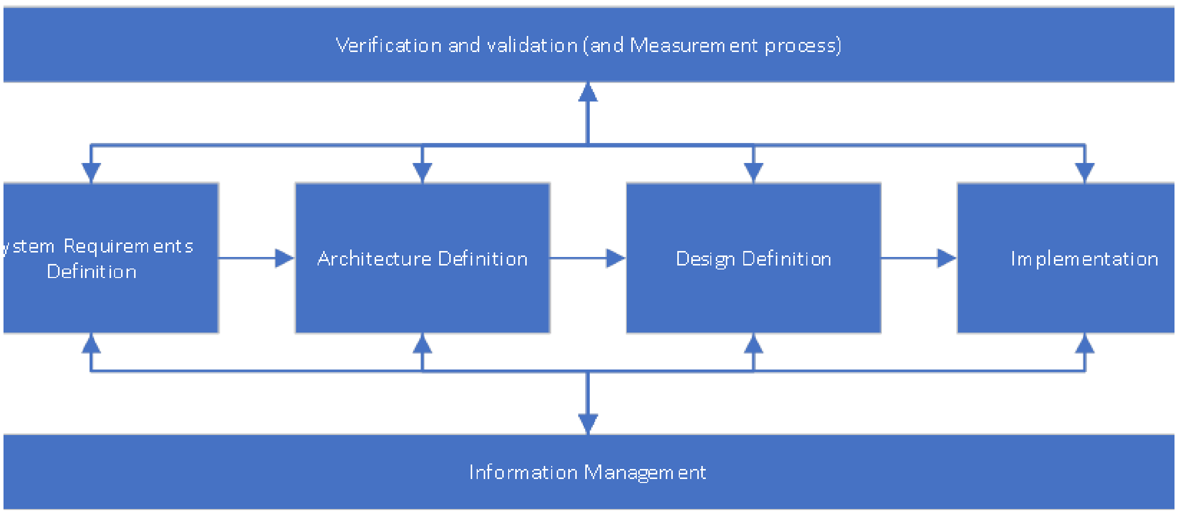

Verification and validation (and Measurement process)
ystem Requirements Definition
Architecture Definition
Design Definition
Implementation
Information Management

Figure 13 UC-03 Engineering process based on ISO 15288

Table 4 Summary of engineering processes, methods, techniques and tools for UC-03

Engineering process	**Engineering methods**	**Techniques**	**Tools**	**License**
System Requirements Definition	Requirements Engineering	Natural language requirements	IBM Doors, Requirements Authoring Tool	Proprietary and APIs based on standards.
Architecture Definition	Logical modelling	Diagramming with SysML/UML	IBM Rhapsody	Proprietary and APIs based on standards.
Design definition	Physical modelling	Diagramming	Altium designer	Proprietary
Implementation	Simulation	Programming, simulation configuration	Altium designer, native code	Proprietary
Verification & Validation (Measurement process)	Quality management	Quality metrics	Verification Studio	Proprietary and APIs based on standards.
Information Management	Knowledge engineering	Ontologies	KnowledgeManager	Proprietary and APIs based on standards.
Information Management	Knowledge management	Traceability discovery	KnowledgeManager	Proprietary and APIs based on standards.

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

According to the INCOSE Systems Engineering Handbook [14], 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” [15] where procurement is mainly referred to in the acquisition process.

The engineering phase, described in [14] and [15] as well, (see Figure 14 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 lifecycle model and organization or project specific restrictions, this process may change.

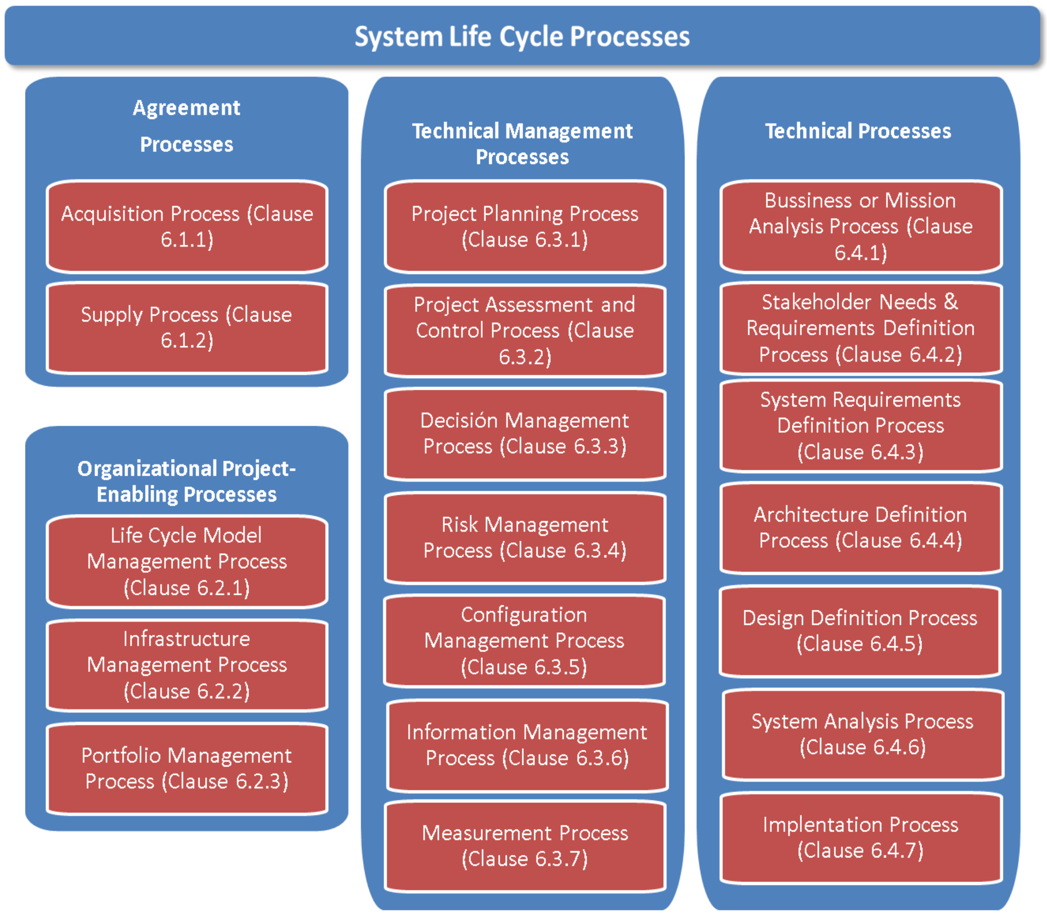

System Life Cycle Processes
Agreement Processes
Acquisition Process (Clause 6.1.1)
Supply Process (Clause 6.1.2)
Organizational Project-Enabling Processes
Life Cycle Model Management Process (Clause 6.2.1)
Infrastructure Management Process (Clause 6.2.2)
Portfolio Management Process (Clause 6.2.3)
Technical Management Processes
Project Planning Process (Clause 6.3.1)
Project Assessment and Control Process (Clause 6.3.2)
Decisión Management Process (Clause 6.3.3)
Risk Management Process (Clause 6.3.4)
Configuration Management Process (Clause 6.3.5)
Information Management Process (Clause 6.3.6)
Measurement Process (Clause 6.3.7)
Technical Processes
Bussiness or Mission Analysis Process (Clause 6.4.1)
Stakeholder Needs & Requirements Definition Process (Clause 6.4.2)
System Requirements Definition Process (Clause 6.4.3)
Architecture Definition Process (Clause 6.4.4)
Design Definition Process (Clause 6.4.5)
System Analysis Process (Clause 6.4.6)
Implentation Process (Clause 6.4.7)

Figure 14 Overview of the Systems and software engineering — System life cycle processes ISO 15288: 2015 [23].

Procurement and 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.

3.2.3.1. 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 15 is run through.

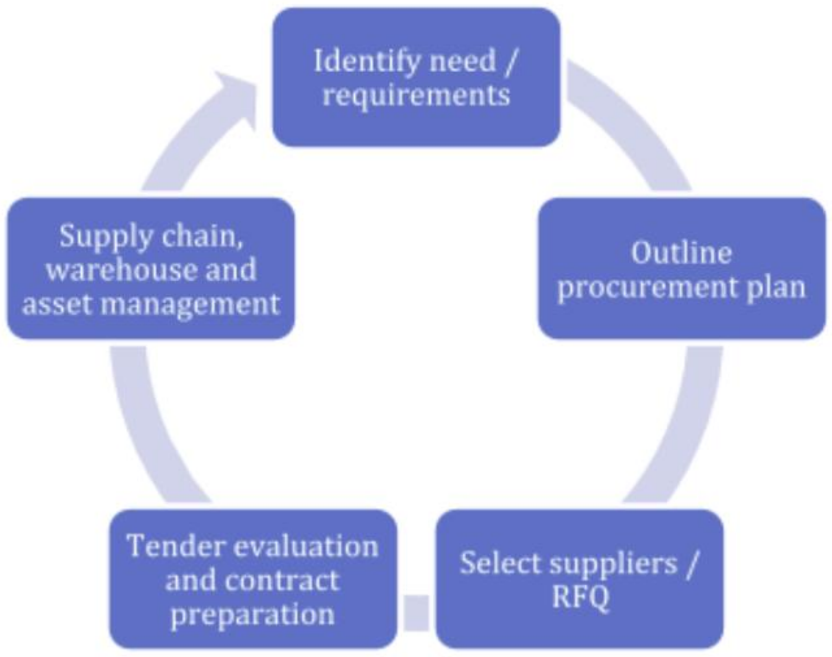

Identify need / requirements
Outline procurement plan
Select suppliers / RFQ
Tender evaluation and contract preparation
Supply chain, warehouse and asset management

Figure 15 Procurement cycle [24]

First of all, dependent on the process to be improved or adapted during the procurement phase, the **need** and corresponding **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.

3.2.3.2. 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.

3.2.3.3. UC tasks related to the “Procurement and Engineering Phase”

3.2.3.3.1. 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 16).

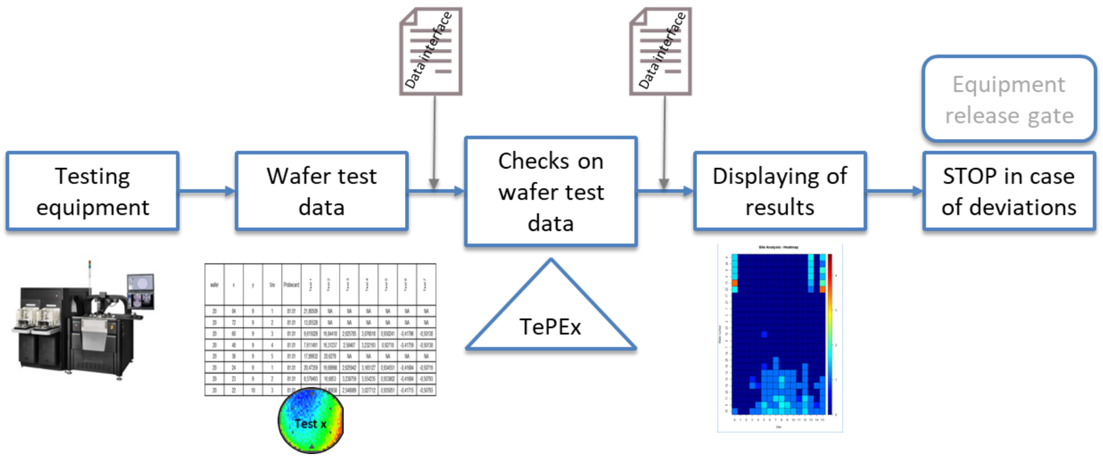


Figure 16 Two data interfaces are needed for TePEx

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 17.

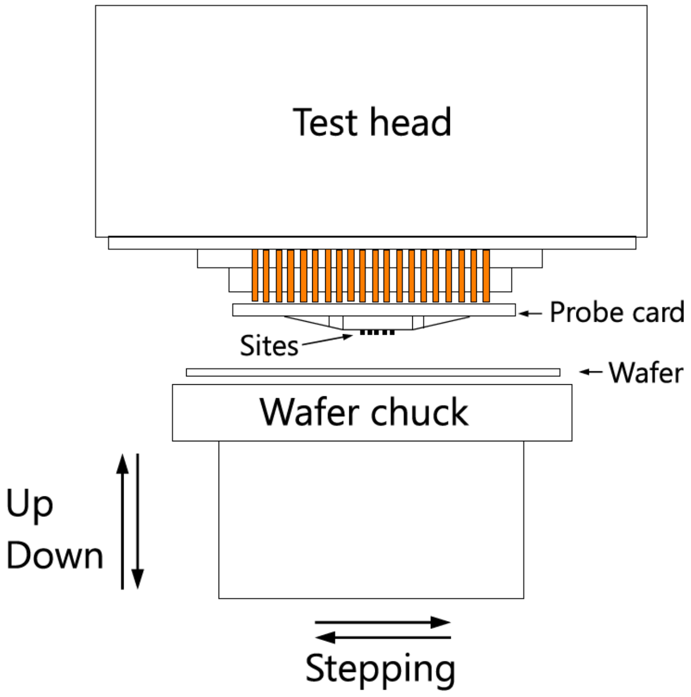


Figure 17 Testing Equipment

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 18).

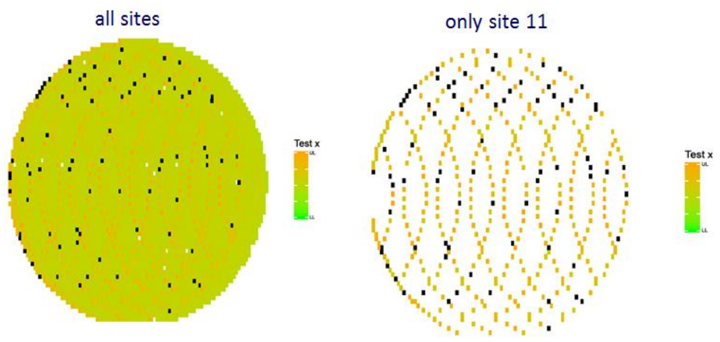


Figure 18 A systematic pattern is visible on the wafermap of the full wafer (left) and can be related to a single site, Site 11 (right).

Since a probe card consists of multiple sites (see Figure 17), 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 16.

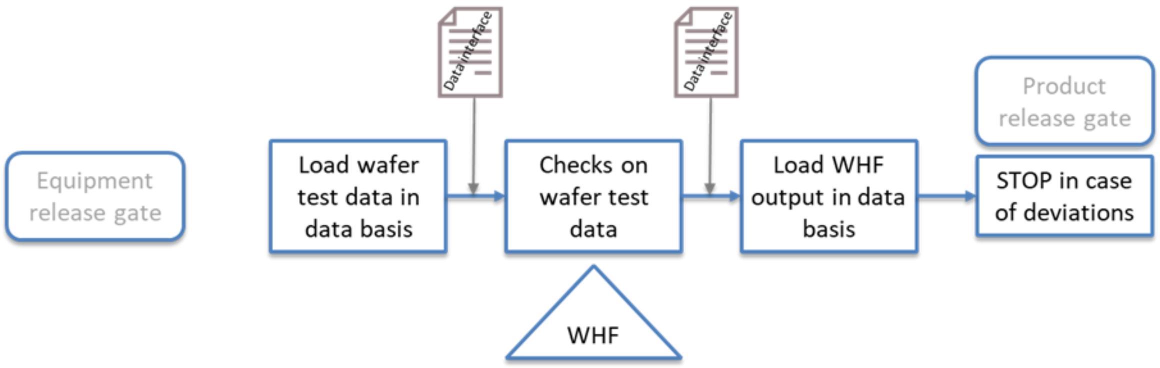

Data interface
Data interface
Equipment release gate
Load wafer test data in data basis
Checks on wafer test data
Load WHF output in data basis
Product release gate
STOP in case of deviations
WHF

Figure 19 Two interfaces are needed for WHF

For the WHF (Figure 19), 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.

3.2.3.3.2. UC-07: CNC Machine Automation

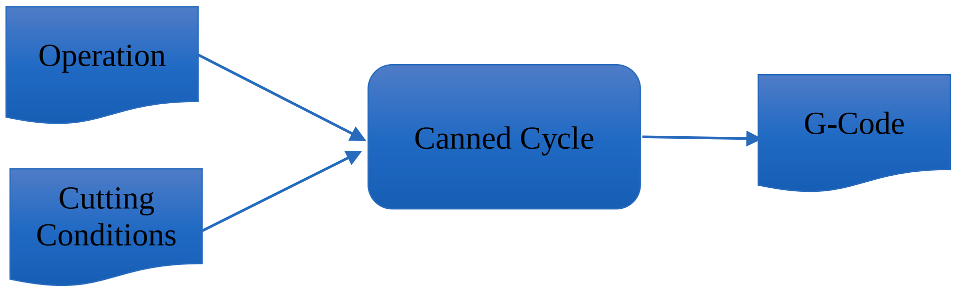
The use case presented under this title includes many scenarios. One of them relates with the Engineering phase Procurement & Engineering. There are different actors in the UC-07, taking part mainly in the deployment and commissioning phase, but there is an important, especially from the point of view 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 20). 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:

1. 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.

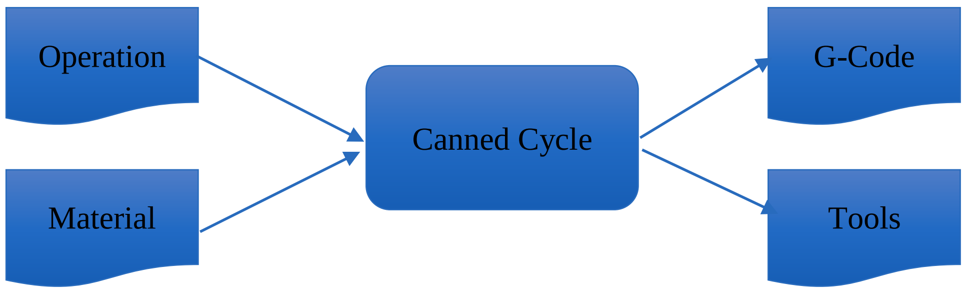
2. Canned cycle: This is the traditional canned cycle. Usually it must be provided as specific G-Code for the target CNC due to the need of doing math inside it to find the points that the cutting tool must follow.

3. 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.



A simple drawing of the above described process follows 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:



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.

3.2.3.3.3.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.

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

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 services/product 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.

3.2.4.1. 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 ArrowHead 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, and allows services to discover and connect on demand. They should, however, be accordingly authorised and registered in the Service Registry.

3.2.4.2. Commissioning

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

1. **Unit tests** - usually performed before deployment to verify whether single units of the whole solution (methods, services) are working properly.

2. **Integration tests** - performed with an aims of identifying any defects on interfaces between services, and assuring proper interaction between the parts of the system.

3. **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.

4. **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.

Examples of tools

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

A. **Docker image** of the ArrowHead Framework core services.
B. **System testing tools**, with the aim of analysing cross-integration and communication between interfaces at the time of the first run.
C. **Configuration tool** that supports the final configuration of a local cloud with the associated components and services.
D. **Code generation tool** for e.g. microcontrollers, Programmable Logic Controllers (PLCs) or sensing modules.

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

The Operation & Management phase (a.k.a. Engineering Process Phase 5) is the dominant phase of the whole collection of engineering phases. It is the productive part of the engineered artefact’s lifecycle, which justifies the existence of the concept. In a production context, it is long lived. Its importance however cannot overshadow that of the other engineering phases as it is itself continuously dependent on the output from the other phases, and in our current theme, of the tools within all phases.

The Operation & Management phase is extremely complex. There is not one asset (person or tool) that is not somehow related to this phase. If such would be the case, that asset ought to be removed through an optimization process. Taking a human perspective, one can easily picture engineers planning a production facility and following its performance. Technicians install, operate, and maintain the production facility. At the same time administrative staff ensures the procurement of components at the best prices at the right time, while sale persons promote the products. At strategical level, managers optimize production plans by chasing waste.

The most interesting part is sharing lunch with any one of these groups. Although all have the same overall goal (i.e., to stay in business by producing efficiently) they have different opinions of the other groups. What might be obvious to one group is missed by another. There is a communication (or understanding) failure that leads to a derailed optimization. This being an analogy to communication failure in between tools (e.g., interoperability) used by the different groups hampers efficiency and productivity.

When one first looks at the engineering phases (Figure 1), they seem to be sequential in time. That might have been the case before when tools used in operation did not need to communicate. Some time ago, the draftsman's pen did not need to communicate with the operator's wrench nor the accountant's ledger. But nowadays, with the industrial Internet revolution, all assets are interconnected to each other. This occurred when micro-controllers became powerful enough to connect to the Internet. The whole concept has propelled concepts such as Industry 4.0. But being connected to the Internet does not mean that the tools can cooperate. Most often they are not yet interoperable, which justifies the AHT project.

This whole phase, Operation & Management, can be summarized by the ISA-95 standard composed of five parts [25]. Each part of the standard is relevant, going from hierarchy to data exchange between different levels (Figure 22). The hierarchy is usually described as a pyramid with the top level (level 4) being the Enterprise Resource Planning (ERP). At that level, the software tools handle company wide data. At level 3, one finds the Manufacturing Execution System (MES) where the tools follow the manufacturing processes. At level 2 is the Supervisory Control and Data Acquisition (SCADA) system, which collects and integrates data from the automation equipment on factory floor. Control of the machines on the factory floor are on level 1 with field devices are at the level below.

The ISA-95 standard is still relevant and being maintained [26]. At its origin, the levels had clear distinctions as the communication at each level was specific and the inter-level communication was limited. With the pervasive use of the Internet and its suite of protocols, this is not the case anymore [27]. New and correct solutions need to be developed to harvest the potential of this emerging situation, which is very complex.

ISA (International Society of Automation) published an illustration of how complex is this situation (Figure 22). The focus here is the dynamics within level 3, the MES level. The software tools need to be exchanging the right information at the right time in a secured manner.

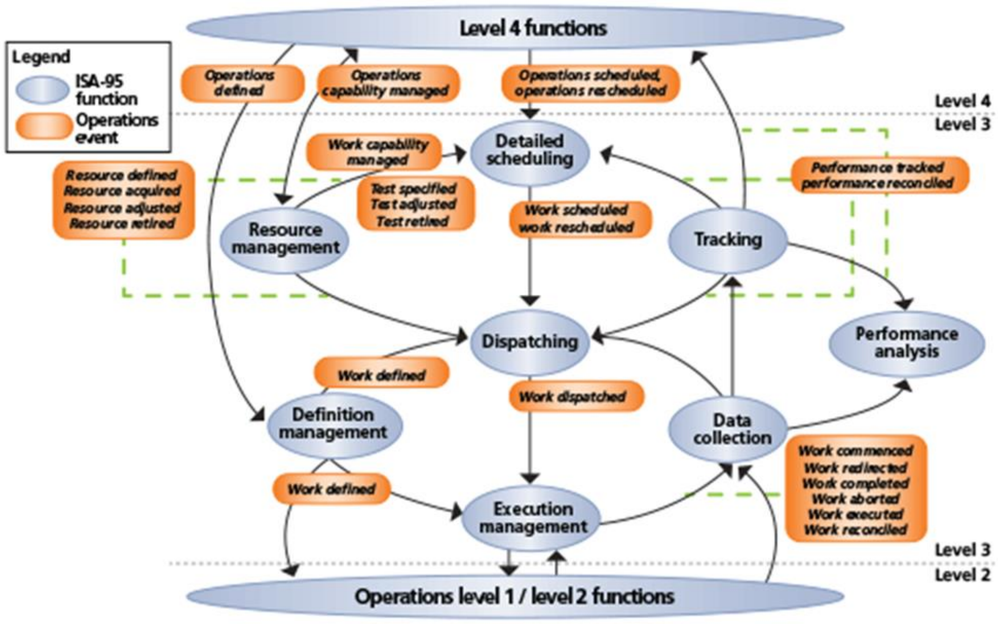
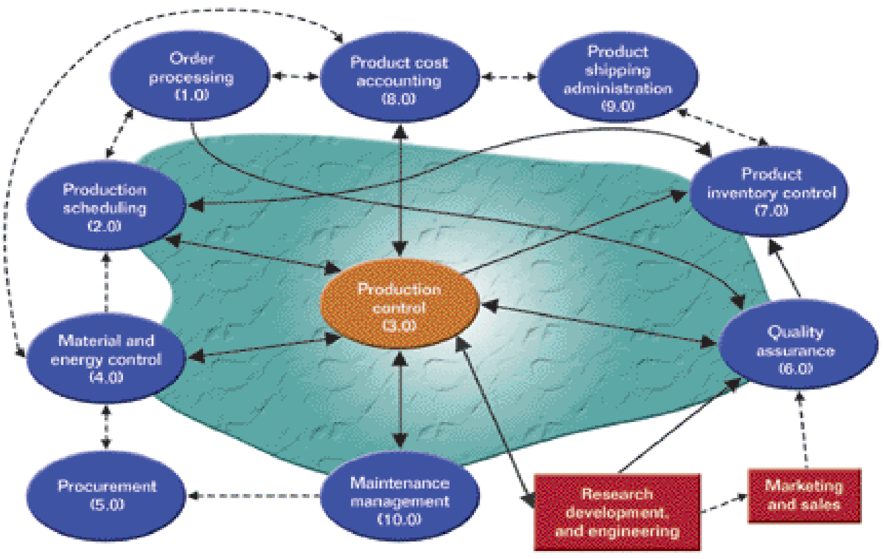


Figure 22 ISA-95 standard.

Efforts to interoperability are not new, which is obvious with standardization endeavours. An example is the CAEX - IEC 62424 standard [28]. CAEX (Computer Aided Engineering Exchange) is a neutral data format that allows storage of hierarchical object information [29]. One appealing aspect of CAEX is its relationship with AutomationML (Automation Markup Language) as it might prove to be of interest with the idea of reconfiguration at runtime within the ArrowHead-Tools project. The goal of AutomationML is to interconnect the heterogeneous tool landscape of modern engineering tools in their different disciplines, e.g. mechanical plant engineering, electrical design, HMI development, PLC, robot control [30].
WP2's general description, is stated as the Digitalization of the Engineering Process, aims to develop a consolidated engineering process model that relies on a service oriented architecture (SOA), which can be implemented using an integration platform based on WP3 (Digitalization framework: Integration & Interoperability) and WP4 (Tools chain architecture) results. One way to interpret this is that the tools offer services over the Internet to each other. Reaching into the Arrowhead Framework, one can get the inspiration that legacy tools are seen as assets that have a SOA interface or wrapper enabling them to exchange information. The cooperation with WP3 and WP4 are compelling as they will be applied to WP7, 8 and 9.
ISA had even earlier published Figure 23 to illustrate the complexity in modern manufacturing, which is relevant to the current deliverable. One can see names of some of the other AHT-EP. The figure elucidates well the description of the proposed ontology in section 3. For example, the tools of the Procurement phase have to interact with the tools of the Operation phase.


Order processing (1.0)
Product cost accounting (8.0)
Product shipping administration (9.0)
Product inventory control (7.0)
Production scheduling (2.0)
Production control (3.0)
Quality assurance (6.0)
Material and energy control (4.0)
Procurement (5.0)
Maintenance management (10.0)
Research development, and engineering
Marketing and sales

3.2.5.1. Example use case

The 22 use cases of the AHT will all demonstrate the digitalization of the engineering processes and the tools' interaction thought a service oriented architecture. To make things concrete in an Operation & Management context, we make us of one such use case (UC-16), specifically the one covered in task 9.4.
Main objective of task 9.4 is to monitor and control processes in the semiconductor production by integrating different sensors. This supports the continuous improvement of the semiconductor production. Besides improving the production processes, energy efficiency is also of interest. Sensors will not be only used in the semiconductor front-end production, but also in facility areas such as special vibration sensors that support improving maintenance processes. In addition to that, Big Data analytics are going to set new production and maintenance strategies.
The state of the art in the semiconductor industry is that sensors are normally purchased together with the semiconductor equipment. Infineon in Dresden (IFD) is taking care of equipment integration including all sensor data. Main objective at IFD for 300mm is to integrate all facilities based on IFD´s own IT integration concept. Concerning a single sensor, data should use standard SECS/GEM interfaces and the equipment must follow 300mm Semi standards.
The next two figures are showing the actual state and the desired target architecture for semiconductor fabs within the ArrowHead-Tools project how sensors should be integrated in the future using all required interfaces. The actual state (Figure 24) shows that the production equipment is connected to the Data Management via the Equipment Automation Framework.

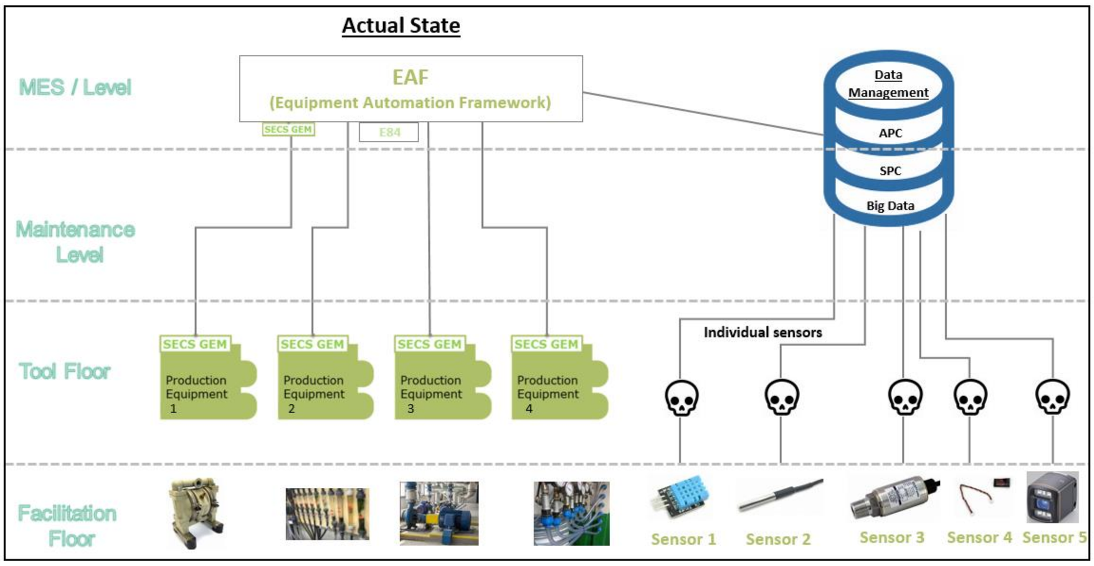


Figure 24 Actual state of UC-16

However, there is no sensor integration, which constitutes a huge effort and is not easy to handle it for the management. In contrast to that, the target state shows how the ArrowHead framework is going to implemented and all sensors will be integrated as shown in Figure 25.

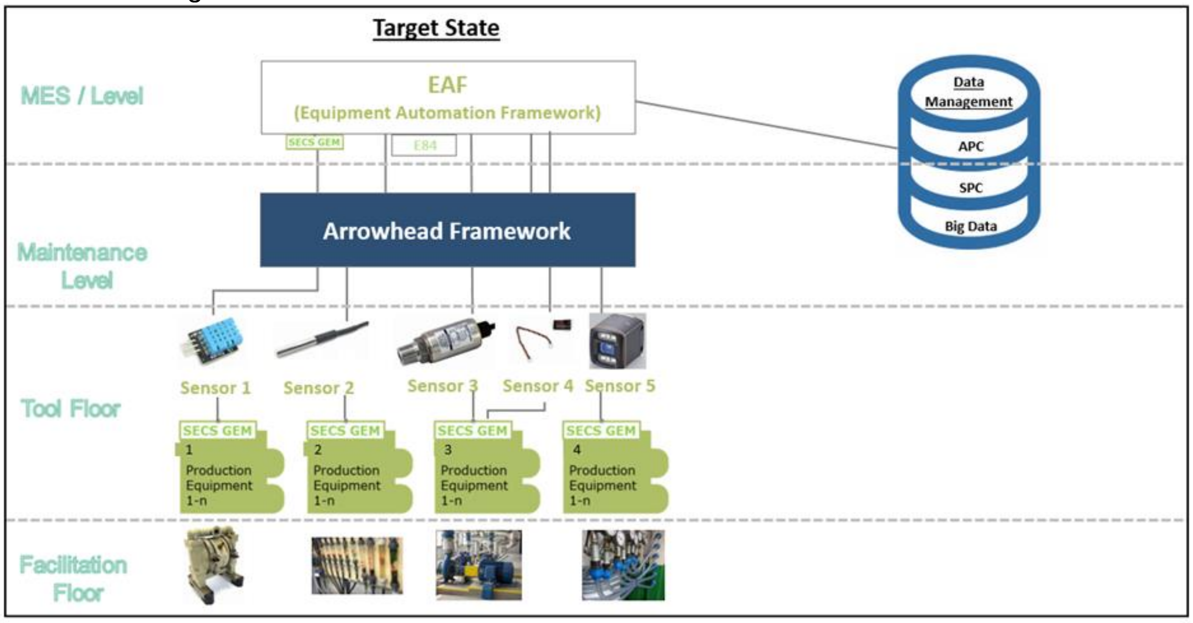


Figure 25 Target state of the UC-16

For other industries, an integration scheme is already provided of the University of Applied Sciences Munich (see Figure 26).

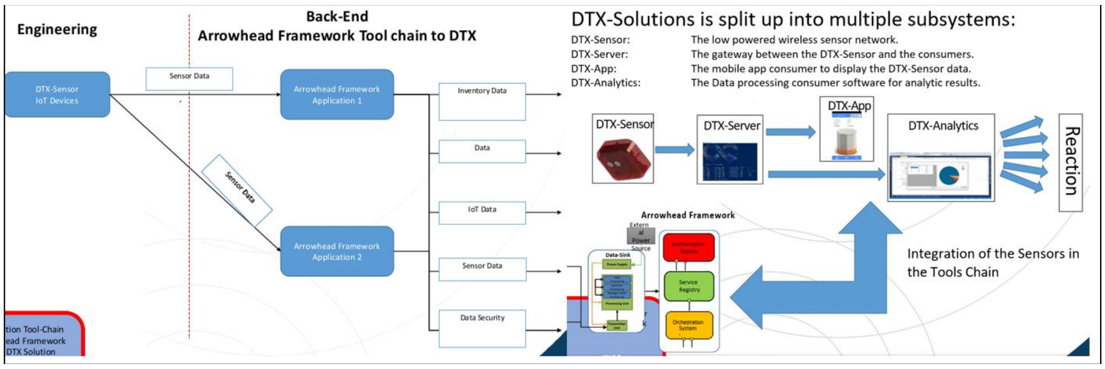

Engineering
Back-End
Arrowhead Framework Tool chain to DTX
DTX-Sensor IoT Devices
Sensor Data
Sensor Data
Arrowhead Framework Application 1
Arrowhead Framework Application 2
Inventory Data
Data
IoT Data
Sensor Data
Data Security
tion Tool-Chain
ead Framework
DTX Solution
DTX-Solutions is split up into multiple subsystems:
DTX-Sensor: The low powered wireless sensor network.
DTX-Server: The gateway between the DTX-Sensor and the consumers.
DTX-App: The mobile app consumer to display the DTX-Sensor data.
DTX-Analytics: The Data processing consumer software for analytic results.
DTX-Sensor
DTX-Server
DTX-App
DTX-Analytics
Reaction
Arrowhead Framework
External Power Source
Data-Sink
Service Registry
Orchestration System
Integration of the Sensors in the Tools Chain

Figure 26 ArrowHead Framework integration scheme

Topics like sustainability, flexibility, efficiency and competitiveness are high-level topics in today’s society, which are in turn driven by important societal issues, such as environmental sustainability, the availability of energy and other raw materials, and rapidly changing market trends. To contribute to these topics, the rising field of Internet of Things (IoT) can be tamed in combination with the approach of a SOA as Carlsson et al. propose in [31]. According to Evans the amount of IoT-devices will increase tremendously from 12.5 billion in the year 2010 to 50 billion in the year 2020 [32]. Setting up a solution needs to fill some gaps:

- Interoperability of a wide range of IoT and legacy devices.
- Automation requirement on latency guarantee/prediction for communication and control computations.
- Scalability of automation systems enabling very large integrated automation systems.
- Multi stakeholder integration and operations agility.
- Security and related safety of automation systems as well as the ease of application engineering.

With the ArrowHead Framework and its local cloud approach, a possible solution to address these issues is presented. The UC-16 is set up, to investigate the possibilities of coupling the ArrowHead Framework with machine learning algorithms to increase a factories efficiency, by detecting machine failures before they disturb the production. This use case is well defined in a Model-Based System Engineering approach, providing a better understanding of the intended system. The report structure follows the Design Research Methodology (DRM). An explanation of the DRM and a rationale why it was used in the described research project can be found in deliverable D9.1 [33].

The integration of new technologies and technological devices, e.g., new sensors, cameras or automation concepts, into existing production sites and IT infrastructures is currently labor intensive. Regularly, this results in isolated applications that run in parallel to existing IT solutions and are therefore difficult to maintain. Furthermore, the missing data integration makes it difficult to develop data analytics that incorporate the information from these devices.

Therefore, a general-purpose standard for data exchange and storage would be beneficial. As a minimum, this would require adapter services to different data sources and formats as well as the support for basic metadata sharing, such as sensor type, location, purpose and data type annotations. The Extended Historian service addresses this need by providing the functionality for scheduled and event-based data acquisition from different sensor types. It will provide an input/output adapter interface and several related adapters will be developed in the test and demonstration phase. This service and its Web-based configuration interface will be available as core functionality of the Arrowhead Framework.

Additionally, several measures within the sub use cases will further contribute to the reduction of engineering costs. For example, the automatic machine diagnostic and different sensor systems mentioned within the architecture chapter e.g. vibrational sensors or optical based sensors will be investigated for easy integration.

3.2.6. Maintenance (EPP6) < SubTask 6 >

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, 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 that 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 to may 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:

A. Healthiness: it is necessary to monitor the services deployed in the SOA, in order to verify if the services are working properly.
B. Deployed services: it is necessary to check which services are deployed in the architecture and which is the version deployed.
C. 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.
D. 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.

E. 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 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.

3.2.7. Evolution (EPP7) < SubTask 7>

ArrowHead-Tools 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.

Example: 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 utilised e.g. for condition monitoring, optimising the operation and performance of the product, and virtual testing of control or operation.

Evolution process
The evolution phase of system or product engineering process consists of several steps (Figure 27). Each of these steps is realised 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.

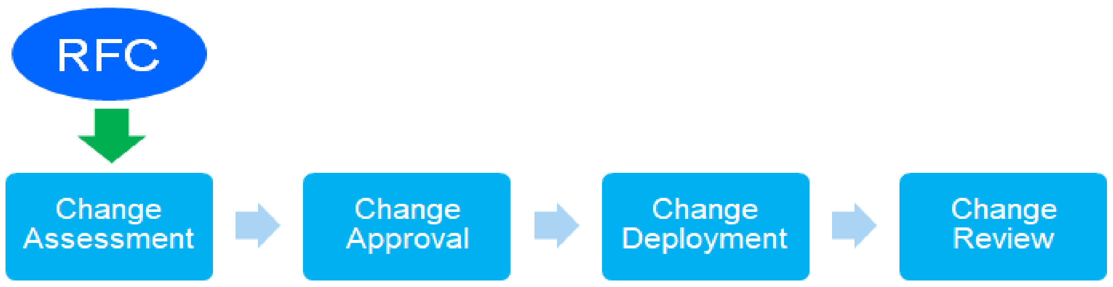


Figure 27 Evolution phase process in product or service engineering procedure

Life cycle and end-of-life

The product needs to be evolutionarily redesigned during its whole life cycle covering as a whole procurement, operation/use, maintenance, modernisation and disposal phases. Thus, the evolutionarily redesigning is kind of a constantly running circular procedure that should be utilising feedback from each of these different phases. For the optimisation 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. Modernise or change to a new, like the previous one?
8. When is time for disposal?

As described above, the evolution phase also includes 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. The management of EOL is answering to the questions 5-8 above. 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.

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

The goal of Training Activity 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 [34] consists of the first iteration.

The ArrowHead platform will be 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.

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

AHT will propose 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

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

In this section, we will analyse the most updated summary of the use cases provided by the UC leaders in the WP124 survey (point A.1) [4]. For each use case we will use info provided in section ***C*** 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/service life cycle. We will provide a picture describing what are the EPPs that each UC can exploit for structuring its own AHT-EP. In doing this analysis we will highlight what are the general AHT objectives and the WP1-WP2 objectives that each UC can potentially match by using each EPP.

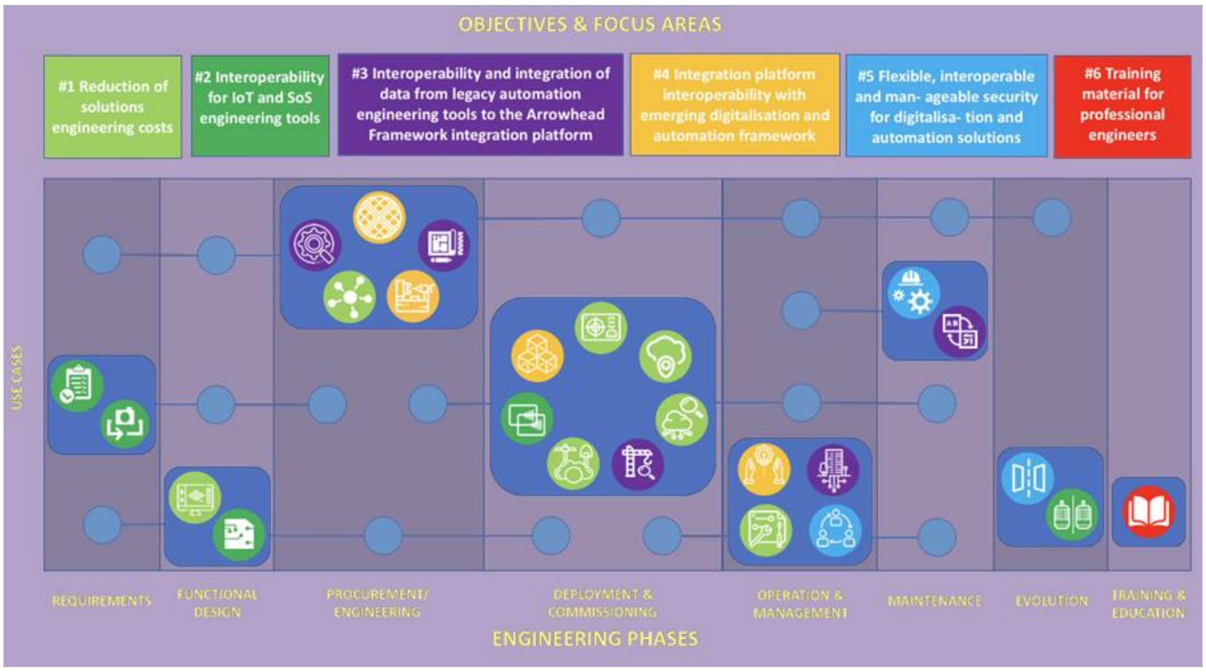
4.1. Updated Use Cases summary and AHT-EP analysis

In the document [4] we collect all the information provided by use case leaders that have been analysed during the definition of the AHT-EP feature, structure and components. Moreover, we provide a short summary for each use case describing the field of application and the main product/service to be developed a more detailed explanation can be found in WP1 deliverable D1.1 [35], WP4 deliverable D4.1 component [36], WP7 D7.1 [37], WP8 D81 [38], and WP9 D9.1 [33].
In the same document, that integrate the present deliverable, we reported all the information of the UCs describing the specific aspects related to the Engineering Process. Using these information we have generated a 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.
Then these tables are further summarized in the UC-EPP picture of the AHT project proposed in Figure 29.
Analysing the inputs collected in the document [4] we discovered that the life cycle of all the use cases can be well described by using the AHT-EP. Indeed, all except UC-02 have declared that the UC-EP steps can be mapped on the AHT-EP. The UC-02 leader expressed the needs to have an explicit EPP for the ***Information Management*** with the following argumentation:
“The main difference between UC-EP engineering processes and the AHP-EP is the notion of Information Management that in this use case, it is explicitly defined as a cornerstone to provide reuse capabilities and traceability management. More specifically, this case focuses on the concept of Knowledge-Centric Systems Engineering as a means to guide the engineering process exploiting all the data, information and knowledge that is generated during the development lifecycle and encoded in the system artefacts.”
During the project we will discuss this aspect, together with the new observations, for further improve the capability of the AHT-EP in supporting UCs with particular needs.
Many UCs have declared that the order showed in the AHT-EP picture is similar to the UC-EP. However, this linear order can be used only for small and easy UCs not requiring feedbacks and interactions with externa EPs from different stakeholders.
An example of such a non-linear EPs is the Engineering process currently adopted in the UC-02 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 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 ArrowHead Tool framework, they can implement and verify a more automated yet secure way of transferring data in the information flow.

4.2. 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/services to be developed in the project.
The analysis is based on the information provided by use case leaders reported in the document [4].
In the DoA, [39] it was proposed an alignment between use cases and the eight phases of the AHT-EP. This alignment was summarised in the picture in Figure 28 where each use case, represented by one of the circular coloured icons, was assigned to the EPP in which the UC is expected to have the major contribution.


OBJECTIVES & FOCUS AREAS
#1 Reduction of solutions engineering costs
#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 digitalisation and automation framework
#5 Flexible, interoperable and man- ageable security for digitalisa- tion and automation solutions
#6 Training material for professional engineers
USE CASES
REQUIREMENTS
FUNCTIONAL DESIGN
PROCUREMENT/ ENGINEERING
DEPLOYMENT & COMMISSIONING
OPERATION & MANAGEMENT
MAINTENANCE
EVOLUTION
TRAINING & EDUCATION
ENGINEERING PHASES
Figure 28 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 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 colour of the main AHT objective that the UC can potentially satisfy.
We have analysed the information provided by the UC leaders [4] for creating a new and more detailed version of this picture (Figure 29) that shows 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 is provided in Figure 30.
On top of the Figure 29, 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.

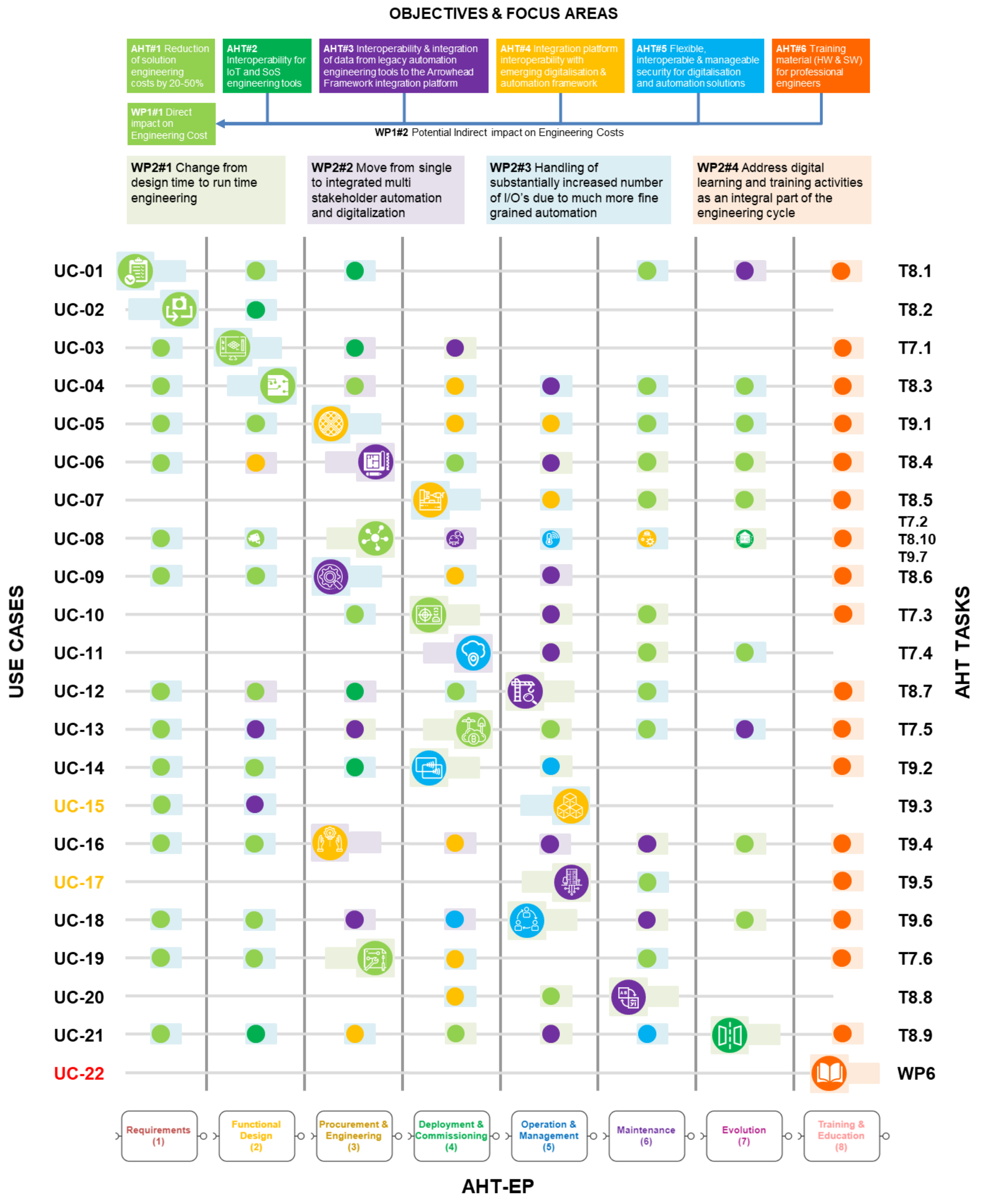
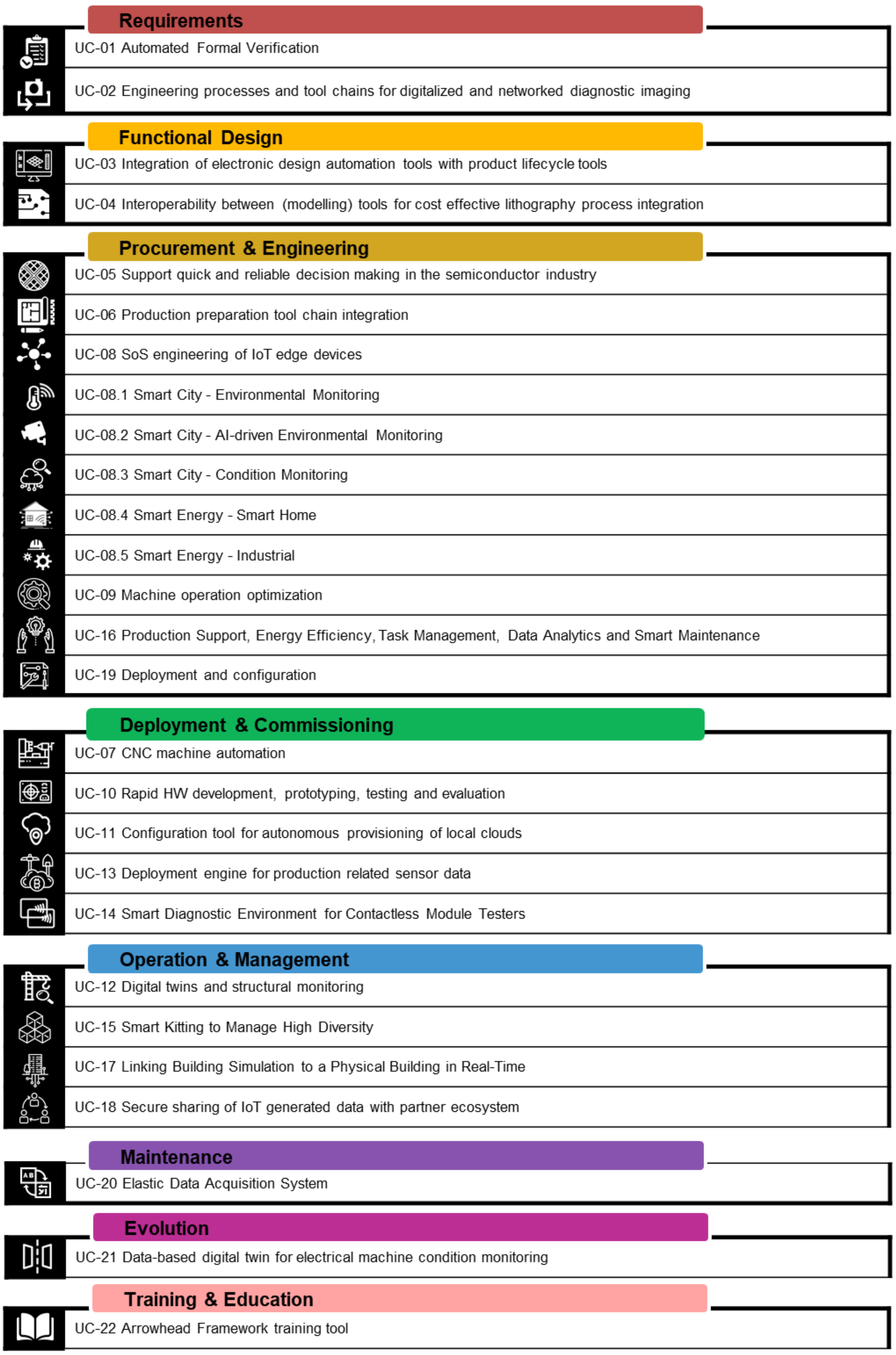

OBJECTIVES & FOCUS AREAS
AHT#1 Reduction of solution engineering costs by 20-50%
AHT#2 Interoperability for IoT and SoS engineering tools
AHT#3 Interoperability & integration of data from legacy automation engineering tools to the Arrowhead Framework integration platform
AHT#4 Integration platform interoperability with emerging digitalisation & automation framework
AHT#5 Flexible, interoperable & manageable security for digitalisation and automation solutions
AHT#6 Training material (HW & SW) for professional engineers
WP1#1 Direct impact on Engineering Cost
WP1#2 Potential Indirect impact on Engineering Costs
WP2#1 Change from design time to run time engineering
WP2#2 Move from single to integrated multi stakeholder automation and digitalization
WP2#3 Handling of substantially increased number of I/O's due to much more fine grained automation
WP2#4 Address digital learning and training activities as an integral part of the engineering cycle
USE CASES
UC-01
UC-02
UC-03
UC-04
UC-05
UC-06
UC-07
UC-08
UC-09
UC-10
UC-11
UC-12
UC-13
UC-14
UC-15
UC-16
UC-17
UC-18
UC-19
UC-20
UC-21
UC-22
AHT TASKS
T8.1
T8.2
T7.1
T8.3
T9.1
T8.4
T8.5
T7.2
T8.10
T9.7
T8.6
T7.3
T7.4
T8.7
T7.5
T9.2
T9.3
T9.4
T9.5
T9.6
T7.6
T8.8
T8.9
WP6
Requirements (1)
Functional Design (2)
Procurement & Engineering (3)
Deployment & Commissioning (4)
Operation & Management (5)
Maintenance (6)
Evolution (7)
Training & Education (8)
AHT-EP

Figure 29 New 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 eac 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 30 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. Coloured 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, coloured with one of the four colour associated with one of the WP2 objectives, and a circle coloured with the AHT objectives colours. The colour assignation has been done by using information collected in the UC analysis [4] 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. We will re-evaluate this mapping during the project for providing a more accurate map in the D2.2.
Comparing the new picture (Figure 29) with the old one (Figure 28) we can note that the main focus of some UC has been moved now that the project has started and partners begin to work on their UCs. In particular, the main focus of UC-07, that in the project conceptualisation was placed in the Procurement & Engineering phase, has now been declared to be in the Deployment & Commissioning phase. In addition, UC-12, UC-16, and UC-19 have been shifted the focus on another EPP now that the operative part of the project has started.
In Figure 31, we aggregated the mapping reported in Figure 29 for generating the per-EPP clustered version of the UC mapping comparable with the mapping done in the DoA represented in the Figure 28.

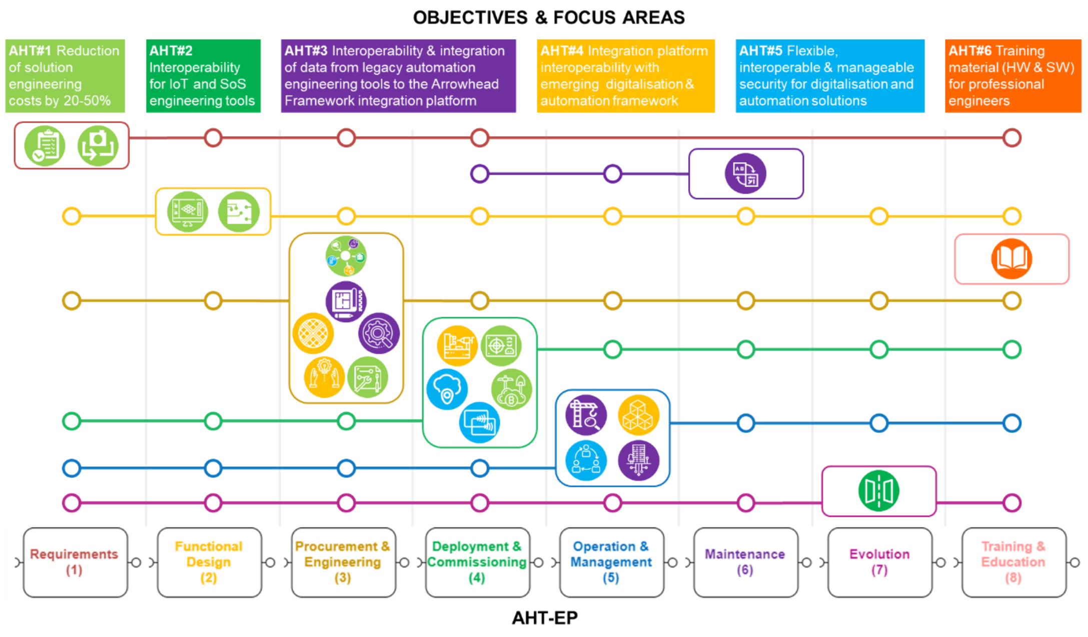

OBJECTIVES & FOCUS AREAS
AHT#1 Reduction of solution engineering costs by 20-50%
AHT#2 Interoperability for IoT and SoS engineering tools
AHT#3 Interoperability & integration of data from legacy automation engineering tools to the Arrowhead Framework integration platform
AHT#4 Integration platform interoperability with emerging digitalisation & automation framework
AHT#5 Flexible, interoperable & manageable security for digitalisation and automation solutions
AHT#6 Training material (HW & SW) for professional engineers
Requirements (1)
Functional Design (2)
Procurement & Engineering (3)
Deployment & Commissioning (4)
Operation & Management (5)
Maintenance (6)
Evolution (7)
Training & Education (8)
AHT-EP

Figure 31 New 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

5. Conclusions

In the deliverable D2.1 we have produced a detailed description of the AHT Engineering Process Phases and their interactions that will be 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 a description of their phase of the engineering

process, highlighting the essence of what the Phase means in the context of service-oriented architecture paradigms. 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* [3]) 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. The WP leaders have established joint WP2-WP1-WP4 collaboration to ensure a coherent and coordinated development of the concepts that will drive the ArrowHead-Tools project.
The *WP124 - Survey* follows 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).
The material collected with the *WP124 Survey* describing the Engineering Process aspects has been analysed 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.
An initial mapping of the UCs phases onto the AHT-EP, highlighting the WP2 and AHT objectives, shows how each UC can potentially match in each Engineering Process Phase. In the *WP124 – 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 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*, 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 preliminary analysis of the UC mapping on the AHT-EP that is proposed as a contribution of the D2.1 deliverable.
This UC map will be updated, corrected, extended, and eventually consolidated by WP2 leader in M24.

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: D2.1 Deliverable Appendix - WP1 WP2 WP4 Use Cases survey -> DA1_WP124_survey.pdf
2. Appendix2: D2.1 Deliverable Appendix - Use Cases analysis for AHT-EP definition -> DA2_WP2_22UC_analysis.pdf

7. References

[1] J. Delsing, Iot automation: Arrowhead framework, CRC Press, 2017.

[2] J. Garcia Represa and J. Delsing, “Autonomous Production Workstation Operation, Reconfiguration and Synchronization,” in *25th International Conference on Production Research, Chicago, August 9-15, 2019.*, Chicago, 2019.

[3] P. Azzoni, G. Urgese and F. Montori, “WP1 WP2 WP4 Use Cases survey,” 2019.

[4] G. Urgese, “Use Cases analysis for AHT-EP definition,” 2019.

[5] M. Iñigo and C. Schmittner, “Deliverable D10.1 Standardisation base line,” AHT Project, 2019.

[6] F. Kevin and 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.

[7] E. Allouis, R. Blake, S. Gunes-Lasnet, T. Jorden, B. Maddison, H. Schroeven-Deceuninck, M. Stuttard, P. W. K. Truss, R. Ward and 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.

[8] M. Hankel and R. Bosch, “The reference architectural model industrie 4.0 (rami 4.0),” ZVEI, 2015.

[9] F. Zezulka, P. Marcon, I. Vesely and O. Sajdl, “Industry 4.0–An Introduction in the phenomenon.,” *IFAC-PapersOnLine,* vol. 49, no. 25, pp. 8-12, 2016.

[10] Y. Lu, K. C. Morris and 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 and M. Crawford, “The industrial internet of things volume G1: reference architecture,” *Industrial Internet Consortium,* pp. 10-46, 2017.

[12] S. Mathur and S. Malik, “Advancements in the V-Model,” *International Journal of Computer Applications,* pp. 29-34, 2010.

[13] S. Balaji and 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.

[14] D. D. Walden, G. J. Roedler, K. Forsberg, R. D. Hamelin and T. M. Shortell, Systems engineering handbook: a guide for system life cycle processes and activities, Fourth edition ed., Hoboken, Ed., New Jersey: Wiley, 2015.

[15] NASA, NASA SYSTEMS ENGINEERING HANDBOOK, nasa/sp-2016 -6105 rev2, 2017.

[16] N. Viola, S. Corpino, M. Fioriti and F. Stesina, “Functional analysis in systems engineering: Methodology and applications,” *Systems engineering-practice and theory,* 2012.

[17] P. B. Kruchten, “The 4+1 View Model of architecture,” *IEEE Software,* vol. 12, no. 6, p. 42–50, 1995.

[18] P. Micouin, Model-based systems engineering: fundamentals and methods, N. U. Hoboken, Ed., London: Wiley, 2014.

[19] A. W. Wymore, Model-based systems engineering, Boca Raton: Chapman and Hall/CRC, 2018.

[20] J. M. Borky and T. H. Bradley, Effective model-based systems engineering, New York: Springer Science+Business Media, 2018.

[21] L. Delligatti, SysML distilled: a brief guide to the systems modeling language, Upper Saddle River, NJ: Addison-Wesley, 2014.

[22] I. C. Society, P. Bourque and R. E. Fairley, Guide to the Software Engineering Body of Knowledge (SWEBOK(R)): Version 3.0, 3rd ed ed., Los Alamitos, CA, USA: IEEE Computer Society Press, 2014.

[23] S. Amira, V. Perelman and D. Dori, “A Project-Product Lifecycle Management approach for improved systems engineering practices,” in *INCOSE International Symposium*, 2008.

[24] “Procurement Process Flow – A Guide to Procurement in Business,” 11 03 2019. [Online]. Available: https://www.codelessplatforms.com/blog/procurement-process-flow/. [Accessed 20 10 2019].

[25] “ISA-95,” [Online]. Available: isa-95.com. [Accessed 25 November 2019].

[26] B. Scholten, The road to integration: A guide to applying the ISA-95 standard in manufacturing, Isa, 2007.

[27] J. Virta, S. Ilkka, T. Antti and 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.

[28] “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. [Accessed 25 November 2019].

[29] T. Holm, L. Christiansen, M. Göring, T. Jäger and 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.

[30] “AutomationML,” [Online]. Available: https://en.wikipedia.org/wiki/AutomationML. [Accessed 25 November 2019].

[31] O. Carlsson, J. Delsing, F. Arrigucci, A. W. Colombo, T. Bangemann and P. Nappey, “Migration of industrial process control systems to service-oriented architectures,” *International Journal of Computer Integrated Manufacturing,* vol. 31, no. 2, pp. 175-198, 2018.

[32] D. Evans, “The internet of things: How the next evolution of the internet is changing everything,” *CISCO white paper,* pp. 1-11, 2011.

[33] C. De Luca and G. Schneider, “Deliverable D9.1 Use case spec for Y1,” AHT Project, 2019.

[34] E. Vaoumorin, “Deliverable D6.1 Training and support material for project internal and external usage,” AHT Project, 2019.

[35] P. Azzoni and Ø. Haugen, “Deliverable D1.1- Requirements, state of the art,” AHT Project, 2019.

[36] M. Tatara, F. Montori and B. Sara, “O3: AHT use case analysis - Use cases analysis,” AHT-Project, 2019.

[37] 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.

[38] G. Peeren and 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.

[39] J. Delsing, “AHT DoA Annex 1 (part B) - Arrowhead Tools for Engineering of Digitalisation Solutions,” AHT Project, 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

9.1. Contributing and reviewing partners

Contributions	Reviews	Participants	Representing partner
X	X	Gianvito Urgese	POLITO
X	X	Jan Van Deverter	LTU
X	X	Andrea Acquaviva	IUNET
X	X	Paolo Azzoni	Eurotech
X	X	Sara Bocchio	ST-I
X	X	Enrico Macii	POLITO
X		Oskar Berreteaga	UMLA
X		Jose María Alvarez Rodríguez	UC3M

X	X	Anja Zernig	KAI
X		Marek Tatara	DAC
X		Arellano Cristobal	IKERLAND
X		Iglesias Aitziber	IKERLAND
X		Jan Westerlund	ABB
X		Janne Keränen	VTT
X		Jari Halme	VTT
X		Emmanuel Vaumorin	Magillem
X		Asma Smaoui	Magillem
X		Michel Iñigo	Mondragon
x		Martin Mischitz	IFAG
x		Germar Schneider	IFD
x		Geran Peeren	PHC

9.2. Amendments

No.	Date	Version	Subject of Amendments	Author
1	2019-10-30	0.1	First draft	Urgese Gianvito
2	2019-11-25	0.5	Second draft	Urgese Gianvito, Jan Van Deverter

9.3. Quality assurance

No	Date	Version	Approved by
1	2019-12-03	1.0	Jerker Delsing