LeArning and robuSt decision SupporT systems for agile mANufacTuring environments

Project Acronym:

ASSISTANT

Grant agreement no: 101000165

Deliverable no. and title	**D2.1 - Initial architecture document**	
Work package	WP 2	Ethic and Human Centric Toolbox
Task	T 2.1	Human Centric Architectural design
Subtasks involved	Task 2.2	
Lead contractor	Institut Mines-Telecom (IMT) Alexandre Dolgui, mailto: alexandre.dolgui@imt-atlantique.fr	
Deliverable responsible	Europa Universität Viadrina Jan-Hendrik Passoth, passoth@europa-uni.de	
Version number	v1.0	
Date	30/08/2021	
Status	Deliverable	
Dissemination level	Public (PU)	

Authors

Participant no.	Part. short name	Author name	Chapter(s)
	JB	Johan Buchholz	All
	BL	Benedict Lang	All
	JHP	Jan-Hendrik Passoth	All
	VX	Evangelos Xanthakis	2,3
	EV	Eduardo Vyhmeister	3

Document History

Version	Date	Author name	Reason
v0.1	04.12.2020	Félicien Barhebwa-Mushamuka	Initial Template
V0.2	01.06.2021	Benedict Lang	Initial draft of structure and first contents
V0.9	06.08.2021	Benedict Lang	First version of the document to be circulated for feedback
V1.0	30.08.2021	Benedict Lang	Final Deliverable; integrated feedback; editing for better readability

Publishable Executive Summary

This deliverable represents the human-centric Architecture for the ASSISTANT project. It combines approaches from responsible research and innovation with the high-level technical architecture that was provided as an input for this document. This deliverable is the first in a series of three. The initial architecture document will be updated twice within the ASSISTANT project.

Within work package 2, there are different approaches that try to ensure the responsible or ethical development of artificial intelligence within the digital twins. This document mostly focuses on the ex-ante approach which raises issues that need to be considered during the process and which are supposed to shape the design process. Ex-post architectural considerations will be included in subsequent deliverables within work package 2.

First, the document contains a review of the research field and positions our work. We introduce different approaches towards responsible design of artificial intelligence that range from ethics to RRI (responsible research and innovation) and introduce the ART-principles that are the base for the methodological approach for the human-centric architecture. ART stands for Accountability, Responsibility and Transparency and provides a lens to look at the interaction of the system and the human that is using the system.

After providing the theoretical background, this document contains the conceptual architecture. This section is extracted from another task within the work package, that is the technical architecture. We provide insights into guiding principles for the interoperability and the distribution of responsibility among different components.

In the chapter human-centric architecture we combine the theoretical approaches with the technical architecture and make first suggestions how to integrate the ART-principles into the further design and development process of ASSISTANT.

As this is just the first deliverable in a series of three, we will outline how we proceed in the future to deepen our analysis and to contribute to the responsible development of AI-components in ASSISTANT.

Table of contents

List of figures

List of tables

1. About this document

This document aims to provide an overview of the steps taken within the ASSISTANT project to ensure a responsible and trustworthy architecture for the artificial intelligence system. The endeavour is based on abstract frameworks in the area of “ethics-by-design” and suggests how they can be applied within the concrete case of Artificial Intelligence in industrial manufacturing. Not only does it raise questions or aspects that might become ethical issues, it also provides suggestions on how to approach these different challenges.

This is a living document insofar as its development is an ongoing, iterative and explorative process. At this point in time, our approach is based on a literature review as well as some initial thoughts on the technical architecture of the digital twins to be used within the ASSISTANT project. However, our approach remains open to modifications in the further process and in upcoming revisions of this document in future deliverables. Visualizations and the identification of different *modules* that require closer attention allow us to reduce the complexity of the overall architecture. The architecture comes with distributed responsibilities and requires the knowledge of different actors for its different parts.

This document is neither a checklist nor a to-do list. We as the authors are convinced that in order to design systems ethically, it is not enough to use off-the-shelf methods and ticking of boxes. Instead, it is necessary to deliberate, to discuss and to co-create among stakeholders. Therefore, this document invites stakeholders to the table to foster these debates.

With this document, we also aim at providing a blueprint for other organizations and projects that want to implement their technical applications in an ethically responsible way. By being transparent about how we approach the task of creating an ethical-by-design artificial intelligence application, we invite feedback, critique and debates about further improvement.

As other deliverables within the ASSISTANT project cover the full list of requirements and therefore also an in-depth description of each individual component, we decided to not replicate this information but instead want to invite you as the reader to refer to Deliverable 7.1.

With this document, we aim to contribute to the knowledge of how abstract frameworks and principles can be translated into concrete adaptations of technical developments. We beliveve that this can have a major impact.

2. Developing a Human-Centric Architecture

Developing a human-centric architecture within the ASSISTANT project is part of a broader effort to ensure that the technologies produced within the project are designed in a responsible way. This document is the first deliverable within work package 2 (Ethic and Human Centric Toolbox) and shows one approach how this responsible creation can be achieved.

The ASSISTANT human-centric architecture in its current version consists of two parts. In the first section, we will discuss different approaches towards human-centric design and the consideration of ethical issues in the development of technology and specifically AI. In the second section, we will outline the technical architecture of ASSISTANT and point out aspects that require attention in the process of development in order to be human-centric and ethical-by-design. In further revisions of this living document, the methods of intervention and their results will be documented. As we have learned from previous projects, a requirement for an ethical-by-design system is that ethical considerations already find their way into the design process to ensure that the end product is responsible.

2.1 Theoretical Framework

This chapter gives an overview of existing approaches towards human-centric architectural design in the context of AI. We provide this overview to be transparent about the assumptions and schools of thought that underlie this document. We will contextualize specific approaches among existing frameworks on human-centric and ethical AI to then develop our own approach.

As we connect different streams of literature in the following section, Figure 1 provides an overview of what will be covered. Even though the graphic suggests a structured order, the different approaches are not that easily distinguished but somehow overlap, and also develop next to as well as influence each other. The figure is rather a table of content than a description of the relations of the different approaches towards each other.

Figure 1 overview of theoretical framework

In recent years, debates on ethical, trustworthy, or responsible AI have been fostered by political actors, researchers, and civil society. A growing number of guidelines to ensure 'ethical' AI demonstrates the relevance of the issue ((Hagendorff, 2020b) discusses a selection of these guidelines, a non-exhaustive overview is provided by(Algorithm Watch, 2021)). However, these rather abstract approaches are hardly translated into concrete results in the development process, as they are not "put into practice" (Hagendorff, 2020a, p. 1). The development of a human-centric architecture document for the concrete context of manufacturing within ASSISTANT is embedded in these broader discussions and debates on ethics, responsible research and innovation (RRI) and other frameworks. We consider responsibility to be the key aspect that links different discourses and approaches towards a development process, which reflects the assumptions and needs of different stakeholders.

Debates on ethics are usually connected to different **theoretical fields of ethics**, such as normative or applied ethics. While the former focuses on the identification of moral standards that can be used to differentiate between wrong and right behaviour, the latter approaches specific controversial issues. Following one of these approaches within the development process of the ASSISTANT architecture would include the definition of standards and boundaries that should not be ignored or crossed. Therefore, such an approach might include abstract recommendations that have to be considered when it comes to the specific context of the ASSISTANT AI systems. While the strength of such approaches lies in being explicit about certain boundaries and values, it is hard to reflect and understand the local situation through *ex ante* definitions of norms and standards. This is why we also extend our perspective towards responsible research and innovation and other approaches. These are to be adapted to each situation in which they are implemented.

2.1.1 Responsible Research and Innovation and ethics by design

One approach to consider and anticipate the consequences of a certain technology in society next to normative ex-ante ethics is the field of **responsible research and innovation (RRI)**. Shaped by contributions from Science and Technology Studies (STS), this approach has been established and prominent in recent projects funded by the EU. It includes continuous reflection on different questions during the research process, involves actors from the research context as well as civil society, such as third sector organizations, in order to align the development process and the outcomes with expectations of society. One aim of this approach is to make the assumptions that are embedded in development processes visible and transparent. This transparency supports the development of teams to ensure that they are on track regarding their responsibilities. It also allows for users and stakeholders of technologies to criticize them.

Within the area of responsible research and innovation, an approach has emerged that is called **ethics by design**. This approach includes a set of best practices that focus on including ethical or responsible deliberations already in the design process. These best practices include organizational aspects – for example, the establishment of an ethics board as well as suggestions for the actual process (Leidner & Plachouras, 2017). This aims at an integration of "ethical decision routines in AI systems" (Hagendorff2020a), that is, values are explicitly integrated into the decision algorithms. This architecture document is supposed to contribute to an ethics-by-design approach in that it not only presents the different technical components but also points towards the integration of values as part of the process.

Such a process can start with **frameworks**. One example of such frameworks is the contribution of the "High-Level Expert Group on Artificial Intelligence", which provides an assessment list for **trustworthy AI**. The authors point out that AI should be lawful, ethical, and robust in order to be trustworthy (High-Level Expert Group on AI, 2019). The framework comes with a range of questions, grouped by issues such as human agency and oversight, technical robustness and safety, privacy and data governance, transparency, diversity, non-discrimination and fairness, societal and environmental well-being, as well as accountability.
As part of an assessment list, these topics provide important foci in the reflection of AI applications.

The strengths of such frameworks lie in reflecting abstract dimensions that should be considered during AI development and use. They are supposed to ensure that socio-technical systems do not contradict specific societal needs and issues, e.g. that AI does not exclude people based on dimensions such as race, gender, education and others.

However, different frameworks by governments, businesses and from the third sector exist and highlight slightly different aspects. And even when the decision towards the use of a framework is taken, the translation of the mentioned issues into the context of development is a highly complex task that requires additional decisions. Therefore, abstract frameworks can only provide starting points for a discussion on trustworthy and human-centric AI, while the actual implementation process requires the consideration of the very local contexts, the associated assumptions and decisions that are taken for granted.

2.1.2 Responsible human-centric AI

In addition to this, **human-centric design** can be considered to be one set of methodologies and strategies to make these abstract frameworks tangible in concrete development projects: it is informed by approaches in design thinking and uses methods from the social sciences such as interviews, group discussions, ethnographic approaches, but also many more (several handbooks, such as (IDEO, 2015; LUMA Institute, 2012) provide an overview on these methods). Human centric design can thus shape a development process according to the needs of humans, as the formulation and reflection of these needs is a core element of this process. However, the use of these methods alone does not guarantee that ethical aspects are considered, as they may not be relevant to the actors involved in the process. Therefore, we combine elements of the mentioned approaches to ensure that the human-centric architecture is ethical-by-design.

More concretely, human-centric design – as we understand it – is an approach to narrow down the scope of values that need to be addressed within design processes. It places the focus on the interaction between humans and technologies. This is specifically helpful in contexts of the development of artificial intelligence systems because a lot of the moral dilemmas that emerge from them are entangled with the non-humanness of the AI. This interaction between humans and technologies is radically changed through artificial intelligence applications, which is a good reason to focus on these interactions.

Human-centric design must not be mixed up with human-centred design, which is defined in the ISO-norm 9421-210 and refers to design processes that include the users and develop products based on their requirements. It is therefore closely related to the value of the product for the user, i.e. the product will be used and will contribute to the company's profit.

Creating a **responsible, human-centric AI** is a complex task that needs to reflect these different streams of literature and thought. We suggest including elements of these different discourses through the notion of responsible AI and the use of a specific framework, which is open for taking up core-concepts from ethical debates, responsible research and innovation, and abstract frameworks. Also, there should be openness to the requirements and needs within the specific context of AI systems in industry. Concrete work in that sense must be eclectic and must put together different components adopted for the specific situation.

2.1.3 ART-principles

One concrete approach towards responsible AI based on Human Centric Design encompasses the ART principles (Dignum, 2019a, 2019b). Dignum's suggestion spells out the abstract idea of the responsible creation of AI through a concrete set of values. This is helpful, as the concrete values allow us to raise questions regarding the process of development and the system to be developed. Therefore, we consider it an adequate starting point for our endeavour to make assumptions explicit in the course of the process. The ART-principles allow us to address issues of accountability, responsibility, and transparency within ASSISTANT.

As part of the project HumanE AI – which was funded within the Horizon 2020 programme – Virginia Dignum authored a report that introduces a methodology to develop what she calls Responsible AI systems. This includes mainly two thoughts. At first, the ART-principles are introduced as a requirement towards AI systems. Secondly, they introduce the methodology of Design for Values that aims at making values and the process of embedding them into software design projects visible and transparent.

The underlying assumption is that processes of software and technology development are full of decisions that the "designers, developers and other stakeholders" have to make, "many of them of an ethical nature". Dignum therefore also highlights the difference between an ethical process of development and the AI system being capable of making its decisions in an ethical way. Whether an AI system can decide ethically is dependent on the values that are embedded within it and therefore result of the development process. Additionally, it is dependent on negotiation processes within societies that define what is ethical and what is not. As we share this understanding in relation to values in design processes and the ideas are already present in the text above, we will not further elaborate on them at this point but will continue with the introduction of the ART principles.

The ART principles for Responsible AI can be summarized as follows: They include three aspects that need to be reflected when pursuing responsible systems. Accountability, Responsibility and Transparency have a close connection to each other and seem to be to some extent overlapping. However, they provide a different focus and complement each other.

- "***Accountability*** *refers to the requirement for the system to be able to explain and justify its decisions to users and other relevant actors*" (Dignum, 2019a). This means that the system needs to be able to be held accountable in relation to humans that interact with it and are affected by it. Therefore, decisions need to be explainable *after* they have been taken.
- "***Responsibility*** *refers to the role of people themselves in their relation to AI systems*" (Dignum, 2019a). Responsibility is different from accountability in that it focuses on the people involved and is not related to the content of the decision: It links to questions of liability, on the one hand, but also to who is capable of behaving morally. Questions of responsibility could be: Who delegates which decisions to the system and how are decisions supervised? The responsibility dimension encourages reflections about the role of different persons within the process of decision-making and system development.
- "***Transparency*** *indicates the capability to describe, inspect and reproduce the mechanisms through which AI systems make decisions [...]*" (Dignum, 2019a). It is therefore a precondition to determine responsibilities and to hold the responsible people accountable. Transparency increases trust, as people do not only have to trust but can ground their faith on a sophisticated understanding of how algorithms work. Making the algorithms transparent allows stakeholders to criticize what is going on.

> Transparency is different from accountability in that it is not necessarily linked to one specific situation that is evaluated ex-post but includes a more general need for openness.

As stated above, the three principles are closely connected to each other, interdependent and intertwined. But still, they address different specific foci.

What Dignum (2019a) generally suggests is to concretize abstract values into more concrete norms that can then again be translated into concrete functionalities. For example, one can define “openness” as a value which is to be translated into norms that could be something like “access for stakeholders” or “being adoptive to stakeholders’ feedback”. In the implementation phase, where actual code is produced, this results in concrete forms, buttons or dashboards that enable control or insight.

In concrete situations, there are also different perspectives that must be brought together. This means that these ethical concerns (e.g. values, norms and functionalities) and domain requirements (e.g. functional and non-functional) influence the actual process of developing the AI. They have an impact on the motives and roles, the goals that are to be achieved and finally the actual plans & actions. Both domain requirements and ethical considerations can be structured hierarchically to match more high-level or more specific aspects of the design process.

Important to note is that – like other system development methodologies – the process is designed to be, on the one hand, iterative and, on the other hand, goes beyond the first going live of the application. It requires management throughout the entire lifecycle of the AI application. Obviously, the creation of responsible AI – following the suggestions of Dignum – does not replace legal compliance and the reflection of the regulatory context in which the system is to be embedded.

2.1.4 ART-principles for the human-centric architecture in ASSISTANT

After providing the theoretical context in which we position our work, we would now like to present the concrete approach that we will use to reach a responsible, Human Centric architecture for the digital twin artificial intelligence applications. We start by introducing the expectations we have towards our approach.

Our expectation towards an approach within the ASSISTANT project is defined by the following criteria:

1. Our approach has to enable explicit deliberations about values. It is our assumption that the development of technical systems is always influenced and shaped by the values and perceptions of the engineers and the developing teams. Values and their embeddedness in technology are a very core idea of Science and Technology Studies.
2. Our approach has to offer the potential to improve processes in which AI systems are produced. It is our assumption that in order to improve technologies, it is necessary to improve the processes in which they are developed.
3. Our approach must enable reflections about potential biases and different perspectives of the people involved. We assume that in order to create a Human Centric architecture, multiple stakeholders need to have a seat at the table.
4. Our approach has to be connected to broader discussions in the field.
5. Our approach has to be concrete and offer tangible instructions for the actors working in the ASSISTANT project.
6. Our approach has to consider specificities of the manufacturing sector in connection to artificial intelligence.

7. Our approach has to function as a blueprint for others that want to adopt our work for their projects.

In the following sections, we will introduce a methodology that matches these requirements. We will then argue why we choose to use this specific approach as the basis for our work within the ASSISTANT project. We are strongly convinced that there is no such thing as an *off-the-shelf method*. This means that we will not implement the ART-principles in a 1:1 way but instead use it as a starting point to not reinvent the wheel from the very beginning.

While the need for an approach towards a Human Centric architecture might sound reasonable on an abstract level, we build on the assumptions explained above. To be more precise, this means that we will shed light on the different components of the architecture and, at the same time, on the architecture as a whole reflecting different sets of values. In doing so, we hope to make the values that are to be embedded in the systems explicit through interventions like workshops and discussions.

At first, it is necessary to define which values are to be chosen within the process. Obviously, it will not be possible to consider all potential values at the same time. Within this specific context of the project, we decided to choose *human-centric design* as the guiding principle.

Within the ASSISTANT project, we are going to base our deliberations for the development of a human-centric architecture on the process suggested by (Dignum, 2019a) and described above.

So far, we have shown where the approach that we are going to apply within the ASSISTANT project comes from and where we position ourselves within the debate. Additionally, we have identified a set of values (ART-principles) that we want to explicitly consider within the project. We will now in the course of the project look at the technical architecture – its first version can be found in the following – to identify methods that can help us facilitate discussions about how to incorporate the values described above into the different components of the architecture. After these discussions have taken place, more and more requirements and features will be derived and integrated in the further revisions of this architecture document.

We conclude that the criteria that we have defined for the design of a process can be met with the approach that we have described above and which we will try to implement in the course of our involvement in the project. Our approach makes values explicit, it focuses on the process, and it is connected to the broader research field and academic literature. We will explore and reflect the specificities of the manufacturing process by looking more closely at the different use-cases and components. Through the progressing architecture document and through transparency about our choices and decisions, we aim to inspire other projects to pick up our threads and to continue what we are going to do.

Additionally, the procedure that we proposed is therefore in line with the criteria that are suggested by (Aldewereld et al., 2015, p. 834): (1) We explicitly defined “global aims” for this architecture document and explained how it is supposed to help us in building a human-centric architecture. (2) We plan to facilitate discussions about which stakeholders to involve in the project and we (3) are going to make decisions explicit in the revisions of this document.

As this is only the initial architecture document, we will potentially extend the theoretical framework during the project. This means that we will inspect and include more approaches and frameworks from the field of the relevant disciplines. For the starting point, however, it is more relevant to decide on a concrete approach than to consider everything that is out there. Our aim within this project is not to provide a holistic overview of existing approaches but rather to contribute to the experiences in bridging the gap between abstract frameworks and concrete developments. This is even more true, as the field is very dynamic and new developments are emerging rapidly.

2.2 Methodology

In this chapter, we describe how we proceeded to develop the Human Centric architecture that is to be found in the next chapter. We do this in order to be explicit and therefore also to be open for feedback for further improvements. This also allows others that are interested in building upon our work to understand our approach.

"Responsible AI is more than ticking of some ethical 'boxes' or the development of some add-on features in AI systems" (Dignum, 2019a, p. 5). We would like to stress this aspect and emphasize that *Responsible AI* requires a process in which the assumptions, decisions taken, and their consequences are reflected both by the developers but also the humans potentially affected. Therefore, the used ART principles take aspects of high-level frameworks into account with which particular issues and concrete problems in the very context of AI design and application can be addressed.

In order to analyse and adjust the developing technical architecture within ASSISTANT, we are going to examine the components and their interplay through the lens of the ART principles. Consequently, we will identify both components and combinations of them that point to potential issues for further discussion. In another step, we will document these issues, group them and discuss selected issues with the developers and, where applicable, with the people affected. These workshops will address the abstract claims from frameworks and focus on Human Centric responses to these identified issues that can be implemented in the further development process.

While this report sets the stage for the further development process, it aims at describing the approach, the central elements of the ASSISTANT architecture and will identify issues for further discussion. The following intermediate Human Centric architecture document, as well as its final version, will describe the course and results of the outlined process.

Within the ASSISTANT project and its work package 2, there are different tasks that are supposed to ensure the responsible and ethical creation of artificial intelligence. This architecture document is supposed to function as an ex-ante approach that points out potential issues in the first place and focuses on the process. The KPIs and evaluation criteria that are defined in another task are supposed to help in examining the success of the project. Additionally, the architecture focuses on the interaction of the complete system with its users and the interaction of the different components with each other. Task 2.4 is supposed to develop methodologies for implementing concrete components responsibly, and therefore uses different frameworks and guidelines than we do for putting together the human-centric architecture. A visualization of the two approaches is shown in Figure 2. The figure shows differences in the approaches in relation to the levels they address, the frameworks they use, the object of their attendance, and the responsibility of the organization within the ASSISTANT project.

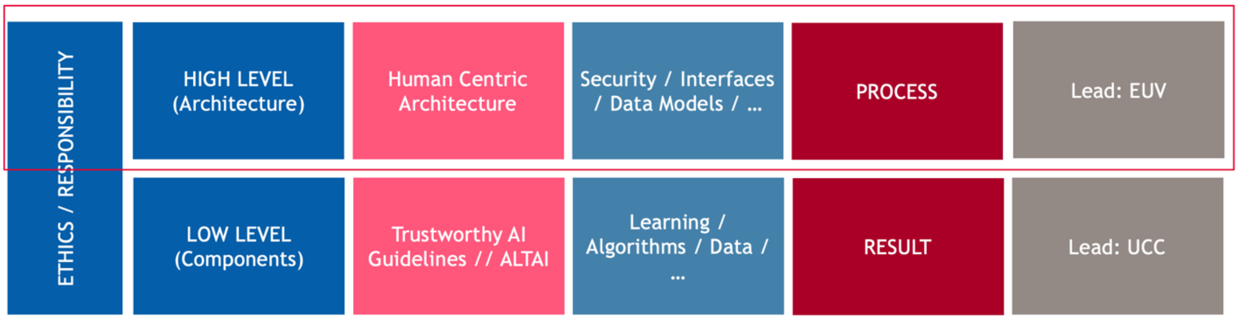

Figure 2 Visualization of the different approaches for responsible design within ASSISTANT

The two approaches not only complement each other in relation to time but also in relation to the objects they address. While the human centric architecture focuses on a higher level and looks at the organization of different components and their interaction, the trustworthy AI guidelines will address what happens inside the components and focus more on the concrete algorithms and machine learning mechanisms. Throughout the project, it will be necessary to combine these two approaches in order to gain a more in-depth analysis.

3. ASSISTANT Human Centric architecture

Within ASSISTANT, there are two architecture documents that both continuously evolve. While the technical architecture is produced within task 2.2, the human-centric architecture is the result of task 2.1, represented by this deliverable. As both documents evolve, the aim is to deepen the integration of both in the course of the project. The endeavour of bringing the two of them more and more together is supposed to actually bridge the gap between theoretical concepts and the actual responsible development of the digital twins.

Two sections from the technical architecture (task 2.2) are attached in their entirety to this document in the appendix.

This chapter, however, presents the fundamentals of the ASSISTANT technical architecture. It provides initial considerations based on the ART-principles to transform this technical architecture into the human-centric architecture. We will introduce the architecture as well as initially apply the methodology defined above.

3.1 The context of Manufacturing

Manufacturing refers to a large-scale production of goods that converts raw materials, parts, and components into finished merchandise using manual labour and/or machines. The finished goods can be sold directly to consumers, to other manufacturers for the production of more complex products, or to wholesalers who distribute the goods to retailers. In the last decades, computer-aided technologies were put into use in the manufacturing domain and the term CAx was introduced. More specifically, computer-aided technologies (CAx) is the use of computer technology to aid in the design, analysis, and manufacture of products. Advanced CAx tools merge many different aspects of the product lifecycle. In industry, product lifecycle management (PLM) is the process of managing the entire lifecycle of a product from its inception through the engineering, design, and manufacture, as well as the service and disposal of manufactured products. From among the high variety of CAx tools, the ASSISTANT project aims at offering a portfolio of applications that will aid in the design phase (designing product processes) but also in the runtime phase with production planning/scheduling and digital twins for execution.

3.2 Digital Twins for Manufacturing

Figure 3 provides an overview of the different components that constitute the ASSISTANT architecture. There are three different digital twins that contribute in different ways to the optimization of the manufacturing process (process planning, production planning and scheduling, and execution). They focus on different functional parts of the overall production process and the desired optimizations. All digital twins share a common data fabric that provides services in relation to data storage services, data control and data analysis. Concrete exchange of data between the digital twins and the data fabric can be found in Figure 4, which depicts the data flow.

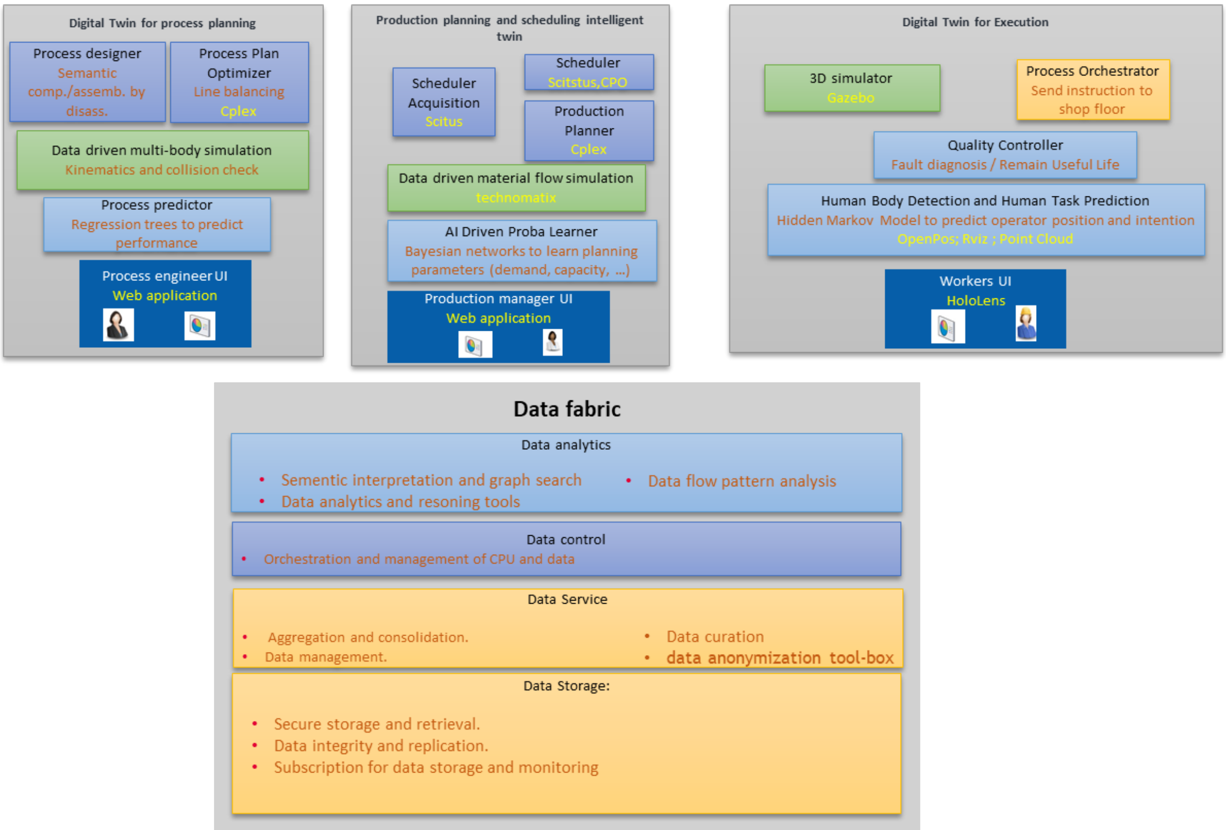

Figure 3 Component Landscape

'Digital Twin for process planning' focuses on the design of the production process. The "process designer" and "process plan optimizer" are responsible for generating the different process graphs. Then, for different production plans and technical changes, the "process predictor" enables the forecast of various KPIs regarding cost, time, and quality. These components are joint through the "process engineer". Process engineer provides a user interface that supports users in generating efficient and effective decisions regarding the process design.

'Production planning and scheduling digital intelligent twin' aims at providing tools for production planning and scheduling based on AI prediction and simulation. The "Scheduler Acquisition" is responsible for generating a constraint model from a set of tables with schedule-related data. "Scheduler" then optimizes the constraint, model while "Production Planner" computes the production plan (quantity to produce per period, quantity to order, and capacity adjustment with overtime). Again, these functionalities are provided to the production manager via a user interface ("Production Manager UI").

The responsibility of 'Digital Twin for Execution' is to utilize the output of the two aforementioned component groups (namely process plans and production schedule) in order to successfully drive production in the shop-floor. "Process Orchestrator" works as the interface of the digital twin with the process and production planning, while "3D Simulator" monitors the real-time behaviour of the system along with "Human Body Detection and Human Task Prediction" and guides the system to accomplish the task at hand uninterrupted. The "Quality Control" module offers a near real-time system performance in order to quickly identify faults and propose countermeasures. Again, these functionalities are offered to the end users via a graphical user interface ("Workers UI").

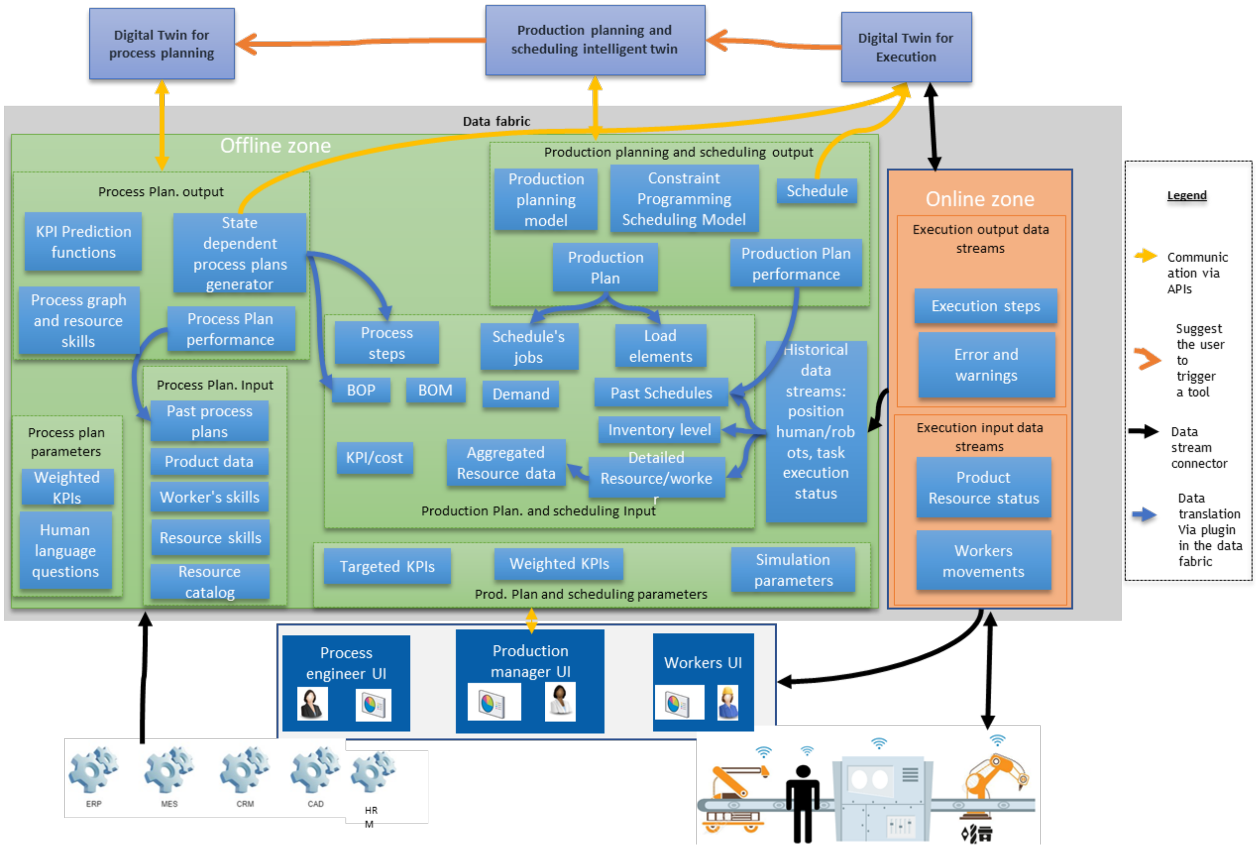

Figure 4 ASSISTANT Data Flow

Above is the initial ASSISTANT domain model along with the envisaged data flow. The initial input ("Process Plan Input") consists of the product and production system data (Resources, Workers, Skills etc.) utilized to produce the "Bill Of Processes/Materials" (BOP/BOM) for a specific product. This data set (BOP/BOM) along with the generated scheduling constraints (resource workload, inventory capacities etc.) and KPIs serving as scheduling criteria are utilized for generating the production schedule by the "Production planning and scheduling intelligent twin". The generated production schedule is then fed into the "Digital Twin for Execution", which generates and stores real-time status data related to the processes and the resources. Apart from the domain model and data flow explained briefly above, the diagram also depicts communication actions through components APIs. Such an action consists of a trigger event, i.e. an end-user presses a button on the relevant UI component which triggers an operation on the data (i.e. generation of production schedule). Similarly, there are some communications triggered by the system, which prompts the end-user to perform another operation. The most common case is the rescheduling of the production and/or choosing another production process based on shop-floor status (i.e. machine malfunction) and/or quality component measurement (i.e. bad quality).

3.3 Requirements engineering and technical design methodology

The first step towards the human-centric architecture is a process of requirements engineering to understand what it is that is to be built in the first place. In order to gather the requirements within the project, an integrated requirements engineering methodology is proposed. Figure 5 depicts this methodology, which was derived from the traditional requirements engineering process. The foundations of the methodology are formed by the actual processes of “End User/Customer” which are described and analysed as current practices. The current practices (“as is” scenario) represent a typical image of today’s mfg processes and contain several challenges. These challenges, also called points of improvement, should constitute the major drivers of the project. Based upon the points of improvement, use cases will be derived which constitute an alternative business process. Ultimately, these use cases should serve as a foundation for the requirements in the project. Within the methodology, two main types of requirements are distinguished: the end-user requirements and the IT requirements.

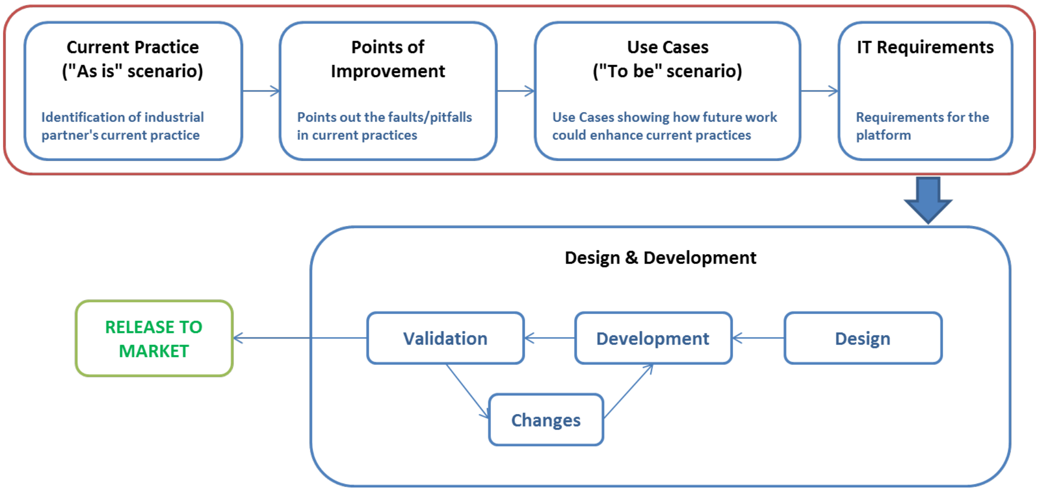

Figure 5 Requirements engineering approach

The end-user requirements are output of process step “points of improvement” and are valid for the entire project. End-user requirements are requirements purely from the end-user point of view. They represent the main business drivers of the project that need to be satisfied by the project outcomes indicating the main pain points in industrial practice at the moment.

The IT requirements are formed based on the use cases (“to be” scenario) and target the development by specifying software & hardware functionalities that should be delivered by the solution to support the “to be” scenario.

3.3.1 “As is” scenario

This process should describe the pilot and each pilot’s current practices. Text format is appropriate for such a description. This is described in the deliverable D7.1.

3.3.2 Points of Improvement

This process should describe the “as is” scenario flaws and describe how the current practices could be improved. It is recommended that the output of this process includes a list of “point of improvements”/flaws in bullets.

Along with the improvements, KPIs should also be described in the current process that could measure the impact of these improvements. The cardinality could be many-to-many since grouping of improvements with a single KPI or a single improvement to affect multiple KPIs is allowed. This section is also covered in the D7.1

3.3.3 Component Use Case Definitions

Within the ASSISTANT project, the term *use case* is originally used for the concrete application of the digital twins with the partners from Atlas Copco, Siemens Energy and PSA. The term *component use case* in the following refers to the meaning that is common within requirements engineering.

Each component use case will be described using the following fields:

Name
Each use case should have a name that clearly describes the main goal of the use case. The name may have to be several words long to be understood. Typically, the name is a verb phrase, for example: Withdraw Cash. The reader should be able to determine the goal of a use case simply by observing its name.
Note: No two use cases can have the same name.

Brief Description
The brief description of the use case should reflect its purpose.

Involved Components
This is the list of the ASSISTANT components that interact to achieve the use case goal in the flow of events.

Precondition
A precondition (assumption) is the state of the system and its surroundings that is required before the use case can be started.

Basic/Alternative Flow of Events
A use case describes the interactions between the ASSISTANT component(s) and the factory system in the form of a dialogue, structured as follows:

- The component 1 <<does something>>
- The component 2 <<does something in response>>
- The system <<does something else>>
- …

Name	Process Manager
Brief Description	
Involved components	
Pre-Conditions	
Basic Flow of Events	
Alternative flows	
Subflows	
Key-Scenarios	
Post-Conditions	
Special requirements	
Relevant pilot case(s)	

Table 1 Template for describing use cases

In Appendix **Error! Reference source not found.**, there is a list of such templates filled in for each component of the ASSISTANT component landscape.

3.3.4 Requirements Catalogue

The system requirements are defined based on the use cases and the pilot cases. The system requirements could be defined using the table presented below. The requirements may be connected to one or more use cases or to none of the use cases. The requirements describe both functionality as a response of the system to some user action (that is presented in the use cases chapter) as well as functionality that may not be “visible” to the user but is expected by the system in order to be able to respond to user needs (non-functional requirements). Table 2 shows the template for the catalogue that is used to gather the system requirements.

ID	Overall Description	Specific requirements						
		Performance	Logical Database	Hardware Constraints	Standards Compliance	Priority (Low/ Medium/ High)	Module that implements this requirement	Relevant pilot case(s)

Table 2 Template for requirements catalogue

In APPENDIX **Error! Reference source not found.**, there is a list with such a template for each component of the ASSISTANT component landscape.

3.4 Conceptual Architecture

The role of the overall architecture of ASSISTANT is to provide a framework for individual sub-architectures to collaborate on a common cause. The key for this approach is to introduce interoperability on the different sub-architectures.

The entire technical architecture can be found in a report from task 2.2 of the ASSISTANT project. While it is not public for everyone, the engineers and project members have access to it and can use it for their technical implementations. Within this document, there is merely a small subset that was extracted from the input from task 2.2 to ensure that this document does not extend too much. For the following revisions of this architecture document, it is planned to include further details as necessary. Thus, the contribution document serves as a repository for informing this architecture document.

3.4.1 Interoperability in a Service Oriented Architecture

Interoperability is defined as how easily a system can share information and exchange data with other systems. Interoperability in a “system of systems” is defined by the standard interfaces each system offers and the standard data representation (data format). As the ASSISTANT architecture is based on SOA (Service Oriented Architecture), it is composed by different services that cooperate to accomplish a common task, thus rendering ASSISTANT as a “system of systems”.

Interoperability requirements are considered a non-functional requirement, but its role is important in contributing to efficient development and the integration of different tasks. ASSISTANT’s interoperability will consist of three main types of interoperability: the information interoperability, the technical interoperability and, finally, the presentation interoperability. Information interoperability defines how information is to be shared among the different stakeholders and is described on section 3.4.1.1. Technical interoperability defines how technical services are shared and connected to each other; this aspect is described in section 3.4.1.2. Finally, presentation interoperability defines a common look-and-feel approach through a common portal-like solution which guides the user to the underlying functionality of the set of services. This kind of interoperability is discussed in section 3.4.1.3.

3.4.1.1 Information Interoperability

Data representation is the main focus of information interoperability. The main requirements imposed on the ASSISTANT solution regarding data representation are the clear, shared expectations regarding the contents, context and meaning of that data. Even though there are standard formats for different data domains (i.e. STEP ISO for 3D representation), we will not focus in this section on the most efficient data representation per domain (which will probably result in a multitude of data representation formats) but on a “global” data representation format capable of supporting ASSISTANT data-related requirements. In the ASSISTANT architecture, we can easily distinguish a major component related to the data itself, the Data Fabric. This component would play the role of ASSISTANT data storage, thus data also from different domains should be represented in a common format.
Data formats can generally be separated into two categories, the schema based and the schemaless based. Schema based formats have the advantage of being able to be considered valid or not according to a predefined data structure (schema), while the schemaless can only be evaluated as well as formed (syntactically correct), but no rules can be applied regarding the structure of these data. Schemaless data, on the other side, provide greater flexibility since they can accommodate any kind of data and can be expanded with less effort than the schema based.

The following section provides some insights regarding standard formats for the ASSISTANT architecture.

In computing, **Extensible Markup Language (XML)** is a mark-up language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable. The W3C's XML 1.0 Specification and several other related specifications (all of them free open standards) define XML. The design goals of XML emphasize simplicity, generality, and usability across the Internet. It is a textual data format with strong support via Unicode for different human languages. Although the design of XML focuses on documents, the language is widely used for the representation of arbitrary data structures such as those used in web services. Several schema systems exist to aid in the definition of XML-based	In computing, **JSON (JavaScript Object Notation)** is an open-standard format that uses human-readable text to transmit data objects consisting of attribute–value pairs. It is the most common data format used for asynchronous browser/server communication, largely replacing XML, and is used by AJAX. JSON is a language-independent data format. It derives from JavaScript, but as of 2017 many programming languages include code to generate and parse JSON-format data. The official Internet media type for JSON is application/json. JSON filenames use the extension JSON became an ECMA international standard in 2013 as the ECMA-404 standard. In the same year, RFC 7158 used ECMA-404 as reference. In 2014	**CSV (comma separated values)** format is a common data exchange format that is widely supported by consumer, business, and scientific applications. Among its most common uses is moving tabular data between programs that natively operate on incompatible (often proprietary and/or undocumented) formats. This works despite lack of adherence to RFC 4180 (or any other standard), because so many programs support variations on the CSV format for data import. Many applications that accept CSV files have options to select the delimiter character and the quotation character.

languages, while programmers have developed many application programming interfaces (APIs) to aid the processing of XML data.	RFC 7159 became the main reference for JSON's internet uses (ex. MIME application/json), and obsoletes RFC 4627 and RFC 7158 (but preserving ECMA-262 and ECMA-404 as main references).	

Table 3 formats for file exchange

3.4.1.2 Technical Interoperability

Technical interoperability is the ability of two or more components/applications to accept data from each other and perform a given task in an appropriate way without the need of additional intervention. The following paragraphs provide general information about the standard communication protocol proposed to exchange data among the ASSISTANT components.

HTTP & Security	REST
The Hypertext Transfer Protocol (HTTP) is an application protocol for distributed, collaborative, and hypermedia information systems. HTTP is the foundation of data communication for the World Wide Web. Development of HTTP was initiated by Tim Berners-Lee at CERN in 1989. Standards development of HTTP was coordinated by the Internet Engineering Task Force (IETF) and the World Wide Web Consortium (W3C), culminating in the publication of a series of Requests for Comments (RFCs). The first definition of HTTP/1.1, the version of HTTP in common use, occurred in RFC 2068 in 1997, although this was obsoleted by RFC 2616 in 1999 and then again by RFC 7230 and family in 2014. A later version, the successor HTTP/2, was standardized in 2015, and is now supported by major web servers. TLS cryptographic protocol will be implemented for providing security to HTTP whenever required, as detailed in D8.3. The Transport Layer Security protocol aims primarily to provide privacy and data integrity between two communicating computer applications. TLS 1.2 was defined in RFC 5246 and further refined in RFC 6176.	Representational state transfer (REST) are Web services providing interoperability between computer systems on the Internet. Using HTTP, as is most common, the kind of operations available include those predefined by the HTTP verbs GET, POST, PUT, DELETE and so on. REST was defined by Roy Fielding in his 2000 PhD dissertation "Architectural Styles and the Design of Network-based Software Architectures" at UC Irvine. Fielding developed the REST architectural style in parallel with HTTP 1.1 of 1996–1999, based on the existing design of HTTP 1.0[7] of 1996. By making use of a stateless protocol and standard operations, REST systems grant fast performance, reliability, and the ability to grow, by re-using components that can be managed and updated without affecting the system as a whole, even while it is running. For these reasons, most ASSISTANT communication interfaces are based on REST API communicating over HTTP.

Table 4 proposed protocols for ASSISTANT

These protocols are compatible with a Service Oriented Architecture where each ASSISTANT component is a separate individual entity that provides an added value in a data flow. The ASSISTANT component landscape as well as the ASSISTANT intercomponent data flow are depicted in the figures below.

3.4.1.3 Visualization Interoperability

The definition of templates for the screens was the initial step for the front-end developments. The templates were designed as a guideline for the implementation of the user interfaces. The existence of some common templates had the goal of achieving a uniformity of the user interface both in terms of appearance and in terms of behaviour. The basic defined templates are presented below.

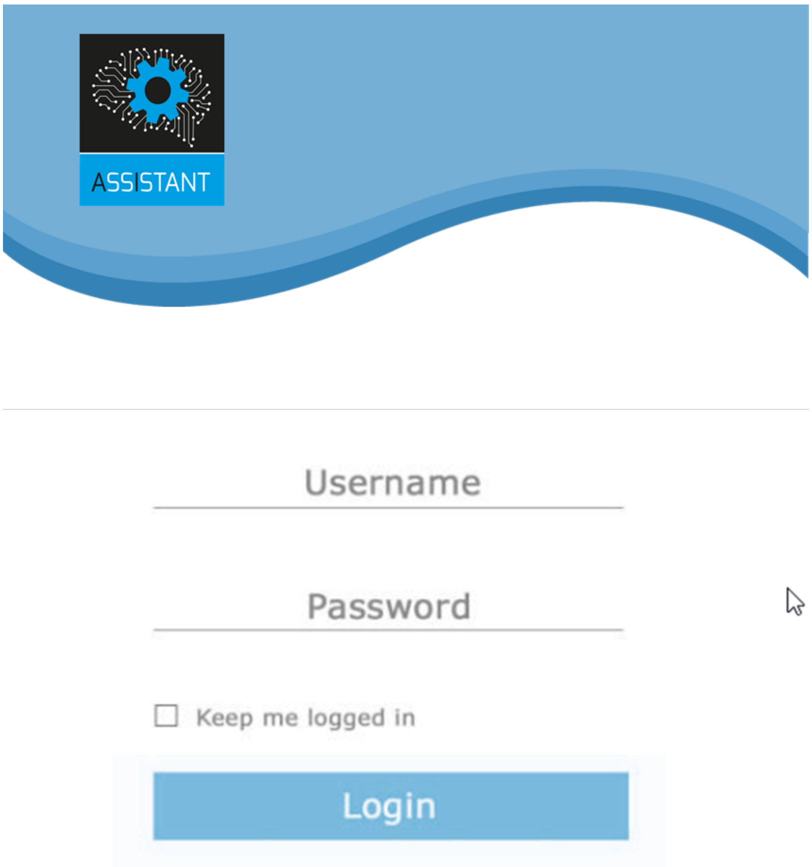

Figure 6 Loginscreen

The log-in screen (Figure 6) template was designed to be used as guideline of the page(s) that will be employed for user log-in to the applications. The welcome/home screen (Figure 7) is the screen that the user "sees" as soon as he logs in the application. It consists of three parts: header, content, and footer. The header contains the logo of the project, the logout button and the menus for navigating inside the application. The footer contains the logos of the developer(s) and the project logo. The content of the welcome screen is a dashboard style containing various action buttons, information, etc. The "Various Screens" template (Figure 8) is the template for the implementation of the other screens of the user interface. It also consists of the header, the footer, and the content section. The content section is divided into two parts: the left part contains the context menu (if applicable), while the right part contains the actual content of the screen.

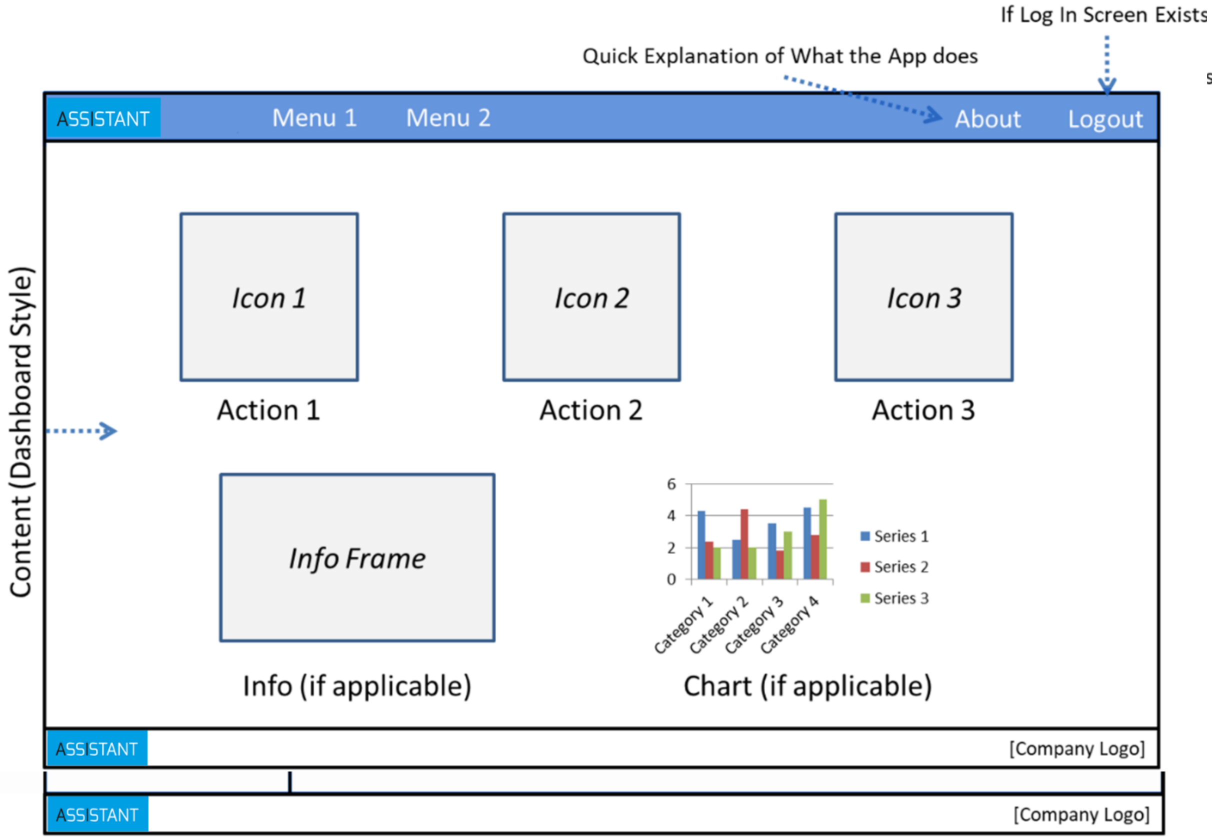

Figure 8 Various Screens

3.5 Security and Encryption

IT security is a set of strategies that prevents unauthorized access to organizational assets such as data. Data security strategies are divided mainly into two categories of strategies that will prevent unauthorized access to data (Access Control System) and data encryption strategies that even if data access fails, data cannot be read. ASSISTANT will implement both of these strategies. For the first category, an RBAC (Role Based Access Control) mechanism based on a central authentication system (CAS) will be developed, while data transition channels will enforce data encryption with a Transport Layer Security (TLS). The access control will be centrally implemented, but each client must make sure that access to security areas and data are properly handled.

3.5.1 Role Based Access Control (RBAC)

Role-based access control (RBAC) is an access-control mechanism defined around roles and privileges. The components of RBAC are permissions, subject and roles. Combined, these three elements form the security policy rules (i.e. role “Administrator” has permission “Creation” on subject “Users”). These policies are to be stored in a database that will implement the RBAC system. The policies will be enforced on service usage and/or data retrieval. Essential for such a system to work is the successful identification of the user and its role. This is called user authentication and is discussed in the next sections.

3.5.2 Central Authentication System (CAS)

The Central Authentication Service (CAS) is a single sign-on protocol for the web which is responsible of identifying and authenticating a user of the system to be the one that the user claims to be (by the use of a username and password). Its purpose is to permit a user to access multiple applications while providing their credentials (such as username and password) only once. It also allows web applications to authenticate users without gaining access to a user's security credentials, such as a password. The name CAS also refers to a software package that implements this protocol. The ASSISTANT platform foresees the use of different modules/applications by different users/companies. To allow one single access point to each user, according to the contract stipulated, i.e. which modules are available to this specific user, as part of the "user friendliness" the CAS service allows only one single access, avoiding multiple insertions of credentials. By doing this, also the security part is being handled in a centralized way by one service.

The CAS server and clients are the two physical components of the CAS system architecture depicted in Figure 9. They communicate by means of various protocols.

CAS Server
The CAS server is Java servlet built on the Spring Framework whose primary responsibility is to authenticate users and grant access to CAS-enabled services, commonly called CAS clients, by issuing and validating tickets. An SSO session is created when the server issues a ticket-granting ticket (TGT) to the user upon successful login. A service ticket (ST) is issued to a service at the user's request via browser redirects using the TGT as a token. The ST is subsequently validated at the CAS server via back-channel communication. These interactions are described in great detail in the CAS Protocol document.

CAS Clients
The term "CAS client" has two distinct meanings in its common use. A CAS client is any CAS-enabled application that can communicate with the server via a supported protocol. A CAS client is also a software package that can be integrated with various software platforms and applications in order to communicate with the CAS server via some authentication protocol (e.g. CAS, SAML, OAuth). CAS clients supporting a number of software platforms and products have been developed.

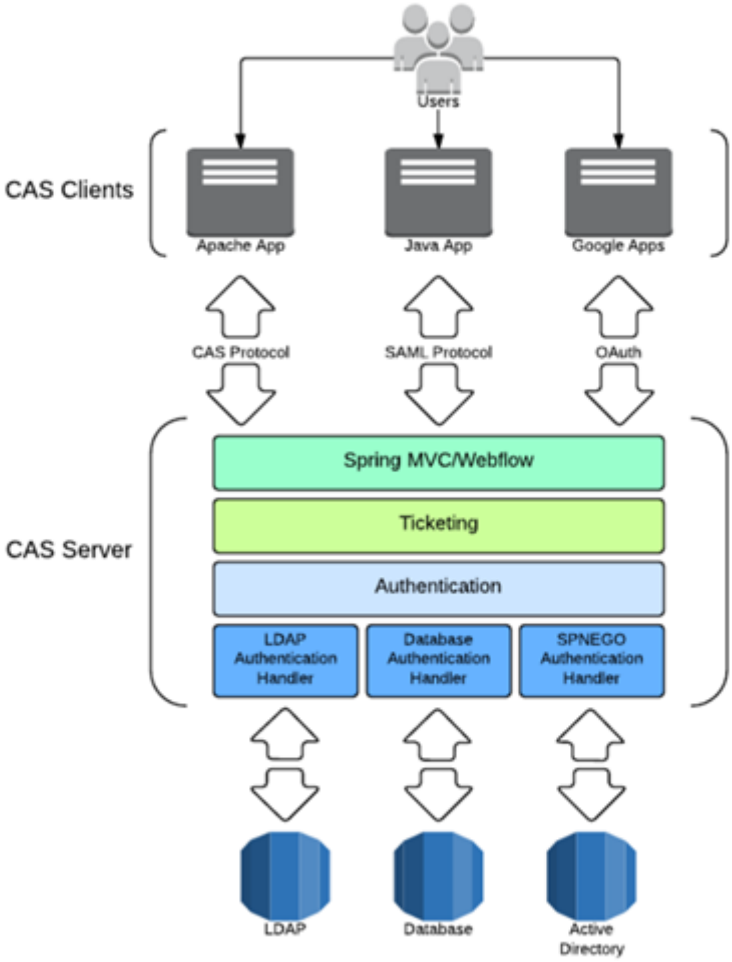

Figure 9 CAS Server Architecture components

3.5.3 Data Encryption

Exchange of data between components will be encrypted via industry standard technologies which make use of symmetric/asymmetric cryptography, such as TLS 1.2. Specific certificates for TLS 1.2 will be created and maintained to enforce data encryption on the communication level.

On the other hand, data storage should encrypt or otherwise obfuscate sensitive data as an extra security measurement.

3.6 Initial reflections on the architecture

As promised in previous parts of this architecture document, we will provide initial reflections on the architecture based on the ART-principles. It is planned to concretize these for the next revisions of the architecture while also coming up with additional aspects that need to be considered. In this first step of developing the human-centric architecture, we look at a very high level that reflects the current stage of the technical architecture. Once the technical architecture advances and provides more and more depth, the analysis will also go into more detail and not only look at the interaction of the system as a whole but it might also address specific components.

These initial reflections mostly focus on potential issues that could arise during the further course of the project and how to cope with them. However, we can also notice that even though we have so far only discussed responsible development and the ethical assessment on a quite abstract level in project meetings, we have experienced very positive reactions from the project partners. This is mentioned here because presenting only potential risk might create an imbalanced perception from a normative perspective. In the discussions that we have been involved so far in the project, we already influenced the mindsets. It is fair to state that we therefore may already have positively impacted decisions that were taken in the design of the architecture and in the writing of Deliverables D3.1, D5.1 and D6.1

3.6.1 Interoperability as negotiation

While interoperability is mostly framed as a technical issue in the architecture, it is important to also reflect how it is established. While potentially all technical artefacts are the process of negotiations and decisions of the developing teams, decisions in a distributed system are more difficult to take in this more complex environment as different actors play roles that are also part of different organizational entities.

These negotiations potentially include biases of the different actors, which may have been formed due to their former experiences, their individual preferences of toolchains and alike. It is important to mention that interoperability is not only a technical requirement within a service-oriented-architecture but it is also a matter of negotiation, a matter of compromises and a matter of the actors coming and working together on a personal level.

- ➔ It is therefore necessary that the deliberations and decisions about the architecture and technical implementation are done in a moderated environment to ensure the fair exchange of arguments and to ensure that the implementation is chosen that works best altogether for the entire system and especially for its users.

3.6.2 The early nature of requirements

It is in the nature of requirements that they are to be gathered as a first step in the project. Depending on them, the project develops in one way or another. They set the directions and give instructions as to what needs to be developed in the first place. For this reason, also in the ASSISTANT project, the requirements are gathered as a first step and are also published as Deliverables (for example, D7.1). Requirements often serve as a contract between the users, who require functionalities, and the developers, who implement functionalities. It is therefore relevant what is put in the requirements documents. Within ASSISTANT, the initial human-centric architecture comes at a point in time where the requirements are mostly already set. In order to develop a system that is ethical by design, it will be necessary to include the insights from the human-centric architecture in the requirements.

- ➔ It is therefore necessary to stress that the requirements that are collected in a very early phase of the document remain flexible. This way, insights from the human-centric architecture and the discussion it facilitates can be included as requirements that are equally to be fulfilled as the functional requirements.
- ➔ It is therefore necessary to rethink methodologies for further projects that acknowledge the lack of synchronicity between the moment when responsibility aspects would be necessary in the project and the moment when we are able to implement them. This is especially true, as the approach we chose is not only supposed to raise issues but also intervene in the development process. Within the ASSISTANT project we therefore plan to conduct a review and evaluation of existing methodologies as well as a refinement of selected integration methods to increase synchronicity.

3.6.3 Distributing responsibility

Responsibility: At first, what is important to mention when we look at the architecture at the level we did in this document, we see that the overall system consists of multiple services that interact with each other. For the user, however, the interaction with the system is an interaction with the system in general, not with the sum of its individual components. Reflections about whom the user can make responsible for certain behaviours of the system therefore become quite complex. This is even more the case as the responsibilities are delegated to different people within the development project.

- ➔ It is therefore necessary to develop a responsibility map that covers the responsibilities of different stakeholders. This has two functions: On the one hand, it makes the responsibilities for the developers explicit and, on the other hand, makes the responsibilities of the stakeholders visible for the end users.

We were able to observe within the early stage of the project that the service-oriented architecture also inspires discussions about where to locate functions. When we speak about multiple services that interact with each other, the definition of how to exactly cut the services is a matter of discussion. Depending on where a certain task – for example, ensuring data quality – is located, responsibility is also re-distributed.

3.6.4 Making decisions of the system explicit

Accountability: In order to be able to criticize decisions that the system provides to the user, it is necessary to understand what happens within the system. Depending on different grades of automation, it is necessary to come up with differently detailed descriptions of the processes that are taking place within the system. If no human oversight is planned, this requirement is even more important.

Here, again, it is necessary to point towards the service-oriented architecture. If different components interact to collaboratively produce a decision, it needs to be clear which component contributes what, and each component itself needs to work towards explainability to establish accountability.

- ➔ It is therefore necessary to develop clear decision trees and visualizations of decision processes that are then delegated to algorithms in the further course of the project. This again serves two functions: On the one hand, it opens up the assumptions for debate. On the other hand, it allows the user to understand the modes of reasoning within the system. These decision trees shall not only contain the contribution that each component delivers from a technical perspective but shall instead focus on the content of the decision itself.

If the functionality is framed as interaction between the different components of the service-oriented architecture and if therefore the focus is shifted towards interoperability, it is necessary to mention at this point that not all questions of accountability can be delegated to the interaction. Also, each component itself contributes functionality for which it needs to be held accountable. Therefore, not only the interaction and interoperability but also the functionality of the components itself need to be designed explicitly.

3.6.5 Operationalizing transparency by keeping privacy

Transparency requires the accessibility of information. Privacy requires that not all data that might be stored is accessible. Additionally, storing logfiles that make processes within the system potentially traceable comes with costs. They are often very technical, they need processing and storing, and they require certain skills to be understood and to be analysed.

- ➔ A strategy for understanding how a concrete decision was taken by the system needs to be developed. The output of this strategy could be a visualization that is not only traceable in cases of conflicts but also presented to the user.

3.7 Assessment criteria based on trustworthy guidelines

Within work package 2 of the ASSISTANT project, there are two different approaches towards the responsible development of the digital twin (see Figure 2). While this document mostly focuses on the development of the architecture, there is also a task that defines and evaluates assessment criteria. At this point in time, a preliminary list of assessment criteria already exists and is included in this document.

While the methodology for the development of a human-centric architecture is explicitly not supposed to define a check-list that needs to be ticked off, the aspects that are raised within the assessment criteria might still inform the development process of the digital twins and can also serve as a starting point for considerations about issues of responsibility.

While the architecture tries to integrate an ex-ante approach, the criteria perform an ex-post approach. Therefore, different frameworks are employed to solve the two different tasks. For the development of the criteria and the assessment, the concrete tools of Trustworthy Guidelines (High-Level Expert Group on AI, 2019) and ALTAI – Assessment List for Trustworthy AI – are used, while the human-centric architecture mobilizes the ART-principles as described above.

ALTAI aims to provide a basic evaluation process for Trustworthy AI self-evaluation. Organizations can draw elements relevant to the particular AI system from ALTAI or add elements to it as they see fit, taking into consideration the sector they operate in. It helps organizations to understand what Trustworthy AI is, in particular what risks an AI system might generate. It raises awareness of the potential impact of AI on society, the environment, consumers, workers and citizens (in particular children and people belonging to marginalized groups). It promotes involvement of all relevant stakeholders (within as well as outside of an organization). It helps gain insight on whether meaningful and appropriate solutions or processes to accomplish adherence to the requirements are already in place (through internal guidelines, governance processes etc.) or need to be put in place (High-Level Expert Group on AI, 2020).

ALTAI is supposed to help in fostering responsible and sustainable AI innovation in Europe. It seeks to make ethics a core pillar for developing a unique approach to AI, one that aims to benefit, empower and protect both individual human flourishing and the common good of society.

ALTAI and the trustworthy guidelines share the understanding of the importance of including values in design processes and of making these values explicit. It therefore makes sense to ensure an exchange between the two processes within the project. This will be established through the ethical management plan that is developed in task 2.3.

An initial component analysis based on these guidelines was performed within other tasks of WP 2. The initial results are placed in the appendix of this document. The considerations inform and shape the further development of individual components and therefore also might have an impact on the overall architecture, which is the reason to include them in the appendix of this initial version of the human-centric architecture document.

While for the developer, it might be easier to adopt concrete instructions instead of questions for the developing process, the assessment is still phrased in the form of questions, as this is the format that is required for the evaluation process. However, it seems possible to translate these questions into concrete instructions for the developers in the current revisions of the human-centric architecture.

4. Further process

This document will be updated, maintained, and revised in the course of the project. In this section, we will discuss the process, planned revisions and the upcoming steps to reflect the different levels (accountability, responsibility, transparency) in the following versions of this document. Additionally, the current version of the document will be mapped within the general process of the development of the human-centric architectural design (this can include a chart or a schematization).

The ART-principles – as discussed above – are the starting point for the development of this architecture. Initial reflections have been shared that shed light on the potential issues that might arise and how to cope with them. It is important to note that the ART-principles function as a starting point. During the concretization and implementation process it might be necessary to include more input or different frameworks in further revisions of the document.

So far, the ART-principles have only been applied to the overall architecture, the technical integration, and therefore on a high level. For the next revision, it is planned to take a closer look at the individual components that constitute the digital twins and reflect on these components from the perspectives of accountability, responsibility, and transparency. Additionally, the requirements for the respective component need to be considered: some of them might change, some of them might be extended or adapted to the new insights that are brought to the surface because of the reflection.

Two more revisions of this document will be published. One will come out during the process and show how practical the approach we chose and described will be and where and how it needs to be amended. The third will then be delivered at the end of the project and provide information in full detail about the concrete implementations.

Within the ASSISTANT project, there has been an amendment in regard to the timeline of the different inputs. Originally, this Deliverable was supposed to be submitted in month 8, which also was the time when the requirements and the conceptual architecture were supposed to be completed. Before the amendment, we planned to base our reflection on a questionnaire that was supposed to replace in-person workshops, which could not take place due to the pandemic. You can find the questionnaire in the appendix. After the amendment, however, we were able to base our reflections directly on the inputs from the other work packages and tasks. The answers we have received will be integrated in the next revisions of the Human Centric Architecture Deliverables.

5. Limits and Reflections

This chapter refers to the limits of the human-centric architecture document. Therefore, considerations on the role of this document within the project, as well as the current state of the living document are pointed out. Additionally, we discuss possible blind-spots of the document as it is created within and applied to a specific context: The authors reflect their disciplinary school of thought as well as its implementation and the specific context of AI in manufacturing.

While we try to be transparent about our approach, we also want to acknowledge that the process we chose is not without limits. By basing the development of the Human Centric Architecture on the ART-principles, we have chosen one specific approach where we could have also decided for others. We have provided reasons for this decision. However, this means that we do not include all the dimensions for reflection that are out in the field. We will keep that in mind and add other frameworks and approaches as they seem necessary and seem fit.

Another limit to the development of the architecture document is that, so far, the project team has not had the chance to meet in person. We hope that in the further course of the project, it will be possible to facilitate workshops and discussions in person. Especially in the very normative context of responsibility and ethics, meetings in person would be very helpful.

6. References

Aldewereld, H., Dignum, V., & Tan, Y. (2015). Design for ValuesInformation and communication technologies in Software DevelopmentSoftware development. In J. van den Hoven, P. E. Vermaas, & I. van de Poel (Eds.), *Handbook of Ethics, Values, and Technological Design* (pp. 831–845). Springer Netherlands. https://doi.org/10.1007/978-94-007-6970-0_26

Algorithm Watch. (2021). *AI Guidelines Global Inventory*. https://inventory.algorithmwatch.org

Dignum, V. (2019a). *Responsible Artificial Intelligence: How to Develop and Use AI in a Responsible Way*. Springer International Publishing. https://doi.org/10.1007/978-3-030-30371-6

Dignum, V. (2019b). *Toward AI Systems that Augment and Empower Humans by Understanding Us, our Society and the World Around Us* (Deliverable No. 1; HumanE AI (Grant Agreement: 761758)).

Hagendorff, T. (2020a). *The missing link in putting AI ethics into practice*. 22.

Hagendorff, T. (2020b). The Ethics of AI Ethics: An Evaluation of Guidelines. *Minds and Machines*, *30*(1), 99–120. https://doi.org/10.1007/s11023-020-09517-8

High-Level Expert Group on AI. (2019). *Ethics guidelines for trustworthy AI* (p. 41) [Report]. European Commussion.

High-Level Expert Group on AI. (2020). *The Assessment List for Trustworthy Artificial Intelligence (ALTAI) for self assessment* (p. 34). European Commission.

IDEO (Ed.). (2015). *The field guide to human-centered design: Design kit* (1st. ed). IDEO.

Leidner, J. L., & Plachouras, V. (2017). Ethical by Design: Ethics Best Practices for Natural Language Processing. *Proceedings of the First ACL Workshop on Ethics in Natural Language Processing*, 30–40. https://doi.org/10.18653/v1/W17-1604

LUMA Institute (Ed.). (2012). *Innovating for people: Handbook of human-centered design methods* (First edition). LUMA Institute.

7. Appendix

7.1 Abbreviations

Table 5 Abbreviations

Abbreviation	Meaning
ASSISTANT	LeArning and robuSt decision SupporT systems for agile mANufacTuring environments
ART-principles	Principles of Accountability, Responsibility, Transparency

7.2 Questionnaire for Initial Reflections

Question 1: General Information About The WP/UC

Before starting to set up the initial structure for the "Ethical-by-design" architecture for ASSISTANT, we would like to get a better understanding of the proposed solutions and the engineering challenges in the work packages and use cases.

1.1 Digital twin solutions

How do you envision digital twin solutions for process planning, production planning or reconfigurable manufacturing?

Can you give us one or more examples of how the problems you plan to solve with the implemented digital twin solution are usually solved in process planning, production planning or reconfigurable manufacturing?

1.2 Existing digital twin solutions in your field

How does your envisioned solution differ from other digital twin solutions in your field? Can you give us one or more examples of existing solutions for digital twin solutions for process planning, production planning or reconfigurable manufacturing or for the use of a data fabric that you are aware of?

1.3 Data handling, instrumentation and integration

Solutions developed in ASSISTANT are supposed to be integrated and orchestrated using a shared data fabric. Can you give us one or more examples of how this integration and orchestration has been handled in previous or similar projects or do you know of good examples of such a shared data fabric?

Please indicate, which type of information and data you are going to process with AI components and your estimation about the sensitivity of that data.

1.4 Data and ML/AI methods and components

Do you already have preferences for certain AI/ML tools and methods or are you planning to reuse and/or adapt previously developed methods in ASSISTANT? Can you give us one or more examples of the use of these methods in process planning, production planning or reconfigurable manufacturing?

Can you give us a rough overview of the type of data you are using with these methods?

Please name the components that you are developing that have a relation to AI

Question 2: Areas affected and initial ethical reflections

To get us started with identifying the aspects of your work packages or use case that need reflection from a Ethics perspective, we would like to ask you to provide us with any thoughts that you have – if any – that could be relevant for your tasks within the following dimensions. We are looking for the components that need to be further examined and discussed in the ongoing process of the project, to ensure that they are designed and developed in an ethically responsible way.

The three dimensions stem from the Human Centered AI framework. You can read more about the framework here.

2.1 Accountability

Accountability refers to the requirement for the system to be able to explain and justify its decisions to users and other relevant actors. To ensure accountability, decisions should be derivable from, and explained by, the decision-making mechanisms used. It also requires that the moral values and societal norms that inform the purpose of the system as well as their operational interpretations have been elicited in an open way involving all stakeholders.

2.2 Responsibility

Responsibility refers to the role of people themselves in their relation to AI systems. As the chain of responsibility grows, means are needed to link the AI systems' decisions to their input data and to the actions of stakeholders involved in the system's decision. Responsibility is not just about making rules to govern intelligent machines; it is about the whole socio-technical system in which the system operates, and which encompasses people, machines and institutions.

2.3 Transparency

Transparency indicates the capability to describe, inspect and reproduce the mechanisms through which AI systems make decisions and learn to adapt to their environment, and the provenance and dynamics of the data that is used and created by the system. Moreover, trust in the system will improve if we can ensure openness of affairs in all that is related to the system. As such, transparency is also about being explicit and open about choices and decisions concerning data sources and development processes and stakeholders. Stakeholders should also be involved in decisions about all models that use human data or affect human beings or can have other morally significant impact.

Question 3: Further Resources / Examples

It is important for us, to take your experiences from various disciplines into account and to learn from (best) practices. If you know any resources from previous projects, use-cases or from your experience, we would like to ask you to send them to us. If you want to describe context for the documents, please add the descriptions here.

The resources can include either architecture documents that reflect on ethical questions or articles from your disciplines that present approaches and experiences towards ethical engineering.

The resources can but do not necessarily have to be related to the *Human Centric Approach*.

Document	Filename	Comments

7.3 Component evaluation dimensions based on Trustworthy Guidelines

The following tables within this section provide an overview of an initial cross-analysis performed by partners in the project based on the Trustworthy guidelines. In addition, these tables show the initial identification of potential risks that need consideration.

The nomenclature used in these tables refers to the trustworthy requirements of (1) Human Agency and oversight (Trust1), (2) Technical Robustness and Safety (Trust2), (3) Privacy and Data Governance (Trust3), (4) Transparency (Trust4), (5) Diversity, Non-Discrimination and Fairness (Trust5), (6) Societal and Environmental Well-Being (Trust6), (7) Accountability (Trust7).

Component Definition	Trust1	*Trust2*	Trust3*	Trust4*	Trust5*	Trust6*	Trust7*
Process Manager UI	x		x	x			x
Process Designer		x		x		x	x
Process Predictor		x		x			x
Process Optimiser		x		x		x	x

Table 6 Trustworthy Guidelines: requirements for Process Planning

Component Definition	Trust1*	Trust2*	Trust3*	Trust4*	Trust5*	Trust6*	Trust7*
Simulation			x		x	x	x
Production Planner		x		x			x
Model Acquisition for Scheduling		x		x			x
Scheduler Optimisation		x		x		x	x
Production Manager UI	x		x	x			x

Table 7 Trustworthy Guidelines: requirements for Production Planning

Component Definition	Trust1*	Trust2*	Trust3*	Trust4*	Trust5*	Trust6*	Trust7*
Streamhandler	x		x				x
Execution Control and Reconfiguration		x		x		x	x
Digital Twin for Execution		x					x
Human Body Detection and Human Task Prediction		x	x	x			x
Human Side Interfaces	x		x				

Table 8 Trustworthy Guidelines: Real-Time Control and its components

Component Definition	Trust1*	Trust2*	Trust3*	Trust4*	Trust5*	Trust6*	Trust7*
Data Fabric		x	Optional*	Optional*		x	Optional*

Table 9 Trustworthy Guidelines: data fabric

*The Data fabric will require to be considered under transparency and accountability if any process of data modification by an AI component is embedded within it. Furthermore, privacy principles should be evaluated if sensitive information is kept within the data fabric or if any information can be linked to private information. In the opposite case, the data fabric should only focus on technical robustness and safety perspectives.

7.4 Component evaluation questions based on ALTAI Framework

Process Planning Feedback
General: 1. If personal data will be manipulated, are these processes aligned to a standard (IEEE, ISO)?. 2. What other stages could be used for data input/output (apart from the UI and data fabric) that could cause security concerns?. 3. What components would process or describe transparency results (i.e. users will be able to check explainability, open communication, and set/see traceability throughout the UI interface)?. 4. Have been established the ***responsibilities*** of the AI components (WP3-WP6) and the interactions involved (i.e. AI results) in architectural design ASSISTANT during its deployment and development? 5. Have been defined a methodology in which users can provide feedback (and tagging) from biased or risk information? 6. Are process/risk KPIs expected to be estimated online and provided to the users? 7. Are process/risks KPIs (if implemented) be formated with tagged information and kept easily accessible and secure (e.g. data fabric)? Process Manager UI: 1. Are end-users made adequately aware that a decision, content, advice, or outcome result from an algorithmic decision (especially important if these decisions, content, advice or outcomes are indirectly provided to end-users – e.g. shop floor workers)? 2. Are the end-users informed that they are interacting with an AI system? 3. Did you put in place procedures to avoid that end-users over-rely on the AI system? 4. Would it require users specific training on how to exercise oversight? If so, what protocols would be used for users to fulfil this training? 5. Have mechanisms been established to deal with privacy or data protection through the different communication channels (UI and data fabric)? 6. Would any personal information be required to manipulate the process manager UI (this implies if GDPR conditions have been set at each stage in case of required personal data)?. 7. Is the user/stakeholders informed of the accuracy of the results? Process Predictor: 1. Has the considerations of tagging – monitoring –documenting the system accuracy (e,g +how good are the predictions based on training /validation sets)? 2. It has been considered saving prediction conditions to avoid reprocessing and, therefore, using energy-consuming optimisation processes? Process Predictor and Process Optimiser: 1. What are the expected results and measures are taken if the AI component fails in its execution? 2. Can adversary results produce considerable damaging consequences (safety, economy, security) to your components or other components of ASSISTANT? 3. What implicates would have a failing component in other components that are dependent on it, including external ones (e.g. if process predictor fails, how will the process optimiser effect)? 4. How is it secured that data used for development, training, and estimations fulfil quality requirements established for each component involved?

5. Has placed the consideration of tagging and documenting the outputs (and its data used for generating those results) and keep them for accountability purposes
6. Does exist a complete fall-back plan in case of irreversible situations for running the system?
7. Are considered to save optimisation results to avoid re-running conditions that are not necessary to be re-evaluated? .

Table 10 Considerations for the Process Planning

Process Planning Feedback
General: 1. If personal data will be manipulated, are these processes aligned to a standard (IEEE, ISO, GDPR)? 2. What other stages could be used for data input/output (apart from the UI and data fabric) that could cause security concerns?. 3. What are the components that would process or describe transparency results (i.e. users will be able to check explainability, open communication, and set/see traceability throughout the UI interface)?. In other words, what component will give the users an explanation of what was developed by the global component and explain how the final result was obtained.? 4. Have been considered generating an individual component for developing transparency that will include analyses in terms of quality and, at the same time, forward this set of cases/results to the data fabric (as an additional tag and result)? 5. Have been established the responsibilities (and under what circumstances) of the components, users, and developers in case they could produce any harm or considerable economical impact (i.e. the system fails)? 6. Have been considered who will be responsible for a failure condition on data transfer. For example, who will be responsible for the receiver of the information or the sender (this analysis should consider users responsibility too).? 7. Have been considered to save simulation/optimisation processes and tag them to avoid reprocessing conditions and avoid using energy-intensive tasks (i.e. tag results in callback results if they already exist with the same sets)? 8. Have been established the ***responsibilities*** of the AI components (WP3-WP6) and the interactions involved (i.e. AI results) in architectural design ASSISTANT during its deployment and development? 9. Have been placed procedures to avoid that end-users over-rely on the AI system? 10. Would it require users specific training on how to exercise oversight? If so, what protocols would be used for users to fulfil this training? Production Manager UI: 1. Are end-users made adequately aware that a decision, content, advice, or outcome result from an algorithmic decision? 2. Are the end-users informed that they are interacting with an AI system? 3. Was it place procedures to avoid that end-users over-rely on the AI system? 4. If humans would participate in the decision making (and affecting WP5 or other processes), would it be required specific training on how to exercise oversight? 5. Have been established mechanisms to deal with privacy or data protection of data through the UI?. For example, would any personal information be required to manipulate the production manager UI for each development, testing, and deployment?. 6. Are the users informed of the system accuracy (from the components that predict or pass through a training process/parameter process)?

Simulation:

1. Have been considered that If simulations are describing/manipulating the same content of production planner and production scheduler information/data, the same considerations applied to them regarding data managing should be applied for the simulation.?
2. Have been considered that If training data are generated by simulation, each AI component results should be tagged with this consideration?
3. Has it been considered that if data generated by simulation will be used, the system users are aware of these considerations?
4. Can simulations be used as a source of explainability for optimal and non-optimal solutions?

Production Planner

1. What are the expected results and measures are taken if the AI component fails in its execution?
2. What implicates would have a component fail in another component that is dependant on it (if process predictor fails, how will be affected the process optimiser)?
3. How is it secured that data used for development, training, and estimations fulfilled the requirements of each component involved?
4. What would be the impact if the overall system crash (i.e. what alternatives are recommended to be used on these cases - Fall-back plan)?
5. Are production planner users (e.g. shop-floor users) aware that an algorithm has generated the production plan?
6. Can users override the main result and modify the plan (would this affect the ERP system directly or managed indirectly)?

Model Acquisition for Scheduling

1. Is it consider to put in place a methodology (algorithm, validation process, etc.) to establish that the models created are not accurate (e.g. place a tolerance and tag the accuracy of the models), especially if users wrongly set tables in the system?.
2. Is the previously defined validation error consider always to be reported to make sure the user does not over-rely on the system?.
3. Is it possible at any stage to modify the values of the tables from the users and, therefore, affect data integrity?
4. Have been considered the impact of allowing to use inaccurate models by the users?

Scheduler Optimisation

1. Have been considered that if reprocessing (run optimisation again) is not under conditions that would change results (e.g. stochastic optimisation), and the algorithms used for optimisation are exact (i.e. non-metaheuristics), the reported solutions would be the same as previous ones (i.e. avoid re-running unnecessary processes)?.
2. Have been analysed under conditions in data will make the optimisation process unable to be run?.
3. Are the users aware of the conditions in data that will make the optimisation process unable to be run?.

Table 11 Considerations for the Process Planning

Real-time control and actuation
General & stream handler: 1. Since the stream handler manages all types of information related to control, is there information that requires GDPR considerations at any stage?. 2. Are these processes (1) aligned to a standard (IEEE, ISO, GDPR)? 3. Does the responsibility and accountability to access such information will be lay on WP5 components or others?. 4. Does the digital twin of execution specifies anonymity for its component (clarification should be made for each component to secure privacy and data governance - i.e. make the WP5 component independent of such type of information)?. 5. Are defined how frequency or tolerances of error acceptable for defining training requirements, which will lead to review technical robustness and safety of the system? 6. Have been established the responsibilities of the AI components (WP3-WP6) and the interactions involved (i.e. AI results) in architectural design ASSISTANT during its deployment and development? Execution Controller: 1. Since The process Orchestrator uses information fed back from WP3 and WP5 through the data fabric, is there any instance in which the input information is validated? 2. Have been defined and specified conditions in which AI-based control systems should be overridden?. Digital Twin of Execution: 1. Has it been considered metrics that will be used to measure and evaluate the system performance and, at the same time, provide the user with dynamic information to check system reliability?. 2. Under what circumstances the digital twin should not be used, and how would the system recognise these conditions? 3. Are the user provided with the information defined in (2)? 4. Until what point is the digital twin model accountable in case of an error that would produce loss or harm? 5. Are determinates the risks involved in the planner to perform or estimate incorrect trajectories?. 6. Are implemented andy metric that defines the accuracy of the representation or models to the actual scenario?. If so, are these given to the users? 7. Were there procedures to avoid end-users over-rely on the AI system been considered in the architecture (consider numeral one and other methods)? Human Body Detection and Human Task Prediction: 1. Can the AI be combined with other sources of information to recognise specific users and at the same time, is the user fully aware of the use of this information? For example, does it involve only human recognition for safety and processing information?. 2. Has it been defined and provided to the user the information that will be handled (together with its purpose)?. 3. Has it been considered that If wearable connected through human body detection should focus on body detection for safety considerations or human-machine

behaviour analyses only, or other purpose specified and informed to the user?.
4. Has it been considered anonymity to avoid track users outside the duties involved on the workstation?.
5. Is there any potential form of attack to which the AI system could be vulnerable and in the long term produce harm (these include data poisoning, model evasion, model inversion, or misuse by the user)?.
6. Has it been considered the use of risk metrics and risk levels specific for the use cases?.

Smart Human Interfaces:

1. Can wearables information be linked to a user to be tracked and check their behaviour on the shop-floor (i.e. other than safety considerations or that clearly specified to the user)?
2. Are users provided with information regarding the possible threats to the AI system before their use (design faults, technical faults, environmental threats)?.

Table 12 Considerations for the Real-Time Control

Data Fabric
General: 1. Is there any process involved in data curation imposed over the data fabric? If so, are those processes AI-based or methodologies that can easily be linked to well-known methodologies? 2. Is there any standard that will be followed for data security and data managing? 3. What are the main processes for backup plans for collected/created information in case of error with the system that causes loss of information (from an architectural point of view)?. 4. Have been established the ***responsibilities*** of the AI assets (WP3-WP6) and their interactions involved (i.e. AI results)? What responsibilities are involved in WP6 in regards to their results and managing?

Table 13 Considerations for the data fabric

7.5 ASSISTANT Component Use Case Definitions

As already mentioned above in the document, this appendix also contains two sections from the technical architecture document from task 2.2. We provide that content because of two reasons: First, the content is relevant to understanding the project and the concrete application that will be build within ASSISTANT. Secondly, this allows us to also document progress of the integration of the technical and the human-centric architecture along the way, as both of them are living documents. The publication of sections of the technical architecture in task 2.2 freezes the current status and allows us to reflect on future changes. You can find those sections in the next pages.

7.5.1 Process Planning

7.5.1.1 Process Manager UI

Name	Process Manager
Brief Description	The process manager as a user interface supports users in generating efficient and effective decisions by applying the 6.
Involved components	The data fabric, the process designer, the process predictor and the process optimizer are involved components that interact with the process designer.
Pre-Conditions	Product and production system are required as input.
Basic Flow of Events	It is a user-specific interface, which, depending on the user's role, enables the control of the digital twin and provides visualisation of process planning artefacts like the resulting process plans, requirements, and skills. In addition, a chatbot supports, for example, process planners by answering their questions. //bullets for process planner, first time building the plan – Login – execute designer and optimizer (see below) – visualize input (data fabric), output of modules stored – manual changes
Alternative flows	– instead of executing he can just look at exisitng ones and can compare
Subflows	-
Key-Scenarios	– when needed: changes, new products, variants, changes to productions, manual changes of the production system o haven szenarios or changes.
Post-Conditions	Store generated visualizations in the data fabric.
Special requirements	-
Relevant pilot case(s)	AC and PSA

7.5.1.2 Process Designer

Name	Process Designer
Brief Description	The process designer developed in T 3.3 takes the three-dimensional files of the product and of the production system with all its resources to build all possible process graphs.
Involved components	The data fabric and the process manager are involved components that interact with the process designer.

Pre-Conditions	The three-dimensional files of the product to be produced and the production system used as well as their characteristics are present in the data fabric.
Basic Flow of Events	The first process designer first analyzes the product and the production system. A precedence graph will result after product analysis. Using the analysis information, the designer compares the resulting product and process requirements with the skills of the production system to assign all possible resources to each assembly, logisics, or monitoring process within the so-called process graph.
Alternative flows	Not applicable
Subflows	-
Key-Scenarios	The user selects the input data as precondition and starts process designing via the process manager. Then the process designer itself automatically develops the task and production skill model as well as the process graphs. Those results can be viewed via the process manager.
Post-Conditions	The production skill model, the task model and the resulting process graph must be stored inside the data fabric.
Special requirements	-
Relevant pilot case(s)	AC and PSA

7.5.1.3 Process Predictor

Name	Process Predictor
Brief Description	For different production plans and technical changes, the *process predictor* enables the forecast of various KPIs regarding cost, time, and quality.
Involved components	The data fabric, process designer and the process manager are involved components that interact with the process designer.
Pre-Conditions	Trained decision tree, change or process plans are available.
Basic Flow of Events	For three-dimensional technical changes, the designer is executed first. Developed process plans, as well as resulting process plans from three-dimensional changes, are then the input of the predictor. For each process in the process plan, a decision tree trained with historical data then predicts the KPIs depending on the assigned resources, tasks, and parts. In the end, the KPI prediction of a process plan is added up using all forecasts on the process step level. Similarly, the impact of textual changes can be predicted. By using natural language processing, the predictor identifies change features of a documented product or production change and a decision tree predicts the KPI categories based on the impact of similar historical changes.
Alternative flows	-
Subflows	-
Key-Scenarios	The user selects the input data as precondition and starts process prediction via the process manager. Then the process designer itself automatically develops the task and production skill model as well as the process graphs. Those results can be viewed via the process manager.
Post-Conditions	Store predicted KPIs
Special requirements	-

Relevant pilot case(s)	AC and PSA

7.5.1.4 Process Optimizer

Name	
Brief Description	
Involved components	
Pre-Conditions	
Basic Flow of Events	
Alternative flows	-
Subflows	-
Key-Scenarios	
Post-Conditions	
Special requirements	-
Relevant pilot case(s)	AC and PSA

evt. Ehsan kontaktieren

7.5.2 Production Planning and scheduling

7.5.2.1 Simulation

Name	Validate decision
Brief Description	Validate decision within the degrees of freedom of a certain use case scenario.
Involved components	Production Planner, Production Scheduler (Model Acquisition & Optimization), Production Manager UI, Data Fabric
Pre-Conditions	All required data is provided in the Data Fabric.
Basic Flow of Events	All alternative flows from below make sense, there is no basic flow.
Alternative flows	– Manual set up and execution: Production Manager UI –> sets up executes –> Simulation – Final validation of production plan: Production Planner –> provides decisions to be finally validated by –> Simulation – Iterative feedback from Simulation to Production Planner: Production Planner –> provides different choices all to be validated by –> Simulation – Generation of training data for model acquisition: Simulation –> is set up and executed several times to provide training data for –> Production Scheduler (Model Acquisition) – Final validation of production schedule : Production Scheduler (Optimization) –> provides decisions to be finally validated by –> Simulation – Iterative feedback from Simulation to Production Scheduler : Production Scheduler (Optimization) –> provides different choices all to be validated by –> Simulation
Subflows	
Key-Scenarios	– Validate an already given schedule regarding OTD, cost, etc. (SE Scenario Schedule Validation) – Validate different choices for release dates, shift models, prioritization logics, etc., regarding OTD, cost, etc. (SE

	Scenario Schedule Optimization) – Validate different choices for make-or-buy split regarding OTD, cost, etc. (SE Scenario Make-or-Buy Proposal)
Post-Conditions	All data calculated by the simulation needs to be stored in the Data Fabric.
Special requirements	-
Relevant pilot case(s)	SE, AC

7.5.2.2 Production Planner

Name	Capacity adjustment and requirement planning
Brief Description	Tools that automatically computes a production plan (quantity to produce per period, quantity to order, and capacity adjustment with overtime)
Involved components	Data fabric, domain model, scheduler, simulation
Pre-Conditions	Required data is available, the simulator is on and it create the required output.
Basic Flow of Events	(1) User connect to production manager UI. (2) Production planning interface send a request to domain model to update data (demand, machine and workers available) (3) Domain model get the data from the correct system (ERP, worker management, MES) (4) The user enter the targeted KPIs (minimize expected costs/ensure a service level of 95%/…) (5) Production planner find a production plan: 5.a Solver find production quantities and extra capacity required to meet the KPI targets 5.b The simulation validates the plan , in case of negative update capacity computation and resolve. 6. The production manager UI displays the plan, and the output of the latest simulation run.
Alternative flows	The loop in (5) stops after a predetermined iteration limit if no feasible plan is found.
Subflows	-
Key-Scenarios	The shopfloor manager wants to adjust it production capacity and place orders to suppliers based on the latest information on customer demand
Post-Conditions	All data calculated by the production planner needs to be stored in the Data Fabric.
Special requirements	-
Relevant pilot case(s)	SE, AC

7.5.2.3 Model Acquisition for scheduling

Name	
Brief Description	The component goal is to acquire a constraint model from a set of table with schedule data
Involved components	data fabric, scheduler optimization
Pre-Conditions	Data must be prepared in accordance with Task 4.3
Basic Flow of Events	1. The user manually launches Model Acquisition via Production

	Manager UI 2. The user selects one or more data tables from Data Fabric, that he wants to process 3. The user can specify, which columns will be outputs (i.e. can they be calculated from entries of other columns in data tables from the same or different rows) during the model acquisition 4. The user starts the process of Model Acquisition, which will result in a set of equations. 5. The user reviews the equations and, if needed, returns to the step 3, to select new set of output columns. When the user is satisfied with the result, the user proceeds to the step 6. 6. The user selects all or part of the equations and stores them into Data Fabric
Alternative flows	-
Subflows	-
Key-Scenarios	
Post-Conditions	All data created by the Model Acquisition needs to be stored in the Data Fabric.
Special requirements	
Relevant pilot case(s)	SE, AC

7.5.2.4 Scheduler's optimization

Name	
Brief Description	The component goal is to optimize a constraint model to create new schedule tables
Involved components	data fabric, simulation, production planner, model acquisition
Pre-Conditions	Data must be prepared in accordance with Task 4.3
Basic Flow of Events	1. The user manually launches Scheduler's Optimization via Production Manager UI 2. The user selects one of the models stored in Data Fabric 3. The user selects an optimization criteria 4. The user starts the process of schedule optimization. 5. The user reviews the results and, if needed, returns to the step 3, to select new optimization criteria. When the user is satisfied with the result, the user proceeds to the step 6. 6. The user stores the selected schedule table or tables into Data Fabric
Alternative flows	In the step 6, the user can use the selected schedule table for the simulation
Subflows	Step 5 should allow for comparisons between two or more schedule tables, that are results from different constraint models or the same model but with different optimization criteria
Key-Scenarios	
Post-Conditions	All data created by the Model Acquisition needs to be stored in the Data Fabric. Schedule table must be created in accordance with Task 4.3
Special requirements	
Relevant pilot case(s)	SE, AC

7.5.2.5 Production Manager UI

Name	
Brief Description	The user interface of the production manager will present the front part of the work-package 4.
Involved components	Production planning tool, model acquisition (scheduling) tool and simulation tool
Pre-Conditions	
Basic Flow of Events	(1) user authentication (2) User input data (keyboard, files), and store in the data fabric (3) Trigger the tool (simulation/planner/scheduler), the tool store the result in the data fabric. (4) Read the result from the data fabric and display.
Alternative flows	-
Subflows	-
Key-Scenarios	The production manager will be able to launch the simulation, planning and production scheduling tools (model acquisition) as well as to visualize their outputs. In this interface, it will also be possible to view the different KPIs with the possibility of adjusting them.
Post-Conditions	
Special requirements	
Relevant pilot case(s)	SE, AC

7.5.3 Real-time control and actuation

7.5.3.1 Streamhandler

Name	Streamhandler
Brief Description	A publish/subscribe infrastructure based on Apache Kafka
Involved components	Components requiring real time access to shopfloor data, Components that produce high volume/velocity shopfloor data.
Pre-Conditions	A topic has been setup for producers to produce messages and consumers to consume the arriving messages
Basic Flow of Events	Consumer registers to the predefined topic Producer sends a message to the predefined topic Consumer is notified and provided the new message Consumer works on the message and notifies infrastructure of the work completion.
Alternative flows	-
Subflows	-
Key-Scenarios	Gathering shopfloor data
Post-Conditions	
Special requirements	
Relevant pilot case(s)	AC, STELLANTIS

7.5.3.2 Process Orchestrator

Name	
Brief Description	This component is responsible for feeding the digital twin for the task at hand.
Involved components	Digital Twin of Execution, Process Designer, Production Planner, Data Fabric, Quality Controller
Pre-Conditions	
Basic Flow of Events	1. Retrievs the product/process/resource assignment from Production planner 2. Retrievs the production process to be executed in the production line from Process Planner 3. The Quality control module will monitor production and provide feedback to the Process Orchestrator regarding the process in a closed-loop manner 4. Triggers the production digital twin to perform the related process
Alternative flows	-
Subflows	-
Key-Scenarios	AC, STELLANTIS
Post-Conditions	
Special requirements	
Relevant pilot case(s)	

7.5.3.3 Quality Controller

Name	
Brief Description	This component monitors the production and provides feedback on the Process Orchestrator regarding the process in a closed-loop manner.
Involved components	Process Orchestrator, Digital Twin of Execution
Pre-Conditions	Quality controller must always be aware of production's current situation
Basic Flow of Events	-
Alternative flows	-
Subflows	-
Key-Scenarios	
Post-Conditions	
Special requirements	-
Relevant pilot case(s)	AC, STELLANTIS

7.5.3.4 Digital Twin of Execution

Name	
Brief Description	The DTE is responsible for providing information regarding the current state of the production.
Involved components	Process Orchestrator, StreamHandler, Data Fabric, Quality Controller
Pre-Conditions	
Basic Flow of Events	Process orchestrator: • Triggers the cell's DTE to perform relative process. • Connection with WP4: retrieves the product/process/resource assignment.

	• Connection with WP3: retrieves the production process to be executed in the production line. Streamhandler: • Collection of data from shopfloor and store them in Data Fabric. • Real time monitoring Data Fabric: • Direct communication to the data storage. • Integration with tools through the exposure of domain models. Quality Controller: • Provision of feedback to the Process Orchestrator
Alternative flows	1. Human Body Detection (HBD) gives input to the DTE and the Human Task Prediction (HTP) about the position of the operator. 2. HTP informs the DTE about the operator's tasks that are executed. 3. DTE informs Process Orchestrator. 4. DTE informs the operator through human side interfaces
Subflows	Gather human side information (HBD, HTP): • Gathering of sensor data • Data reasoning • Extraction of human state (position, current task execution) Provide information to human: • Process Orchestrator provides information to DTE about task execution • Transfer information to human side interfaces • Visualize current state Gather Robot information: • Robot controller sends Robot Status to DTE • Execution feedback is transferred to Execution Controller Provide Robot information: • Execution Controller sends task • Task splits into *resource, action, and part* • Task resolves to machine command • Command is sent for execution
Key-Scenarios	
Post-Conditions	
Special requirements	
Relevant pilot case(s)	STELLANTIS, AC

7.5.4 Secure and intelligence data fabric

7.5.4.1 Data Fabric

Name	Data Fabric
Brief Description	The ASSISTANT data fabric is a data management system that provides a unified interface to data access and storage, and abstracts the resource management details of data provisioning
Involved components	The data fabric is implemented in a layered architecture realized as a set of distributed services. The data fabric builds on widely available

	and technology neutral tools such as JSON-based REST services and integrates with (but does not depend on) other components of the ASSISTANT architecture.
Pre-Conditions	Data fabric services are deployed on infrastructure resources, components have access to the data fabric services via networks and APIs. Clients of data fabric services (human end-users and software tools) are authenticated using a shared security infrastructure.
Basic Flow of Events	1) clients are authenticated in the security infrastructure 2) clients initiate and drive interactions with data fabric services via the ASSISTANT domain models and / or the data fabric APIs and service interfaces 3) data is stored and (optionally) processed in the data fabric services, potentially resulting in multiple new data sets 4) metadata for all new data is generated and published by the data fabric services to facilitate usage of data 5) clients make use of data fabric search and query services in conjunction with metadata to organize, identify, access, retrieve, and use data outside the data fabric (likely through the domain models)
Alternative flows	similar to the basic flow, but time series data is routed to the system via the Intrasoft StreamHandler
Subflows	-
Key-Scenarios	A (digital twin) tool defines a domain model for its data and uses this and the data fabric to abstract data management and provisioning
Post-Conditions	Data is available and persistently stored / archived in the data fabric
Special requirements	-
Relevant pilot case(s)	SE, AC, PSA

7.6 ASSISTANT Component Interface Requirements

7.6.1 Process Planning

7.6.1.1 Process Designer

7.6.1.1.1 Overall description

The *process designer* developed in T 3.3 takes the three-dimensional files of the product and of the production system with all its resources to build all possible process graphs. These process graphs consider all ways to produce the product with the given production system. To do so, submodules first analyze the product and the production system. A precedence graph will result after product information. Using the analysis information, the designer compares the resulting product and process requirements with the skills of the production system to assign all possible resources to each production or monitoring process within the so-called process graph. The digital twin for process planning must identify the processes necessary to produce a part, predict the process parameter of each process, select the optimal process plans by comparing different resource allocations and provide a suitable interaction platform for the various roles in production (e.g., process planners, production planners, developers, and operators).

7.6.1.1.2 External interface requirements

User interfaces	not applicable
Hardware interfaces	not applicable
Software interfaces	Data Fabric, Process Manager
Communications interfaces	The process manager triggers the transfer JTS, JSON, XML

7.6.1.1.3 Performance requirements

The first outputs of the process designer are assembly, logistics, and monitoring processes, and dependent product and process requirements as well as capabilities of the production system, which are described in the so-called process graph. For this to be represented at different levels of granularity, the underlying **processes must be represented at a granular level** (cf. Defliverable 3.1 - 5.1.3). That includes representations of process levels considering sequences of activities needed to produce a product, operational levels, task levels, and functional levels.

During process design, process planners generate and collect all necessary skills, processes, and their sequence or application to produce a product. The processes, requirements and skills are stored inside a process graph. The **process designer must generate reliable process graphs** realistically mapping this step. The mapping includes a process plan prediction. Cost, time, and quality parameter predictions must be enabled for each step in the process plan. These include, for example, product, production and reconfiguration costs. Those predictions have to depend on the detailed threefold structure: the process to be executed, the product to be produced, and the resource to be used. Additionally the mapping includes a change prediction. Cost, time, and quality parameters should be predicted for textual and geometrical changes. Those predictions must depend on change characteristics. Specific performance indicators on computing time, required storage space, capacities, response times, real-time capability, and others are to be detailed in the further course of the project.

7.6.1.1.4 Logical database requirements

Regarding the logical requirements for information to be stored in and provided by databases, the following can be said: The type of data and information used by several modules in WP3 are among others CAD files of the produced products, CAD files of the used production systems, and MES data. This data is generated and uploaded into the database by different users and has to be accessible to several parties accordingly. The frequency of use can vary from several requests a day to only a few requests per year. Data entities and their relationships heavily depend on the existing data structure in the respective companies and the specific ways CAD and MES data is stored. Integrity constraints are limited to the requirement that the CAD file used as an input for the process designer are up to date and correct, in terms of being a physically correct model of the real product/process to be analysed. As of now, no specific data retention requirements could be identified for the process planner.

7.6.1.2Process Predictor

7.6.1.2.1 Overall description

For different production plans and technical changes, the *process predictor* enables the forecast of various KPIs of the magical triangle, including cost, time, and quality. For three-dimensional technical changes, the designer is executed first. Developed process plans, as well as resulting process plans from three-dimensional changes, are then the input of the predictor. For each process in the process plan, a decision tree trained with historical data then predicts the KPIs depending on the assigned resources, tasks, and parts. In the end, the KPI prediction of a process plan is added up using all forecasts on the process step level. Similarly, the impact of textual changes can be predicted. By using natural language processing, the predictor identifies change features of a documented product or production change and a decision tree predicts the KPI categories based on the impact of similar historical changes.

7.6.1.2.2 External interface requirements

User interfaces	Not applicable
Hardware interfaces	Not applicable
Software interfaces	Process Designer, Data Fabric, and Process Manager
Communications interfaces	JTS and JSON

7.6.1.2.3 Performance requirements

The process optimizer must enable predictions of various parameters of processes since they influence the optimal process plan choice. This includes KPIs regarding cost, time, and quality, that should be predicted for each process step. Those are the base to select an optimal process plan. The representation of the process parameter prediction can be specified as follows. The predictor must map process plan predictions. Cost, time, and quality parameter predictions must be enabled for each step in the process plan. These include, for example, product, production, and reconfiguration costs. Those predictions have to depend on the detailed threefold structure: the process to be executed, the product to be produced, and the resource to be used. Furthermore, the DTPP must map change predictions. Cost, time, and quality parameters should be predicted for textual and geometrical changes depending on change characteristics. Specific performance indicators on computing time, required storage space, capacities, response times, real-time capability, and others are to be detailed in the further course of the project.

7.6.1.2.4 Logical database requirements

Regarding the logical database requirements of the predictor, the same requirements as for the process designer (cf. 3.1.1.4) account. Additionally, it can be emphasized that especially JTS, JSON files, and equivalents will serve as main input and output data.

7.6.1.3 Process Optimizer

7.6.1.3.1 Overall description

The *process optimizer* selects the optimal process planning by evaluating the value of an objective function. The submodules generate possible process plans, add secondary tasks, evaluate the objective function using the predictor and validate the process plans via simulations. During Optimization, the submodules are iteratively executed to derive the process plan with the highest objective value.

7.6.1.3.2 External interface requirements

User interfaces	Not applicable
Hardware interfaces	Not applicable
Software interfaces	Process Predictor, Data Fabric, and Process Manager
Communications interfaces	JSON and XML

7.6.1.3.3 Performance requirements

The process optimizer needs to support selecting the optimal process plan by choosing a process plan with optimal parameters. To enable a realistic representation of process optimization, the following requirements must be met. Firtsly, the optimizer must **map the users KPI value within the optimization**. The optimal process plan must satisfy the user in terms of boundary clarified by him/her for KPIs. If a user sets a KPI acceptance range/limit, the optimal process plan should respect the proposed limit from the user. The tool should inform the user if no process plan matches his expectations. Secondly**, the optimizer must facilitate the automated generation of possible process plans**. The results of the optimizers must therfore allow for automatically suggesting a (scalable) process plan The optimizer must include approaches to automatically suggest possible resource assignments to the processes and must include resulting transportation processes. The suggested plan must respect the constraints/targets set by the user. Thirdly, **the optimizer's results must facilitate suggesting robust process plans** and include the best alternative process plans for probable machine breakdowns and operator illness. That is process plans that remain valid when the parameters change. This must be enabled by predicting the KPIs. The tool for automatic process plan generation must account for all flexibility of the line. This adjustment allows spreading the load along the line to react to possible parameter variation like processing time or machine failure, and to maintain performance. Specific performance indicators on computing time, required storage space, capacities, response times, real-time capability, and others are also for this module to be detailed in the further course of the project.

7.6.1.3.4 Logical database requirements

Regarding the logical database requirements of the predictor, the same requirements as for the process designer (cf. 3.1.1.4) account. Additionally, it can be emphasized that especially JTS, JSON files, and equivalents will serve as main input and output data.

7.6.1.4 Process Manager

7.6.1.4.1 Overall description

The *process manager* supports users in generating efficient and effective decisions by applying the DTPP. It is a user-specific interface, which, depending on the user's role, enables the control of the digital twin and provides visualisation of process planning artefacts like the resulting process plans, requirements, and skills. In addition, a chatbot supports, for example, process planners by answering their questions.

7.6.1.4.2 External interface requirements

User interfaces	Mouse click or textual user intend within the process manager
Hardware interfaces	Not applicable
Software interfaces	– data fabric: itself here retrieval of production and product data – process designer: provided by process predictor, to call module and provide location of data – process predictor: to call module and provide location of data – process optimizer: to call module and provide location of data
Communications interfaces	Python or C#-based communication interfaces

7.6.1.4.3 Performance requirements

The process manager has to support users in controlling the process planning. The applicable requirements in this regard are as follows. Firstly, **the tool must enable the timely execution of process planning** allowing the execution of all process planning phases described in Deliverable 3.1 (WP3). Secondly, **the tool must support the user by analysing the process planning itself**. The analysis includes the visual and textual evaluation or description of the inputs, outputs, and the individual process planning steps. Thirdly, **the tool must allow user-specific permission rights**. Constructors, process planners, production planners and operators should have different rights to execute and analyse the process planning. Accordingly, the tool has to automatically check and grant or deny access to the analysis or execution. Specific performance indicators on computing time, required storage space, capacities, response times, real-time capability, and others are again to be detailed in the further course of the project.

7.6.1.4.4 Logical database requirements

Regarding the logical database requirements of the predictor, the same requirements as for the process designer (cf. 3.1.1.4) account. Additionally, it can be emphasized that especially

Windows Forms, APIs and JSON files (or equivalent) will serve as main input and output data formats.

7.6.2 Production Planning and scheduling

7.6.2.1 Simulation

7.6.2.1.1 Overall description

Simulation allows to validate decisions within the degrees of freedom of a certain use case scenario, e.g.
- Validate an already given schedule regarding OTD, cost, etc. (SE Scenario Schedule Validation)
- Validate different choices for release dates, shift models, prioritization logics, etc., regarding OTD, cost, etc. (SE Scenario Schedule Optimization)
- Validate different choice for make-or-buy split regarding OTD, cost, etc. (SE Scenario Make-or-Buy Proposal)

To this purpose, this component will calculate a material flow simulation with all the required data input and providing all calculated data output. A major benefit is the detailed view on the production flow that allows to agree on the "best" decisions from a business target perspective. The time and effort to create simulation runs are relevant objectives. The goal is to reduce this effort significantly by automating this process and by finding the most relevant simulation experiments.

7.6.2.1.2 External interface requirements

User interfaces	– User can set up (load and edit) and execute manual simulation runs. – User can view simulation results. – User can compare different simulation runs.
Hardware interfaces	-
Software interfaces	Production planner Model Acquisition and Optimization for scheduler Data Fabric External tools: – Simulation tools (e.g. Tecnomatix Plant Simulation) – python libraries (e.g. for creating lightweight simulations to be integrated in an easier way than commercial tools)
Communications interfaces	Interface between simulation and production planner : – Production planner triggers simulation, signaling the simulation's input data is available in the data fabric – Simulator triggers the production planner that the simulation's output data is available in data fabric.

	Communicate by read/write data from the data fabric. Communication with Production Manager UI (for manual triggering) and scheduler are similar to production planner

7.6.2.1.3 Performance requirements

Multi-user is out-of-scope. Multi-core (4 cores) should be considered.
Amount of information to be handled and response duration in time depend strongly on the industrial use case and will be specified later.

7.6.2.1.4 Logical database requirements

Cannot be (fore)seen yet. Need an integrated view from all components.
Data entities and their relationships will be specified in Task 4.3.
Regular data management (read/write/update).

7.6.2.2 Production Planner

7.6.2.2.1 Overall description

Tool for production planning, including capacity and requirement planning. Given input data from the domain model, and KPI targets inputted by the user through the production planner interface, the tool automatically suggest decisions related with the adjustment (shift length) of the production capacity, subcontracting, and orders to place to suppliers. See D4.1 for more information.

7.6.2.2.2 External interface requirements

User interfaces	Production planner will not provide any graphical interface, but it will interact with the production planner UI. The production planner UI allows: – The user to control the outputted plan by entering targeted KPI values, or by giving a priority to the objectives. – The user to visualize/compare different production plans. – The tool should show the impact of uncertain date (worst case scenario, average,….)
Hardware interfaces	-
Software interfaces	**Interface with the material flow simulation:** The objective is to validate/evaluate the production plan. The precise information transferred between the two software may change during the project, but a first version is given below. The information toward the simulation includes the production load per process step and per period, the extra work required, the quantity of outsources production. The message return by the simulation gives the end date of each production load. **Interface with Production Manager UI:** Production planner -> production manager UI: send the production plan, and this include the matrix of: – Production quantity for each item in each period – Inventory Level for each item in each period

	– Number of overtime required per resource – Purchase quantity for each component/subcontracted item. **Interface with Production manager UI :** – Target values for KPIs – ordering of the KPIs **Interface with the domain model** to get the input data and store the output data.
Communications interfaces	**Communication with data fabric:** Communicate by REST services provided by the datafabric to get each input data required for production planning (resources with capacity, Flexible BOM, capacity consumption per operation on each resource, costs, targeted KPIs values, ordering of the KPIs, ...) Communicate by REST services provided by the datafabric to store each output data (production quantity per period, inventory level per item and period, quantity ordered to suppliers/subcontractors) **Communication with production manager UI:** Production planner will provide a rest service for the production manager to request a production run. (1) Production Manager UI store the targeted KPI values and the ordering of the KPIs to the data fabric. (2) Production Manager requests a run from production planner. (3) Production planner computes the production plan and it stores it posts the results in the data fabric. (4) Production planner respond to the request to inform computation are done. **Communication with simulation:** Simulation provides a rest service for the production manager to request a simulation run. (1) Production planner post the production quantity per period in the data fabric, as well as the parameters of the simulation run. (2) Production planner requests a simulation run. (3) Simulation get the simulation's input data from the data fabric (3) Simulation runs and post the simulation output in the data fabric. (4) Simulation responds to the request to inform computation are done.

7.6.2.2.3 Performance requirements

The tool should be able to handle 1/10 simultaneous users; The tool will consume a heavy load on the processor (ideally, 100% for several hours) as well as memory (several GO in RAM). It might handle some mega octet of data during communication with other softwares, and some logs maybe quiet heavy (close to 1 G0). The response time will be lower than few hours with updates every few minutes.

7.6.2.2.4 Logical database requirements

Regular data management (read/write/update).

7.6.2.3 Model Acquisition for scheduling

7.6.2.3.1 Overall description

Model Acquisition is aimed to obtain a constraint model from a set of tables with schedule data. Adherence to a precise structure of data is not required. The Model Acquisition will find relations between tables and functional relations beween columns within tables to use it to generate a constraint model. User can select output columns, i.e. the columns which values are the result of formula that takes inputs from some other column or columns.
The user can later review the obtained model and select or deselect constraints, change output columns and send the model to the scheduler’s optimization, the simulation or the production planner.

7.6.2.3.2 External interface requirements

User interfaces	Graphic UI with the support of M/KB. User can select tables and input/output columns User can review and select acquired constraints User can run different acquired models and compare their results
Hardware interfaces	-
Software interfaces	Data Fabric, scheduler’s optimization
Communications interfaces	Python or Java

7.6.2.3.3 Performance requirements

Single-user access. A computer with multi-core CPU and large amount of RAM is advised. Exact requirements could be specified later, when the scope of an industrial case is known

7.6.2.3.4 Logical database requirements

A requirement for an access to a relational database, specified in accordance with the Task 4.3

7.6.2.4 Scheduler / Optimization of scheduling

7.6.2.4.1 Overall description

Optimization of scheduling is aimed to take any model obtained by the Model acquisition and create a schedule table from it. The user can select one of the criteria and launch this tool.

After the optmization process, the newly created schedule table can be tested in the Simulation or stored directly in the Data Fabric

7.6.2.4.2 External interface requirements

User interfaces	Graphic UI with the support of M/KB. User can run different acquired models and compare their results
Hardware interfaces	-
Software interfaces	Data Fabric, model acquisition, simulation
Communications interfaces	Python or Java

7.6.2.4.3 Performance requirements

Single-user access. A computer with multi-core CPU and large amount of RAM is advised. Exact requirements could be specified later, when the scope of an industrial case is known

7.6.2.4.4 Logical database requirements

A requirement for an access to a relational database, specified in accordance with the Task 4.3

7.6.2.5 Production Manager UI

7.6.2.5.1 Overall description

The production manager User Interface will give the user the possibility to communicate with the modules developed in WP4 while hiding the complexity of these modules. The following outputs from WP4 modules will be managed by the production manager UI:

Production planning

- Production plan and purchasing plan.
- Planned inventory/back-order level.
- Planned resource/capacity consumption.
- Target KPIs values
- Impact of uncertainty on production plan

Scheduling (Model acquisition)

- Model and schedule visualization
- Optimization cost and main KPIs
- Gantt chart (schedule) and visualisation of resource utilisation

Simulation

- Input load data (customer orders and due date)
- Initial factory state/condition
- Input planning/scheduling decisions
- Input simulation parameters
- Output production flow
- Main KPIs
- Output comparison of different simulation runs

Three levels architecture is needed:

1. display level: Display outputs, KPIs

2. Control level: Read data from (3) and transfer it to (1). This level will also send additional constraints to involved modules based on KPIs adjustement
3. Data level

7.6.2.5.2 External interface requirements

User interfaces	Thin client (web application)
Hardware interfaces	Not applicable
Software interfaces	Production planning, model acquisition (scheduling), simulation software modules (. exec if possible) -> WP4 backend
Communications interfaces	Json and Java script modules Communicate through REST Service to get the data from the data fabric (transform if needed) Use the REST service provided by production planner to trigger a production planning run. The production planning service will require authentication

7.6.2.5.3 Performance requirements

The following performances are expected:

- Adjustable KPIS: The user can specify the characteristic of the solution to create, i.e., providing upper or lower bound for the KPI, assigning weights to the optimization objectives, enter additional constraint on the plan/schedule, etc.
- Plot tables with several rows and columns based on the planning horizon and scheduling horizon.
- Plot textual form of the acquired model with link back to the data
- UML diagram in the model (if aplicable)

7.6.2.5.4 Logical database requirements

Not applicable, inputs to the production manager UI will be provided by Json format

7.6.3 Real-time control and actuation

7.6.3.1 Streamhandler

7.6.3.1.1 Overall description

Streamhandler is a high-performance distributed streaming platform for handling real-time data based on Apache Kafka.

7.6.3.1.2 External interface requirements

User interfaces	-
Hardware interfaces	-
Software interfaces	Kafka API
Communications interfaces	Kafka Consumer/Producer/Connectors. Connector bridges for various data sources.

7.6.3.1.3 Performance requirements

Up to 5 seconds from producing a message until message reaches consumer.

7.6.3.1.4 Logical database requirements

None

7.6.3.2 Process Orchestrator

7.6.3.2.1 Overall description

This component will serve as the entry point of the WP5 developments since it will be responsible for triggering the cell's digital twin to perform the related process. It will also be indirectly connected to the task scheduler/planner by retrieving the product/process/resource assignment and the Process Planner by retrieving the production process to be executed in the production line. The Quality control module will monitor production and provide feedback to the Process Orchestrator regarding the process in a closed-loop manner. Once a defect from the Quality Control module is identified or a problem during a process execution occurs, the Execution Controller in an online fashion will communicate the error to the Process Planner for evaluating and providing alternative process plans as well as to the Production Scheduler for adapting the schedule accordingly (i.e., continue with other products and/or resources).

7.6.3.2.2 External interface requirements

User interfaces	
Hardware interfaces	
Software interfaces	REST API
Communications interfaces	REST API

7.6.3.2.3 Performance requirements

7.6.3.2.4 Logical database requirements

7.6.3.3 Quality Controller

7.6.3.3.1 Overall description

The combination of historical and streamed statistical analyses and machine learning methods will use the information to evaluate and predict both the state of the process and the state of the products.

7.6.3.3.2 External interface requirements

User interfaces	-
Hardware interfaces	-
Software interfaces	Process orchestrator: • Quality controller takes as input information regarding the state of the process and the state of the products. DTE: • Exchange of data regarding the status of the production
Communications interfaces	

7.6.3.3.3 Performance requirements

Quality control is a component that is continuously running during the production and provides information regarding the state of the process and the products, which leads to the need of the real-time response time.

7.6.3.3.4 Logical database requirements

Sensor data, event data

7.6.3.4 Digital Twin of Execution

7.6.3.4.1 Overall description

The DTE will represent the whole workstation. It must interact directly with the real world and more specifically it takes as input information about the workstation area layout, the resources and the different parts that exist in the real world. The proposed DTE is hardware agnostic and could integrate seamlessly multiple robots and sensors.

7.6.3.4.2 External interface requirements

User interfaces	AR application, Web application
Hardware interfaces	Robots (ROS), Sensors (ROS), AR headset (Web socket), Android devices (Web socket)
Software interfaces	• Process Orchestrator TBD message format and content • StreamHandler TBD message format and content • Data Fabric TBD message format and content • Quality Controller TBD message format and content
Communications interfaces	ROS, bridges, REST, bridges for MQTT, bridges for OPCUA, PROFITNET

7.6.3.4.3 Performance requirements

UI response time: ranges between 1 to 3 seconds
HTP response time: real time
HBD response time: around 0.05 seconds

7.6.3.4.4 Logical database requirements

Sensor data, event data (streamhandler)

7.6.4 Secure and intelligence data fabric

7.6.4.1 Data Fabric

7.6.4.1.1 Overall description

7.6.4.1.2 External interface requirements

User interfaces	The data fabric does not provide end-user oriented GUIs but does integrate with other project data visualization tools
Hardware interfaces	The data fabric services are deployed as virtualized components (virtual machines and / or containers) and does not make use of platform-specific hardware features or interfaces
Software interfaces	The data fabric provides APIs for all services and integrates with the ASSISTANT domain model tools for abstraction
Communications interfaces	Data fabric services are accessible over networks using standardized service technologies, e.g., JSON-based REST services

7.6.4.1.3 Performance requirements

The data fabric services must operate within reasonable parameters for application-specific data acess and query response time, and requires suitable amounts of storage (memory and disk / persistent storage) and network capacity for caching, access, and transfer of data. The data fabric is designed for multi-tenant environments and concurrent access to the system services. As the data fabric services only provides limited capabilities for data processing within the data fabric itself, not hard requirements are placed on compute capacity.

7.6.4.1.4 Logical database requirements

The data fabric is designed to make use of and abstract multiple types of data stores, including databases. The fabric itself requires platform capabilities for storing and efficiently associating (text-based) metadata to store data. Typically the metadata will be several orders of magnitude smaller than the payload data (at least for large data sets) and as such the data fabric needs capabilities but not a lot of capacity for this.