Deliverable 3.1

Spezifikation des Anwendungsfalls

(Specification of the Application Scenario)

Dokument Version	:	Final
Abgabe	:	31.01.2021
Dissemination Level	:	Öffentlich
Beitrag zu	:	WP3
Dokument Inhaber	:	BOC
Dokument Name	:	ComplAI-D3.1 Spezifikation Anwendungsfall
Revision	:	1.0

Projektakronym	:	compl@i
Projekttitel	:	Collaborative Model-Based Process Assessment for trustworthy AI in Robotic Platforms
Grant Agreement n.	:	33755860
Call	:	IDEEN LAB 4.0 (2019)
Projektdauer	:	12 Monate, beginnend mit 01/02/2020
Website	:	https://complai.innovation-laboratory.org/

Revision History

REVISION	DATE	INVOLVED PARTNERS	DESCRIPTION
0.1	12/04/2020	BOC	Initial, commented table of content, Content collection
0.2	19/04/2020	BOC	Extension: Case Refinement, Architecture
0.4	21/04/2020	BOC	Use Case Discussion and possible AI scenarios
0.5	31/04/2020	BOC	Initial Version for project internal reasons
0.8	15/12/2020	BOC	Revisting the document and transforming it to a public deliverable
1.0	31/01/2021	JR, JKU	Final Approval as public deliverable

List of Contributors:

Wilfrid Utz (BOC), Anna Sumereder (BOC), Damiano Falcioni (BOC), Harald Kühn (BOC)

Bernhard Dieber (JR), Benjamin Breiling (JR)

List of Reviewers:

Stefan Schumann (JKU)

Benjamin Breiling (JR)

Kurzfassung

Dieses Dokument behandelt die Definition und Spezifikation von Demonstrationsszenarien, um die Demonstrationsfälle zu konkretisieren und ein gemeinsames Verständnis für die Motivation und Ziele dieses Projekts zu schaffen. Um dies zu erreichen, werden Beobachtungen und Anforderungen beschrieben. Weiters ist dieses Dokument der erste Teil von zwei für das gesamte Arbeitspaket. Es dient als Grundlage für die folgende Demonstration der Anwendungsfälle.

Die Demonstrationsszenarien stehen in Verbindung mit den Labor-Experimenten, den cyber-physischen Systemen und den Geräten, welche von den Projektpartnern – BOC und JR – zur Verfügung gestellt werden. Basierend auf diesen verteilten Umgebungen haben sich verschiedenste Anforderungen ergeben, insbesondere die Notwendigkeit eines Netzwerkes. Zusätzlich haben mehrere Beobachtungen ergeben, dass eine Transformation der erwähnten Systeme erfolgen muss, da intelligente Umgebungen notwendig sind. Intelligenz in diesem Kontext bedeutet, (1) dass Wissen und Expertenwissen von menschlichen Stakeholdern externalisiert wird und (2), dass kognitive Fähigkeiten erforderlich sind sowohl für die Modellrepräsentation als auch für die Funktionalitäten, um den Modellwert und Ausführung positiv zu beeinflussen. Stakeholder mit unterschiedlichen Hintergründen sollen unterstützt werden, um die Komplexität zu begreifen, die Relationen und Abhängigkeiten des Systems zu verstehen, sowie eine transparente Sicht auf den Einfluss von Design-Entscheidungen zu bekommen.

Basierend auf den Beobachtungen zielt dieses Projekt insbesondere auf die ethischen, rechtlichen und sicherheitsrelevanten Aspekte eines cyber-physischen Systems ab. Dabei soll ein modellierungsbasierter und wissensgetriebener Ansatz die Interaktionen in dem Umfeld erleichtern. Drei Roboterplattformen dienen als cyber-physische Systeme für die Anwendungsszenarien (a) ein Dobot Magician Roboter Arm, (b) eine Makeblock mBot mobile Platform und (3) ein System, welches die Kooperation zwischen Menschen und Roboter ermöglicht. Alle diese Plattformen haben verschiedene Anforderungen und führen zu unterschiedlichen Herausforderungen, insbesondere in Bezug auf KI und System-Transformationen.

Aufgrund der identifizierten Herausforderungen für jedes Demonstrationsszenario durch Dekomposition der einzelnen Anwendungsszenarien wurde eine Architektur erstellt. Diese besteht aus drei Bereichen, welche in Abstimmung mit dem modellierungsbasierten Ansatz gewählt wurden. Die Bereiche sind (1) die Design-Umgebung, welche Modellierungswerkzeuge umfasst um Wissensrepräsentationen von domainspezifischen Anwendungsszenarien zu erstellen, (2) die Ausführungsumgebung, welche abstrakte Gerätefunktionalitäten und Interaktionsschnittstellen zur Verfügung stellt, und (3) die Compliance-Umgebung, welche ethische, rechtliche und sicherheitsrelevante Überlegungen für die Design- und die Ausführungsumgebung zur Verfügung stellt.

Um anschauliche und transparente Anwendungsszenarien erstellen zu können, wurde die Supermarkt-Domäne gewählt. Folgende Fragen dienen als Motivation für die Auswahl von Anwendungsfällen:

(1) Wie wird die Roboter-Ausführung mittels KI und modelliert und wie kann man deren Compliance bewerten?

(2) Wie kann ein Kriterienkatalog erstellt, modelliert und bewertet werden? und

(3) Wie operieren übereinstimmende Modelle auf der Roboterplattform?

Die Szenarien wurden gewählt im Hinblick auf folgende Herausforderungen: die Externalisierung von Wissen, die Selektion von domänspezifischen Kriterien ausgerichtet auf ethische, rechtliche und sicherheitsrelevante Aspekte, sowie auf technologische Einschränkungen insbesondere während der Ausführung wurden beachtet.

Im nachfolgenden Dokument werden mehr Details sowie ein Prototyp beschrieben: "D3.2".

Executive Summary

The document at hand is the initial deliverable of work package 3 and aims at the definition and specification of the demonstration scenarios by raising question and requirements in order to concretize the demonstration cases and establish a common understanding of the project's motivation and objective. Furthermore, this document covers the first phase of the work package by specifying the application scenarios, which will be the foundation for phase two – the demonstration of the application scenarios.

The demonstration scenarios are related to laboratory experiments, and cyber-physical system setup and devices provided by the partners – BOC and JR. Based on this distributed settings, various requirements, in particular a connection network, came up. Moreover, a set of different observations created the need for transforming the mentioned systems towards intelligent environments, where intelligence can be seen as (1) externalization of knowledge and expertise of human stakeholders, and as (2) cognitive capabilities that are based on the model representation and model-value/processing functionalities. Stakeholders from different backgrounds are supported to grasp the complexity, understand system relations and dependencies as well as the impact of design decisions transparently.

Based on the observations, the project specifically addresses the ethical, (criminal) legal, and security/safety aspects of the CPS environment, considering a model-based/knowledge-driven approach of interaction in the environment. Three robotic platforms serve as CPS for the application scenarios (a) a robot arm, (b) mobile platforms, and (c) a human/robot cooperation. All of those platforms have different requirements as well as create different challenges with respect to the transformation towards AI systems.

Based on the challenges identified per demonstration case resulting from the decomposition of the application scenario, the compl@i architecture has been identified. It is separated in three environments in accordance with the model-based approach. Those are (1) design environment that encompasses modelling tools for creating knowledge representations of domain-specific application cases, (2) execution environment that abstracts device functionality and provided interaction endpoints, and (3) compliance environment that imposes ethical, legal and security/safety considerations on the design as well as the execution environment.

As domain for the application scenarios, a smart supermarket case was chosen in order to create demonstrative and transparent application scenarios. Following questions served as a motivation for selecting the application cases: (1) How to model AI, robotics and assess compliance? (2) How to create criteria catalogues that can be modelled and assessed? and (3) How to operate compliant models on a robotic platform?. The scenarios were chosen as they address the challenges of knowledge externalization, the selection of domain-specific criteria, aligned with ethical, legal, security/safety and technological constraints that are imposed on the system as well as on the interpretation level during execution.

More details and a prototype will be discussed in the follow up deliverable “D3.2 “.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

AI	Artificial Intelligence
API	Application Programming Interface
CPS	Cyber Physical System
DMN	Decision Model and Notation
HTTP	Hypertext Transfer Protocol
IoT	Internet of Things
REST	Representational State Transfer
SDK	Software Development Kit
SOAP	Simple Object Access Protocol
UI	User Interface

1. Introduction

The document at hand is the initial deliverable on work package 3 of the compl@i project and aims at the definition and specification of the demonstration scenarios by raising question and requirements in order to concretize the demonstration cases and establish a common understanding of the projects motivation and objective.

The demonstration scenarios are based on the background provided by JR and BOC, specifically related to laboratory experiments, and cyber-physical system setup and devices provided by the partners.

An observation within these experiments is the need to transform those systems towards **intelligent** environments. Intelligence is understood in this context twofold:

- as **smart functionality** that is defined and handled applying a knowledge-based approach. This implies that the knowledge and expertise of human stakeholders is externalized, and specific aspects are supported by artificial intelligence technologies like image recognition. Nevertheless, human intervention and decision is considered as a defining element.
- as a **cognitive capability** that is realized based on the model representation and model-value/processing functionalities. Stakeholders from different backgrounds are supported to grasp the complexity, understand system relations and dependencies as well as the impact of design decisions transparently.

Based on these observations, the project specifically addresses the ethical, (criminal) legal, and security/safety aspects of the CPS environment, considering a model-based/knowledge-driven approach of interaction in the environment. The challenges of these aspects with respect to **transformation towards AI systems** are formulated and discussed below, motivating the selection of application cases for evaluation:

1. *How to model AI, robotics and assess compliance?* This question addresses the challenge on the knowledge externalization. Current modelling techniques do not provide the necessary semantic expressiveness to cover all specific aspects, but they are considered as silos. Specific needs are covered, whereas the transformative nature is only partially considered.
2. *How to create criteria catalogues that can be modelled and assessed?* This challenge is concerned with selection of domain-specific criteria, aligned with ethical, legal, security/safety and technological constraints that are imposed on the system. A systematic technique and approach could include (a) structuring the criteria for compliance assessments, (b) integrating these catalogues on design, modelling as well as robotic layer of the system and (c) providing the possibility so that it can be instantiated and executed during the different phases of abstraction, decomposition and execution.
3. *How to operate compliant models on a robotic platform?* As a technical challenge, this question targets the interpretation level during execution. As the criteria imposed affect the design as well as the implementation and deployment phases (through human intervention and knowledge), the formalized knowledge needs to be reflected with appropriate mechanism on the robotic platform. For this reason, revisions including temporal aspects need to be considered as well as the contextual environment settings that influence the operation.

In order to pave the way for a clear understanding, three aspects should be defined in more detail. Our interpretation is presented below:

- **Digital Environment** – A socio-technical ecosystem that uses digitization technology like IoT, edge computing, networks, etc. to create a digital representative of real world artefacts – a so-called digital twin – in order to apply digital processing technology like AI, Big Data Analytics and so on in order to create added value.
- **Modelling** – The process that applies methods to "system under studies" in order to create a conceptualised representation – simplified, formalised, intentional focus – of real world artefacts enabling the processing via methods and algorithms – query, simulation, transformation, etc. –in order to gain insights.
- **OMiLAB** – Innovation Corner Approach Concept, methodology and instruments reflecting three levels of abstractions – physical, model and application domain layer – and three perspectives – information technology, infrastructure and

project domains – in order to facilitate co-creative digital optimisation for innovation or co-creative digital transformation for disruption.

1.1 Relation to Work Package

According to the project plan, this work package is structured in 2 phases:

1. Specification of application scenarios, and
2. Demonstration of the application scenarios.

This deliverable is related to the first phase and covers the tasks defined, performed during the online partner meeting held on April 7-8, 2020, leading to the documentation of the scenario within this document.

1.2 Document Structure

At the beginning, summaries in German as well as in English are provided in order to get an overview of the deliverable at hand.

This deliverable is structured in the following chapters: this introductory section provides the context of the work performed and results achieved. Chapter 2 introduces the OMILAB as an experimentation space (conceptually and physically/virtually) to operate and run experiments. The demonstration scenario for the transformation towards AI-systems is sliced into 3 sub-scenarios discussed in chapter 3. The “version” without AI is contrasted with the case developed for artificial intelligence. Chapter 4 targets an initial architectural proposal discussed in the project team and agreed upon. Chapter 5 provides a concise conclusion including further steps and considerations.

2. Demonstration Scenario Setting

The demonstration scenario developed is structured according to the approach developed in the OMILAB setting. This setting is briefly introduced below based on [1],[2]. The concrete physical setting at the BOC Digital Innovation Laboratory is used as a foundation to derive the requirements, provide prototypical solutions and co-creatively design and assess the feasibility of the project challenges.

2.1 OMILAB Digital Product Design Framework

The OMILAB considers 3 layers that are introduced below, as a design space for digital products.

- **Business Layer** – The business layer is applicable to define and clarify the application aspect of the novel idea. Methods and tools to co-create the design of novel business models are established and result in a concept refinement. This layer is understood as an externalisation possibility and space for conceptual analysis, at a high-level of abstraction.
- **Conceptual Modelling Layer** – The conceptual modelling layer is responsible to depict and assess the concepts identified and decompose them on a formal level. Conceptual modelling languages on different levels of expressiveness are positioned on this layer to describe the cases.
- **Proof-of-Concept Layer** – The proof-of-concept layer is responsible to assess and evaluate whether the idea can be realized and is feasible to operate properly. Abstraction techniques are applicable to elevate the functional capabilities of this layer towards conceptual modelling.

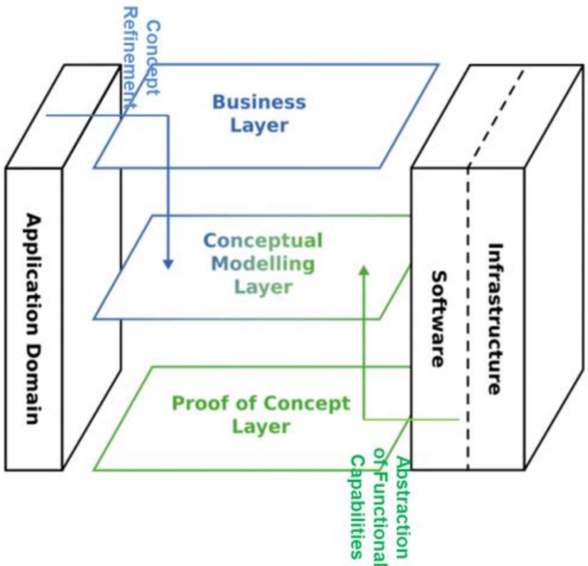

Figure 1: Digital Product Design Framework[2]

As visualized in Figure 1, the three layers are complimented with cross-cutting dimensions of software and infrastructure, required on all layers to design, model, develop and deploy experiments. In addition, the experiments are driven by the application domain that defines the approach, methods and tools selected for the specific instance.

The conceptual architecture of the OMiLAB Digital Product Design Framework is instantiated for various domains and application cases, resulting in a concrete representation of each layer and dimension. The core concerns within the

[1] D. Götzinger, E. T. Miron, and F. Staffel, "OMiLAB: An open collaborative environment for modeling method engineering," in *Domain-Specific Conceptual Modeling: Concepts, Methods and Tools*, Cham: Springer International Publishing, 2016, pp. 55–76.

[2] D. Bork, R. A. Buchmann, D. Karagiannis, M. Lee, and E.-T. Miron, "An Open Platform for Modeling Method Conceptualization: The OMiLAB Digital Ecosystem," *Commun. Assoc. Inf. Syst.*, pp. 673–679, 2019.

framework are connectivity and requirements from a conceptual, interaction and technical viewpoint. Technical connectivity is realised through standardised communication interfaces of the software-based, physical and user interaction components. Artefacts on each layer expose interfaces, define the required and provided communication structure and format for operations. The deployment mechanism reflects these connectivity issues and establishes a network of functionalities that are useful and adequate for the purpose defined.

On a conceptual level, connectivity considerations are related to the structure and behaviour of the modelling constructs, representing the knowledge of the domain. The language and corresponding metamodel defines the syntax, semantic and notation of the vocabulary applied, including dependency links and relations within and beyond a layer (in a networked manner).

Interaction is considered as a connectivity issue related to the human and machine intervention on design and execution level. Machine interaction targets aspects of data and information interchange between layers and elements on the layers, for example:

1. the sensing information of a CPS on the feasibility layer is transformed and abstracted to model artefacts,
2. the actuation information triggers functionality, or
3. the design artefacts impact the conceptual modelling and so on.

Human interaction streams are defined to design and model the system and establish the transformation required as decomposition (top-down) and abstraction of capabilities (bottom-up). These interactions can be classically triggered and conducted through user interfaces and tools, as well as beyondby using means of voice or image/video recognition as push or pull mechanisms.

2.2 Assessment embedded in OMILAB Framework

Based on the challenges of the project, the architecture of the OMiLAB is extended to consider compliance concerns as security/safety, ethical and legal aspects within the operation of such a system. This means that the layers and artefacts above become aware of specific criteria. The vision defined is to include domain-specific assessment logic during design and operation of the system, clarifying responsibilities and enabling interventions according to the input of experts in the field. These assessments impact the design and modelling layer predominantly (assuming a top-down approach).

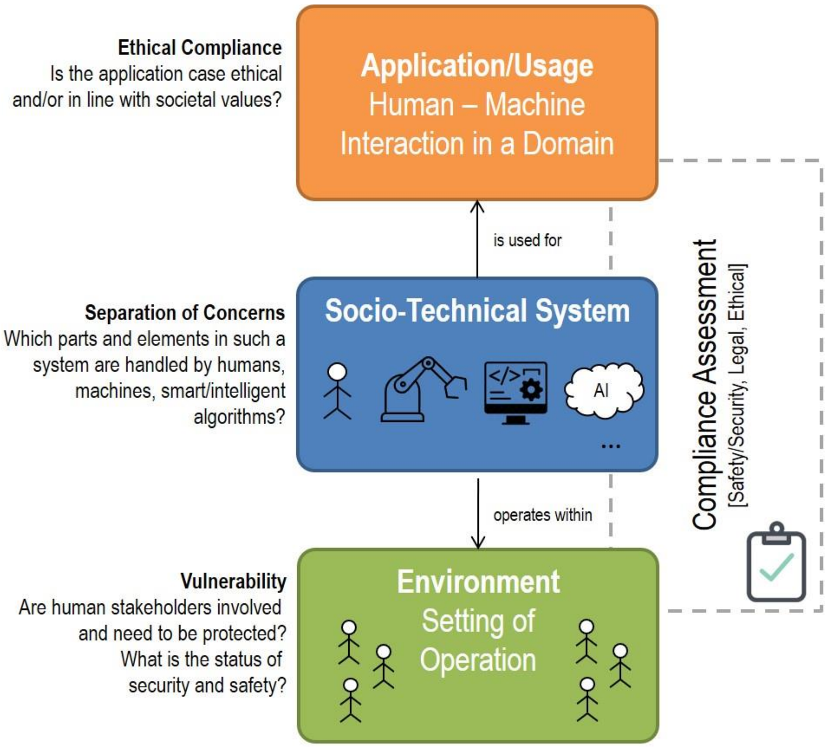

Figure 2: Compliance Assessment of Socio-Technical System

Figure 2 shows graphically the system boundaries, elements and components of the socio-technical system considered for the definition of the application scenarios:

- **Application/Usage** – The top part establishes the context, in which the system is applicable for and used within. It represents the domain and characteristics of the systems for two purposes: a) domain representation of the application case to identify the challenges and functional needs and b) context of the compliance assessment from an ethical viewpoint. The second purpose is concerned with whether the application is in line with societal values and implicit rules of the environment and culture the system will operate within. The Compliance assessment on this level is concerned with the question whether human stakeholders in the application are affected and/or targeted.
- **Environment** – On the bottom, the environment defines the setting of the operation as environmental conditions. The vulnerability of actors is in scope on this level and it is assumed that the context of the system impacts the compliance demand (laboratory setting vs. experiment under real-life conditions). For example, robotic engineers, users or any other human actors interacting within the environment are considered as human stakeholders in this context.
- **Socio-technical System** – The part in the centre refers to – as an organisational approach – the interaction of human actors with technology within a complex workplace setting. Infrastructure, technology as services as well as application and hardware components are combined for a specific purpose and goal. Human behaviour within such a complex system is driving the provision of technology. In specific by applying concepts like the human-centred approach or human in the loop.

Our assessment system for compliance builds upon a criteria catalogue that is structured based on the classification of security/safety, ethical and legal dimensions. Criteria are filtered according to the concrete characteristics of the application and its domain as well as the robotic experiment layout, including software components and AI functionalities supporting the design and execution.

Analogous to aspect-oriented programming, the criteria and resulting assessment influences the socio-technical systems and imposes the specific criteria as questionnaires and checklists during the system definition and implementation as well as filtering and listing to the behaviour during the execution (e.g. service calls, request and response structure, environment sensors, orchestration audit logs, user interventions). From a specification point of view, the catalogue acts as a design guideline imposed on the responsible stakeholders.

The knowledge on the criteria is formally described in extension building blocks that are embedded in domain-specific metamodels within the conceptual modelling layer.

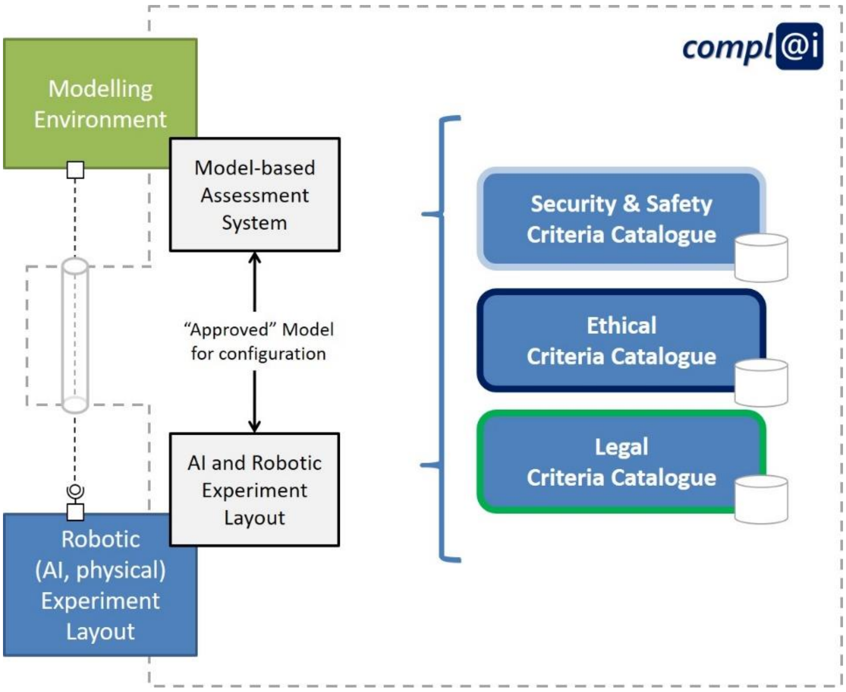

Figure 3: Model-based Assessment System for Robotic Experiments

The architectural discussion is graphically shown in Figure 3. The classification of assessment and compliance criteria are defined according to the above-mentioned classification – security/safety, ethical and legal dimensions – and understood as a database of compliance objectives and guidelines. These objectives act on the socio-technical system, formally described in models and enacting the robotic experiment layout. The imposed guidelines can have different characteristics of importance, for example obligatory, highly recommended or optional based on their content and context. They are related to the assets on both levels, modelling as well as robotics, and define the enactment of those (e.g. checklists to formalisation as code). Therefore, a translation of unstructured textual guidelines, experiences, and interpretations of legal context information into a formal model representation is needed, represented in the graphic above as "Model-based Assessment System". The system includes flexibility, re-use and adaptivity requirements.

The compliance assessment results act as a filter on the communication channel towards the robotic experiment layout and within. This implies that any interaction is assessed, evaluated, and verified for "approval", whereas approval is understood in a broad sense. The approval logic is related to the temporal validity and revision of criteria defined (as they might potentially change along the evolution of legal frameworks, societal values, etc.).

A key question is, if also a criteria catalogue for the technological aspects must be included in the model-based assessment system. And further on, how such an additional catalogue would interaction with ethical and legal criteria. Although in Figure 3 ethical and legal criteria catalogue are defined separately, there are some interconnections between those two. In this project, it is planned to combine ethics and law in order to pave the way for providing a clearer direction of allowed operations. It is necessary to identify out how and to which extent technology must be integrated here.

2.3 Laboratory Environment: OMiLAB at BOC and Robotics Lab at JR

The application case developed, and experimental cases are tested initially within the OMiLAB Node at BOC. BOC offers expert knowledge on conceptual modelling. To gather and include the extensive knowledge in the robotics field, the expertise of JR is included in the development case setup. This combination allows for local testing and evaluation of specific aspects and provides the scope for a novel, distributed development approach of such experiments.

Figure 4 shows how the distributed setting of laboratory experiments looks like on a high level. BOC focuses on conceptual modelling, whereas JR concentrates on robotic systems. The connection of both locations via the internet serves as a laboratory environment for compl@i. In addition to those infrastructure requirements, ethical and legal experts provide further value applied based on the laboratory environment and its stakeholders.

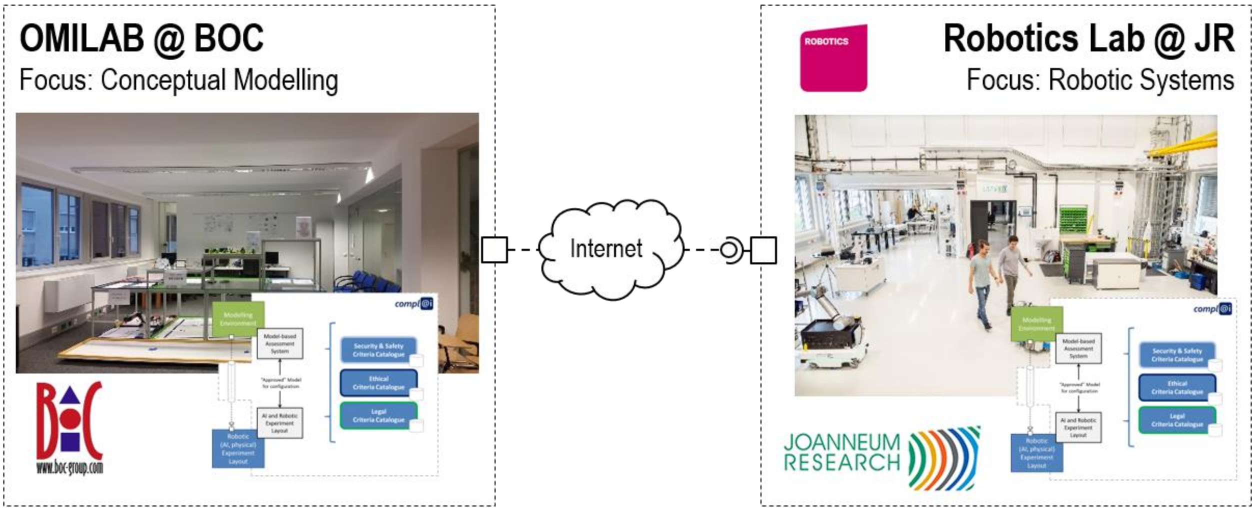

Figure 4: Distributed Setting of Laboratory Experiments

3. Initial Application Scenario

The application scenario for compl@i is decomposed in three dedicated cases that individually showcase the applicability of the approach. Furthermore, those scenarios are used as a foundation to derive develop demonstration prototypes and formulate research questions.

3.1 Overview of Application Scenario Domain

As everybody is familiar with a supermarket scenario, this demonstrative domain was chosen for the application scenarios. There is a set of underlaying concepts enabling AI and robotic application scenarios. Following three platforms exist: social, virtual and physical. Their preconditions and model-based approaches are presented below:

1. **Social Platform** – This platform mostly represents the traditional approach of shopping. The preconditions consist of capabilities of the actor. It is assumed that a human actor has a capability to (a) interpret informal and semi-formal models, (b) interact between physical and digital world, (c) complete incomplete and vague models. A model-based approach is followed, as this is easy to understand for humans, simple, ideally provides a graphical representation and no virtualization necessary. Furthermore, incomplete knowledge can be easily dealt with by using hints, heuristics or images. Any kind of adaptivity is automatically performed by the human actor. For example, if 6 eggs are not available, 10 eggs with a long BBD=best before date are chosen.
2. **Virtual Platform** – The virtual platform requires some preparations, for instance an online shop needs to virtualize the provided products, a user account has to be established, an APIs needs to be available and a workflow engine has to be programmed. Here, a model-based approach can be supportive in the context of formal correct and verifiable sequences of actions that can be executed by a workflow engine using the provided online shop APIs. For example, late binding of products to the shop order or rule-based workflows can be seen as means of adaptivity.
3. **Physical Platform** – For the physical platform robot and sensor devices for the appropriate moves must be installed in order to pick up the provided products. Furthermore, these robot and sensor devices must be secured and work in a legal and ethical correct way. Here, a model-based approach can be supportive in the context of formal correct and verifiable sequences of actions that can be execute by workflow engines and robots in combination with sensors. Adaptivity can be included by means of late binding using rule-based workflows in conjunction with smart sensors.

Quality, ethical, legal, security and safety checks can be conducted for the different platforms. For example on the social platform, the customer checks if there are all eggs inside the package and he makes sure that they are not broken, whereas on the virtual platform a service level agreement can be used. Technology supports the quality check on the physical level, for instance by using cameras.

However, not only various platforms, but also different abstraction layers, presented in Figure 5, complicate the supermarket domain, looking trivial at the first glance. The abstract workflow is without platform binding and represents a sequence of robotic actions in order to fulfil the domain specific goal. For instance, in business process modeling notation. The technical workflow remains platform independent by calling external operations or generic subprocesses. It represents a sequence of executable capabilities of a robotic platform. The platform specific abstraction layer in form of concept models is characterized by a specific IoT adaptor that encapsulations platform specific APIs. Platform specific commands and operations are used on the robotic platform, the IoT adaptors and the workflow engine layer.

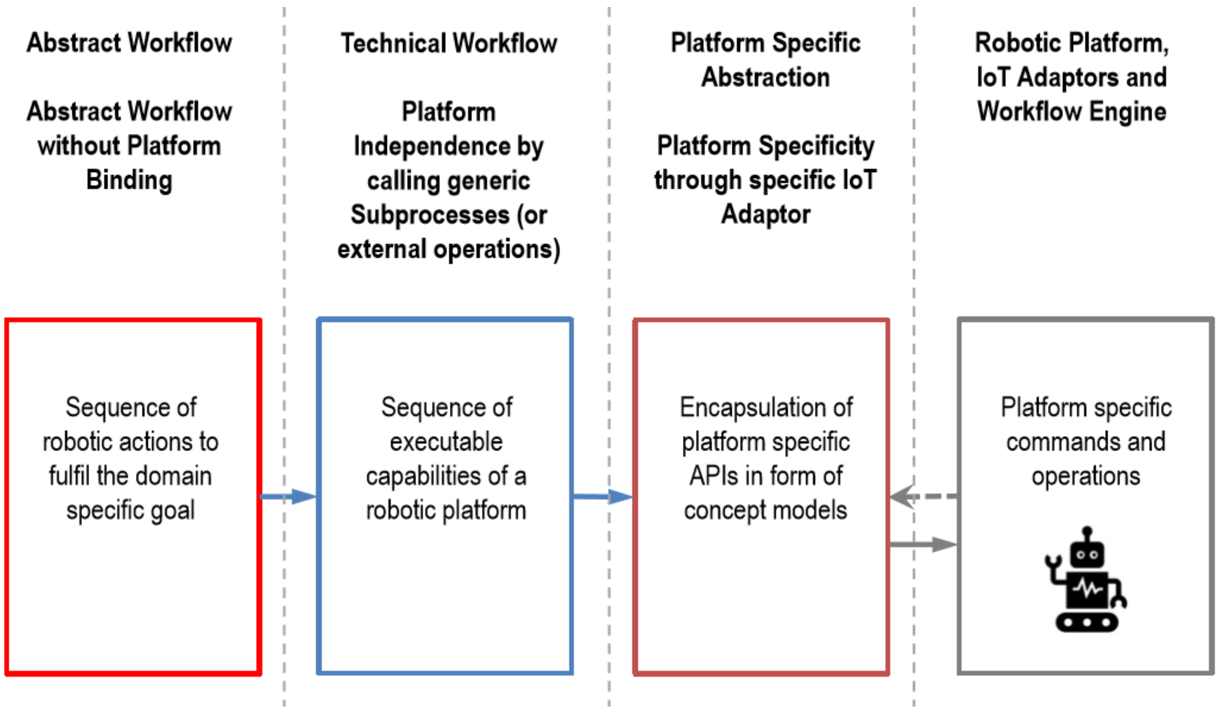

Figure 5: Abstraction Layers required in the Application Scenarios

In order to manage all of the mentioned aspect, workflows operating on robots are introduced. A Workflow executes and orchestrates a sequence of SW APIs. In the context of robots each SW API is an IoT Adaptor that triggers a pre-programmed actuation – a certain move. The resource allocation or binding of a workflow is typically understood to link a certain SW API to a particular action of the workflow a particular IoT Adaptor – hence a certain move – is bound to a workflow task. The supermarket scenario is used to demonstrate different ways of binding a SW API to a workflow action.

Following three bindings are used and presented in more detail in the second phase of this work package. In specific, those workflow bindings are discussed in “D3.2“.

- Fixed Binding – The workflow and all of the moves are predefined, when the user selects the workflow, the user gets exactly the sequence of moves.
- Pre-Binding: The workflow and all of the moves are decided just before starting with the first move. This allows to react in case a certain move – picking up of a particular object – is not possible and allows to consider alternatives – picking up another object instead.
- Late-Binding: The workflow and all of the moves are decided just while the workflow is executed. This allows to react on situations just before the actual move is performed.

3.2 Overview of Experimentation Cases

These cases are defined based on domain complexity and establish different requirements towards the assessment service, which is required for the ethical, legal and safety/security criteria.

1. **Robot Arm** – This case targets the operation of robotic arms that support for example manufacturing processes. The arm itself is statically positioned and operates according to a defined process. This process is either implemented directly on the system or interpreted via a model-based interaction. One key question is how the user can interact with the robot. Using various workflow models and binding options is required to make the interaction more flexible (e.g. fixed workflow, before execution or regularly during the execution). In specific human actors, like users or robotic

engineers, must be able to interact with the robot in a safe and secure way. Therefore, main considerations must be related to emergency stops, flexibility and environmental influences as well as dependencies.
The challenge observed in this case relates to the way, environmental conditions and settings are provided to the system, including position information, types and nature of work artefacts and operations possible. Dedicated escalation strategies are required in case the environmental setting is changed (re-implementation/setup), is interrupted (safe operation) or planned or urgent manual intervention takes place. These requirements are summarized as the execution semantic the robot arm is aware of and can operate upon. Flexibility is needed and defined as a goal for this scenario, that impacts the time and effort required to change the operation capabilities of the system.
Figure 6 shows an example of a robot arm that picks up small mechanical parts and assembles them in a rack, which can be further used.

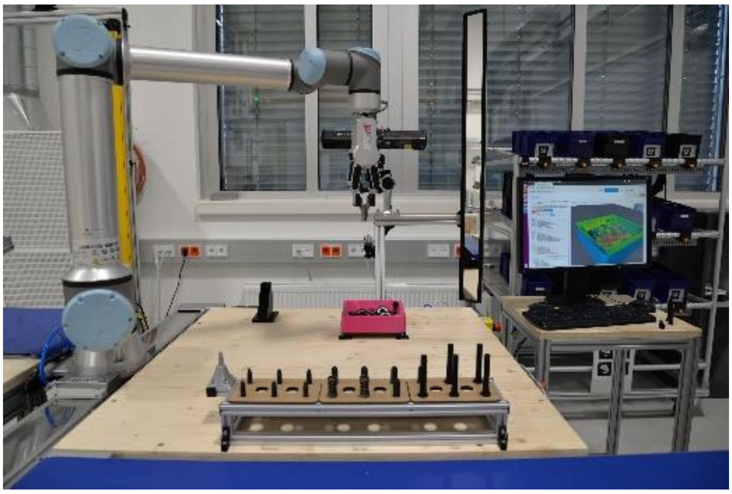

Figure 6: Assembly Robot Arm at JR

2. **Mobile Platforms** – Autonomous vehicles and self-driving mobile platforms are in focus of this case. The vehicles act in an autonomous way, within the boundaries of the environment and perform pre-defined tasks. Considering multiple vehicles within the environment, autonomy requirements result in the need to collaboratively resolve issues, apply negotiation strategies before performing tasks and reacting to preference settings according to interest functions. Essential is to identify the requirements of those mobile robots, their level of autonomy as well as any constraints and prerequisites influenced by the environment. Furthermore, the role of humans and possible points of interaction must be considered.
The challenge observed for this case relates to competition, coordination, and cooperation of intelligent actors/agents within the system. Goals on individual and environment levels need to be reflected and aligned with the capabilities of (potentially) different devices. Task allocation happens according to an interest function that is either controlled centrally or on the edge (responsibility considerations).
Figure 7 shows a sample of an autonomous vehicle at JR.

Figure 7: Autonomous vehicle at JR

3. **Cooperative Human/Robot Assembly Scenario –** For this case, the term robot is understood in a virtual meaning (as defined in the literature on robotic process automation). Software-based robots support the human processes on different degrees of involvement. Such an interaction setting between human and machine actors have been classified in as a) conventional, autark(self-sufficient)/co-existing, synchronized, cooperative or collaborative, ordered by the degree of interaction intensity. Novel work settings are the result of such interaction alternatives. The question here is how a robot and a human can cooperate in an assembly scenario with various involvement degrees and physically distributed environments.
   The challenge is to have a co-creation scenarios, whereas this co-creation is specifically investigating in the field of digital design thinking. For digital design thinking, software robots support the creativity process in a transparent way and enable the facilitator and participants to concentrate on the domain challenge, rather than on technical aspects of the method or tool applied. This ranges from co-creation support in distributed settings towards the involvement of software actors that perform certain assessment (image recognition, validation, matching, extension).
   A possible sample setting includes humans and physical devices cooperating with software accessing a camera can be seen Figure 8.

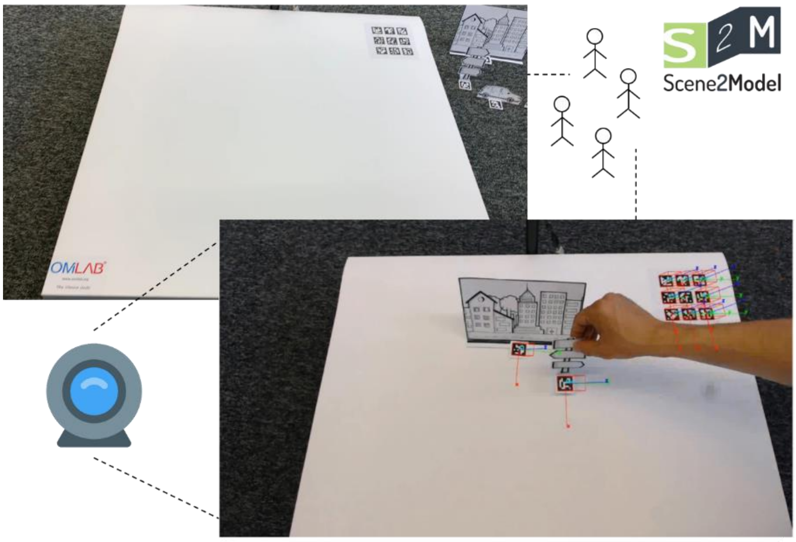

Figure 8: Digital Design Thinking: Co-Creation (Human/Software)

In the following subsections each case is discussed in detail, introducing the aspects relevant for compl@i to establish a common understanding on the challenges perceived by raising questions and requirements. It is required to find a sample domain in order to narrow down the potential legal issues to a management level. Therefore, the cases are related to a supermarket scenario. As everybody is familiar with supermarkets and their characteristics, this seems to be a suitable domain for the application scenarios.

3.3 Robot Arm

The robot arm scenario targets the objective of flexibility in the operation of robotic systems. Artificial intelligence functionality is deemed important to reach a state where the robotic device is aware of its environmental conditions and can react to changes within the setting it operates within. Based on the experimental setting in the OMiLAB, the following system architecture can be observed.

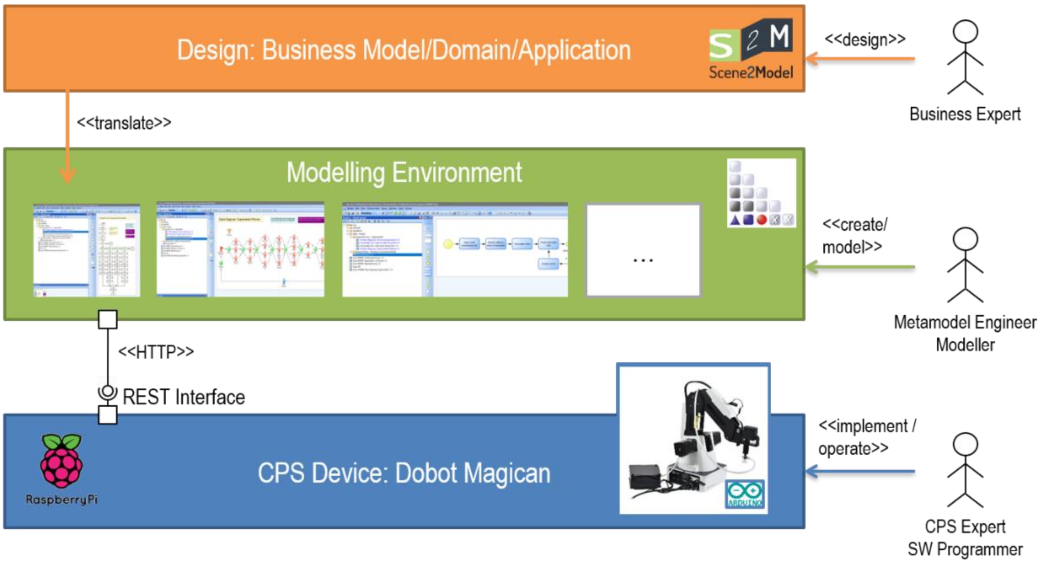

Figure 9: Robot Arm in OMiLAB – Model-Based CPS Design/Operation

Figure 9 shows the approach how the robotic arm experiment is defined. The graphical representation emphasises on two aspects:

1. the user type/skill involved for the specific layer and
2. the representative technologies applied to support the operation of the layer.

A concrete application scenario in the context of the supermarket domain and the architecture are described below.

3.3.1 Robot Arm Demonstration Case – Smart Supermarket

The design of the application case is performed on the top level. The layer is defined by the "business expert". In this context, business expert is defined in a broad sense – any stakeholder that is concerned with the use and application requirements of a novel system (unrelated to technical realisation) can design the basic application case.

The smart supermarket case has been selected as an indicative scenario for the application of the robot arm system. Within a smart supermarket, it is assumed that the actual collection process is not handled by the person shopping, but a robot performs the collect and delivery tasks based on the preference and/or shopping history of a specific customer. As a storyline, the logic for this scenario is briefly introduced below.

1. Ordering: the customer performs his ordering task and defines his/her high-level need (e.g. "I need snacks", "I need fruits to increase my health."). This order is either broken down into a pre-defined package of goods that are

associated with the preference set or characterised by flexible interaction with the user during the collection step. So called workflow bindings can offer various interaction alternatives.

2. Collection: a robotic arm within the supermarket collects the goods within the preference set and prepares them for delivery. One challenge is, if a human shopper would have chosen the exact same product as the robot arm. Furthermore, when considering safety and security, it must be clarified if humans and robots use the same supermarket in collaboration or if there are separated environments.
3. Collect and/or Delivery: the package prepared is either collected by the customer or directly delivered to the customer's home.

The robot arm demonstration mainly focuses on step one and two of the above-mentioned scenario.

3.3.2 Modelling Environment

The modelling environment translates the business requirements into a formal representation. This representation is required for three main aspects:

- **Knowledge Representation** – The domain knowledge described in abstract terms, for example with designs, sketches or prose, is refines by using a formal representation that externalises the logic described, its dependencies and correlations.
- **Analysis Support on Model Level** – Model-value functionality supports the designer to assess the feasibility, quality and quantitative facets of the business model (e.g.: search for deadlocks, simulation, …).
- **Enactment/Operation of Models** – Following a model-based approach, the models design has inherent operation capabilities. This can take different flavours ranging from interpretation of the conceptual model by an execution engine (e.g.: workflow engine) to transformation into source code for compilation or service invocation techniques (discover, bind, execute).

For the demonstration the following concrete instantiations in the modelling environment are available

a. ***Modelling using a Standard Notation*** – The BPMN 2.0 notation is used to describe the "collect and package" case described above. The semantics of BPMN allow the definition of the logical and message flow and provide means to export the logic into an executable skeleton (BPMN DI), which is a formal description that can be read by machines as well as humans who are experienced in this field.
b. ***Modelling using Flowcharts*** – Flowcharts are a best practice from software engineering and used to define task sequences within a system. Logical flow elements are provided and execution semantic is embedded within the metamodel. This means that a verified model defined can be interpreted as a graph and executed directly. The IoT client is connected in this case with the CPS device capabilities, defined as abstract, re-usable patterns.
c. ***Modelling using Petri Nets*** – Petri Nets are used as a formal language to verify the syntax and simulate the behaviour of a system. They can be used to represent the scenario, directly coupled with the IoT client implemented in a scripting language. The main difference compared to flowcharts is the animation features and the ability of interruption during the execution that Petri Nets support.
d. ***Domain-Specific Modelling Language*** – To get nearer to the domain expert, modelling languages that are closer can be considered (e.g. burger builder or supermarket modelling language). Those modelling languages are very often more abstract than the ones mentioned above. As domain-specific vocabulary is established, the business expert can be involved directly.

Observations: The aspect of knowledge externalisation and applicability for criteria of assessment are considered to be relevant. The domain establishes the boundaries of the case developed, exposed on the model-based assessment service and criteria catalogue (from a metamodel level – language elements and their meaning/relations and model level for the domain context). A challenge will be to find out if there is a difference in the criteria catalogue for the modelling environment and the execution environment or if both environments can be handled by the same catalogue only using adapted assessments.

3.3.3 The Dobot Magician Device

As an experimentation device, the Dobot Magician (https://www.dobot.cc/dobot-magician/product-overview.html) is used, shown in Figure 10. As an educational robot, it enables the user to have immediate results without the need of experience with robotic systems and high investment costs. The device is usable out-of-the-box and provides various interface modules (e.g. sucking cup, grabber, pen, laser) to be mounted on the arm for different domain-specific cases.

To control the arm, the device expert interacts with the device using the software components directly (UI-based approach) or a software engineer builds upon the available standard software development toolkits (SDK) delivered. For the Dobot Magician, these SDKs are available for the programming language Java and Phython. Internally, these SDKs translate commands to the low-level C derivate for Arduino. The embedded Arduino board and its firmware control the operations.

Extension in the OMiLAB: For the OMiLAB realisation of the experiment, software extensions have been realized to establish connectivity. As the device itself is not directly exposing an interface over a network protocol, it is connected to a Raspberry Pi computer that operates the implemented IoT adaptor (as a wrapper of the SDK functionalities). The selected communication protocol for the experiment is REST by using the HTTP protocol.

Figure 10: Dobot Magician
(Retrieved from https://www.dobot.cc/dobot-magician/product-overview.html)

Observations: Considering this stack of hardware and software components, the following considerations impact the compliance assessment within the project:

- Hardware as well as software components are used that are provided by the official vendor, not under the control of the CPS expert or software programmers.
- Open-source components are used to realize the IoT adaptor that exposes the REST interface. These components are dynamically loaded during the compilation of the software. The same applies for the Raspberry Pi device, that is a community effort to realize affordable minicomputers.

3.3.4 AI Challenge – Flexibility

The challenge introduced above relates to flexibility. Within the experimental setup, this challenge can be observed in the following peculiarity:

- *Position and Type Information* – In the current setup, the robot arm is not aware of position information and types/semantic of items to collect. This means that the position and the type information is handed over from the model level. This means that the arm performs the action requests received as service calls in a stateless and unaware manner. In case the experiment product area is either empty (nothing in stock of the supermarket) or wrong types are located at wrong positions, it will continue to execute.
- *Change of Experiment/Domain* – In case the experiment is changed/adapted, the models need to be re-aligned to the specific domain, which results in efforts and costs imposed on the modelling expert (and potentially also the CPS expert).

Based on these two observations, AI capabilities are proposed to provide contextualisation on model level. This has the following implication:

- *Models become Runtime/Operation Aware* – Model awareness should be created by observing the operation canvas of the robot arm, the context and environment setting is monitored. A convolutional neural network is trained with the domain of the application case and provides input to the model level. As such, the model or model interpretation logic can operate on this information and react to changes in a flexible manner.

- *Domain Awareness* – As already introduced, domain-specific training of the neural network is required. For this purpose, pre-trained networks can be re-purposed and extend with domain elements that are derived from the semantics of the domain-specific metamodel or model content.

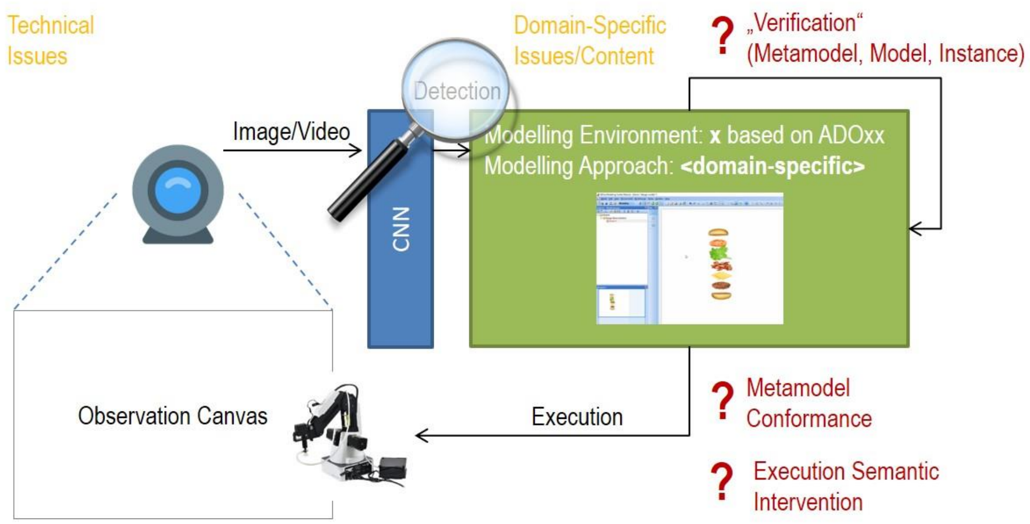

Figure 11: Context-Awareness for CPS-Flexibility

Figure 11 shows graphically how AI-based image recognition supports the flexibility goal of the case. Image and video streams from a camera observing the operation canvas are continuously assessed by a convolutional neural network (CNN). The results of this analysis are provided as contextual information to the modelling environment and impact the metamodel and model content (at runtime). During the interpretation of a model for execution, this information is used to validate whether specific operations are possible; in case of an observed deficiency, mitigation and escalation strategies are implemented on model level or as an aspect in the execution client calls.

3.3.5 Prototyping

In the following, two prototypes dealing with detection/image recognition and execution semantics are presented to get a better feeling for possible AI technologies used in the application scenarios.

Detection/Image Recognition

To verify these statements, a prototypical demonstrator has been realised. The prototype uses the pre-trained convolutional neural network yolo9000[3] [3]. The network builds on the darknet implementation and is pre-training for 9000 classes of objects. The training source and data is transparently available for the user. Nevertheless, detection challenges can be observed that relate to the nature of a neural network as a heuristic of recognition and is currently applied as a blackbox (in a sense of training strategy and processes). Figure 12 shows a sample of how object detection, modelling and execution environment are related to each other.

[3] J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," *Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017*, vol. 2017-January, pp. 6517–6525, Dec. 2016.

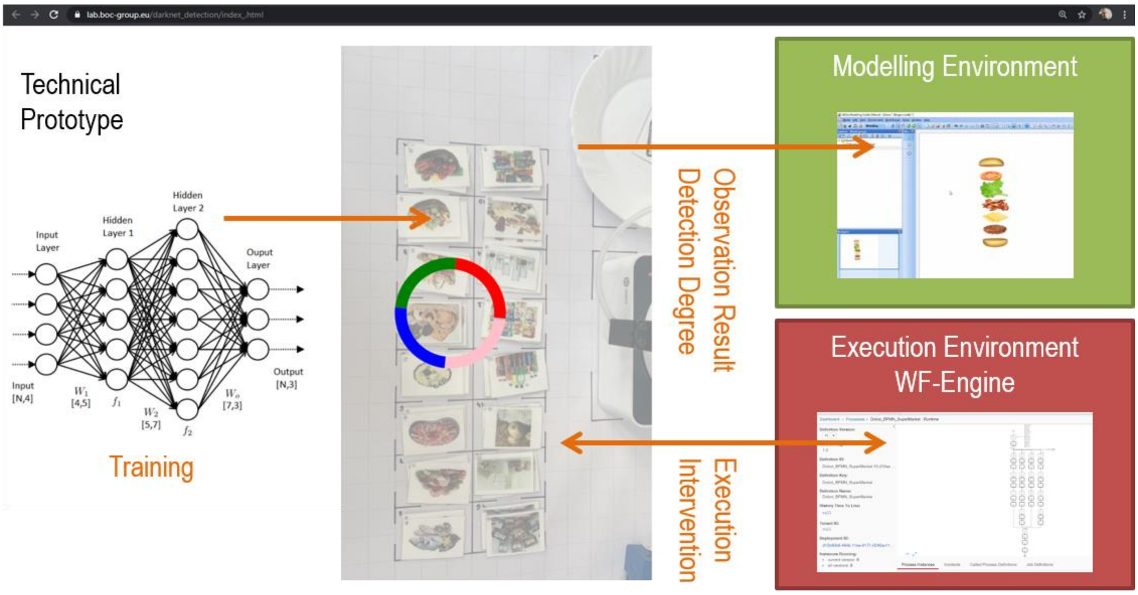

Figure 12: Object Detection for Modelling/Execution System

Transparency is required to assess and evaluate such a socio-technical system component. Furthermore, reliability and responsibility concern must be addressed.

Execution Semantics

A second prototype in this case targets execution semantics, which considers in more detail how programs are executed. Expanding on a “in-model” translation of execution graphs, the implementation uses standard workflow technologies to operate the conceptual model. Concretely, for this case the BPMN DI export capabilities has been used to transform a business graph into a workflow graph and trigger the execution. The relationships between modelling environment, execution environment and CPS device is shown in Figure 13.

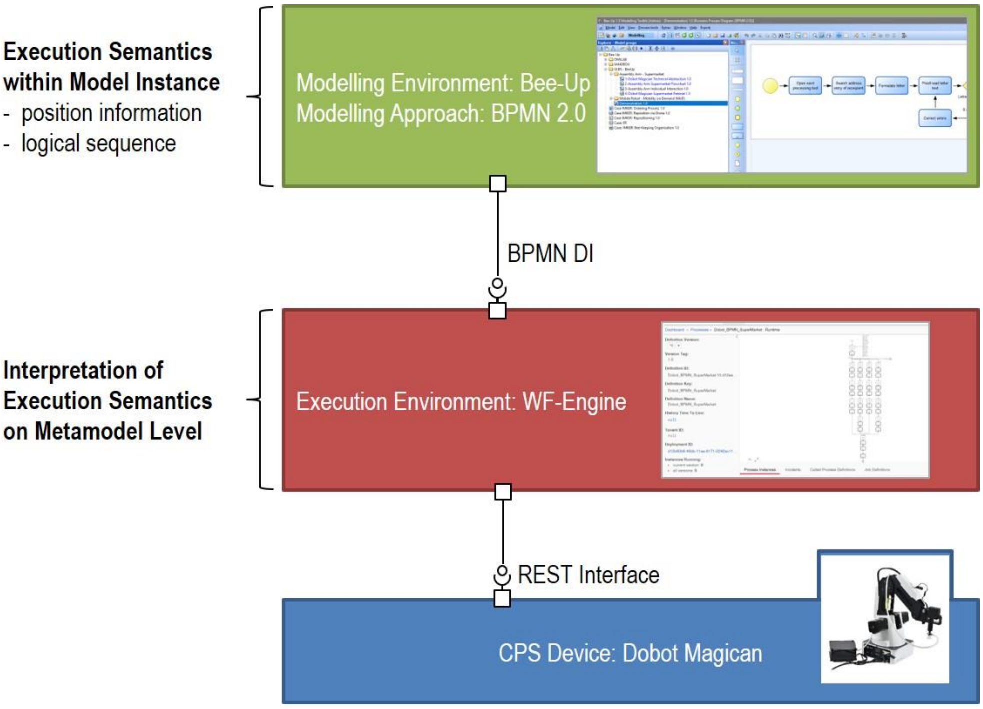

Figure 13: Workflow-based Execution

This technique uses graph re-writing techniques for the transformation, and the applicability for the architecture has been assessed:

- Business Graph Representation – This format is used to represent the graph and required extension capabilities. This means, that results are embedded within the graph during design. This information can be on a descriptive, meta-information level (for revision control) or also included as additional guidelines for the implementation responsible on workflow level.
- Workflow Graph Representation – The transformation towards execution is performed on the workflow level; resources in the form of service calls are allocated and interaction requirements defined. From a development process point of view, the guidelines imposed are confirmed and related to the assets on this level.
- Execution Graph – The technical implementation is executed, observing the environment conditions according to specific implementations (e.g. person entering the experiment area results in a stop of execution).

A screenshot of the implementation is shown below in Figure 14, currently operating the robot by using BPMN 2.0. Extensions are required to support rule representations (e.g. DMN 1.1 or similar) and to assess the semantics returned from visual recognition, for instance by using dedicated escalation strategies during the application of AI techniques.

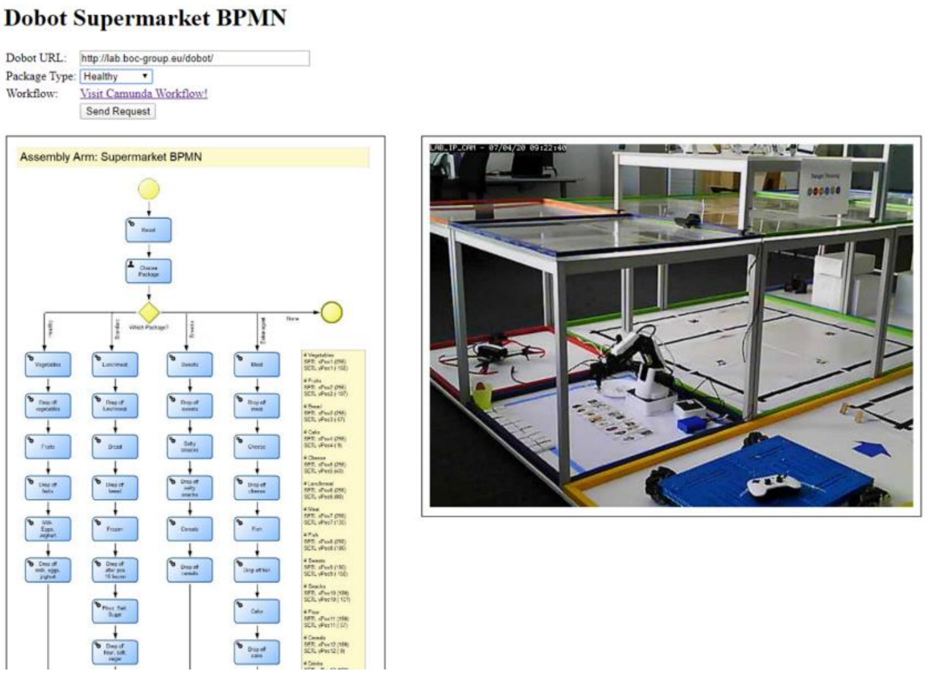

Figure 14: BPMN DI based Workflow Execution

3.3.6 Overview of Case Challenges

The main challenges of this case are summarized below, targeting the flexibility objective.

ID	Challenge	Description
RA1	AI Technique Selection	The identification of an AI technique to resolve flexibility concerns during design and execution time, on a conceptual level in relation with compliance rules seems to be not trivial. How does the selection process act in the design/definition of trained models? How are pre-trained/open-source results impacting the solution?
RA2	Language Extension	Extensions to modelling languages are necessary to capture goal settings and decision rules in case of escalations, adaptation and uncertainty. For instance, DMN can serve as a potential

		candidate to enhance flexibility and derive goal patterns within a domain specific representation.
RA3	Impact of Compliance on Architectural Components	It is required to define the impact of compliance assessments (on type level) on the system and its components.

Table 1: Robot Arm Case Challenges

3.4 Mobile Platforms

Mobile platforms represent a specific challenge as vehicles act autonomously and react to sensor streams in a controlled or uncontrolled manner. The key characteristic observed for this demonstration case relates to the aspect of cooperation, learning and autonomy within a system of multiple autonomous entities. Currently, the experiments in the OMiLAB with respect to mobile platforms follow two streams of work:

- Environmental Recognition – This includes assessing the environment as perceived by the mobile platform/entity and collecting information on the surrounding environment.
- Controlled Actuation – This is assuming a traditional approach, where a controlling entity is responsible for defining the interaction streams, triggering the actions and listening to reactions of the entity.

These two streams limit the way of collaboration between the participating entities that can accomplish operations.

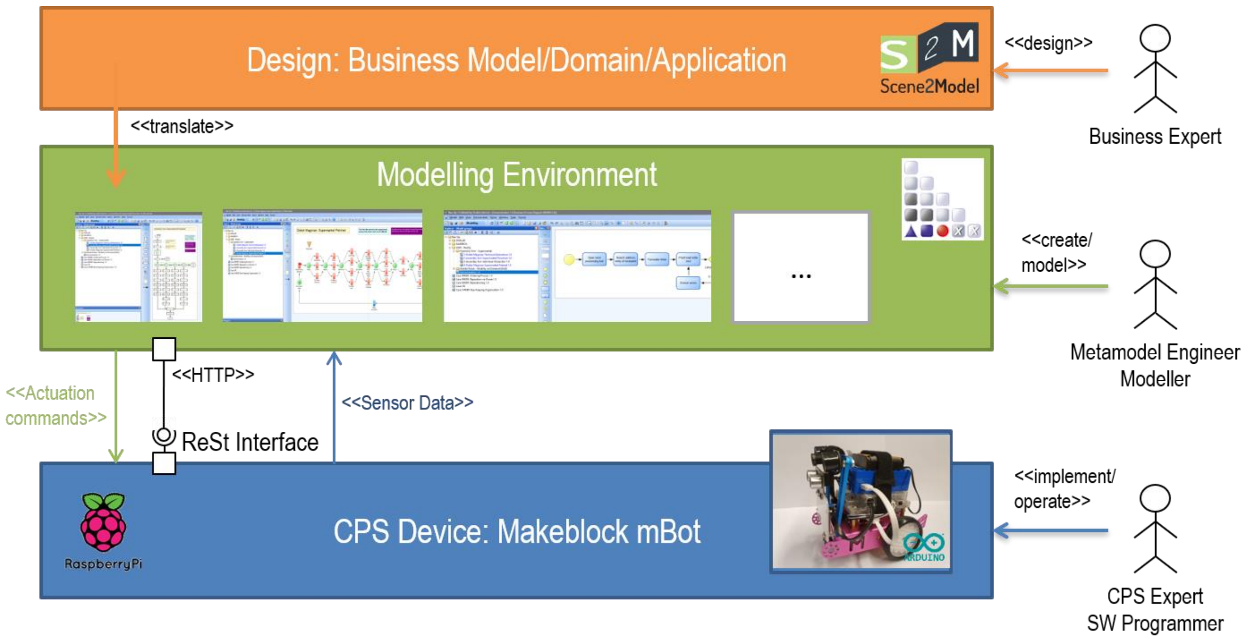

Figure 15: Mobile Platforms in the OMiLAB – Model-Based CPS Design/Operation

Figure 15 shows the interaction streams, similar as for the robot arm, but emphasising the sensor/actuation interaction. As an experimentation device, (multiple) Makeblock mBots are available in the lab (https://www.makeblock.com/mbot). These mBots are equipped with an Arduino board that is custom made for the physical hardware and provides access to all sensors, motors and devices attached. Moreover, to establish connectivity and provide computing power, a Raspberry Pi is "carried" by the mBot and exposes the functionality of the mBot via REST interfaces of the web. Sensor streams are established in parallel to the REST interface as web sockets that continuously stream the data of the sensor back to the

Raspberry Pi and forward it to a controlling entity. For the current implementation, this controlling entity is within the modelling environment. The modelling environments are extended to support the IoT adaptor of the device and can trigger actuation (movement, sensing, displaying) actions.

3.4.1 Demonstration Case – Competing Cars

As a motivational case, an innovative application case is introduced that targets competing cars. Assuming a car sharing system (as well known and established in different cities around the globe), the case discusses whether and how it is possible to extend the knowledge of the participating, individual cars so that these cars can act by themselves, capture the knowledge of the environment surrounding them and potential clients. In contrast to the current interaction scheme where the client is in charge and performs the selection, the car becomes a learning entity that proposes itself as a potential ride based on criteria such as the route, preference of the client (from the past and current), the current position and traffic conditions. From an interaction point of view, the user triggers the process and then the system interacts with him/her in an intelligent manner. The potential system actions are discussed below

1. Receive the trigger (current position of user and requested destination) from the user
2. Assess environmental conditions
    a. Current position relative to user and destination, including local knowledge on allowed routes as well as temporal limitations
    b. Traffic and weather conditions
    c. User preference and behaviour (from the past) and current condition the use is in (e.g. transporting goods, intoxicated)
3. Calculate and simulate the potential offerings that could be made to the user, this happens in parallel by all participating entities, the internal goal set could be dynamically adjusted and reflects and results in a decision that the car makes whether to offer mobility services or not
4. Reflect the underlying pricing model based on the conditions and offering
5. Interact with the customer (as a negotiation) and propose a price and intelligent offering (e.g. pick-up and drop-off location)

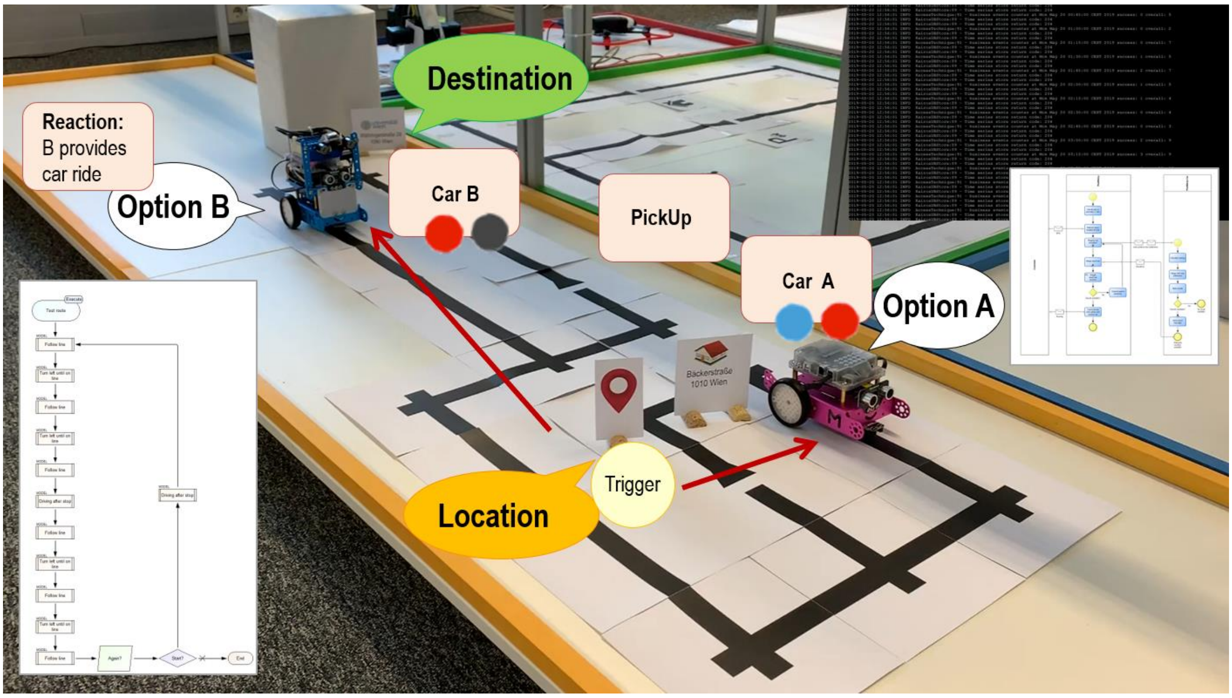

Figure 16: Competing Cars: Experimental Setup

The interaction flow has been realized within a laboratory setting, showcasing the application case. In Figure 16, this setup is shown: 2 mBots perform the local assessment (on a limited set of information) and perform a simulation on the feasibility of an offering to be made to client. In the laboratory case, the user confirms the offering received via text message and the selected car executes the offering following a model-based implementation.

Observations: As a domain for the application scenarios a supermarket was chosen. For this reason, the above-mentioned scenario idea about car sharing, must be transferred to a supermarket domain using smart shopping carts. Such autonomous shopping carts do have to consider similar challenges related to interaction and collaboration with humans as well as other carts.

3.4.2 Modelling Environment

The modelling environment in its basic version above, operates on a process-based simulation, running on the specific entities. The goal model of each actor (as an interest function) is established centrally, by a controller, and deployed on the mBots. The Bee-Up Modelling Tool is used as a starting point.

In a future extended scenario that involves AI services and transforms the logic into an agent-based setting, a combination of modelling methods and metamodels are required to represent the domain semantics. The candidates required are shown below, aiming at an intelligent, multi-agent system of mobile devices, as discussed in[4] and shown graphically in Figure 17.

Figure 17: Classification of Agents [4]

The classification presented influences the selection of modelling methods and metamodels by considering following aspects:

- Goal models are used to define the interest of each entity in the system
- Decision and reasoning capabilities are needed
- Optimization logic applied on graphs, networks and goal settings (also in concurrent environments)

Observations: A core concern with respect to compliance relates to the aspects of data (as sensor streams), their quality and provenance, as the autonomous entity relies in all of its actions on the incoming and perceived stream of information.

[4] J. Rocha, I. Boavida-Portugal, and E. Gomes, “Introductory Chapter: Multi-Agent Systems,” in *Multi-agent Systems*, 2017.

In case of an autonomous setting, these streams are uncontrolled and established individually, whereas a controlled approach results in a level of certainty and assigned responsibility.

3.4.3 Device – Makeblock mBot

The mBot, produced by MakeBlock is an educational mobile platform that aims to provide quick and efficient realisation of prototypical solutions based on standard protocols and technologies. The basic hardware required to operate the mBot as a mobile platform is shown below in Figure 18, from a software point of view, the same approach as discussed for the robotic arm is followed:

- Physical Interaction – The firmware is developed in a custom manner and provides means for serial communication. These serial commands are received and sent to/from the attached Raspberry Pi to/from the mBot mCore board.
- Java-based Interfaces – The interfaces (as a REST webservice, SOAP endpoint and web-socket) are exposed from the Raspberry Pi that runs a Java server and provides the interfaces to the network, where the device is attached to.

This setup is considered flexible as requirements on physical level are realised close to the hardware (on mCore, Arduino level), whereas complex, processing intense logic runs on the Linux-based Raspberry Pi micro-computer. The connection between both fields is stable via a cable USB-serial connection.

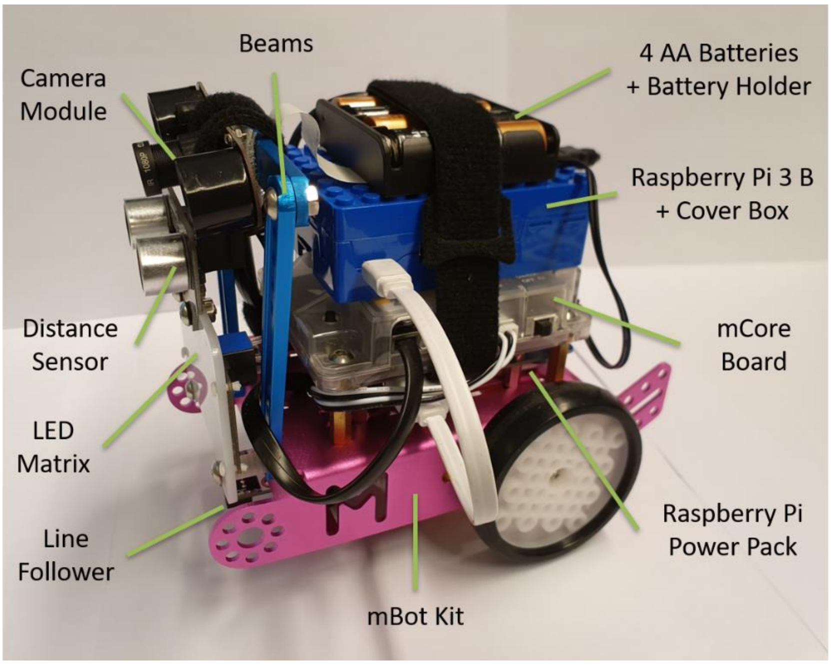

Figure 18: mBot Hardware Elements

3.4.4 AI Challenge – Optimization of Task Allocation

As introduced above, the optimization of task allocation in a connected system is the challenge followed for this demonstration experiment. Multiple mBots, potentially with different sensor adaptors (to emulate different car types and capabilities) need to collaborate and negotiate with an individual user for an offering. In this way the task is allocated within a network of entities that act in an intelligent manner. Considering a supermarket scenario with assisting shopping carts, also the interaction between shopping cart and human shopper must be considered.

Observations: The distinction between controlled and autonomous is regarded as a 1) technical challenge as functionality, that is normally centrally executed is run on the edge and negotiation logic via goal sets needs to be implemented and b) a compliance challenge as the uncertainty resulting from autonomy and individual logic on each device might result in a complexity overhead.

3.4.5 Prototyping – Agent-based Operation

For the prototypical setup, two scenarios are proposed to demonstrate the correlation between autonomy/control and decision making in an intelligence system.

- Orchestrator-based Approach – As the traditional architecture of such a system, an orchestrator is available and performs the assignment of tasks to the participating and registered entities. In such a setting, the orchestrator holds the knowledge of the system and is deemed responsible for decision making. Functionalities to evaluate the interest function and trigger the retrieval of data streams are centrally controlled.
- Distributed Approach – In this architecture, the functionalities belong to the device and collaboration is a result of the network the devices act within. Local processing is considered to be beneficial as individual aspects are processed and evaluated locally. Nevertheless, uncertainty with respect to the network nodes and their interaction is assumed.

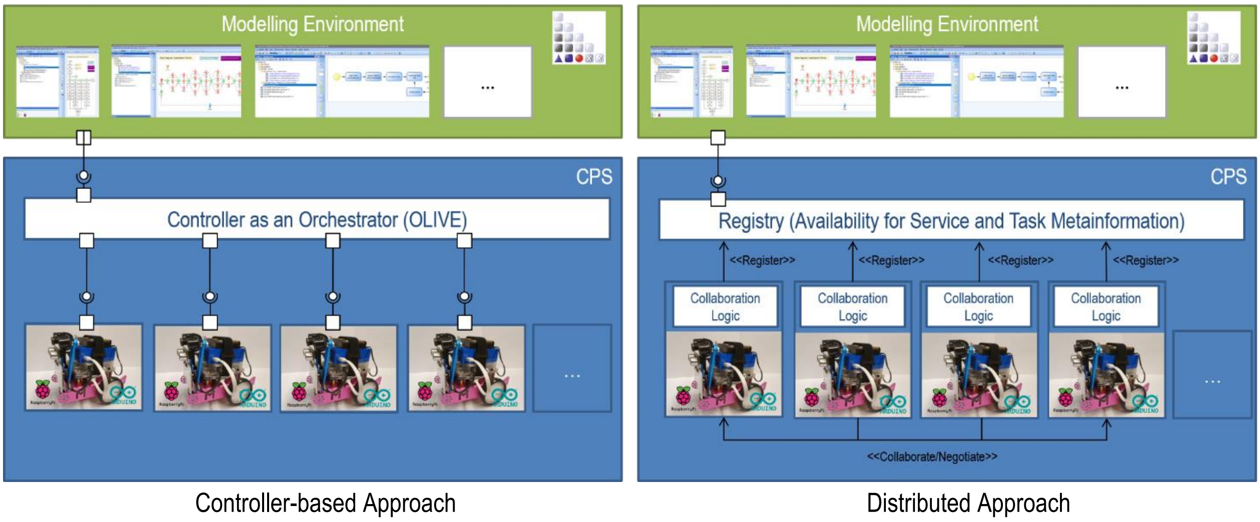

Figure 19: Agent-Oriented Architectures (Simplified)

Figure 19 shows the difference in the architecture graphically. In the orchestrator-based approach, the logic of task allocation is assigned to the controller entity, whereas in a network-based setting, this controller does not exist and is replaced by a registry to distribute potential tasks. The allocation is performed as a result of negotiation and collaboration logic on each device.

3.4.6 Overview of Case Challenges

The main challenges of this case are summarized below, targeting the multiple agent objective.

ID	Challenge	Description
MP1	Intelligence Agent Architecture	Identification of an adequate architecture that reflects autonomous considerations and correlates those with uncertainty within such a system. The execution of an interest function on the edge needs to be supported, however the controller of a decision must be identified clearly and traceable.

MP2	Language Extension	Language requirements for each device (and its specificities) as well as global goal function might be reasonable. The modelling environment is based on the results discussed in literature on multi-agent systems ([5]).
MP3	Compliance in Uncertain Situations	The data quality and provenance aspects are identified as a challenge of this use case, besides the technical as well as the architectural aspects must be considered.

Table 2: Mobile Platform Case Challenges

3.5 Cooperative Human/Robot Assembly

The cooperative human/robot assembly demonstration experiment targets the interaction of human and robotic actors within a common environment. The case selected is related to digital design thinking – a novel approach to support design thinking with technology for different use cases, ranging from recognition, model transformation, validation or distribution to intelligence functionality within the environment.

3.5.1 Demonstration Case – Digital Design Thinking

Digital design thinking has been developed as an approach and tool environment to support distributed teams in their design challenges. As a – typical – haptic approach, this requires the presence of participants. In most cases, the processes and outcomes are not accessible for members/stakeholders that are not present during the workshop. Methods (e.g. storyboarding, scenes, value-proposition mapping) are well-received, but lack the formalisation to enable additional functionality upon the results achieved. Digitalisation of the process and its artefacts has been considered for the novel techniques, discussed in detail in[6],[7]. The outcome of this endeavour is the Scene2Model tool, that implements design thinking methods and provides tool support to participants. The results (tool, paper figures, technology) are openly accessible at https://austria.omilab.org/psm/content/scene2model/.

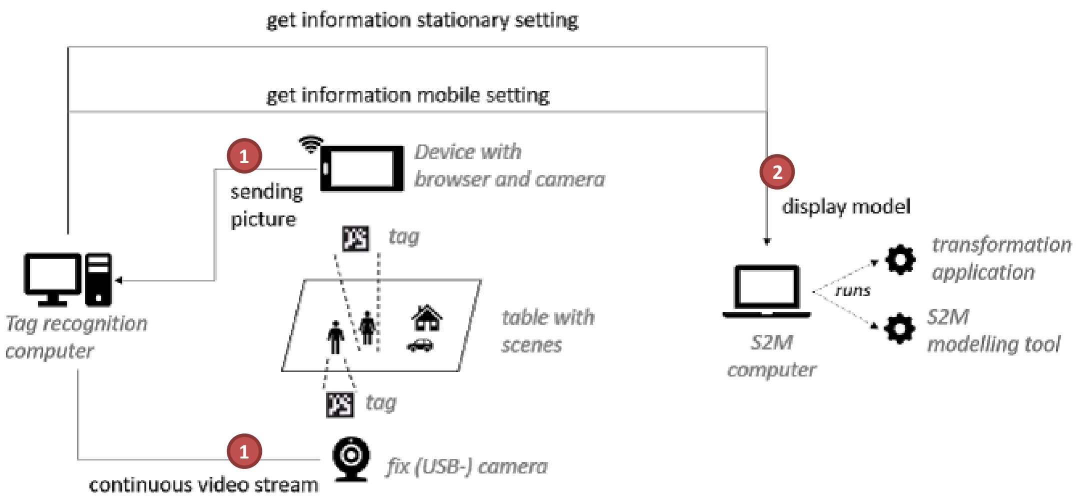

Figure 20: Scene2Model Architecture

[5] J. Barbosa, P. Leitao, A. Ferreira, J. Queiroz, C. A. S. Geraldes, and J. P. Coelho, "Implementation of a Multi-Agent System to Support ZDM Strategies in Multi-Stage Environments," in *Proceedings - IEEE 16th International Conference on Industrial Informatics, INDIN 2018*, 2018.
[6] E.-T. Miron, C. Muck, and D. Karagiannis, "Transforming Haptic Storyboards into Diagrammatic Models: The Scene2Model Tool," in *Proceedings of the 52nd Hawaii International Conference on System Sciences*, 2019.
[7] M. Walch, *A conceptual modelling approach for design and use in cyber-physical environments: the s*IoT modelling method*. Vienna, Austria: University of Vienna, PhD Thesis, 2019.

Figure 20 shows the architecture that has been deployed within the OMiLAB. Images of the design thinking workshop results are captured using a steady camera mount or mobile setup. The objects used within the storyboard are annotated (physically) with QR tags and elevate the images sent to the tag recognition system. The tag recognition system performs two tasks: a) identification of the objects by the QR ID and validation of the input against a common knowledge base (formalized as an ontology) and b) capturing the individual object position on the canvas. The results are captured by the Scene2Model tool that parses the stream of information and builds a conceptual/iconic representation of the outcome. The detailed components are introduced below as the baseline for the demonstration case.

3.5.2 Modelling Environment – Scene2Model

The modelling environment consists of the Scene2Model tool and necessary adaptors to the tag recognition system. Both systems share a common knowledge base, that is responsible to identify types (in Scene2Model "modelling classes"), the meaning of the constructs recognized (as properties) and potentially constraints within the dependencies of actors. The metamodel of Scene2Model is unaware of the actual object but only implements generic types that support a classification of elements and consequently, a harmonized notation. The concrete example implemented and provided as a default in Scene2Model is the catalogue of objects provided by the SAP's Scenes approach[8]. Figure 21 shows an example scene created with the Scene2Model tool.

The SAP Scenes basic catalogue (accessible online at https://experience.sap.com/designservices/resources/scenes) consists of 10 categories of elements (vehicle, characters, device, speech bubble, sign, arrow, office furniture, building, background, characters as icons) resulting in 126 graphical elements that can be used in workshops. Extension packs provide domain-specific elements for specific industries or applications.

In addition to recognition and model setup (that reduces the manual documentation effort), Scene2Model is equipped with concepts to annotate scenes of a storyboard with formal concepts in relation to enterprise modelling best practices. This includes assets such as processes, infrastructure elements and so on that are considered as additional knowledge for reasoning and deduction on one hand (relation between design thinking constructs and enterprise modelling constructs) and as input for transformation on the other hand.

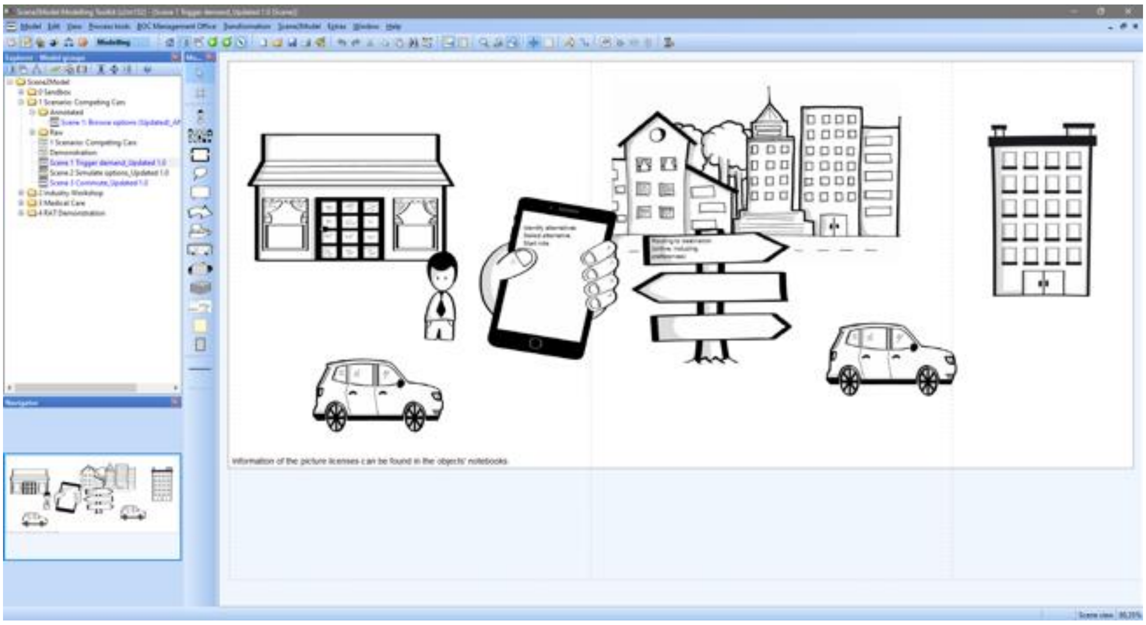

Figure 21: Scene2Model Example Scene

[8] SAP User Experience Design Services, "Scenes," 2019. [Online]. Available: https://experience.sap.com/designservices/approach/scenes. [Accessed: 07-May-2019]

3.5.3 Device – Camera Stream

The camera system used within the laboratory uses a standard high-definition USB web-camera. The model (Logitech C920) has been calibrated to recognize QR codes produced with the Aruco Library (https://docs.opencv.org/trunk/d5/dae/tutorial_aruco_detection.html, more specifically the results and implementation discussed in[9] and [10] is available online at https://www.uco.es/investiga/grupos/ava/node/26). The QR code detection library has been selected to have a sufficient number of tags available for arbitrary elements and optimize the recognition process also in varying lighting conditions. Within the lab the camera is mounted at a fixed position, supporting the workshop environment. However, there is also a mobile setting planned.

3.5.4 AI Challenge – Safe Co-Creation

The challenge is a co-creation within digital design thinking. This means that the design process and task – performed using digital design thinking techniques – is supported by various software agents and/or virtual robots that interact with the human participants during the workshop. Examples for these agents are intelligent functionalities that trigger tool actions, perform validation and verification, retrieve and crawl contextual data for specific cases, dynamically link the design results to pre-existing enterprise assets and so on. The robotic device as a software-based service becomes an integral part of the process, co-creates and extends the scenes and storyboards specified and supports domain experts. An important consideration is related to safety, which is defined for this case in the context of functionality that is performed by an autonomous system needs to be traceable, explainable and revision safe.

3.5.5 Prototyping – Distributed Haptic Modelling

An initial prototype with this objective in mind has been discussed and presented in[11]. The prototype targets the abstraction of actors in the design thinking process as human or software-based actors, assigning specific functionalities to each participating node and operations. These operations are either local (haptic modelling) or remote (performed by a system remotely and interpreted at connected nodes locally).

The initial challenge that the paper linked above addressed was related to distribution. How can remote human teams participate in design thinking workshops beyond video calls and chat systems? The idea to distribute not only the results but establish software-based communication channels, where each team produces and/or receives projections of other results has been introduced. A sample of a remote deign thinking service architecture can be seen in Figure 22.

---

[9] F. J. Romero-Ramirez, R. Muñoz-Salinas, and R. Medina-Carnicer, "Speeded up detection of squared fiducial markers," *Image Vis. Comput.*, vol. 76, pp. 38–47, Aug. 2018

[10] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and R. Medina-Carnicer, "Generation of fiducial marker dictionaries using Mixed Integer Linear Programming," *Pattern Recognit.*, vol. 51, pp. 481–491, Mar. 2016.

[11] W. Utz, "Support of Collaborative Design Thinking using ADOxx," in *Proceedings - 2019 International Conference on Innovation and Management*, 2019.

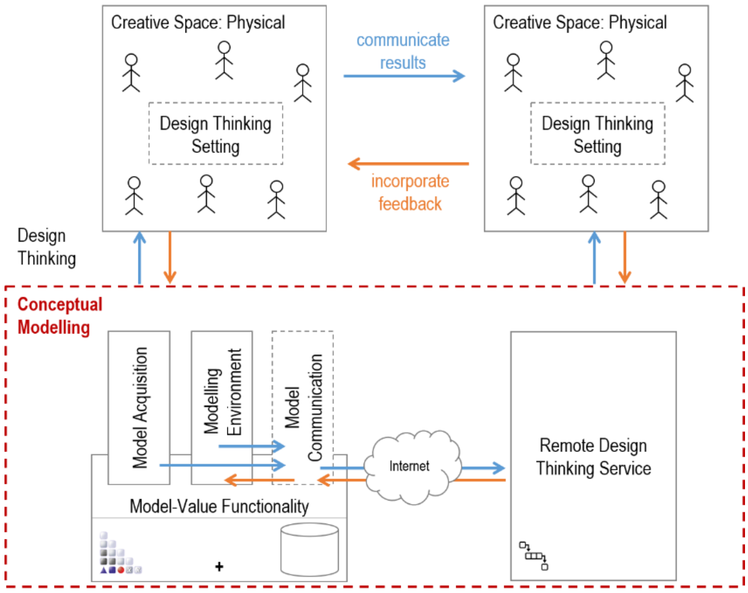

Figure 22: Remote Design Thinking Services

Observations: An abstraction of this distribution challenge relates to the formulation of services that are contacted to provide additional (model) value to the system. This includes functionalities to a) collect information from systems/data streams/enterprise assets, b) provide intelligence capabilities to retrieve community feedback, criteria assessments and interaction channels or c) validate and support the design process through semantic/knowledge graph deduction. Expanding on these aspects, an interesting challenge relates to the dynamics and evolution of the metamodel and corresponding models. As these models represent domain knowledge.

3.5.6 Overview of Case Challenges

The main challenges of this case are summarized below, targeting the co-creation and collaboration objectives.

ID	Challenge	Description
CA1	Distributed System Architecture and related Trustworthiness of Services	Safe co-creation builds on services that are adequate and useful from a functional viewpoint. Characteristics to identify the trustworthiness of services as minimal requirements need to be identified as output from assessment criteria and catalogues.
CA2	Metamodel Extension	Formulate the language required to support dynamic, domain-specific extension of existing Scenes implementations.
CA3	Classification of Collaboration Streams as Input towards Explain Ability of AI Functionality	Define and classify the collaboration streams between human actors and machines/services/functionalities by considering ethical, legal, security and safety aspects.

Table 3: Cooperative Assembly Case Challenges

4. Architecture Blueprint (Proposal)

Based on the challenges identified per demonstration case (see chapter and subchapters above) resulting from the decomposition of the application scenario, the architecture has been identified. The blueprint of the architecture is presented and discussed below. As an initial outcome, the application case has informed the design and specification of the architecture, graphically shown as a UML deployment diagram in Figure 23. The architecture is separated in 3 environments in accordance with the model-based approach presented earlier.

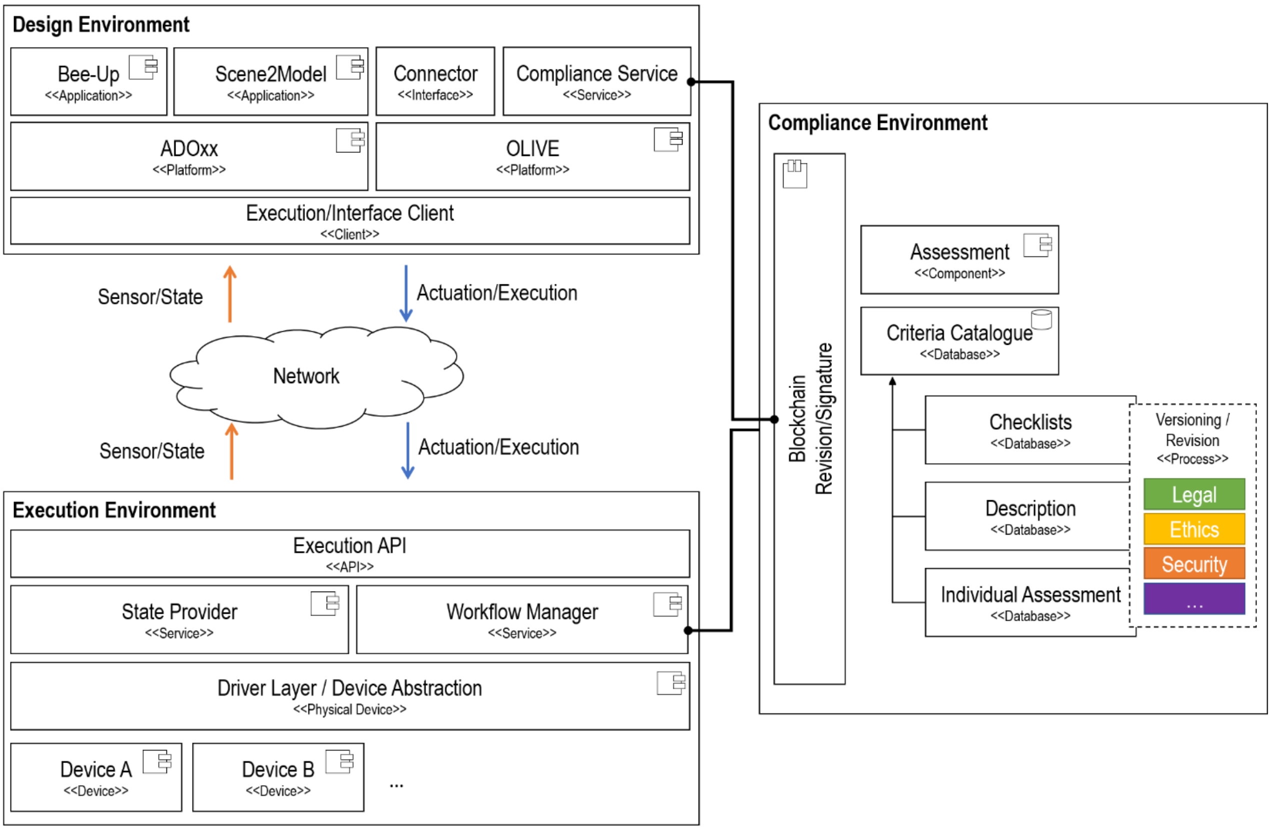

Figure 23: High-Level Architecture

1. **Design Environment** – The design environment encompasses all design functionalities. In specific, modelling tools required to create the informal and formal knowledge representation of a domain-specific application case. The tools provided in this environment enable design thinking and conceptual modelling. The common platform ADOxx provides the needed flexibility and adaptivity features for the tools.
   Functionality and specifically model-value functionalities are provided via the ADOxx platform (https://www.adoxx.org/live/home) that enables binding of services and execution, whereas complex service structures (e.g. orchestrated and choreographed) are realized using the OLIVE framework (https://www.adoxx.org/live/olive) as an execution environment for micro/web-services. This is specifically important for realizing the connectors to the compliance and execution environment.
2. **Execution Environment** – The execution environment is responsible for the abstraction of device functionalities and the provision of endpoints for interaction. The environment exposes an API so that the model-based environment can interact with. This API is divided into passive components that provide the state of the device and active ones that support actuation and triggering of functionality. A workflow manager is available to translate service calls and provide means for sequencing logic steps of interaction and the integration of communication streams to the other environments. Both building blocks are based on the abstracted device functionality.
3. **Compliance Environment** – This building block is responsible to impose ethical, legal and security/safety considerations on the design as well as the execution environment. A distributed ledger is foreseen to guarantee

revision safe and trustworthy persistence of compliance results, whereas the assessment service is exposed as a dynamic combination of questions, checklists and individual assessment techniques. The assessment service is responsible for the composition, parsing of individual items and assessment scales and the provisioning to the modeller or CPS expert.

5. Conclusion

The document at hand is the initial deliverable of work package 3 and aims at the definition and specification of the demonstration scenarios by raising question and requirements in order to concretize the demonstration cases and establish a common understanding of the project's motivation and objective.

An observation within these experiments is the need to transform those systems towards intelligent environments. Smart functionality can be used to define and apply a knowledge-based approach. This implies that the knowledge and expertise of human stakeholders is externalized, and specific aspects are supported by artificial intelligence technologies like image recognition. Furthermore, cognitive capabilities are based on the model representation and model-value/processing functionalities. Stakeholders from different backgrounds are supported to grasp the complexity, understand system relations and dependencies as well as the impact of design decisions transparently.

Based on these observations, the project specifically addresses the ethical, (criminal) legal, and security/safety aspects of the CPS environment, considering a model-based/knowledge-driven approach of interaction in the environment. The challenges of these aspects with respect to transformation towards AI systems were formulated and discussed in the document and motivated the selection of application cases for evaluation. As domain for the application scenarios, a smart supermarket scenario was chosen. Three robotic platforms serve as CPS for the application scenarios (a) a robot arm, (b) mobile platforms, and (c) a human/robot cooperation.

Following main observations were made:

- Knowledge externalisation and applicability for criteria of assessment matter
- The stack of hardware and software components impact the compliance assessment
- A core concern with respect to compliance relates to the aspects of data (as sensor streams), their quality and provenance
- The distinction between controlled and autonomous is regarded as technical and compliance challenge
- Distribution requires the formulation of services that are contacted to provide additional (model) value to the system

Following key challenges were raised in this document:

- Identification of an AI technique to resolve flexibility concerns during design and execution time
- Extension of the modelling languages to capture goal settings and decision rules in case of escalations, adaptation and uncertainty
- Definition of the impact of compliance assessments (on type level) on the system and its components
- Identification of an adequate architecture that reflects autonomous considerations and correlates those with uncertainty
- Identification of language requirements for each device (and its specificities) as well as a global goal function
- Data quality and provenance aspects, as well as technical and architectural aspects
- Identification of the trustworthiness characteristics of services identified as output from assessment criteria and catalogues
- Identification of language requirements to support dynamic, domain-specific extension of existing implementations
- Definition and classification of collaboration streams between human actors and machines/services/functionalities by considering ethical, legal, security and safety aspects

More details and a prototype will be discussed in "D3.2".