# Deliverable 4.1

## Modellbasierter Prototyp eines Assistenzsystems

(Model-based Prototype for an Assistant System)

| | | |
|---|---|---|
| Dokument Version | : | Final |
| Abgabe | : | 31.01.2021 |
| Dissemination Level | : | Öffentlich |
| Beitrag zu | : | WP4 |
| Dokument Inhaber | : | BOC |
| Dokument Name | : | ComplAI-D4.1 Modellbasierter Prototyp eines Assistenzsystems |
| 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** | 09/11/2020 | BOC | Initial, table of content, content collection |
| **0.2** | 10/11/2020 | BOC | Very first draft version |
| **0.3** | 16/11/2020 | BOC | Extension, revision |
| **0.4** | 01/12/2020 | BOC | Writing and Revision of draft version, extension, appendix |
| **0.5** | 20/12/2020 | BOC | Concluding the document for internal review |
| **0.8** | 10/01/2021 | BOC | Ready for Review |
| **1.0** | 31/01/2021 | BOC, UNIVIE | Final Review of Deliverable |

## List of Contributors:

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

## List of Reviewers:

Laura Crompton (UNIVIE)

Robert Woitsch (BOC)

## Kurzfassung

Das modellbasiertes Assistenzsystem verbindet auf der einen Seite den Kriterienkatalog – dieser wird im Arbeitspaket 5 erarbeitet – mit den modell-basierten Ansteuerung der Roboter – diese wird in Arbeitspaket 3 erarbeitet.

Dieses Dokument stellt also die Verbindung von (a) den im D5.1 erläuterten Abfrageschemen um ethische-, rechtliche- security[1]- und safety-Aspekte mittels geeigneten Fragenkatalog zu beurteilen sowie von (b) den in D3.2 erläuterten modell-basierten Ansteuerung von Robotern dar.

Das modellbasierte Assistenzsystem ermöglicht ein Assessment von Modellen – die in unserem Projekt Roboter ansteuern und in D3.2 ausgearbeitet sind - nach verschiedenen Kriterien – die in unsrem Projekt ethische-, rechtliche-, securtiy- und saftey Fragestellungen beantworten, die in D5.2 ausgearbeitet sind.

Dafür ist die sogenannte Konzeptualisierung der Fragenbögen sowie die Konzeptualisierung der Roboteransteuerung – wie Workflows, Petri-Netze oder Flow-Charts - notwendig, um dann beide Konzeptualisierten Schemen miteinander zu verbinden. Der dahinterliegende Mechanismus ist das Meta-Model Matching[2], das unterschiedliche Metamodelle miteinander verbindet.

Dieses Dokument stellt ein eigens entwickeltes Metamodell für das Beurteilen von Modellen gemäß domänen-spezifischen Kriterien vor. Dieses Metamodell wurde prototypisch auf der für akademische Zwecke offen zugänglichen ADOxx Plattform realisiert (ADOxx.org). Somit kann der Mechanismus erprobt werden um ein Modell - das für eine Ansteuerung eines Roboters verwendet wird - mittels Fragebögen zu beurteilen und danach so zu signieren, dass eine Veränderung des beurteilten Modells unmöglich ist.

Der Prototyp besteht aus zwei Hauptkomponenten: (a) der Bewertungskomponente sowie (b) der Signatur und Zertifikatskomponente.

Die Bewertungskomponente erlaubt die Überprüfung von Modellen mit Hilfe von Fragebögen, welche in Abstimmung mit Domänen-Experten erstellt wurden. Die Signatur.- und Zertifikatskomponente ermöglicht zunächst eine digitale Unterschrift und anschließend eine Überprüfung der Modelle.

Das Prototyp Paket wird zum Download zur Verfügung gestellt. Dieses kann getestet werden, indem entweder die vorgefertigten Prototypen verwendet werden, oder ob entweder der Kriterienkatalog oder die Ansteuerungsmodelle verändert werden. Alle Materialien für eine Hands-on Erfahrung werden bereitgestellt.

[1] „Security“ und „Safety“ wird hier auch im Deutschen mit den Englischen Begriffen verwendet, um beide Merkmale des „Sicherheits“-Begriffs, nämlich (1) die sichere Ausführung und (2) der Sicherheit gegenüber Fremdübernahme zu verdeutlichen.
[2] Kühn, H.: Methodenintegration im Business Engineering. PhD Thesis, University of Vienna, April 2004.,

# Executive Summary

The model-based assistance system connects on the one side the assessment criteria catalogue – which is worked out in workpackage 5 – with the models that are used to configure and steer robots – that have been worked out in workpackage 3.

This document hence connects (a) the assessment criteria catalogue that has been worked out in D5.1 and identifies appropriate questionnaires to assess ethical-, legal-, security- and safety-issues with (b) the model-based configuration and steering of robots that have been worked out in D3.2.

This model-based assessment system enables the assessment of models – in our project we were using models that steer and configure robots as worked out in D3.2 – according different criteria. In our project we use a criteria catalogue addressing ethical-, legal-, security- and safety issues worked-out in D5.1.

A pre-requisite for such a linkage is the conceptualisation of both, first the questionnaires and the corresponding criteria catalogue as well as those models – like workflows, petri-nets or flow-charts - that are necessary to steer and configure a robot in order to enable the linkage of those conceptualisations. The applied mechanism is the so-called meta-model matching[3] that enables the linkage of different meta models.

This document introduces the developed meta model enabling the assessment of models according domain-specific criteria. The metamodel has been developed as a prototyped using the ADOxx platform which is freely available for academic purpose (ADOxx.org). Hence, it is possible to test the algorithm that allows the assessment of a model that is used to steer a robot via a questionnaire and by digitally signing the result. The digital Signature ensures that, once a model has been signed, it cannot be changed.

The prototype consists of two core components: (a) the assessment component as well as (b) the signature and certification component.

The assessment component allows to check a model and its content based on a questionnaire that is designed for specific purpose of the models.

Second, the signature and certification service enables to check the validity of a model by means of a digital signature, hence the change of a digitally signed model is not possible. The certification service enables the check, if a model has been signed.

The available prototype package is provided as a download. It can either be installed and tested with the provided prototype models, or with adapted assessment and steering models. The prototype package provides all necessary material to test the prototype.

[3] Kühn, H.: Methodenintegration im Business Engineering. PhD Thesis, University of Vienna, April 2004.,

# Table of Contents

Kurzfassung ... 3
Executive Summary ... 4
List of Figures ... 6
List of Tables ... 7
List of Abbreviations ... 8
1. Introduction ... 9
1.1 Overview of Model-based Assessment System ... 9
1.2 Relation to Work Package ... 10
1.3 Document Structure ... 10
2. Architecture of the Assistant System ... 11
2.1 Technical High-Level Architecture ... 11
2.2 Conceptual High-Level Architecture ... 14
2.2.1 Meta Model for Questionnaire Component ... 14
2.2.2 Meta Models for related modelling languages ... 14
2.2.3 Meta-Model Merging – using the "Reference" Pattern ... 16
2.2.4 Converting Assessment Criteria into Questionnaire Models ... 16
2.3 Secure environment for the compliant development of robot work flows ... 17
2.3.1 Compliance Environment ... 17
2.3.2 Distributed Ledger and smart contracts ... 18
3. Questionnaire Assessment Prototype ... 20
3.1 Requirement Exploration ... 20
3.2 Assessment Architecture ... 21
3.3 Introductory Sample using the Prototype ... 22
4. Signature and Certification Prototype ... 26
4.1 Requirement Elaboration ... 27
4.2 Signature and Certification Prototype Architecture ... 27
4.3 Introductory Sample using the Prototype ... 28
5. Prototype Package ... 30
5.1 Prerequisites ... 30
5.2 Getting Started ... 31
6. Conclusion ... 33

## List of Figures

Figure 1 Model-based assessment system within the project context ... 10
Figure 2 High-Level Technical Architecture ... 11
Figure 3 Meta Model Questionnaire ... 14
Figure 4 Meta Model Petri Net ... 14
Figure 5 Root Element metamodel fragment ... 15
Figure 6 Core objects of BPMN ... 15
Figure 7 Mapping between Criteria Catalogues and Questionnaire Models ... 16
Figure 8 The distributed ledger architecture of the compliant development and execution environment ... 18
Figure 9 A simplified flow of the development within the compl@i environment ... 19
Figure 10 Model questionnaire procedure ... 20
Figure 11 Olive-based assessment architecture ... 22
Figure 12 Model questionnaire ... 23
Figure 13 Transformation of questionnaire format ... 23
Figure 14 Questionnaire selection ... 24
Figure 15 Association of model objects and questionnaire ... 24
Figure 16 Selection of Model to be Assessed ... 25
Figure 17 Presenting associated questionnaires and the assessment ... 25
Figure 18 Model signing and certification procedure ... 26
Figure 19 Olive-based signature and certification architecture ... 28
Figure 20 Model signature service ... 28
Figure 21 Sign a model ... 29
Figure 22 Verify the signature ... 29
Figure 23 Prototype package structure ... 30
Figure 24 Starting Bee-Up as standalone application ... 31
Figure 25 Model questionnaire service ... 32
Figure 26 Model signature service ... 32

## List of Tables

Table 1 Assessment application requirements.......... 21
Table 2 Signature and certification application requirements.......... 27

## List of Abbreviations

| | |
|---|---|
| AI | Artificial Intelligence |
| API | Application Programming Interface |
| BPMN | Business Process Modelling Notation |
| HTML | Hypertext Markup Language |
| ID | Identifier |
| JSON | JavaScript Object Notation |
| LDAP | Lightweight Directory Access Protocol |
| REST | Representational State Transfer |
| SOAP | Simple Object Access Protocol |
| UI | User Interface |

# 1. Introduction

The model-based assessment systems connect domain-specific criteria – in our project we worked out ethical-, legal-, security- and safety criteria – with models that need to be assessed against those criteria. In our project those models are BPMN business processes and workflows, Petri-Nets and UML Flow charts that have been worked out to steer the behaviour of a robot.

The challenge how criteria have to be represented to sufficiently asses a certain aspect is elaborated in D5.2 that lists the criterion catalogue. The challenge how a model can define or influence the behaviour of a robot, is elaborated in D3.2 that shows which abstraction layers and which form of autonomous behaviour can be defined in workflows.

This document addresses the challenges:

- How can assessment criteria be conceptualised and how can the answer to those criteria be interpreted by computers in order to rate that a model is either approved or not.
- How can a model – which can be potentially in any form – be assessed in an efficient – wrt. to user handling - and effective – with respect of the quality of the result – way.
- How to link assessment criteria and models for robot steering purposes.
- How to introduce a digital signature that ensures the authentication of a signature and
- What is the generic architecture and how can it be realised on a prototype platform like ADOxx.

First, the technical and conceptual architecture is introduced. The technical architecture relies on the open platform ADOxx and the open source Microservice Framework OLIVE and the conceptual architecture is based on the meta model matching patterns, where a so-called semantic lifting approach is applied to reference two meta models.

This document explains first the theoretical background and second demonstrates how those findings have been realised in form of prototypes. Those prototypes are provided for download, and due to the open character of ADOxx and OLIVE can be evolved by any interested party.

Before explaining the concepts in more detail, we introduce the high-level architecture to provide an overall orientation.

## 1.1 Overview of Model-based Assessment System

Figure 1 introduces the overview of the model-based assessment system by linking the different criteria with models that are used to operate robot behaviour. After the assessment a digital signature is applied on the models, which are then sent to the robots.

The prototypes demonstrates that:

1. The model can be checked with regard to legal, ethical and security & safety issues. Essential at this point is to find out how those catalogues can be modelled and assessed. In particular, the foundation for this challenge is tackled in D5.1, which covers the collection of assessment catalogues. This might be reasonable in order to avoid that a robot hurts people or that decisions taken by the AI cannot be followed by humans.
2. The next challenge deals with modelling AI and robotics as well as the assessment of compliance. Those issues are mainly handled in the document at hand. Approved models can be signed in order to avoid fraud. As the assessed models might be shared between different stakeholders, it ensures that an assessment is transparent. The models are signed using a hash code, which means that any change in the model results in a mismatch of the public available part of the signature and the hash code that is calculated from a textual (XML) representation of the model.
3. Third challenge is about operating compliant models on a robotic platform. Deliverable 3.2 deals with those aspects more specifically. Before using the models on the robotic platform their validity can be ensured by means

of certification checks. It ensures that only valid models are sent to the robot platform. A technical mounting of the robot is necessary that this validation check is performed before execution.

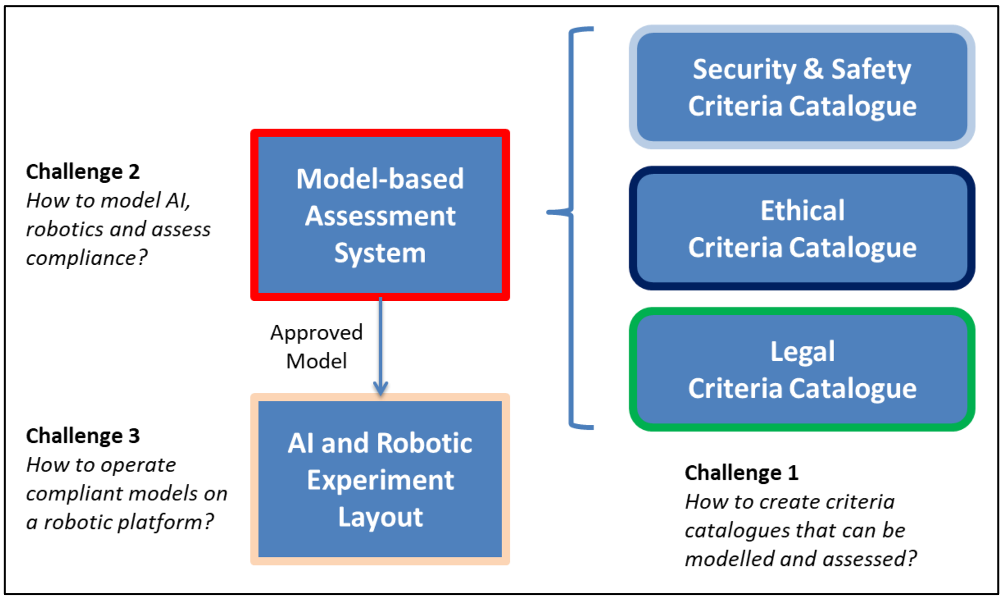

Challenge 2
How to model AI, robotics and assess compliance?
Model-based Assessment System
Approved Model
Challenge 3
How to operate compliant models on a robotic platform?
AI and Robotic Experiment Layout
Security & Safety Criteria Catalogue
Ethical Criteria Catalogue
Legal Criteria Catalogue
Challenge 1
How to create criteria catalogues that can be modelled and assessed?

**Figure 1 Model-based assessment system within the project context**

## 1.2 Relation to Work Package

The deliverable at hand is part of work package 4, which deals with a feasibility analysis by means of a prototype for a model-based assistant system. This is the only deliverable in work package 4 and it summarizes the results achieved in task 4.1 and 4.2. In particular, task 4.1 covers the development of a model-based assistant system, whereas task 4.2 is about AI assessment for assistant systems. The results are tightly connected to work package 5 that includes the creation of a criteria catalogue for ethical, legal and security issues.

## 1.3 Document Structure

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

This document starts with an introduction in order to provide an overview of content as well as a high-level contextual project overview. Afterwards, in Section 2 the architecture of the assistant system and its relationship to other components of the project is outlined. Section 3 covers the assessment of any criteria including a descriptive sample. Furthermore, the process of signature and certification is detailed in Section 4. The prototype package is introduced in Section 5. Finally, a short conclusion summarizes the results.

# 2. Architecture of the Assistant System

## 2.1 Technical High-Level Architecture

The high-level technical architecture is introduced in Figure 2 can be seen as a composition of three environments:

- The Design Environment is based on the Meta Modelling Platform ADOxx and provide a set of graphical modelling tools as well as the corresponding connectors.
- The Execution Environment is based on the different technology stack of the robot environments, which can be abstracted as outlined in the high-level architecture, but where a platform-specific aspect is involved.
- The Compliance Environment consists of different criteria catalogues that are expressed in a special format, so they can be read by the assessment tool. The corresponding signature is then placed on a Distributed Ledger or a Blockchain so it can be ensured that the digital signature is authentic.

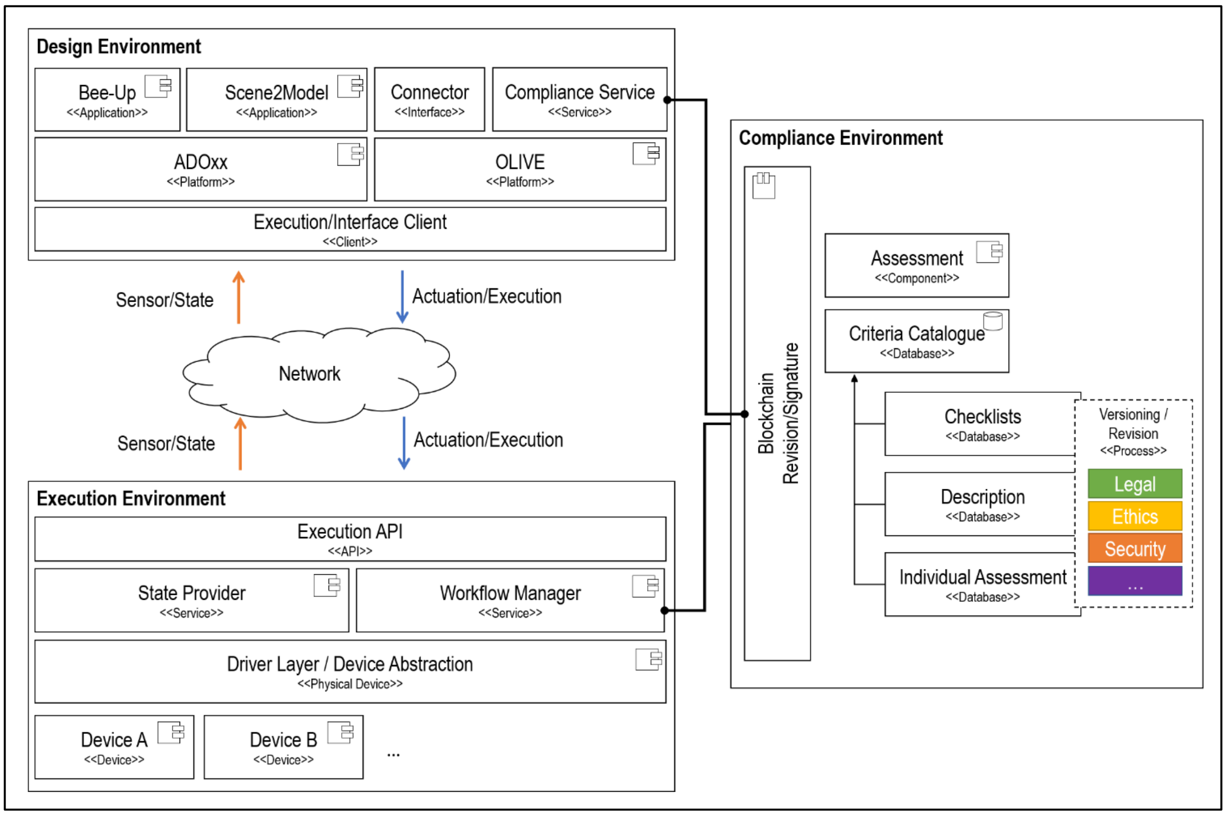


**Figure 2 High-Level Technical Architecture**

The Compliance Environment receives input from the Design Environment in form of models. The criteria catalogue is provided form external experts – in case of project, those experts were the legal and ethical partners who provide their contribution in form of a criterion catalogue in D5.1.

In the following we elaborate the Compliance Environment that links the models that have to be assessed with the assessment critiera, and furthermore ensure a transparent and secure hand-over of the assessed models to the Execution Environment.

The Sequence of using the Compliance Environments is described as following:

1. **Transformation of Assessment Criteria into Excel Sheet:**
   The first step is the collection and transformation of the assessment criteria from an informal and textual description – like the sample set in D5.1 – into a semi-formal structure that can be used to generate a questionnaire model. Task of this survey, was to identify typical assessment criteria and how they are expressed as well as addressing the challenge to transform them into a computer readable format to create a questionnaire model.
   First, we state that we do not limit the expert in formulating the assessment criteria, hence the original input – mainly in form of tables in a text document – is not restricted. This ensures that advisors can formulate in the most appropriate way, without the need to reflect model-based or technical limitations. This text then needs to be manually transferred into "questions" and "assessment criteria" that can be represented in a conceptual model. This step is performed by transforming the assessment criteria form a text file into an Excel Sheet and extending the pure criterion with questionnaire specific meta information.
2. **Generating Questionnaire Model out of the Excel Sheet:**
   The second step is the (semi-) automatic generation of a questionnaire model based on the provided Excel sheet input. This transformation is performed as the Design Environment provides several built-in features that can be used to approve the questionnaire model.
   Some key characteristics are:
   - The Excel Sheet has only a limited number of references, whereas a modelling tool can interlink potentially without limit. Hence, the interconnections and the re-use of questions are much more powerful.
   - The Model represents the questionnaire in a graphical way; hence co-creation can be performed where experts, non-experts as well as domain-experts can co-create the questionnaire using the graphical and intuitive visualisation. Furthermore, collaboration features like chats, comments and change histories are typically provided by state-of-the-art modelling environments.
   - Models can be stored in form of reference patterns, hence questionnaire models that have been successfully used in one application scenario or domain can be re-used for other application scenarios and other domains. A common model-repository typically provides features like a central storage, a user authentication, user or role specific access rights and the use of generators that read the repository content and documents, transfers or export the models in different forms. It has to be stated, that each use case requires personalised adaptation of the provided reference patterns.

   Outcome of this phase is a well-designed and from all experts co-creatively approved questionnaire including a threshold and score calculation. This means, the questionnaire model is approved to:

   a. Correctly represent the questions and assessment criteria raised by the experts, which are understandable for the targeted end user.
   b. The scores are approved, which means that a score defines if a question is sufficiently successfully answered – green code; insufficiently answered – yellow code or sufficiently unsuccessfully answers – red code.
3. **Preparing the use of Questionnaire Model within Assessment Tool:**

   Once the questionnaire model is completed and approved it can be used in the assessment tool. Within the project we implemented an open source Microservices within the OLIVE framework that interprets the questionnaire model and provides the corresponding user interface to answer the questions.

   Before starting the questionnaire, we need to align the concrete model which is actually assess with the questionnaire model. As explained in more detail in the next section, we propose a conceptualisation of both (a) the questionnaire in form of a questionnaire model and (b) the model to be assessed in form of an executable workflow on a robot platform. The conceptual linkage is performed with a semantic lifting approach, using the name of the model objects as the glossary and linking them via a mapping.

Technically, we realise this linkage with a user interface that selects the questionnaire model on the one side and the execution model on the other side and stores this mapping within the assessment tool. Hence, this linkage can be accessed only vial the assessment tool in the current prototype implementation. There are approaches to store this mapping also in a centralised repository or to change the models and store this mapping in the models, but those alternatives would reduce the current flexible approach. Hence, for a commercial outlook, we consider the integration of those aspects as extension services of the Design Environment, but for the purpose of this feasibility survey, we haven chosen the light weight and flexible approach.

4. **Usage of the Questionnaire Model within Assessment Tool:**

   A user interface is provided for the usage of the assessment tool. This user interface enables the selection of the correct model and questionnaire and perform the assessment by answering the questions. There is an immediate colour code in form of green, yellow or red, that indicates if the particular question has been sufficiently answered.

   The answered or additional information that is uploaded, is stored in the assessment tool by using a database.

   This database is also used to digitally sign a model, by using a non-cryptographic signature in order to sign the hash-code of the text representation of the model. In order to sign the model, we provide an authentication system, where users and user roles can be specified, and the digital signatures can be verified in a so-called single source of truth.

5. **Verification of a Model:**
   The final aim is to use the signed model by the Execution Environment; hence the Execution Environment needs to verify if a model has been successfully signed or not. This verification is performed by sending a request to the verification interface, which checks if the textual representation of the model is the same as the textual representation which has been signed. In case this check is positive, the verification service returns that the signature is verified, in the other case it returns that the signature of the model is not verified.

After introducing the technical sequence, how the components in the high-level architecture are used, we discuss in the next section the conceptual architecture to introduce in more detail, how the questionnaire model and the criterion catalogue are linked.

## 2.2 Conceptual High-Level Architecture

### 2.2.1 Meta Model for Questionnaire Component

The “Questionnaire Metamodel” is depicted using the CoChaCo notation which is design tool for a meta-model[4]. The “Questionnaire Metamodel” consists of two major concepts, which are Questionnaire and Question. The Questionnaire is connected to the Question with a Contains connector. The main attributes are depicted for each meta model concept. The questionnaire modelling library is based on ADOxx, which means that some commonly used attributes, such as the object name are available due to the pre-defined ADOxx meta model. The question answer type attribute can have various facets such as, short text or multiple choice for instance.

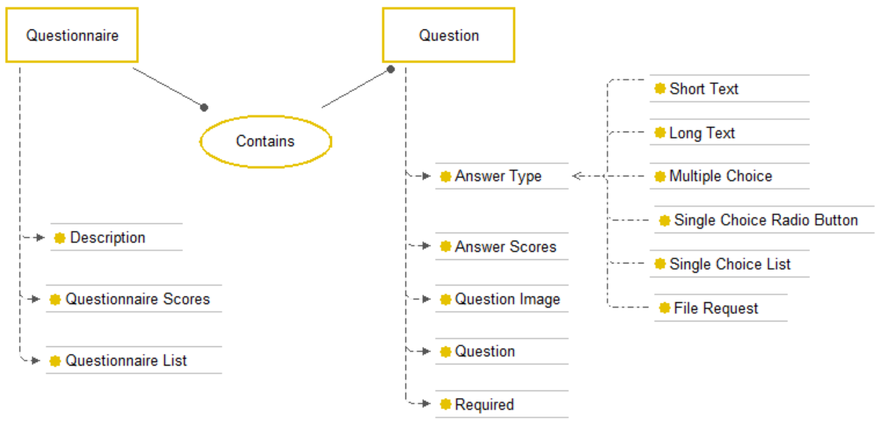


**Figure 3 Meta Model Questionnaire**

Throughout the project, different modelling languages, such as BPMN, Petri Net or Flowchart were used to model the workflows. The following explanations provide some more insights on the meta models of those languages and pave the way for integrating the different approaches.

### 2.2.2 Meta Models for related modelling languages

The well-known Petri Nets consists of places and transitions that are connected with arcs. Additionally, a Place can have a Mark in form of a token. Places as well as Transitions can be named. A Transition has additionally the attribute Ready that outlines if a Transition can be fired or not.

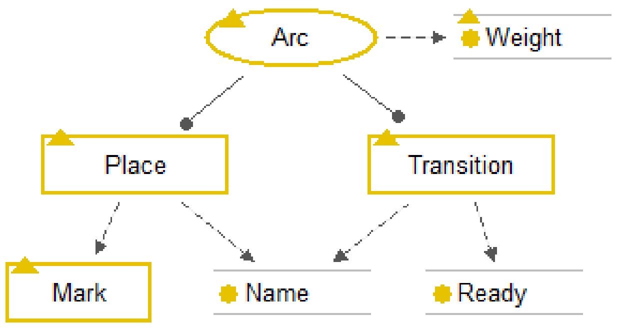


**Figure 4 Meta Model Petri Net**

[4] www.omilab.org/activities/cochaco.html (last visited 09/02/2021)

The Business Process Model Notation (BPMN) is a standard modelling language developed by the Object Management Group (OMG). BPMN 2.0[5] is recent version of this standard released in 2011. BPMN is among other aspects created to ease collaboration between business and software people. It provides a solid meta model, which defines modelling constructs, their semantics and data properties that can be used as a foundation for developing executable process models. Due to the complexity of the meta model, the meta model can be divided in fragments (such as Root Elements, Container, Flow Elements, Lanes, Resource Roles, Data Elements, Artifacts, and Messages). The following graphical visualization shows only a fragment of the whole meta model, as otherwise the description would go beyond the scope of this project. Further details on the BPMN meta model can be found in "D3.2 Learnpad"[6].

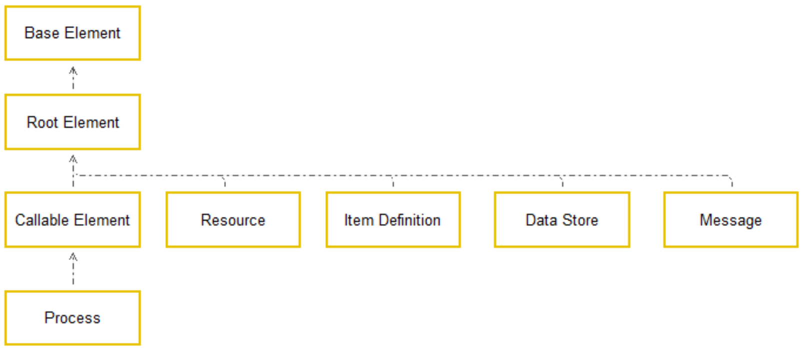


**Figure 5 Root Element metamodel fragment**

BPMN also provides a graphical notation, which enables domain experts to design and analyse business processes using graphs similar to flowcharts. The core objects that are used for the projects purpose of BPMN can be found in the following picture. We used events (start and end) as well as activities (task and subprocess) in order to depict the process from a high-level domain specific point of view.

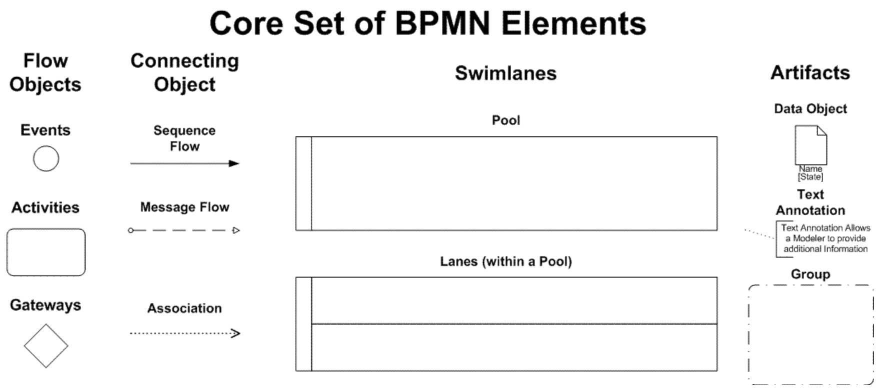


**Figure 6 Core objects of BPMN**[7]

Flowchart diagrams contain elements that are similar to the core set of BPMN elements, therefore we go without an additional meta model here. The major objects are start/end, arrows, input/output, process and decision.

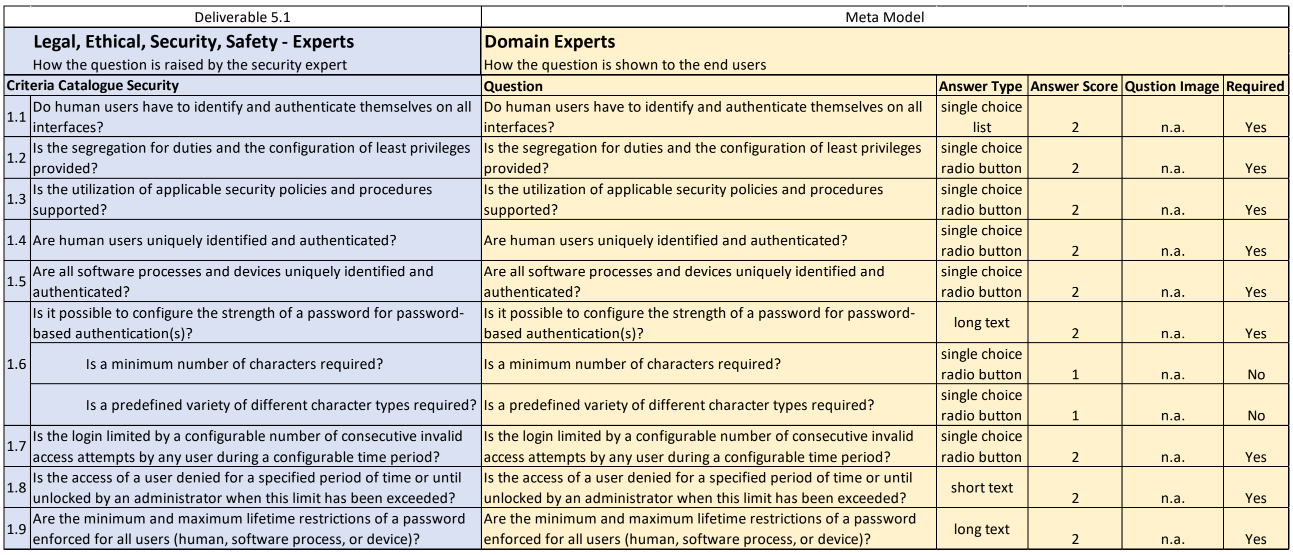
[5] Business Process Model OMG. (2001). Notation (BPMN) 2.0. Object Management Group: Needham, MA, 2494:34, 2011.
[6] D3.2 Learnpad. (2015). Retrieved from http://www.learnpad.eu/index.php.
[7] BPMN Core Elements (2021). Retrieved from https://www.omg.org/bpmn/Samples/Elements/Complete_BPMN_Elements.htm.

### 2.2.3 Meta-Model Merging – using the “Reference” Pattern

The challenge was to connect BPMN, Flowchart and Petri Net models with the questionnaire model. Here we revisited the pattern-oriented integration approach from Kühn[8]. The described integration patterns was used to integrate heterogenous modelling languages to ensure the flexibility that the assessment can be performed not only by the prototype models that were used during this project, but is generic in such a way that future robot execution models can also be assessed. The questionnaire and other relevant modelling languages are loosely coupled by using a reference pattern[9]. The reference pattern is characterized by bringing together complementary metamodels so that different situations and perspectives can be shown. A reference link connects exactly one element of the main meta model with one element of another meta model.

### 2.2.4 Converting Assessment Criteria into Questionnaire Models

After describing how the criteria catalogue can be conceptually linked with executable models, the corresponding link from the assessment criteria to the text document provided in D5.1 is elaborated. For the purpose of bridging the gap of textual description of the criteria catalogue to the questionnaire model, we introduce a mapping file, which we used in form of an Excel Sheet. The criteria provided by legal, ethical, security and safety experts are captured in a textual way in the assessment criteria catalogue D5.1. In order to formalize the criteria and to map the questions to the introduced questionnaire meta model, a tabular approach was chosen, as can be seen in the following Figure 7.



| Deliverable 5.1 | | Meta Model | | | | |
|---|---|---|---|---|---|---|
| **Legal, Ethical, Security, Safety - Experts**<br>How the question is raised by the security expert | | **Domain Experts**<br>How the question is shown to the end users | | | | |
| **Criteria Catalogue Security** | | **Question** | **Answer Type** | **Answer Score** | **Qustion Image** | **Required** |
| 1.1 | Do human users have to identify and authenticate themselves on all interfaces? | Do human users have to identify and authenticate themselves on all interfaces? | single choice list | 2 | n.a. | Yes |
| 1.2 | Is the segregation for duties and the configuration of least privileges provided? | Is the segregation for duties and the configuration of least privileges provided? | single choice radio button | 2 | n.a. | Yes |
| 1.3 | Is the utilization of applicable security policies and procedures supported? | Is the utilization of applicable security policies and procedures supported? | single choice radio button | 2 | n.a. | Yes |
| 1.4 | Are human users uniquely identified and authenticated? | Are human users uniquely identified and authenticated? | single choice radio button | 2 | n.a. | Yes |
| 1.5 | Are all software processes and devices uniquely identified and authenticated? | Are all software processes and devices uniquely identified and authenticated? | single choice radio button | 2 | n.a. | Yes |
| 1.6 | Is it possible to configure the strength of a password for password-based authentication(s)? | Is it possible to configure the strength of a password for password-based authentication(s)? | long text | 2 | n.a. | Yes |
| | Is a minimum number of characters required? | Is a minimum number of characters required? | single choice radio button | 1 | n.a. | No |
| | Is a predefined variety of different character types required? | Is a predefined variety of different character types required? | single choice radio button | 1 | n.a. | No |
| 1.7 | Is the login limited by a configurable number of consecutive invalid access attempts by any user during a configurable time period? | Is the login limited by a configurable number of consecutive invalid access attempts by any user during a configurable time period? | single choice radio button | 2 | n.a. | Yes |
| 1.8 | Is the access of a user denied for a specified period of time or until unlocked by an administrator when this limit has been exceeded? | Is the access of a user denied for a specified period of time or until unlocked by an administrator when this limit has been exceeded? | short text | 2 | n.a. | Yes |
| 1.9 | Are the minimum and maximum lifetime restrictions of a password enforced for all users (human, software process, or device)? | Are the minimum and maximum lifetime restrictions of a password enforced for all users (human, software process, or device)? | long text | 2 | n.a. | Yes |

**Figure 7 Mapping between Criteria Catalogues and Questionnaire Models**

On the left-hand side – coloured in blue – the expert questions how they are listed in D5.1 can be seen, while on the right-hand side, the questions are leveraged so that they can be automatically transformed in a specific questionnaire model. The figure presents a segment of the whole criteria catalogue and its mapping to the questionnaire meta model. As short meta model recap that was introduced in the aforementioned section – a question element consists of the question itself, an answer type, a score, a required mark and optionally a picture. The answer type can be specialized by short text, long text, multiple choice, single choice ratio button, single choice list or file request.

While the expert questions in the presented sample snippet are raised by a security expert, the question may need to be rephrased that a domain expert – who is no legal, ethical or security expert - is capable to follow the question and hence answer it correctly. Additional concept-like answer types, scores and specific details of questions need to be added as a configuration of the questionnaire. In this sample, a one-to-one mapping of the security to the domain questions is used,

[8] Kühn, Harald & Karagiannis, Dimitris. (2005). Strategie-, Prozess- und IT-Management: Ein Pattern-orientierter Integrationsansatz. 1483-1502. 10.1007/3-7908-1624-8_78.
[9] Kühn, H.: Methodenintegration im Business Engineering. Dissertation, Universität Wien, April 2004.

as the domain expert considered these questions understandable for the end user. However, in case there would be for instance law specific terms included, a rephrasing might be necessary for more transparency. As the domain expert is familiar with the domain requirements, he can rate the answer scores so that they best fit the domain/scenario requirements. For instance, some sub-questions may be less important compared to the main question, therefore, the score may be lower. The domain expert can define the possible answer types related to the questions in the context of the application scenario.

The graphical representation of this questionnaire model allows the integration of domain experts as an intermediary between law, ethics, security and safety experts on the one side and the end users on the other side. Therefore, interdisciplinarity is supported by a graphical and co-creative questionnaire modelling and approval capabilities.

## 2.3 Secure environment for the compliant development of robot workflows

This section describes how the assessed executable model is transferred towards the Execution Environments elaborating the use of distributed ledger technology. This is seen as an extension of the assessment services, in order to ensure a distributed and reliable usage of the assessed and signed models.

The following text has successfully been published as a paper on workflow integrity[10] and for completeness reasons the relevant parts rephrased and copied in the following sub-section.

Our overall goal is to establish a framework of tools to verify robot applications already in their development against various criteria from the sides of safety, security, legal or ethics. We want to present the concept of a cryptographic infrastructure that enables the seamless documentation of actions throughout the development of an application from the start of development until the execution of the application.

We foresee an environment that guides an application developer through a catalogue of criteria to be addressed during application design. The check for compliance to the criteria should be automated wherever possible, if this is infeasible, we can fall back to questionnaires asking if certain actions have been performed or requirements are met. This interaction and the outcomes should continuously be documented such that before the application is released into production, it can be digitally signed and fed into the workflow verification system presented here. The whole infrastructure is further referred to as the "compl@i environment".

### 2.3.1 Compliance Environment

The compl@i environment provides an infrastructure for verifying workflows in terms of ethical, legal, safety and security aspects (and potentially more in future). The verification process is realized in form of a distributed ledger to ensure integrity, authenticity and non-repudiation of the artifacts resulting from the verification process as well as the development process itself.

[10] Breiling, B.; Dieber, B.; Pinzger, M.; Rass, S. A Cryptography-Powered Infrastructure to Ensure the Integrity of Robot Workflows. *J. Cybersecur. Priv.* **2021**, *1*, 93-118. https://doi.org/10.3390/jcp1010006

### 2.3.2 Distributed Ledger and smart contracts

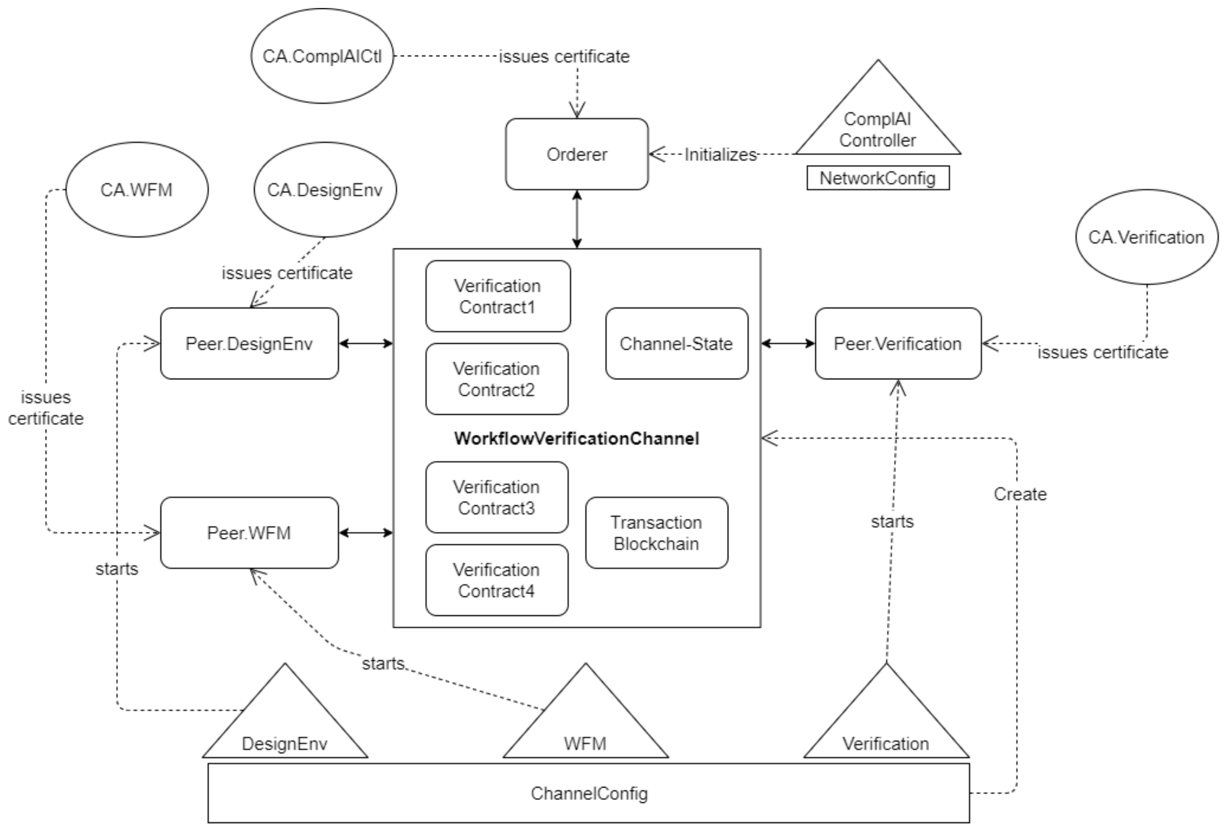


**Figure 8 The distributed ledger architecture of the compliant development and execution environment**

For the compl@i environment we utilize the open-source framework HyperLedger Fabric[11], which provides a distributed ledger software package including the necessary tools for setting up the environment appropriately. In this context the information shared by the peers is grouped in channels which ensures privacy of sensitive information. The channel access can be restricted by the use of certificates. Every channel consists of a world state, which is serialized and stored in a database and a block chain which stores the transactions committed on the state. Later modification on past transaction entries lead to an invalid block chain and can be recognized by each peer which guarantees data integrity. Further, the transactions contain a digital signature of the issuer which ensures authenticity on the one hand and non-repudiation on the other hand. In addition to that, participants can initiate smart contracts in order to agree on how the ledger state can be modified. A smart contract defines an asset which the participants can work on as well as methods which can be called when committing a transaction on the ledger. If a peer wants to execute a transaction, it first has to be endorsed by a set of channel peers. If the endorsement policy is fulfilled the transaction is sent to an orderer node which defines an order of incoming transactions and stores them into blocks for the transaction block chain. After that, the block is sent out to each peer which then store the block in their replication of the transaction block chain. Figure 8 shows a concept for a distributed ledger for the compl@i environment.

[11] https://www.hyperledger.org/use/fabric

Again, we have the client and the robot as actors along with a verification system. The picture in Figure 9 provides a logical view on the compl@i distributed ledger, which means that channel state, contracts and block chain are illustrated as shared information within a channel. Additionally, each participating entity has a peer node operating on the channel. These nodes serve as an entry point for client application which want to operate on the ledger. Up to now four entities were identified during the design process:

- **The Design Environment:** What the client uses to compose the robot application
- **The Execution Environment:** The extended execution entitity as presented in this work
- **The Compliance Environment** itself with
    - **The ComplAIController** responsible for setting up the ledger and running the orderer node
    - **The Verification Entity** responsible for for verifying workflows in terms of ethical, legal safety and security aspects

Each entity has its own Certificate Authority in order to issue access rights to external clients and additional peers if necessary.

The verification process (as illustrated in Figure 9) of a robot program is implemented in form of a smart contract, which is installed on the channel. When a new development is started, a new channel is opened. During the composition of the application, the verification environment provides the developer with supporting information on the requirements. This can be drawn either from automated verification methods as far as possible or from pre-developed questionnaires or from domain experts that may judge the application (such as typically done in safety certification). Each transaction independent if tool output, questionnaire or expert input is logged within the blockchain infrastructure. As the figure shows, the verification process can be triggered from the program composition environment (part of the client). Then the verification will be performed until all inputs from questionnaire, automated evaluation and external experts has been addressed. At some point, the application can be approved and it is digitally signed for execution on the robot. The robot verifies that the signature of the workflow is valid and executes it. This part corresponds to the work presented here.

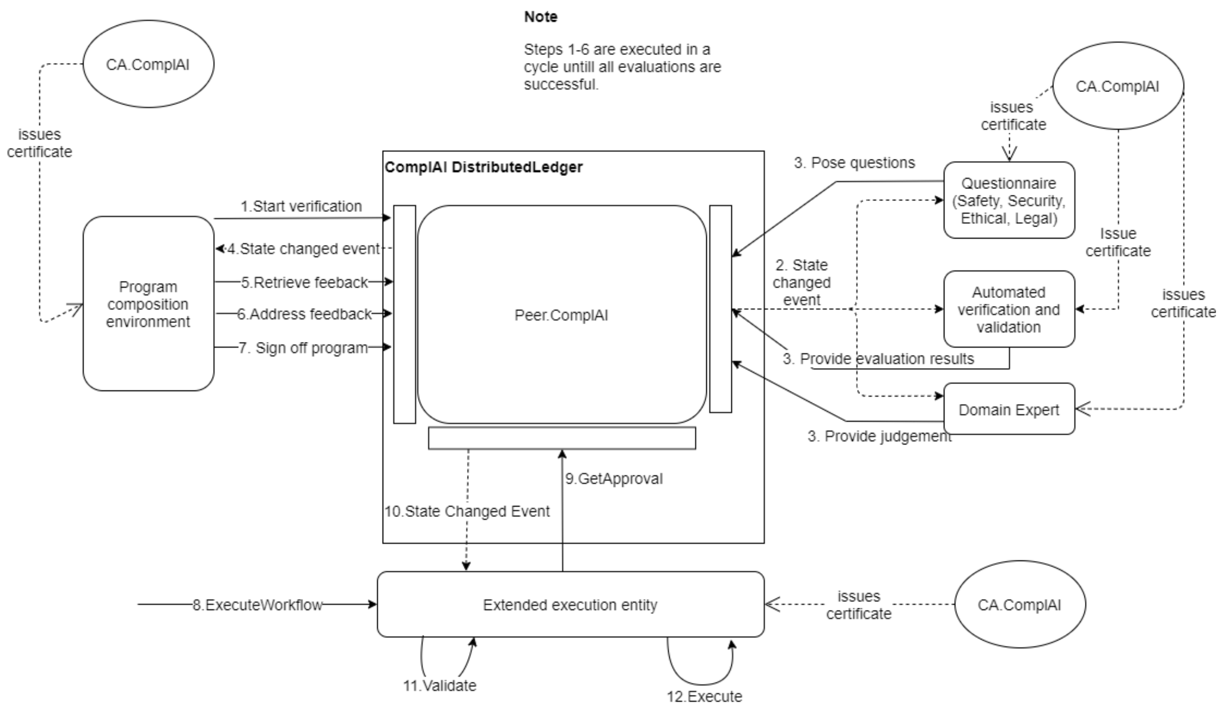
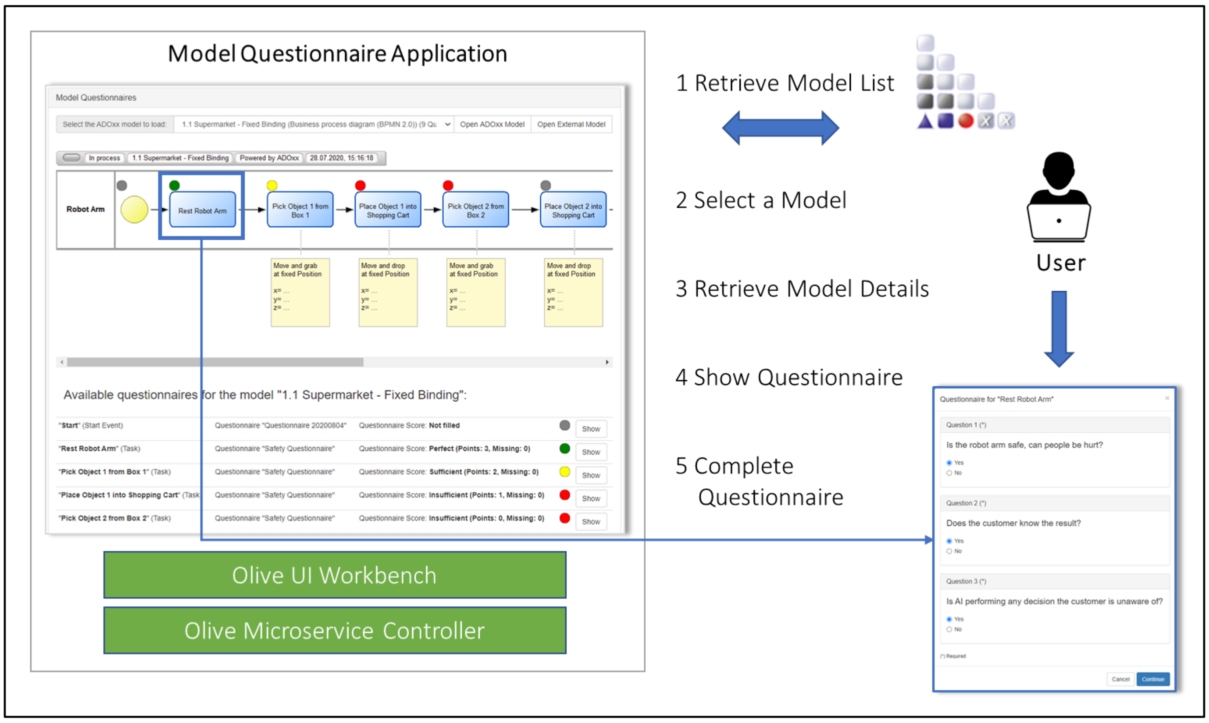


**Figure 9 A simplified flow of the development within the compl@i environment**

# 3. Questionnaire Assessment Prototype

Figure 10 introduces the intended user interaction to link a questionnaire with an executable model and start the assessment. In a first step, the user selects a model in the assessment application. The application visualizes the model and an assessment questionnaire can be associated with every task of the respective model. Users and/or experts can open the questionnaire and answer the questions. The application page is regularly updated with the latest questionnaire results. Specific color codes indicate the results of the evaluation and show where information is required.



Model Questionnaire Application
Model Questionnaires
Select the ADOxx model to load:
1.1 Supermarket - Fixed Binding (Business process diagram (BPMN 2.0)) (9 Qu
Open ADOxx Model
Open External Model
In process
1.1 Supermarket - Fixed Binding
Powered by ADOxx
28.07.2020, 15:16:18
Robot Arm
Rest Robot Arm
Pick Object 1 from Box 1
Place Object 1 into Shopping Cart
Pick Object 2 from Box 2
Place Object 2 into Shopping Cart
Move and grab at fixed Position x= ... y= ... z= ...
Move and drop at fixed Position x= ... y= ... z= ...
Move and grab at fixed Position x= ... y= ... z= ...
Move and drop at fixed Position x= ... y= ... z= ...
Olive UI Workbench
Olive Microservice Controller
1 Retrieve Model List
2 Select a Model
3 Retrieve Model Details
4 Show Questionnaire
5 Complete Questionnaire
User

Available questionnaires for the model "1.1 Supermarket - Fixed Binding":

| | | | | |
|---|---|---|---|---|
| **"Start"** (Start Event) | Questionnaire "Questionnaire 20200804" | Questionnaire Score: **Not filled** | ● | Show |
| **"Rest Robot Arm"** (Task) | Questionnaire "Safety Questionnaire" | Questionnaire Score: **Perfect (Points: 3, Missing: 0)** | ● | Show |
| **"Pick Object 1 from Box 1"** (Task) | Questionnaire "Safety Questionnaire" | Questionnaire Score: **Sufficient (Points: 2, Missing: 0)** | ● | Show |
| **"Place Object 1 into Shopping Cart"** (Task | Questionnaire "Safety Questionnaire" | Questionnaire Score: **Insufficient (Points: 1, Missing: 0)** | ● | Show |
| **"Pick Object 2 from Box 2"** (Task) | Questionnaire "Safety Questionnaire" | Questionnaire Score: **Insufficient (Points: 0, Missing: 0)** | ● | Show |

Questionnaire for "Rest Robot Arm"

Question 1 (*)

Is the robot arm safe, can people be hurt?

[✓ Yes] No

Question 2 (*)

Does the customer know the result?

[✓ Yes] No

Question 3 (*)

Is AI performing any decision the customer is unaware of?

[✓ Yes] No

(*) Required

Cancel Continue

**Figure 10 Model questionnaire procedure**

## 3.1 Requirement Exploration

The generic use case described above raises some requirements. Two types of models must be provided to the assessment Application. Those are a list of ADOxx[12] models, as well as external models. The selection of any model without restrictions based on modelling method or language is possible. Furthermore, in order to extend the model and its objects with colour codes, they must be converted into a picture. An administrative section is given to define questionnaires and associate those with model objects. The questionnaire must be evaluated based on score aligned with the questionnaire content and context. After retrieving the questionnaire, the user must be able to store his answers.

[12] https://www.adoxx.org/live/home (last visited on 10/11/2020)

Table 1 presents a detailed list of requirements including the essential input and output, as well as some relevant notes.

| Requirement | Input | Output | Notes |
|---|---|---|---|
| model list retrieval of all ADOxx models | no direct input | ADOxx model list including model name and ID | the ADOxx SOAP interface is used |
| model image retrieval of ADOxx model | model ID | model image in Base 64 format | the ADOxx SOAP interface is used |
| model object retrieval of ADOxx model | model ID | list of model objects | the ADOxx SOAP interface is used |
| questionnaire retrieval of ADOxx questionnaire model | model ID | questionnaire in JSON format | use a REST service, in specific the GET method, for returning the questionnaire associated to the selected model |
| questionnaire evaluation functionality | model ID, result JSON with questionnaire answers | evaluation result as JSON file | use a REST service, in specific the POST method, to retrieve the associated questionnaires and calculate a score, storage of score in a database, retrieval of model details, present details and score to client |
| questionnaire adding functionality | questionnaire ID, questionnaire JSON file | no direct output | use a REST service, in specific the POST method |
| questionnaire association functionality | questionnaire ID, model object ID | no direct output | use a REST service, in specific the POST method |

**Table 1 Assessment application requirements**

In the following we introduce how the current open-source prototype – which is available for download – has been realised.

## 3.2 Assessment Architecture

The assessment architecture, visualized in Figure 11, explains in more detail, how the identified requirements have been realized in the prototype implementation.

First, we distinguish between front end components, so-called Micro Frontends that can be composed to a common user interface. The use of HTML5 widgets that can be configured to a user-specific user interface is well known and common in todays software engineering. Current trends in Microservices – a different philosophy of providing computer capabilities – consequently lead to the idea of Micro Frontends as a form of corresponding user interface. A well-known representative of Micro Frontends is the MS TEAMS platform that allows the integration of user interfaces in the Web-Browser like way.

Although this is not relevant when describing the mechanisms of the assessment or validation component, we want to emphasise the fact that we consider the new trends in software development and respectively see our concept appropriate also for future development. The definition and combination of the widgets is facilitated by the Olive[13] UI Workbench, which is open-source.

Beside the Micro Frontend aspects, we consider the connection and orchestration of Microservices, in our case the access to the Design Environment using an ADOxx SOAP API, the verification services or the aforementioned Distributed Ledger environment as relevant, and we applied the open-source framework OLIVE to establish the connection and the orchestration between those services. More concretely the Olive micro service controller is used to create all services, which require and use an ADOxx connector based on the ADOxx SOAP server features, a MySQL connector in order to communicate with the database that contains questionnaires and associations, as well as an orchestration feature for combining all services. As mentioned, MySQL is used to collect all questionnaires as well as the associations between the model object and the questionnaire. This was explained in previous section as the reference between questionnaire models and executable models and a central storage that can be accessed.

[13] https://www.adoxx.org/live/olive (last visited on 10/11/2020)

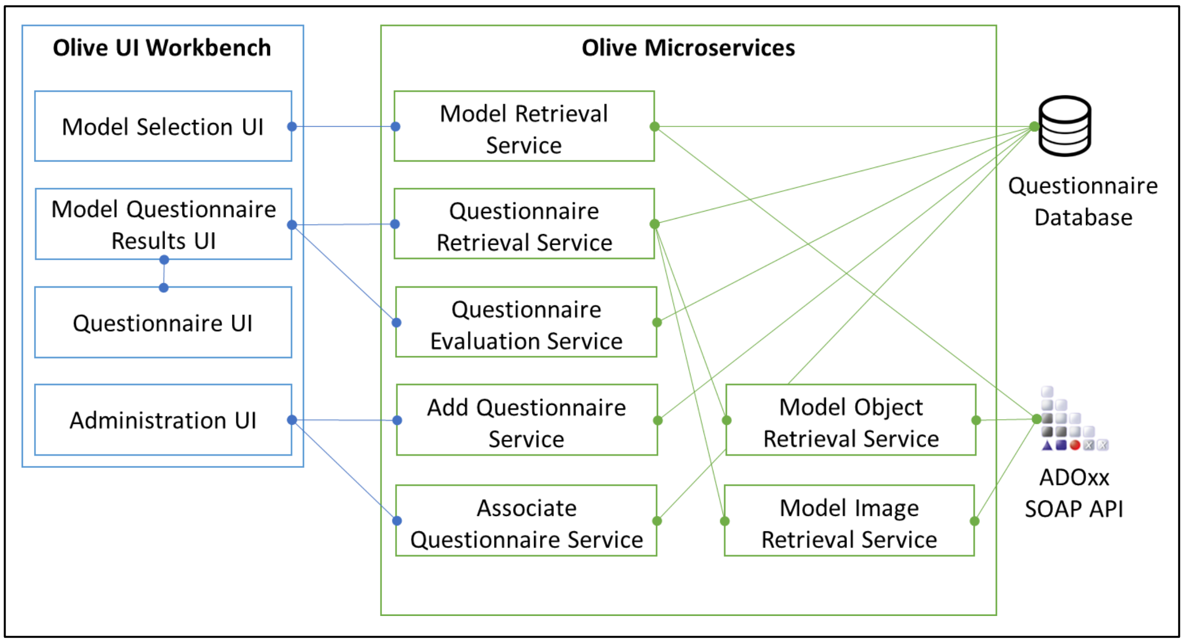

Olive UI Workbench
Model Selection UI
Model Questionnaire Results UI
Questionnaire UI
Administration UI
Olive Microservices
Model Retrieval Service
Questionnaire Retrieval Service
Questionnaire Evaluation Service
Add Questionnaire Service
Associate Questionnaire Service
Model Object Retrieval Service
Model Image Retrieval Service
Questionnaire Database
ADOxx SOAP API

**Figure 11 Olive-based assessment architecture**

It is composed of two major Olive components, the UI workbench and a set of microservices. ADOxx serves as a modeling component by providing a SOAP API interface. A database for the question and the answers serves as a data storage.

There are four UI components: administration, model selection, model questionnaire results closely connected with the questionnaire UI. The foundation for using the UI components is a set of microservices that are targeted at the specific components. The Olive microservice collection serves as an intermediary between Workbench, database and ADOxx.

The data layer indicated on the right side realize the flexible approach, by separating the questionnaire databased, that contains the converted and ready to use questionnaires, as well as the model that need to be assessed, that are independently accessed by the ADOxx interface.

## 3.3 Introductory Sample using the Prototype

A questionnaire modelling language, shown in  Model questionnaire, allows to create a questionnaire that contains the aforementioned questions that have been extracted from D5.1 and transformed to a questionnaire model. Those questions can be of different types and can either be domain and / or model independent, such as if a certain decision is performed by AI or a human, or it can be domain or model specific, such if a certain move of a robot can be “compliantly” performed.

In the following, we list different concrete questions, where (1) is domain and model dependent, (2) is domain dependent, as well as (3) domain and model independent:

1. Is the robot arm for picking object x safe, or can customers be hurt?
2. Does the customer know the result, when using service x?
3. Is AI performing any decision the user is unaware of?

In this sample, the question type is single-choice. In case to drill-down the analysis, the question types can be detailed ranging from open answers – where human expert interpretation is foreseen - to multiple choice answers. The answers can be scored individually, which is initially filled out in the Excel Sheet we presented in 2.2.4.

The resulting score for the overall questionnaire is used in connection with the definition of color codes. Those codes are presented for each model that is assigned to the questionnaire.

The detailing of questions and answer types in combination with the evolution of scoring algorithms is considered as a co-creative process to evolve the quality of the assessment. Full-fledged modelling capabilities like the graphical representation, the documentation, the analysis, version history, extension using additional files or collaboration between experts is supported to finally end up with a quality approved questionnaire.

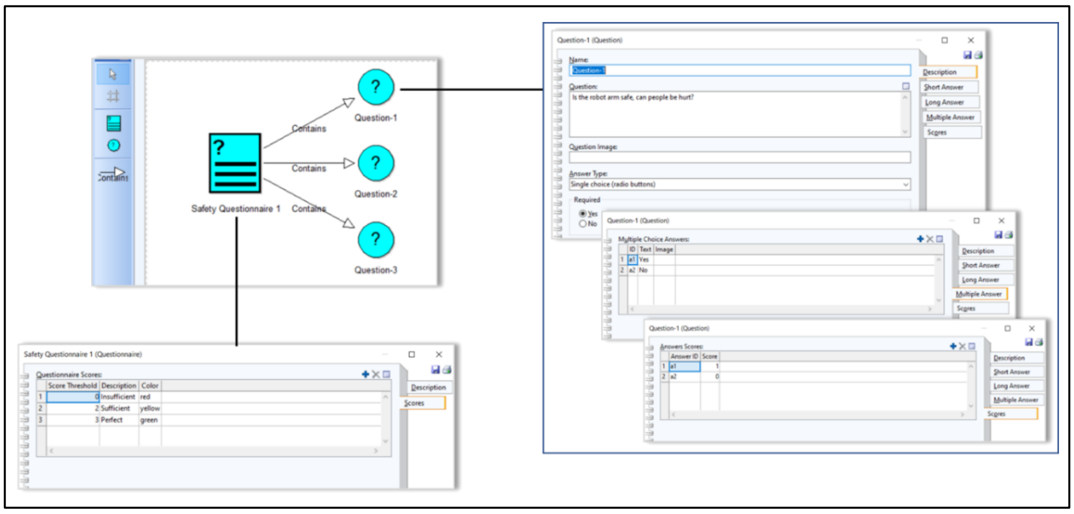
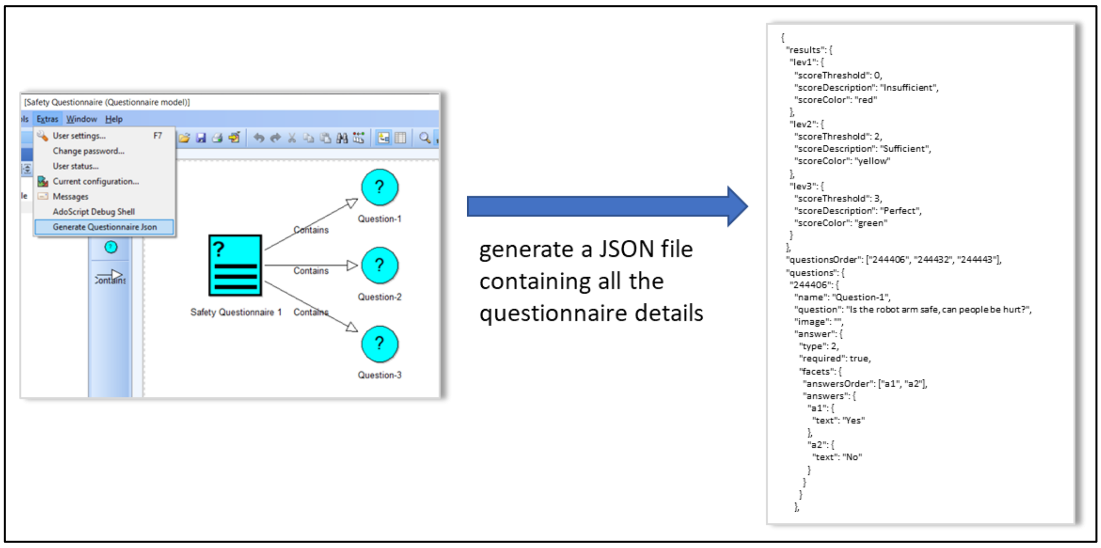


**Figure 12 Model questionnaire**

In order to raise the flexibility of this prototype, the questionnaire model is presented in JSON format, hence the model is transformed into the required format by using an export feature in ADOxx, see section 2.2.4



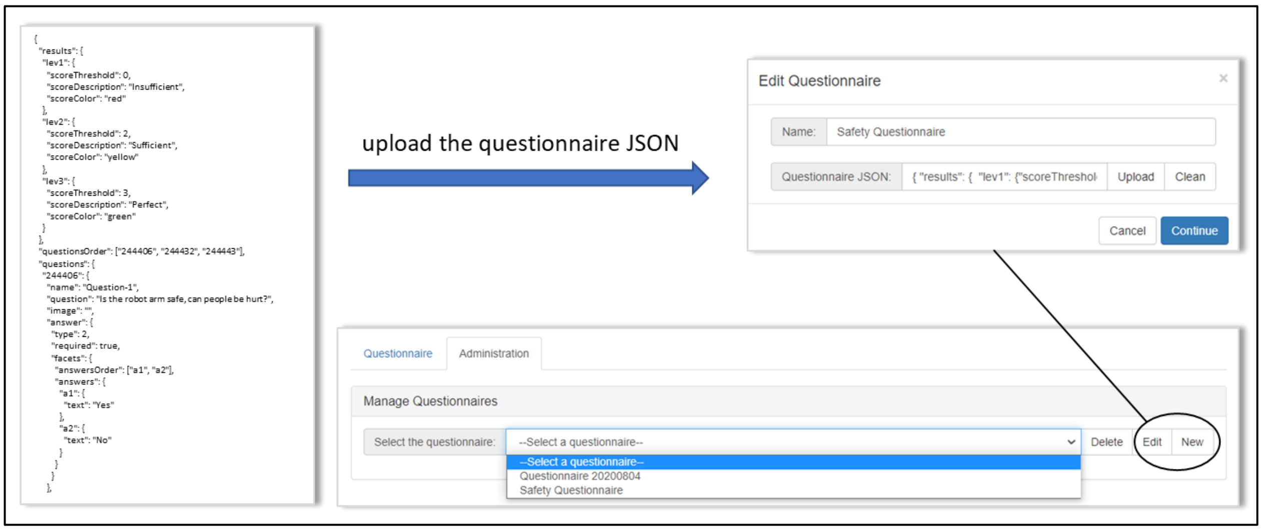
```
{
 "results": {
  "lev1": {
   "scoreThreshold": 0,
   "scoreDescription": "Insufficient",
   "scoreColor": "red"
  },
  "lev2": {
   "scoreThreshold": 2,
   "scoreDescription": "Sufficient",
   "scoreColor": "yellow"
  },
  "lev3": {
   "scoreThreshold": 3,
   "scoreDescription": "Perfect",
   "scoreColor": "green"
  }
 },
 "questionsOrder": ["244406", "244432", "244443"],
 "questions": {
  "244406": {
   "name": "Question-1",
   "question": "Is the robot arm safe, can people be hurt?",
   "image": "",
   "answer": {
    "type": 2,
    "required": true,
    "facets": {
     "answersOrder": ["a1", "a2"],
     "answers": {
      "a1": {
       "text": "Yes"
      },
      "a2": {
       "text": "No"
      }
     }
    }
   },
```

**Figure 13 Transformation of questionnaire format**

The linkage between model to be assessed and the questionnaire model, which is conceptually described in 2.2.3 is implemented in the prototype in form of an “administration component” that allows to link a questionnaire with a model. A new questionnaire can be uploaded in JSON format, as presented in Figure 13

Both ADOxx as well as external models can be loaded, as presented in Figure 14 to complete the linkage between the model and the questionnaire. In case the model is available via ADOxx, additional features like the model picture are shown, and the questionnaires and the corresponding color-codes is “laid over” the model.

To detail the analysis of model, the uploaded questionnaires can be associated not only with the whole model but also with each individual object in the model. It is also possible to have no questionnaire for a specific model object, for instance a certain activity of the workflow like the setup of the robot, which is irrelevant for the assessment but have a detailed questionnaire for the follow-up activity like the reset movement of a robot.



```
{
 "results": {
  "lev1": {
   "scoreThreshold": 0,
   "scoreDescription": "Insufficient",
   "scoreColor": "red"
  },
  "lev2": {
   "scoreThreshold": 2,
   "scoreDescription": "Sufficient",
   "scoreColor": "yellow"
  },
  "lev3": {
   "scoreThreshold": 3,
   "scoreDescription": "Perfect",
   "scoreColor": "green"
  }
 },
 "questionsOrder": ["244406", "244432", "244443"],
 "questions": {
  "244406": {
   "name": "Question-1",
   "question": "Is the robot arm safe, can people be hurt?",
   "image": "",
   "answer": {
    "type": 2,
    "required": true,
    "facets": {
     "answersOrder": ["a1", "a2"],
     "answers": {
      "a1": {
       "text": "Yes"
      },
      "a2": {
       "text": "No"
      }
     }
    }
   }
  },
```

upload the questionnaire JSON

Edit Questionnaire

Name: Safety Questionnaire

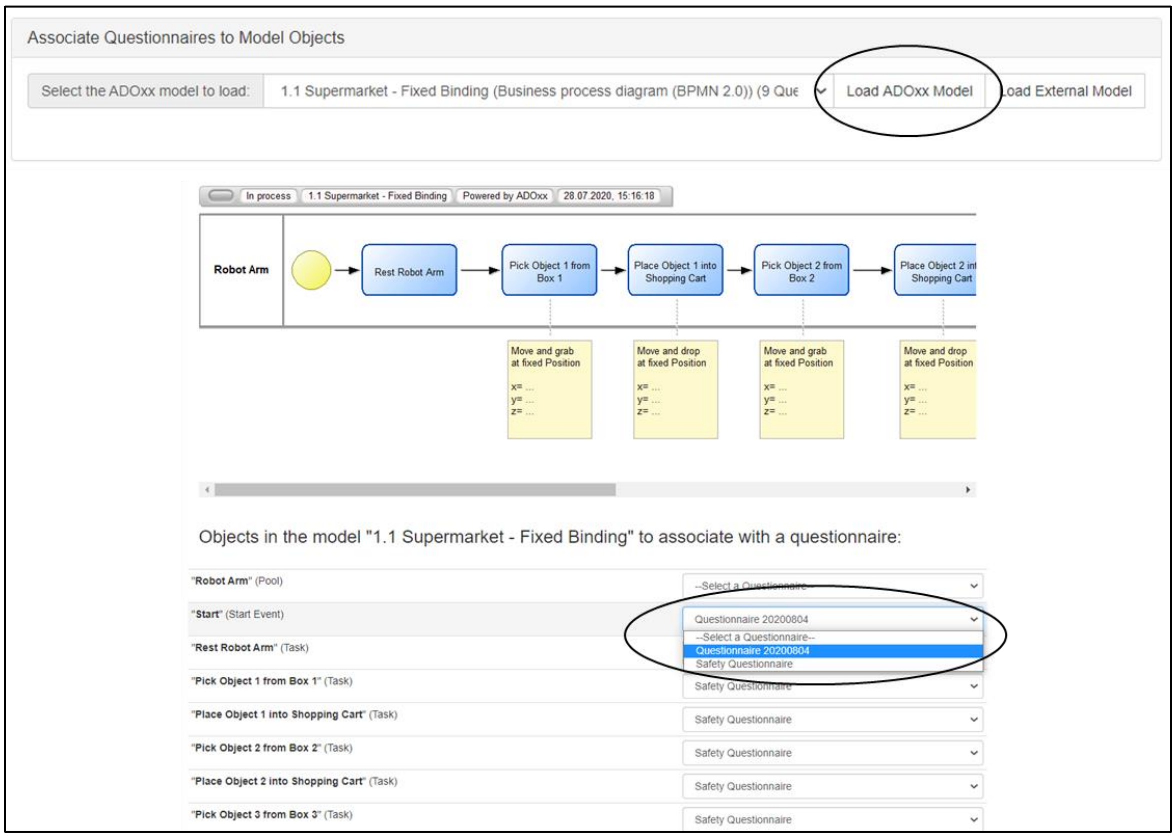
Questionnaire JSON: { "results": { "lev1": {"scoreThreshol

Manage Questionnaires

Select the questionnaire: --Select a questionnaire--

- --Select a questionnaire--
- Questionnaire 20200804
- Safety Questionnaire

Figure 14 Questionnaire selection



Associate Questionnaires to Model Objects

Select the ADOxx model to load: 1.1 Supermarket - Fixed Binding (Business process diagram (BPMN 2.0)) (9 Que

Load ADOxx Model
Load External Model
In process
1.1 Supermarket - Fixed Binding
Powered by ADOxx
28.07.2020, 15:16:18
Robot Arm
Rest Robot Arm
Pick Object 1 from Box 1
Place Object 1 into Shopping Cart
Pick Object 2 from Box 2
Place Object 2 int Shopping Cart
Move and grab at fixed Position x= ... y= ... z= ...
Move and drop at fixed Position x= ... y= ... z= ...
Move and grab at fixed Position x= ... y= ... z= ...
Move and drop at fixed Position x= ... y= ... z= ...

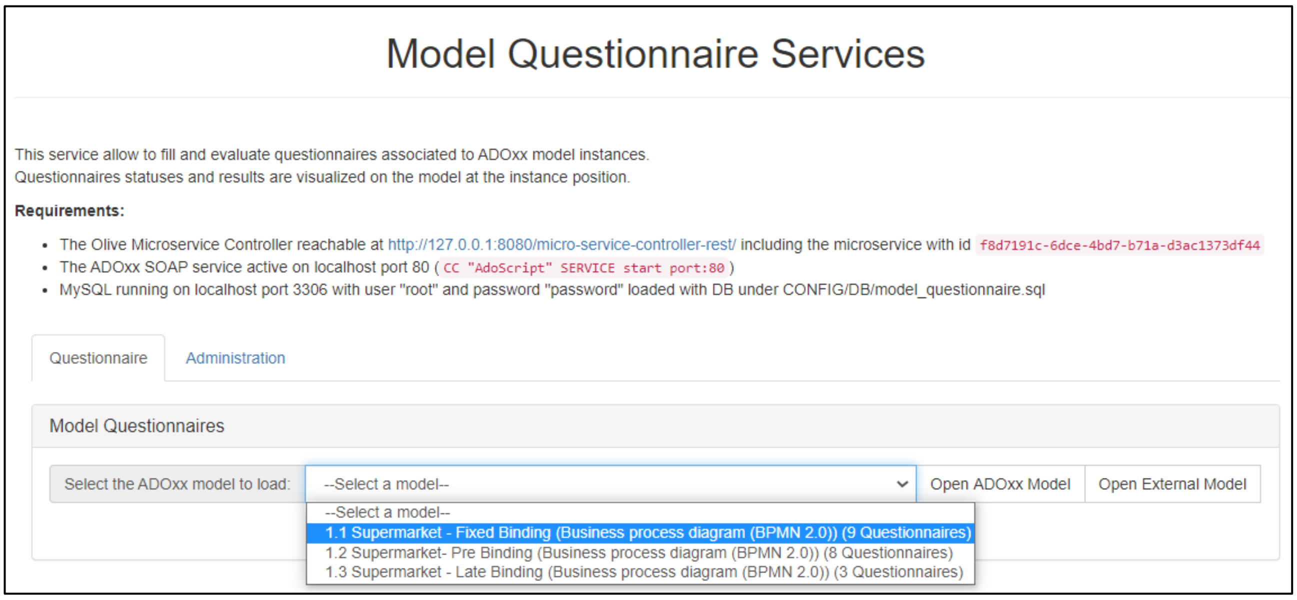
Objects in the model "1.1 Supermarket - Fixed Binding" to associate with a questionnaire:

| | |
|---|---|
| "Robot Arm" (Pool) | --Select a Questionnaire-- |
| "Start" (Start Event) | Questionnaire 20200804 |
| "Rest Robot Arm" (Task) | |
| "Pick Object 1 from Box 1" (Task) | Safety Questionnaire |
| "Place Object 1 into Shopping Cart" (Task) | Safety Questionnaire |
| "Pick Object 2 from Box 2" (Task) | Safety Questionnaire |
| "Place Object 2 into Shopping Cart" (Task) | Safety Questionnaire |
| "Pick Object 3 from Box 3" (Task) | Safety Questionnaire |

- --Select a Questionnaire--
- Questionnaire 20200804
- Safety Questionnaire

Figure 15 Association of model objects and questionnaire

After associating models with questionnaires, change back to the questionnaire component. Figure 15 shows the list of uploaded models and the number of assigned questionnaires. Select the model used above.



# Model Questionnaire Services

This service allow to fill and evaluate questionnaires associated to ADOxx model instances.
Questionnaires statuses and results are visualized on the model at the instance position.

**Requirements:**

- The Olive Microservice Controller reachable at http://127.0.0.1:8080/micro-service-controller-rest/ including the microservice with id `f8d7191c-6dce-4bd7-b71a-d3ac1373df44`
- The ADOxx SOAP service active on localhost port 80 ( `CC "AdoScript" SERVICE start port:80` )
- MySQL running on localhost port 3306 with user "root" and password "password" loaded with DB under CONFIG/DB/model_questionnaire.sql

Questionnaire | Administration

Model Questionnaires

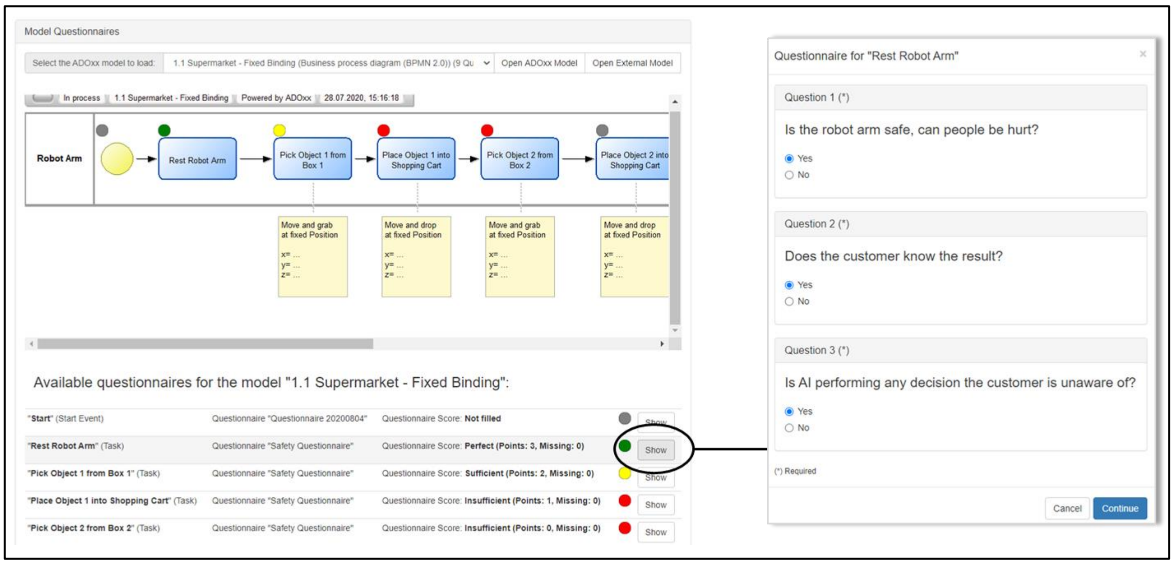
Select the ADOxx model to load: --Select a model-- | Open ADOxx Model | Open External Model

- --Select a model--
- 1.1 Supermarket - Fixed Binding (Business process diagram (BPMN 2.0)) (9 Questionnaires)
- 1.2 Supermarket- Pre Binding (Business process diagram (BPMN 2.0)) (8 Questionnaires)
- 1.3 Supermarket - Late Binding (Business process diagram (BPMN 2.0)) (3 Questionnaires)

**Figure 16 Selection of Model to be Assessed**

Figure 16 depicts what the user interface for a model assessment can look like. The grey dots indicate that the corresponding questionnaire was not filled out yet. This can be done by going through the model object list on the bottom and showing the questionnaire. After filling it out, the result can be saved. The other coloured dots show the overall score of the questionnaire.

In this sample, the "Reset Robot Arm" indicates a green code, indicating that the current available information is sufficient for a compliant approval.



Robot Arm
Rest Robot Arm
Pick Object 1 from Box 1
Place Object 1 into Shopping Cart
Pick Object 2 from Box 2
Place Object 2 into Shopping Cart
Move and grab at fixed Position x= ... y= ... z= ...
Move and drop at fixed Position x= ... y= ... z= ...
Move and grab at fixed Position x= ... y= ... z= ...
Move and drop at fixed Position x= ... y= ... z= ...

Model Questionnaires

Select the ADOxx model to load: 1.1 Supermarket - Fixed Binding (Business process diagram (BPMN 2.0)) (9 Qu | Open ADOxx Model | Open External Model

In process | 1.1 Supermarket - Fixed Binding | Powered by ADOxx | 28.07.2020, 15:16:18

Available questionnaires for the model "1.1 Supermarket - Fixed Binding":

| | | | | |
|---|---|---|---|---|
| "**Start**" (Start Event) | Questionnaire "Questionnaire 20200804" | Questionnaire Score: **Not filled** | ● | Show |
| "**Rest Robot Arm**" (Task) | Questionnaire "Safety Questionnaire" | Questionnaire Score: **Perfect (Points: 3, Missing: 0)** | 🟢 | Show |
| "**Pick Object 1 from Box 1**" (Task) | Questionnaire "Safety Questionnaire" | Questionnaire Score: **Sufficient (Points: 2, Missing: 0)** | 🟡 | Show |
| "**Place Object 1 into Shopping Cart**" (Task) | Questionnaire "Safety Questionnaire" | Questionnaire Score: **Insufficient (Points: 1, Missing: 0)** | 🔴 | Show |
| "**Pick Object 2 from Box 2**" (Task) | Questionnaire "Safety Questionnaire" | Questionnaire Score: **Insufficient (Points: 0, Missing: 0)** | 🔴 | Show |

Questionnaire for "Rest Robot Arm"

Question 1 (*)

Is the robot arm safe, can people be hurt?

[✓ Yes] No

Question 2 (*)

Does the customer know the result?

[✓ Yes] No

Question 3 (*)

Is AI performing any decision the customer is unaware of?

[✓ Yes] No

(*) Required

Cancel | Continue

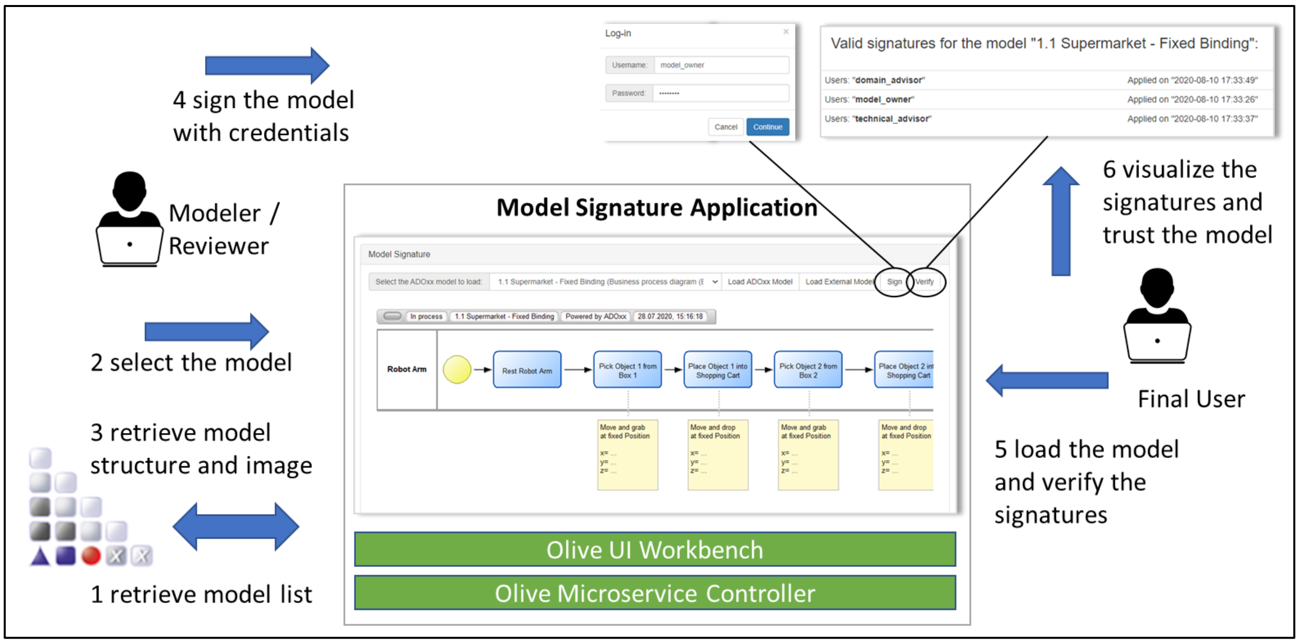
**Figure 17 Presenting associated questionnaires and the assessment**

## 4. Signature and Certification Prototype

Once a model is sufficiently assessed and finally approved, it needs assurance that the model is not changed afterwards. We propose to digitally sign the model in such a way, that first the signatory can be identified – using authentication - and second, that changes in a model that have been performed after the model has been signed can be tracked – using hashing of the textual representation of the model. Both can be performed via digital signatures. We analyzed three possible technologies:

- ***Cryptographic signatures using Smart Cards:*** This principle uses physical artefacts like a card reader and a personalized smart card. The signatory has a physical card or other means of personal verification and can sign the model by using its smart card in combination with the corresponding hardware and software interaction.
- ***Cryptographic signatures using Remote Sign Authority:*** This principle uses a centralized trusted third party that ensures the correctness of the signature. A precondition is that all applications refer to the same trusted third party. In most cases such trusted third parties offer their service in form of a business model, only rare providers perform this service for free, or in form of open source (e.g. SignService).
- ***Non-Cryptographic Signature***: This principle uses a so-called single source of truth, which is often an infrastructure that is operated for the specific purpose of an application. In our case this means, that we install the full infrastructure for the authentication and the digital signature and ensure that it is used correctly.

For the purpose of an open-source proof-of-concept prototype, we decided to install the whole infrastructure on our own and provide an independent non-cryptographic signature. This has been installed to enable prototyping but does not limit future realisation Figure 19 shows a sample how different roles can be allocated to completely sign a model. In our case, we consider a domain-expert, a technical-expert or a knowledge-engineer, who assess the model according their expertise. The allocation of roles for assessments can be customized for the specific purpose of the application. Hence, all three aforementioned roles may be necessary to sign, or there are overruling signatures, or different roles are configured for signatures. This enables that responsible persons need to sign the models and hence can fulfill their duties in their daily tasks by ensuring that only approved models are executed on the robots.


4 sign the model with credentials
Log-in
Cancel
Continue
Modeler / Reviewer
2 select the model
3 retrieve model structure and image
1 retrieve model list
Model Signature Application
Model Signature
Select the ADOxx model to load:
Load ADOxx Model
Load External Model
Sign
Verify
In process
Powered by ADOxx
Robot Arm
Rest Robot Arm
Pick Object 1 from Box 1
Place Object 1 into Shopping Cart
Pick Object 2 from Box 2
Place Object 2 int Shopping Cart
Move and grab at fixed Position
Move and drop at fixed Position
Olive UI Workbench
Olive Microservice Controller
6 visualize the signatures and trust the model
Final User
5 load the model and verify the signatures

Username: model_owner

Password: ········

1.1 Supermarket - Fixed Binding (Business process diagram (E

1.1 Supermarket - Fixed Binding | 28.07.2020, 15:16:18

x= ...
y= ...
z= ...

Valid signatures for the model "1.1 Supermarket - Fixed Binding":

| | |
|---|---|
| Users: "**domain_advisor**" | Applied on "2020-08-10 17:33:49" |
| Users: "**model_owner**" | Applied on "2020-08-10 17:33:26" |
| Users: "**technical_advisor**" | Applied on "2020-08-10 17:33:37" |

**Figure 18 Model signing and certification procedure**

On the other side a final user, which may be an operator of machines, or which may be machines as well, can access the model repository and check, if all signatures are available. The verification service hence only approves, if a certain model

– which is sent via a textual representation – has been signed by authenticated signatories. In case the model has been changed after signing, the verification service responds a negative answer.

## 4.1 Requirement Elaboration

In the following we elaborate on the key requirements based on aforementioned scenario.

Any ADOxx model without restrictions concerning the modelling method or language can be selected, hence we realise a model-independent approach, where a BPMN Workflow can be assessed in the same manner as a Petri-Net or Flow-Chart model. An authentication system is necessary to enable the signature of models. A unique signature for each model must be generated and all of those signatures must be stored with connection to the respective model. This ensures that the model cannot be exchanged, or changed afterwards. Several signatures can be applied on one model. Finally, all signatures must be checked before the model is considered as "correctly signed". It has to be stressed that the signature only ensures that an expert who has the allocated signature role, approved the model. However, there are no further assurances than the signature that has been given. This means, if a modelling expert approves a corrupted model, the model is considered as correct as the modelling expert has approved it.

Table 2 presents a list of requirements including the essential input and output, as well as some relevant notes.

| Requirement | Input | Output | Notes |
|---|---|---|---|
| model list retrieval of all ADOxx models | no direct input | ADOxx model list including model name and ID | the ADOxx SOAP interface is used |
| model structure retrieval of ADOxx model | model ID | model structure in JSON format | the ADOxx SOAP interface is used |
| signing functionality | credentials, model ID, model version, model in BPMN / ADL format | confirmation message | use a REST service, in specific the POST method, for generating a SHA-256 hash of the provided file, LDAP and database details are provided by a configuration file, credentials as well as hash and version must be checked, safe the signature to the database |
| verification functionality | model in BPMN / ADL format | list of users that signed the model in JSON / XML format | use a REST service, in specific the POST method, for generating a SHA-256 hash of the provided file that can be checked with the details within the database |

**Table 2 Signature and certification application requirements**

In the following we explain the high-level architecture of the prototype implementation.

## 4.2 Signature and Certification Prototype Architecture

The signature and certification prototype architecture, depicted in Figure 19, was defined based on the identified requirements. It is composed of Olive components and the UI workbench.

Basically, we distinguish between UI widgets, Olive, a database and an authentication system. There are two UI widgets provided for (a) signing models and (b) checking the model signature. The Olive UI workbench is used to define and combine widgets and provide a homogeneous user interface that consists of several Micro Frontends.

The Olive microservice controller is used to orchestrate the individual services. The ADOxx soap server is used to establish an ADOxx connector and receive the models that need to be signed. MySQL is used to realise the database that keeps track of the signatures, and the LDAP authentication provides features of the authentication. As a storage for user authentical credentials ApacheDS is used as LDAP.

ADOxx serves as a modeling component by providing a SOAP API interface. A database acting as a signature repository is integrated as well as LDAP repository for the credentials. Basically, there are two major UI components. Those are 1) sign model and 2) check model signature. The foundation for using the UI components is a set of microservices that are targeted at the specific components. The Olive microservice collection serves as an intermediary between Workbench, database and ADOxx.

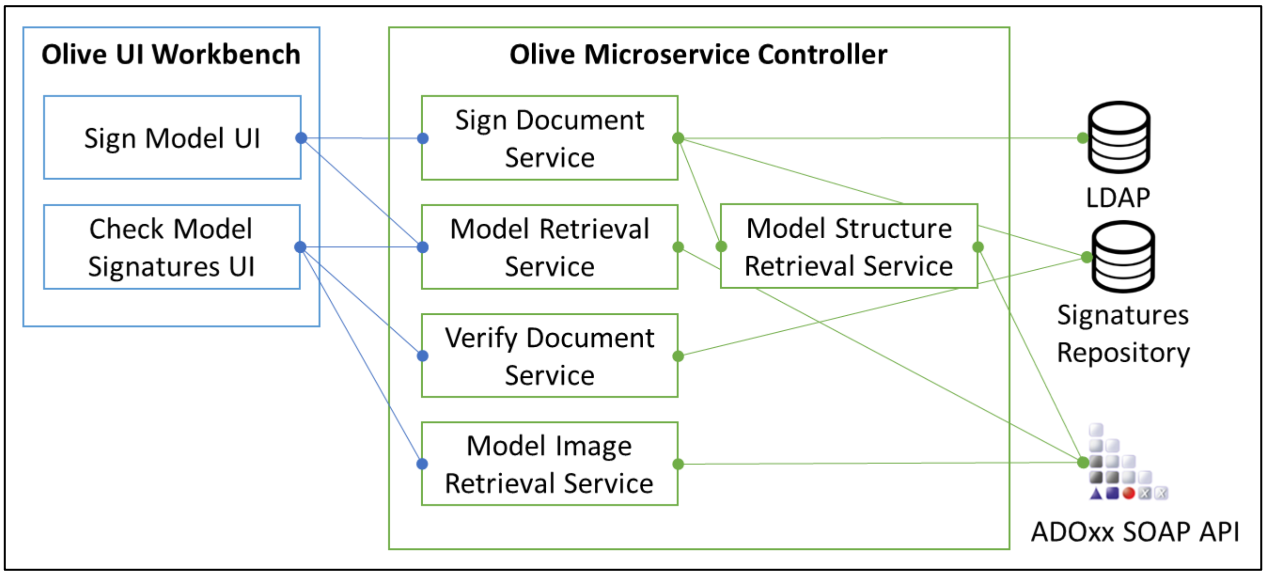
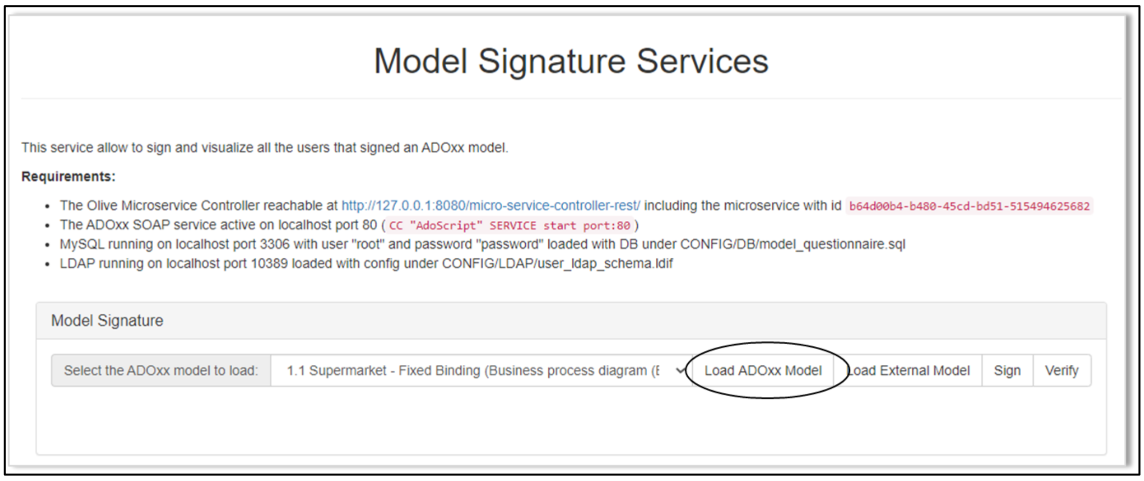


**Figure 19 Olive-based signature and certification architecture**

## 4.3 Introductory Sample using the Prototype

The signature service allows to load an ADOxx or an external model, as shown in Model signature service. In this introductory sample a BPMN process which is used for steering a robot via a fix-binded workflow as described in D3.2 is selected.



### Model Signature Services

This service allow to sign and visualize all the users that signed an ADOxx model.

**Requirements:**

- The Olive Microservice Controller reachable at http://127.0.0.1:8080/micro-service-controller-rest/ including the microservice with id `b64d00b4-b480-45cd-bd51-515494625682`
- The ADOxx SOAP service active on localhost port 80 ( `CC "AdoScript" SERVICE start port:80` )
- MySQL running on localhost port 3306 with user "root" and password "password" loaded with DB under CONFIG/DB/model_questionnaire.sql
- LDAP running on localhost port 10389 loaded with config under CONFIG/LDAP/user_ldap_schema.ldif

Model Signature

| Select the ADOxx model to load: | 1.1 Supermarket - Fixed Binding (Business process diagram (E | Load ADOxx Model | Load External Model | Sign | Verify |
|---|---|---|---|---|---|

**Figure 20 Model signature service**

After selecting, the model it is loaded and presented as in Figure 21, and it is ready to sign. For signing the model, the sign button is used, and the correct credentials related to the specific role of the user must be inserted. A short confirmation message confirms that the signing process worked.

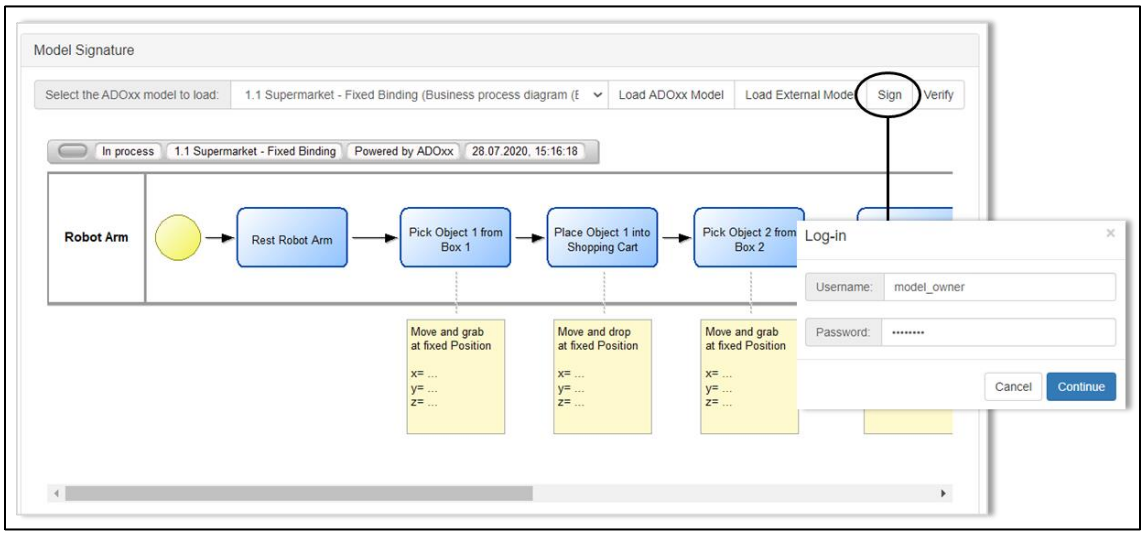
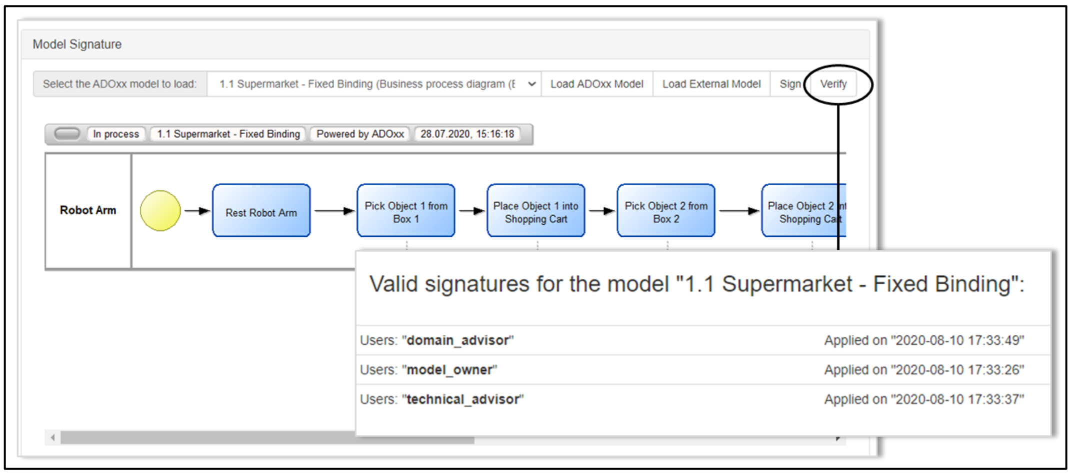

Model Signature
Select the ADOxx model to load:
1.1 Supermarket - Fixed Binding (Business process diagram (E
Load ADOxx Model
Load External Model
Sign
Verify
In process
1.1 Supermarket - Fixed Binding
Powered by ADOxx
28.07.2020, 15:16:18
Robot Arm
Rest Robot Arm
Pick Object 1 from Box 1
Place Object 1 into Shopping Cart
Pick Object 2 from Box 2
Move and grab at fixed Position x= ... y= ... z= ...
Move and drop at fixed Position x= ... y= ... z= ...
Move and grab at fixed Position x= ... y= ... z= ...
Log-in
Username: model_owner
Password: ••••••••
Cancel
Continue

**Figure 21 Sign a model**

To verify if a model has all signatures, either the verify button on the user interface – as depicted in Figure 22 – or by directly accessing the interface of the verification service from via a software invocation. The foreseen procedure is, that the workflow engine that executes the workflows on the robot platforms, is mounted in such a way, that it automatically accesses the verification service before workflow execution in order to ensure that only signed models are executed.

For demonstration purpose we provide also the user interface, that displays the different signatures – in our case the "domain advisor", the "model advisor" and the "technical advisor" – as well the corresponding time stamp. In case the model is changed, then the so-called hash-code of the model in combination with the signature does not match anymore, hence the signature will not be verified.


Model Signature
Select the ADOxx model to load:
1.1 Supermarket - Fixed Binding (Business process diagram (E
Load ADOxx Model
Load External Model
Sign
Verify
In process
1.1 Supermarket - Fixed Binding
Powered by ADOxx
28.07.2020, 15:16:18
Robot Arm
Rest Robot Arm
Pick Object 1 from Box 1
Place Object 1 into Shopping Cart
Pick Object 2 from Box 2
Place Object 2 into Shopping Cart

Valid signatures for the model "1.1 Supermarket - Fixed Binding":

| | |
|---|---|
| Users: "**domain_advisor**" | Applied on "2020-08-10 17:33:49" |
| Users: "**model_owner**" | Applied on "2020-08-10 17:33:26" |
| Users: "**technical_advisor**" | Applied on "2020-08-10 17:33:37" |

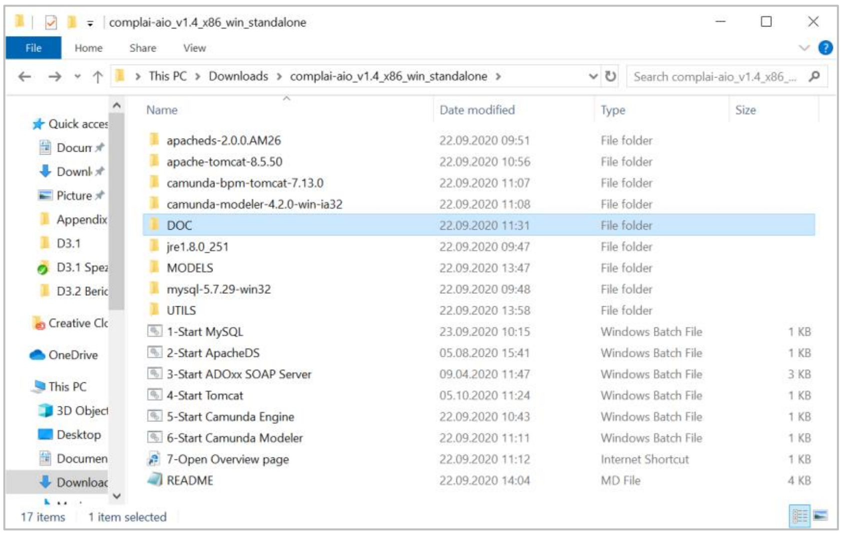
**Figure 22 Verify the signature**

# 5. Prototype Package

This chapter provides information on how to download and install the aforementioned prototypes.

## 5.1 Prerequisites

First, the prototype package must be downloaded from here: https://adoxx.org/live/web/complai/. Please note that the package must be unzipped with 7Zip[14]. The unzipped package includes libraries, models as well as executable files, see Figure 23 In general, users use the files in the DOC and the MODELS folder as well as the files numbered from 1 to 7.



| Name | Date modified | Type | Size |
|---|---|---|---|
| apacheds-2.0.0.AM26 | 22.09.2020 09:51 | File folder | |
| apache-tomcat-8.5.50 | 22.09.2020 10:56 | File folder | |
| camunda-bpm-tomcat-7.13.0 | 22.09.2020 11:07 | File folder | |
| camunda-modeler-4.2.0-win-ia32 | 22.09.2020 11:08 | File folder | |
| DOC | 22.09.2020 11:31 | File folder | |
| jre1.8.0_251 | 22.09.2020 09:47 | File folder | |
| MODELS | 22.09.2020 13:47 | File folder | |
| mysql-5.7.29-win32 | 22.09.2020 09:48 | File folder | |
| UTILS | 22.09.2020 13:58 | File folder | |
| 1-Start MySQL | 23.09.2020 10:15 | Windows Batch File | 1 KB |
| 2-Start ApacheDS | 05.08.2020 15:41 | Windows Batch File | 1 KB |
| 3-Start ADOxx SOAP Server | 09.04.2020 11:47 | Windows Batch File | 3 KB |
| 4-Start Tomcat | 05.10.2020 11:24 | Windows Batch File | 1 KB |
| 5-Start Camunda Engine | 22.09.2020 10:43 | Windows Batch File | 1 KB |
| 6-Start Camunda Modeler | 22.09.2020 11:11 | Windows Batch File | 1 KB |
| 7-Open Overview page | 22.09.2020 11:12 | Internet Shortcut | 1 KB |
| README | 22.09.2020 14:04 | MD File | 4 KB |

**Figure 23 Prototype package structure**

Furthermore, ADOxx must be downloaded and installed[15]. For this reason, a personal licence key is required that can be requested for free in the context of academic purposes at ADOxx.org. Afterwards the required modelling libraries - BPMN, Bee-Up and the Questionnaire library - are imported into the ADOxx development toolkit and the corresponding sample models may be imported in the ADOxx modelling toolkit. The BPMN library and models are available inside the package folder MODELS/BPMN. More details for Bee-Up and the standalone version can be found online[16]. The questionnaire library and the models are available inside the folder MODELS/QUESTIONNAIRES.

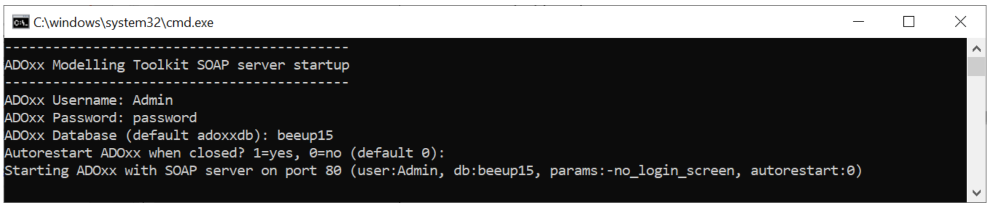
If you are not familiar with ADOxx, please check the following links for detailed instructions on importing libraries in the development toolkit (credentials for the user “Admin” and the password “password”), creating a new user and importing the models in ADOxx: import a library, create a user and import models.

[14] https://www.7-zip.org/ (last visited on 02/11/2020)
[15] https://www.adoxx.org/live/download-guided (last visited on 02/11/2020)
[16] https://austria.omilab.org/psm/content/bee-up/info (last visited on 02/11/2020)

## 5.2 Getting Started

In order to start the portal, following steps must be executed:

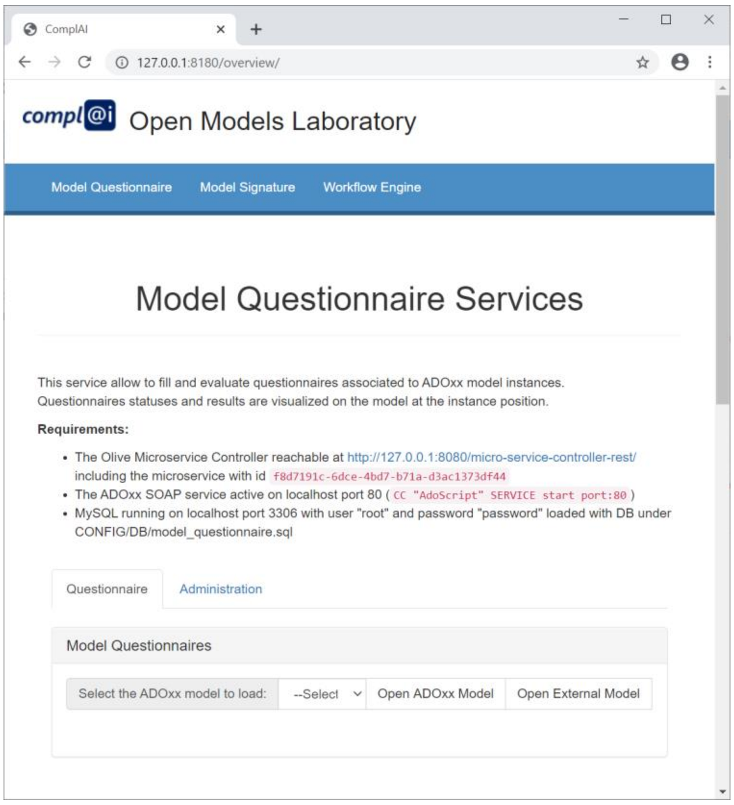
1. Execute the "1-Start MySQL.bat" file in the package folder
2. Execute the "2-Start ApacheDS.bat" file in the package folder
3. Execute the "3-Start ADOxx SOAP Server.bat" in the package folder
   a. insert the credentials for the created username and password for the ADOxx modelling toolkit, use the default database if you did not select a specific name during the ADOxx installation process.
   b. or "Admin", "password" and "beeup15" if you prefer the Bee-Up standalone version, see Figure 24
   c. ADOxx can be used to create questionnaire models as well as domain models



```
C:\windows\system32\cmd.exe
-------------------------------------------
ADOxx Modelling Toolkit SOAP server startup
-------------------------------------------
ADOxx Username: Admin
ADOxx Password: password
ADOxx Database (default adoxxdb): beeup15
Autorestart ADOxx when closed? 1=yes, 0=no (default 0):
Starting ADOxx with SOAP server on port 80 (user:Admin, db:beeup15, params:-no_login_screen, autorestart:0)
```

**Figure 24 Starting Bee-Up as standalone application**

4. Execute the "4-Start Tomcat.bat" file in the package folder
5. Execute the "5-Start Camunda Engine.bat" file in the package folder
   a. not explicitly required for the assistant system, integrated in this package as the workflow engine is needed for operating robots (explained in D3.2)
6. Execute the "6-Start Camunda Modeler.bat" file in the package folder
   a. not explicitly required for the assistant system, integrated in this package as the workflow engine is needed for operating robots (explained in D3.2)
7. Open the link "7-Open Overview page" in the package folder
   a. prefer Chrome for using the dashboard webpage
   b. go to the "Model Questionnaire" tab when using the assessment component, see Figure 25
   c. go to the "Model Signature" tab when using the signature and certification component, see Figure 26
   d. the created questionnaire and domain models are the foundation for using the questionnaire and signature service
   e. notice that the tab "Workflow Engine" is not needed in this prototype for an assistant system
8. Finally, when you are finished with assessment, signing and certification, terminate the execution closing the command console opened in point 1, 2, 3, 4, 5 and 6 (Ctrl+C) as well as the dashboard webpage



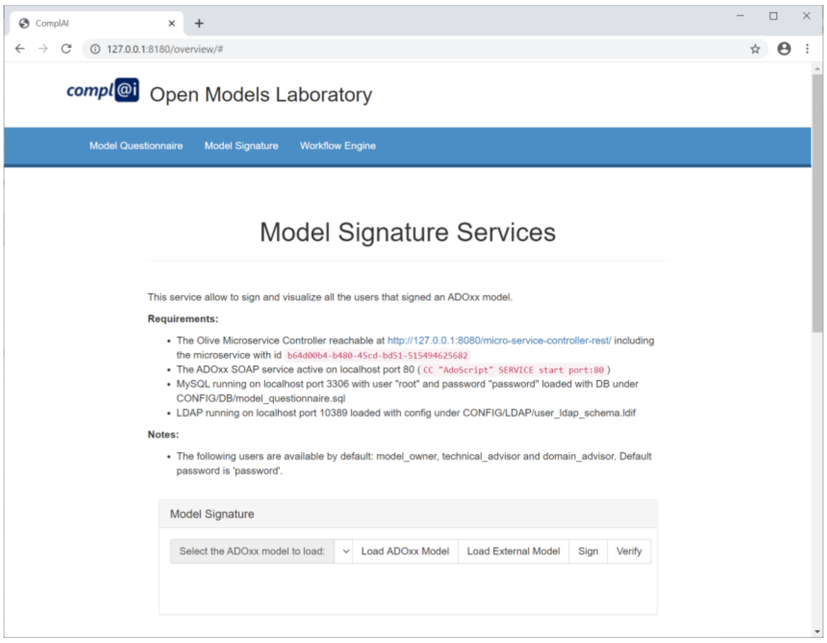
Model Questionnaire Services

This service allow to fill and evaluate questionnaires associated to ADOxx model instances.
Questionnaires statuses and results are visualized on the model at the instance position.

**Requirements:**

- The Olive Microservice Controller reachable at http://127.0.0.1:8080/micro-service-controller-rest/ including the microservice with id `f8d7191c-6dce-4bd7-b71a-d3ac1373df44`
- The ADOxx SOAP service active on localhost port 80 ( `CC "AdoScript" SERVICE start port:80` )
- MySQL running on localhost port 3306 with user "root" and password "password" loaded with DB under CONFIG/DB/model_questionnaire.sql

**Figure 25 Model questionnaire service**



Model Signature Services

This service allow to sign and visualize all the users that signed an ADOxx model.

**Requirements:**

- The Olive Microservice Controller reachable at http://127.0.0.1:8080/micro-service-controller-rest/ including the microservice with id `b64d00b4-b480-45cd-bd51-515494625682`
- The ADOxx SOAP service active on localhost port 80 ( `CC "AdoScript" SERVICE start port:80` )
- MySQL running on localhost port 3306 with user "root" and password "password" loaded with DB under CONFIG/DB/model_questionnaire.sql
- LDAP running on localhost port 10389 loaded with config under CONFIG/LDAP/user_ldap_schema.ldif

**Notes:**

- The following users are available by default: model_owner, technical_advisor and domain_advisor. Default password is 'password'.

**Figure 26 Model signature service**

# 6. Conclusion

The deliverable introduces the conceptual and technical architecture of the model-based assistant service as well as introduces how the prototype has been implemented during the project and how the available open-source prototype can be downloaded and used for proof-of-concept purpose.

The core idea is to link the assessment criteria – which are elaborated in more detail in D5.1 – with execution models – which are elaborated in more detail in D3.2.

The following pre-conditions have been considered:

- The development of the questionnaire has to be independent form the development of the execution models
- The approved execution models have to be digitally signed in such a way, that a signature can be verified but the model cannot be changed afterwards without noticing.

Two main components of the model-based assistant system are presented:

First, the proposed model-based assessment component deals with the transformation of textually written questions and answers like those listed in D5.1, which are transformed into a questionnaire model identifying different levels of questions and answers as well as different scoring calculations. The model-based assessment component deals therefore with the generation and evolution of questionnaires as well as with the actual assessment by human experts answering those questionnaires.

Second, the signature and certification component enable to sign models that have been successfully assessed by using a non-cryptographic digital signature using authentication of either different roles or persons following the widely used LDAP framework. Before using a model for robot execution, the model is verified by checking if all necessary signatures have been singed and therefore approved the model.

The Olive microservice framework has been used to connect and orchestrate the different components. This open-source framework can be extended to fit the provided prototypes in different environments.

Finally, the prototype package can be downloaded and tested including the sample models presented in this document. A hands-on instruction is provided to test the prototypes and to experiment with the concepts.