

COGITO

CONSTRUCTION PHASE
DIGITAL TWIN MODEL

cogito-project.eu

D6.4 –
Adaptive
Processes /
Workflow
Modelling and
Simulation-based
Optimisation
Module
v2



This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 955310

D6.4 – Adaptive Processes/Workflow Modelling and Simulation-based Optimisation Module v2

Dissemination Level:	Public
Deliverable Type:	Demonstrator
Lead Partner:	BOC-AG
Contributing Partners:	UEDIN, QUE, NT
Due date:	31-10-2022
Actual submission date:	27-10-2022

Authors

Name	Beneficiary	Email
Robert Woitsch	BOC-AG	robert.woitsch@boc-group.com
Damiano Falcioni	BOC-AG	damiano.falcioni@boc-group.com
Anna Sumereder	BOC-AG	anna.sumereder@boc-group.com
Amy Wilson	UEDIN	amy.l.wilson@ed.ac.uk
Gail Robertson	UEDIN	gail.robertson@ed.ac.uk
Zohreh Kaheh	UEDIN	zkaheh@exseed.ed.ac.uk
Chris Dent	UEDIN	chris.dent@ed.ac.uk
Panagiotis Moraitis	QUE	p.moraitis@que-tech.com
Martin Straka	NT	straka@novitechgroup.sk

Reviewers

Name	Beneficiary	Email
Jochen Teizer	DTU	teizerj@byg.dtu.dk
Athanasios Tsakiris	CERTH	atsakir@iti.gr

Version History

Version	Editors	Date	Comment
0.1	BOC-AG	05.08.2022	ToC
0.2	BOC-AG	12.09.2022	Process Modelling Section Update
0.3	BOC-AG	19.09.2022	Sandbox Experiment Section Update
0.4	BOC-AG	26.09.2022	Simulation Sections Update
0.5	BOC-AG	03.10.2022	Integration Sections Update
0.6	UEDIN, QUE, NT	06.10.2022	Optimisation, SLA, Integration Sections Update
0.7	DTU, CERTH	10.10.2022	Internal review
0.9	BOC-AG, UEDIN	19.10.2022	Internal review comments addressed
1.0	BOC-AG, Hypertech	27.10.2022	Submission to the EC Portal

Disclaimer

©COGITO Consortium Partners. All right reserved. COGITO is a HORIZON2020 Project supported by the European Commission under Grant Agreement No. 958310. The document is proprietary of the COGITO consortium members. No copying or distributing, in any form or by any means, is allowed without the prior written agreement of the owner of the property rights. The information in this document is subject to change without notice. Company or product names mentioned in this document may be trademarks or registered trademarks of their respective companies. The information and views set out in this publication are those of the author(s) and do not necessarily reflect the official opinion of the European Communities. Neither the European Union institutions and bodies nor any person acting on their behalf may be held responsible for the use, which may be made, of the information contained therein.

Executive Summary

The current document demonstrates the final version of the COGITO Process Modelling and Simulation based Optimisation (PMS) modules, an ecosystem of applications/microservices provided in the form of Software as a Service (SaaS). The module enables the definition of processes in the construction domain, eased by applying a template-oriented approach towards reusability and simplification. At the same time the continuous process optimisation and refinement based on forward-looking simulation and real-time execution monitoring is facilitated. The COGITO PMS module provides its spectrum of features through different components.

The Modelling component – allowing the definition of a particular construction process –, offers an environment where all related resources and metrics can be represented. Among those are the associated Key Performance Indicators (KPIs), the needed resources in form of equipment and workers as well as the linkage to information available from Building Information Models (BIM) data that is derived from the COGITO Digital Twin Platform (DTP). The processes created in this component are then exported and sent to the COGITO Workflow Management Automation Tools (WODM) module that will take care of their execution.

The developed process models provide several benefits such as the systematic identification of security issues and KPIs. In addition, the modelling of the construction-related processes is leveraged by a physical Sandbox Experiment component where the stakeholders can (a) collaboratively reason over the process via design thinking¹, (b) test the required digitisation strategies in an isolated and safe environment and (c) execute the process in an abstracted version of the construction site that can generate sample log data; the latter is used to test other PMS components at an early stage of the project when real execution logs are usually unavailable.

Before releasing the construction workflow for actual, on-site execution, algorithms are introduced on top of the process – such as means of simulation – to check conformance with expected times and costs, analyse and detect systemic inconsistencies in the process model that could prevent it from reflecting the real process or potentially block the execution workflow. In order to better tune the simulation parameters, real-time data from the Sandbox Experiment is used when there is none from the actual process execution so far.

Based on the results of the simulation, the Optimisation component will try to minimise time and cost by reorganizing the process, redistributing workers and equipment. The selected optimisation result can then be reflected in the process model for its finalisation.

The interactions between the different components and required services are handled by the Microservice Integration component, a microservice-based framework integrated in the ADOxx platform where the Modelling component is built on, supporting a low-code definition of microservices working with process model information. This component will be utilised to enable the data exchange across the different PMS internal components and the other COGITO modules, e.g., the Digital Twin platform and the workflow execution engine WODM.

All components included in the PMS are demonstrated in this document using an integration test use case and a pre-validation site use case that are relevant also within the scope of the final pilot sites, i.e., related to the construction of a railway track. Relevant use case models were defined in the Modelling component, presented, and tested in the physical Sandbox Experiment component after which they were simulated and optimised respectively in the Simulation and the Optimisation components, supported by microservices created in the Microservice Integration component.

¹ <https://www.omilab.org/nodes/design-thinking.html>

Table of Contents

Executive Summary.....	3
Table of Contents.....	4
List of Figures.....	6
List of Tables.....	8
List of Acronyms	9
1 Introduction.....	10
1.1 Scope and Objectives of the Deliverable	10
1.2 Relation to other Tasks and Deliverables.....	11
1.3 Structure of the Deliverable	12
1.4 Updates to the first version of the PMS	12
2 PMS – Modelling Component.....	13
2.1 Prototype Overview	13
2.1.1 Process Modelling Environment.....	13
2.1.2 KPIs Modelling Environment	18
2.2 Technology Stack and Implementation Tools	21
2.3 Input, Output and API Documentation.....	22
2.4 Application Example	22
2.4.1 School Sample Use Case.....	22
2.4.2 Railway Construction Use Case.....	25
2.5 Licensing.....	27
2.6 Installation Instructions.....	27
2.7 Development and integration status.....	28
2.8 Requirements Coverage.....	28
2.9 Assumptions and Restrictions	31
3 PMS – Sandbox Experiment Component.....	32
3.1 Prototype Overview	32
3.1.1 Architecture.....	35
3.2 Technology Stack and Implementation Tools	35
3.3 Input, Output and API Documentation.....	36
3.4 Application Example	36
3.5 Licensing.....	39
3.6 Installation Instructions.....	39
3.7 Development and integration status.....	40
3.8 Requirements Coverage.....	40
3.9 Assumptions and Restrictions	41
4 PMS – Simulation Component.....	42

4.1	Prototype Overview	42
4.1.1	Architecture	45
4.2	Technology Stack and Implementation Tools	46
4.3	Input, Output and API Documentation.....	46
4.4	Application Example	46
4.5	Licensing.....	48
4.6	Installation Instructions.....	48
4.7	Development and integration status.....	48
4.8	Requirements Coverage.....	48
4.9	Assumptions and Restrictions	50
5	PMS – Optimisation Component	51
5.1	Prototype Overview	51
5.2	Technology Stack and Implementation Tools	55
5.3	Input, Output and API Documentation.....	55
5.4	Application Example	56
5.5	Licensing.....	61
5.6	Installation Instructions.....	61
5.7	Development and integration status.....	62
5.8	Requirements Coverage.....	62
5.9	Assumptions and Restrictions	64
6	PMS – Microservice Integration Component	65
6.1	Prototype Overview	65
6.1.1	Architecture.....	65
6.2	Technology Stack and Implementation Tools	66
6.3	Input, Output and API Documentation.....	67
6.4	Application Example	67
6.5	Licensing.....	68
6.6	Installation Instructions.....	69
6.7	Development and integration status.....	69
6.8	Requirements Coverage.....	69
6.9	Assumptions and Restrictions	71
7	Conclusions	72
	References.....	73

List of Figures

Figure 1 - PMS Architecture	10
Figure 2 - Process Modelling Environment main interface.....	14
Figure 3 - Process Modelling Environment canvas.....	14
Figure 4 - Locally Installed Process Modelling Environment	15

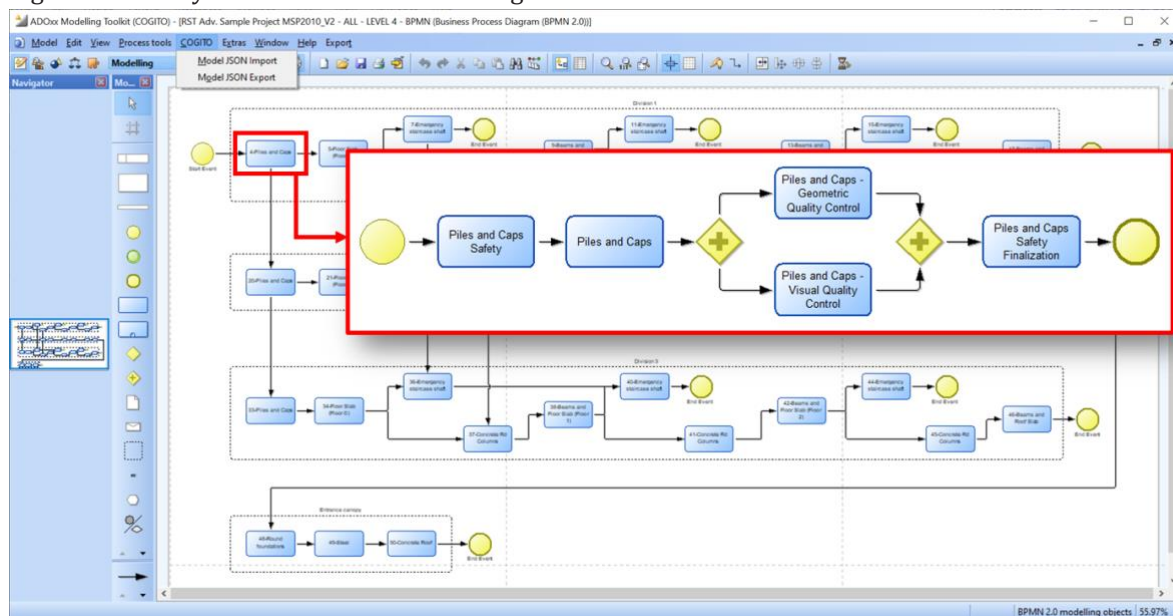


Figure 5 – Extension of Imported Model with Construction Relevant Tasks.....	16
Figure 6 – Extension of the Task Object in BPMN.....	17
Figure 7 – SLA Model Type and SLA Objects.....	18
Figure 8 – Resource Model Type and Resource Objects.....	18
Figure 9 - KPI Model Sample.....	19
Figure 10 - Goal (Left) and KPI (right) attributes.....	20
Figure 11 - KPI Data Calculation Model Sample	20
Figure 12 - Data (Left) and Metric (Right) attributes	21
Figure 13 - BPMN Model for School Sample.....	23
Figure 14 - Extended Process Model with Safety and Control Subprocesses	23
Figure 15 - Parallel Process Path	24
Figure 16 - KPIs identified following the Rules	24
Figure 17 - Railway Track Construction Template Process.....	25
Figure 18 - Railway Track Construction Process Instances.....	26
Figure 19 - Railway Track Construction Process Workflow.....	26
Figure 20 - Railway Track Construction KPI Models	27
Figure 21 - Impressions from a Real-World Track Construction Scenario at Pre-Validation Site.....	32
Figure 22 - OMILAB Innovation Corner Conceptual Perspective (left) and Real Perspective (right)	33
Figure 23 - Steps towards Designing the Physical Experiment	34
Figure 24 - Experiment Architecture.....	35
Figure 25 - Basic Workflow corresponding to the Experiment	36
Figure 26 - Real World vs. Experiment Setting.....	37
Figure 27 - Physical Experiment Execution Sample.....	37
Figure 28 - Construction Experiment Entry Page.....	38
Figure 29 - Equipment and Worker Logging.....	38
Figure 30 - Execution Logs.....	39
Figure 31 - Simulation Mechanisms for Multi-Distributions.....	42
Figure 32 - Process Task List for Simulation.....	43

Figure 33 - Calculation of weighted Time considering Risks	43
Figure 34 - Simulation General Results	44
Figure 35 - Simulation Detailed Results	44
Figure 36 - Simulation Engine Architecture.....	45
Figure 37 - Introduce Wait and Check Tasks including a Decision	47
Figure 38 - Introduce Parallelism.....	47
Figure 39 - Transformed Process Model compliant with BPMN	47
Figure 40 - Custom Simulation Page	48
Figure 41 - Total Duration for different values of α (for TW=100 m ²).....	61
Figure 42 - Total Cost for different values of α (for TW=100 m ²)	61
Figure 43 - OLIVE Microservice Controller Architecture.....	66
Figure 44 - Microservice Management Page	68

List of Tables

Table 1 - Libraries and Technologies used in the PMS Modelling component	21
Table 2 - Requirements on Computing Systems for the PMS Modelling component	28
Table 3 - Requirements about Workflow Planning, Execution and Monitoring for the PMS Modelling component	29
Table 4 - Functional, Non-Functional Requirements and Interfaces for the PMS Modelling component	30
Table 5 - Libraries and Technologies used in the <i>PMS Sandbox Experiment component</i>	36
Table 6 - Requirements on Computing Systems for the PMS Sandbox Experiment component	40
Table 7 - Requirements about Workflow Planning, Execution and Monitoring for the PMS Sandbox Experiment component	40
Table 8 - Libraries and Technologies used in the PMS Simulation Component	46
Table 9 - Requirements on Computing Systems for the PMS Simulation component	49
Table 10 - Requirements about Workflow Planning, Execution and Monitoring for the PMS Simulation component	49
Table 11 - Functional, Non-Functional Requirements and Interfaces for the PMS Simulation component	50
Table 12 - Libraries and Technologies used in the PMS Optimisation component	55
Table 13 - Available workforce and their cost	56
Table 14 - Type of workers required for each activity	56
Table 15 - Productivity of workers for each activity (m ² per h)	57
Table 16 - Maximum available workers per type for each activity	58
Table 17 - Total duration (h) for each activity carried out by different types of workers (where $w_1=0.8$ and $w_2=0.2$)	58
Table 18 - Number of workers per type for each activity (where $w_1=0.8$ and $w_2=0.2$)	59
Table 19 - Total duration (h) for each activity carried out by different types of workers (where $w_1=0.2$ and $w_2=0.8$)	59
Table 20 - Number of workers per type for each activity (where $w_1=0.2$ and $w_2=0.8$)	60
Table 21 - Requirements on Computing Systems for PMS Optimisation component	62
Table 22 - Requirements about Workflow Planning, Execution and Monitoring for the PMS Optimisation component	63
Table 23 - Functional, Non-Functional Requirements and Interfaces for the PMS Optimisation component	63
Table 24 - Libraries and Technologies used in <i>PMS Microservice Integration component</i>	67
Table 25 - Requirements on Computing Systems for the PMS Microservice Integration component	69
Table 26 - Requirements about Workflow Planning, Execution and Monitoring for the PMS Microservice Integration component	70
Table 27 - Functional, Non-Functional Requirements and Interfaces for the PMS Microservice Integration component	70

List of Acronyms

Term	Description
BIM	Building Information Model
BPMN	Business Process Modelling Notation
COGITO	Construction Phase diGItal Twin mOdel
DTP	Digital Twin Platform
GAMS	General Algebraic Modelling System
H&S	Health and Safety
IoT	Internet of Things
KPIs	Key Performance Indicators
MINLP	Mixed Integer Nonlinear Programming
OLIVE	OmiLab Integrated Virtual Environment
OMiLAB	Open Model Initiative Laboratory
PMS	Process Modelling and Simulation
SaaS	Software as a Service
SAML	Security Assertion Markup Language
SLA	Service Level Agreement
WODM	Workflow Management Automation Tools

1 Introduction

1.1 Scope and Objectives of the Deliverable

This deliverable demonstrates the final version of the COGITO Process and Workflow Modelling and Simulation-based optimisation (PMS) module. The PMS module allows to define, simulate, and optimise both the construction business process model as well as the operative workflow model. This gives the possibility to identify process steps that are critical for the successful implementation of the project. Processing capabilities such as optimization algorithms are provided to minimize time and costs KPIs, before and during the construction project execution. The combination with real-time data from the process execution and workers' feedback through the COGITO Workflow Management Automation Tools (WODM) module allows to adapt the simulation model to the actual process occurring at the construction site, thus improving the accuracy of the decision support instruments.

In the PMS module the project manager defines the construction process starting from a list of available templates enriching it with information coming from the COGITO Digital Twin Platform (DTP), namely equipment/team availability and Building Information Model (BIM) data. The construction process model is refined through feedback from the Simulation and Optimisation components and upon finalisation it is released to the COGITO WODM module for execution at the construction site. Here, real-time data and additional feedback from workers are collected and used to recalibrate the simulation inputs and to continuously improve the construction process model.

The COGITO PMS module is released as an ecosystem of different components each providing a specific feature and cooperating with the others through a specific integration component. Figure 1 provides a high-level overview of this ecosystem.

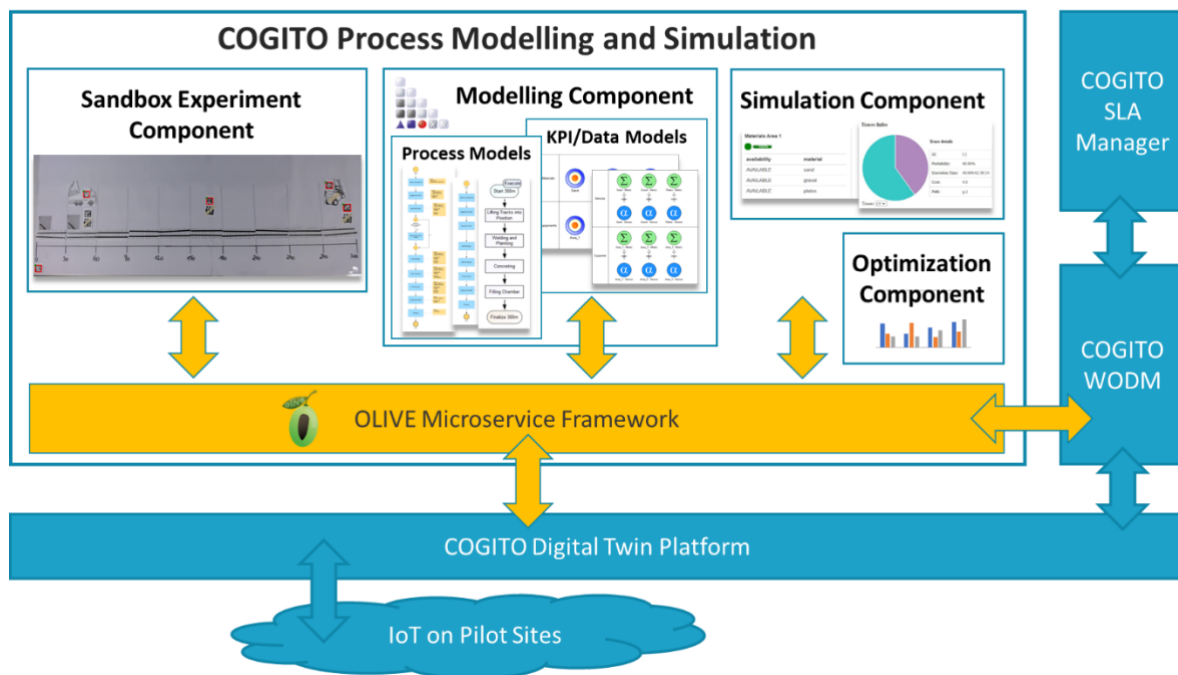


Figure 1 - PMS Architecture

The Modelling component of the PMS introduces the meta-models and the tools for modelling the construction processes; it is created around ADOxx, a meta-modelling platform freely available for academic use via the world-wide community ADOxx.org². As such, most of the COGITO prototypes introduced in this deliverable are therefore released as open use in this community. The ADOxx platform enables the definition of domain-specific meta-models and the subsequent generation of a modelling toolkit specific for

² <https://www.adoxx.org/>

the defined meta-models. In our case we adapted the meta-model of the BPMN (Business Process Modelling Notation) modelling language to also include Key Performance Indicators (KPIs) and other aspects relevant to the construction domain, i.e., equipment, teams and BIM data. Models span across different abstraction layers – ranging from high-level template processes over specific process instances to executable workflows – providing the foundation for advanced mechanisms such as the simulation and the optimisation of the process models explained in the following sections. The domain complexity is divided into business processes dealing with stakeholders and process tasks, as well as into workflows capturing the technical details of the data exchange and the execution. In order to simulate and optimise the overall construction processes, different levels of model abstraction are used to investigate several options.

To collaborate and reason over the construction process, a physical Sandbox Experiment component is available as an enhancement to the Modelling component, focused on a representation of the use case in a design thinking approach³. The experiment is also used in order to have a sandbox environment to evaluate the possibilities that digitisation patterns (like the tracking of equipment) may enable; upon identification of an interesting pattern, it allows to test out different digitisation technologies supporting this pattern (like image recognition or location tracking based on markers). This last step is particularly useful to generate production-like data (e.g., the location of equipment and workers) at an early stage of the project when such data is usually unavailable, nonetheless, useful to test other PMS components such as the Simulation and Optimisation components.

The Simulation component is demonstrated using the process model information obtained from the Modelling component combined with historical execution logs from the Sandbox Experiment component in order to retrieve average execution times for each process activity. Using the simulation, the times and costs of the processes can be estimated, evaluating the different paths in the process and checking design problems that will cause errors upon execution in the workflow engine. Parallel to the simulation, the Optimisation component applies mathematical models to the process model information in order to evaluate different schedule alternatives which are valid for the process and would ultimately optimise times and resources. Additional functionalities like the KPI evaluation dashboard are demonstrated in the context of the Sandbox Experiment component using a microservice created in the Microservice Integration component.

The Microservice Integration component is based on the framework OLIVE⁴, on which all the services required by each demonstrated component have been defined. Such services can be

- (a) specific for the components, like the markers' location storage and retrieval for the Sandbox Experiment and the calculation of average execution time and weather-related delays for each task in the Simulation component, or
- (b) generic for the PMS interconnection with the other COGITO modules – in particular with the COGITO WODM (in order to send the workflow to be executed after being extracted from the Modelling component and to receive real-time data to evaluate the defined KPIs) and with the COGITO Digital Twin Platform (in order to retrieve as-planned 4D-BIM data, as well as resources location, quality and health and safety issues related to the task being executed).

For each component of the COGITO PMS module the technical details are reported as well as a demonstration using a use case common to all COGITO modules (used as testing scenario of modules interconnection capabilities), concerning the construction of a school building, for which BIM data and schedules are available. Additionally, for some modules, a demonstration using available data of the pre-validation site use case about railway track construction is provided.

1.2 Relation to other Tasks and Deliverables

This deliverable is the final version of the PMS demonstrator building upon D6.3.

³ <https://www.omilab.org/nodes/design-thinking.html>

⁴ <https://www.adoxx.org/live/olive>

The data produced by the PMS component described in this deliverable will be consumed by the COGITO WODM module for the workflow execution as described in D6.5 of Task 6.3 and the generated log data will be used to evaluate KPIs. Additionally, the SLA modelling feature of the PMS modelling component is based on the concepts defined in D6.1 of Task 6.1.

The interactions between the components described in this deliverable depend on the COGITO use cases defined in Task 2.4 and described in the final version of the COGITO architecture in D2.5. Additionally, the stakeholder requirements described in Task 2.1 are considered for the creation of each component besides the functional and non-functional requirements described in D2.5.

Finally, the exchanged data between the PMS and other COGITO modules is aligned with the COGITO ontology as defined in D3.3 of Task 3.2.

1.3 Structure of the Deliverable

The deliverable devotes one chapter to each of the COGITO PMS module's components. All chapters share the same structure, starting from an overview of the component, followed by the implementation details (in terms of technology stack and APIs) and concluding on the use of the component having as sample the same use case. Finally, the developing and integration status will provide details on the features planned to be released in the second version of each component. The license type of the released application, installation instructions, and any applied assumptions and/or restrictions are provided when applicable. Chapter 2 demonstrates the Modelling component of the PMS module; all the models created for the exemplarily defined construction sample are presented and explained. Chapter 3 describes the Sandbox Experiment component, an extension of the Modelling component with focus on design thinking, use case reasoning and digitisation approaches testing. Chapter 4 introduces the Simulation component where the construction process is simulated in terms of times and costs. Chapter 5 describes the Optimisation component, used to evaluate possible alternatives in terms of the process schedule that may reduce costs and times. Finally, Chapter 6 demonstrates the Microservice Integration component, used to generate all the microservices enabling the communication between the different PMS components and other COGITO modules.

1.4 Updates to the first version of the PMS

This document extends the D6.3 "Adaptive Processes / Workflow Modelling and Simulation-based Optimisation Module v2" with the last improvements in the COGITO PMS components as in the following:

- Introduction of the community version of the PMS Modelling component including support for construction specific attributes and model types.
- Integration in the PMS Modelling component of import/export functionalities from/to the COGITO DTP.
- Alignment of the application scenario for the PMS Modelling, Sandbox Experiment, Simulation, and Optimization components with the use case of the school construction process, as a common demonstration scenario between all the COGITO modules.
- Update of the pre-validation site use case in the application scenario of the PMS Modelling component.
- Introduction of rules for identification of KPIs in the PMS Modelling component.
- Introduction of rules to align with the BPMN semantic the processes imported from the COGITO DTP in the PMS Modelling component, in order to be valid for the PMS Simulation component.
- Improvement in the algorithm of PMS Optimization component using piecewise linear simplification that speeds up computations and will potentially enable further extensions to the model.
- Addition of equipment to the PMS Optimisation component model description and discussion of how this might be implemented in practice.
- Introduction of microservices for the execution of the PMS Optimization component and its integration with the PMS Modelling component.
- Introduction of microservices for the integration of the PMS components with the different COGITO modules, in particular the DTP and WODM.

2 PMS – Modelling Component

This chapter demonstrates the final version of the Modelling component part of the PMS module, used to support the creation of process models specific to the construction domain, including additional information about the relevant KPIs and business goals, the schedule of the needed resources (in terms of workers and equipment) for each process activity, and the relations with BIM and health and safety data retrieved from the COGITO DTP. In particular, the usage of process templates facilitates and accelerates the process modelling by building upon pre-defined processes that can be altered according to a specific construction process. Such process models, after refinement through feedback from the Simulation and Optimisation components, are delivered to the COGITO WODM module for execution on the construction site, where data about execution status and additional feedback from workers (like notes and pictures) are collected and used to continuously improve the construction process model – i.e. assigning additional resources or updating the activities schedule.

2.1 Prototype Overview

BOC's ADOxx meta-modelling platform⁵ serves as a foundation for the Modelling component of the COGITO PMS. In order to capture the relevant aspects of a construction process, the BPMN model type is used as foundation, extended with KPI information and attributes specific to the construction domain (such as equipment required, workers available and BIM details). In particular, the following specialised modelling environments are introduced:

- **Process Modelling Environment:** The process environment is subdivided in an online cloud version and a locally installed extended BPMN version. The online modelling environment is based on the ADONIS Community Edition and follows the OMG⁶ standard for BPMN 2.0. It is used for modelling and retrieving the overall construction process as well as for processing templates, instances and workflows relevant to the portrayal of major construction tasks. In addition, there is the local process modelling environment based on the ADOxx meta modelling platform – free for research purposes – where the BPMN was extended to fulfil requirements related to the construction domain and the interconnection with other COGITO modules.
- **KPI Modelling Environment:** The KPI modelling environment allows for the definition of goals related to business challenges – in this case in the construction domain – specify them with targeted KPIs and describe the related data assets. A relationship between the construction processes and the KPIs may be created through a mapping between the corresponding models created in the modelling environments. Furthermore, models from the KPI modelling environment serve as a basis for the creation of monitoring dashboards presenting the status of the defined KPIs.

In the following sections and subsections, more details for each modelling environment are provided.

2.1.1 Process Modelling Environment

2.1.1.1 Cloud based modelling environment based on ADONIS Platform

The construction Process Modelling environment is provided in the form of a cloud-based solution based on the community version of the ADONIS platform, a framework built on the ADOxx meta-modelling platform, compliant with the BPMN2.0 standard, allowing to abstract the construction process at many levels. In particular, the construction processes are first designed in the form of generic templates valid for different use cases in the construction domain. Then a specific instance of the construction process for the needed use case is refined, applying use case-specific choices that can be evaluated at process design time. Finally, an executable workflow is created from the instantiated construction process, adding details on the roles required for the different tasks, the services that need to be executed for each automatic task of the process and the KPIs associated with each activity.

⁵ <https://www.adoxx.org/live/home>

⁶ <https://www.omg.org/bpmn/>

The solution provided is a full business process management suite with capabilities not only related to the creation of models (“Design & Document” section in Figure 2) but also allowing to manage their release cycle (“Control & Release” section in Figure 2) and check their correctness (“Read & Explore” section in Figure 2). In this demonstration we will focus on the design features of the construction process templates, instances and workflow models in the BPMN2.0 standard model-type (Design & Document” section in Figure 2).



Figure 2 - Process Modelling Environment main interface

In the “Design & Document” section of the Process Modelling environment it is possible to explore all existing models in a folder-tree view, visualise them and create new ones using the BPMN2.0 modelling canvas (Figure 3). Here the interface allows to create all BPMN objects, with some assistive features like next object and connectors suggestions or automatic alignment. The tool provides export features in the BPMN2.0 standard format as well as the generation of images and reports with details of the model objects.

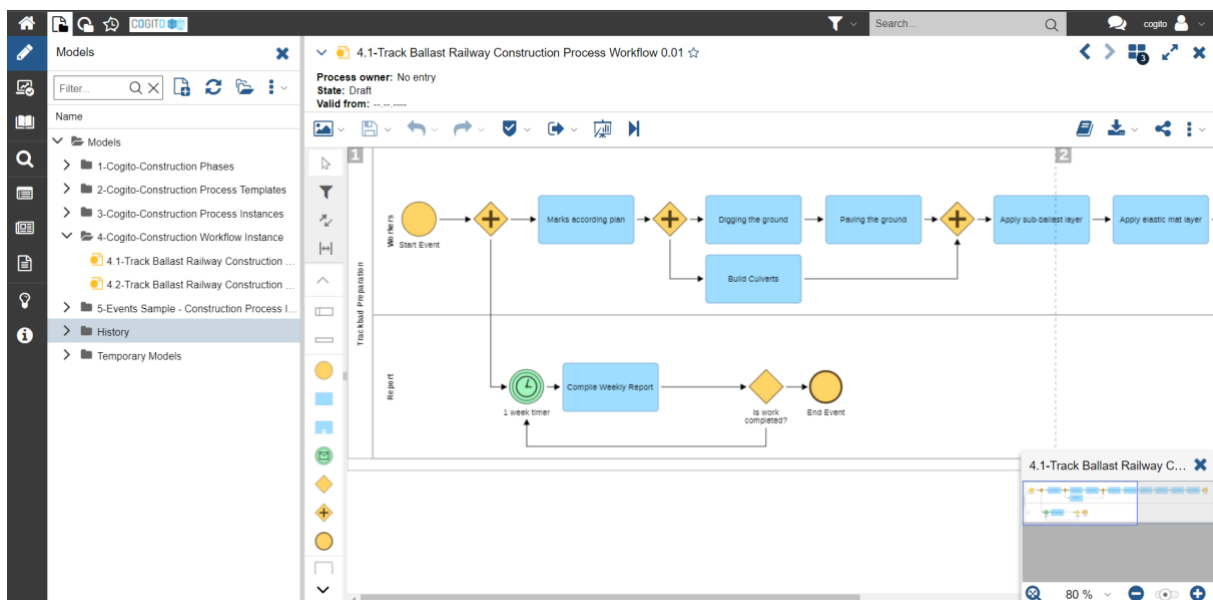


Figure 3 - Process Modelling Environment canvas

In addition to the standard BPMN2.0 attributes, each object in the process model also has the possibility to include some additional attributes relevant within the scope of the construction domain. In particular, it is possible to associate the resources representing specific equipment and the responsible roles for the

activity execution to a BPMN task. References to the Goals and KPIs specified in the KPI modelling environment described in Section 2.1.2 are also available in each BPMN task.

2.1.1.2 Local modelling environment based on community version of the ADOxx Platform

In addition to the cloud based modelling environment a local environment based on the ADOxx Bee-Up Modelling tool⁷ has been released and made freely available to the ADOxx Community for research purposes and extensions. Bee-Up is a hybrid modelling tool that encompasses five different modelling languages (BPMN, EPC, ER, UML, and Petri Nets) and several auxiliary languages (Flowcharts, etc.). For this construction process modelling environment, the BPMN model type is used and adapted to support the model exchange format of the COGITO Digital Twin Platform and the additional attributes introduced in COGITO and relevant for the construction domain.

In particular, the modelling environment support import and export functions in the JSON format required by the COGITO DTP, through the COGITO specific menu. The JSON Import functionality generate a BPMN process from the construction project initial schedule information returned by the COGITO DTP in JSON format (Figure 4). After refinements with resources, BIM elements, and spaces allocation, and the introduction of safety and quality control activities for each task, the process can be exported in the same JSON format required by the COGITO DTP in order to be further processed by other COGITO modules like the execution engine WODM.

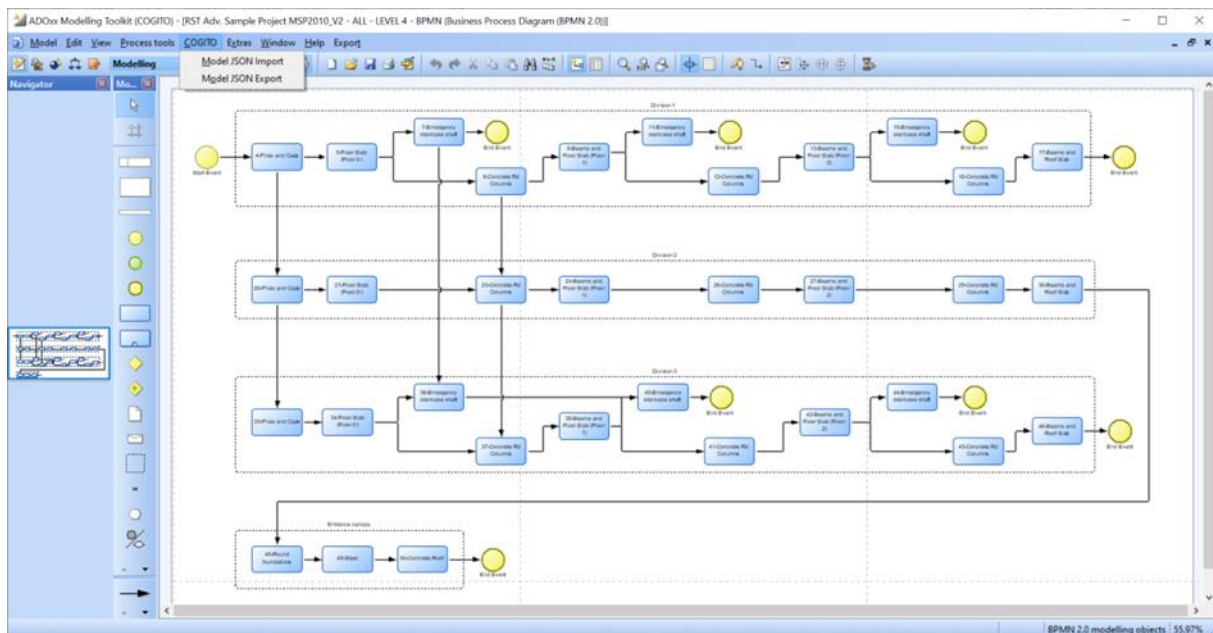


Figure 4 - Locally Installed Process Modelling Environment

Building upon the imported processes, each of the tasks is refined by introducing corresponding template including safety and quality control tasks. This means that one task is specified in more detail by dividing it in four tasks – for instance by using a subprocess as shown in **Figure 5**. At the beginning of each construction task, specific safety related activities are performed (like the installation of protection barriers), then the construction task is executed and afterwards quality control with a focus on visual and geometric quality control is executed. At the end the applied safety measures are remove from the working environment.

⁷ <https://bee-up.omilab.org/activities/bee-up/>

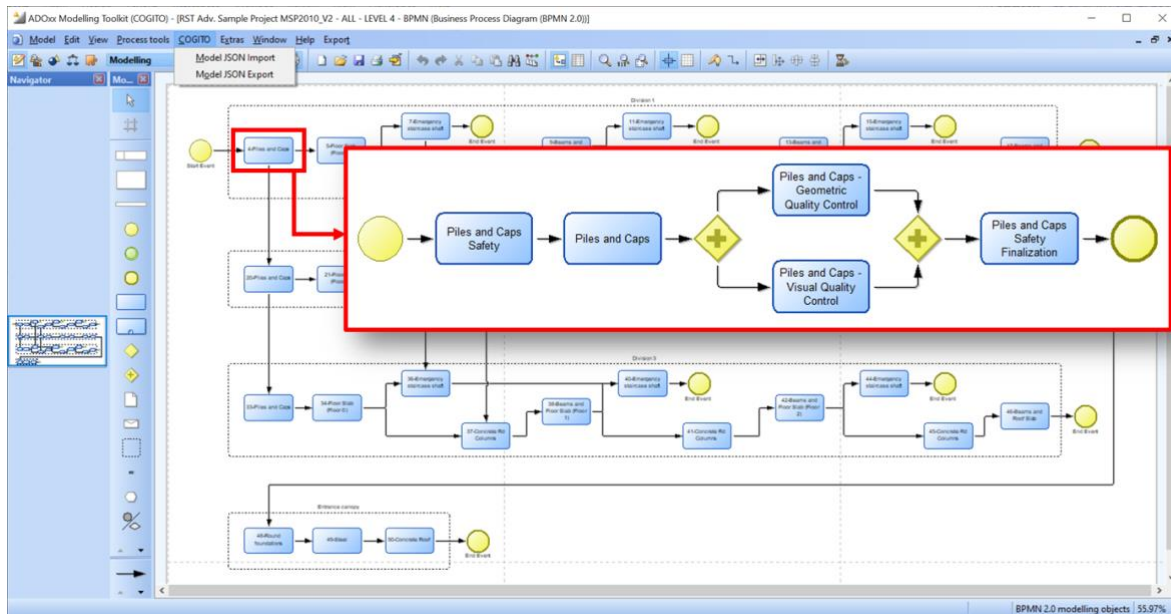


Figure 5 – Extension of Imported Model with Construction Relevant Tasks

Several new attributes for the **BPMN task objects** (see Figure 6) were introduced so that the characteristics of construction processes could be better depicted. These additions were received by the COGITO DTP input. Among the new attributes were COGITO project specific task IDs, task types, starting/end date, resources depicted in corresponding Resource models, location tracking markers, as well as health and safety indicators. In particular, following attributes were introduced for each BPMN task:

- Task Id (string): Id of the task as provided by the COGITO DTP.
- COGITO Task Type (enumeration: normal task, geometric quality control, visual quality control, safety): the COGITO specific task type used to distinguish between working related, control related, and safety related tasks.
- Start Date (date and time): task start time from schedule.
- End Date (date and time): task end time from schedule.
- Resources (record table)
 - Resource (object reference to a resource object): reference to the specific resource needed for the task.
 - Quantity (integer): number of resources needed.
- Zones (record table):
 - Zone Name (string): Name of the BIM zones involved in the task.
 - Description (string): Description of the BIM zones involved in the task.
- Elements (record table):
 - ID (string): BIM element id associated with the task.
 - Element Name (string): BIM element name associated with the task.
 - Zone (string): BIM zone associated with the specific element.

During the task execution the collected data related to resources tracked location and health and safety issues occurred, can be retrieved from the COGITO DTP and stored in the following attributes in order to be taken into account for the optimization of the construction process:

- Time_Step (string): steps to query for H&S data.
- Time_Window (string): time frame to query for H&S data.
- Health_and_Safety (record table)
 - Description (string): description of the H&S issue.
 - Severity_Level (string): severity of the H&S issue.
 - Mitigation (string): mitigation measures for the H&S issue.
 - Safety_Zone (string): safety zone involved in the H&S issue.
 - Construction_Zone (string): construction zone involved in the H&S issue.

- Mitigation_Zone (string): mitigation zone involved in the H&S issue.
- Location_Tracking (record table)
 - Resource_Name (string): unique id of the tracked resource.
 - Group (string): group/type of the tracked resource.
 - Time (string): timestamp of the tracking.
 - Zone (string): BIM zone where the resource has been tracked.
- Import H&S (button): allows import H&S information from the COGITO DTP into the previously described Health_and_Safety attribute.

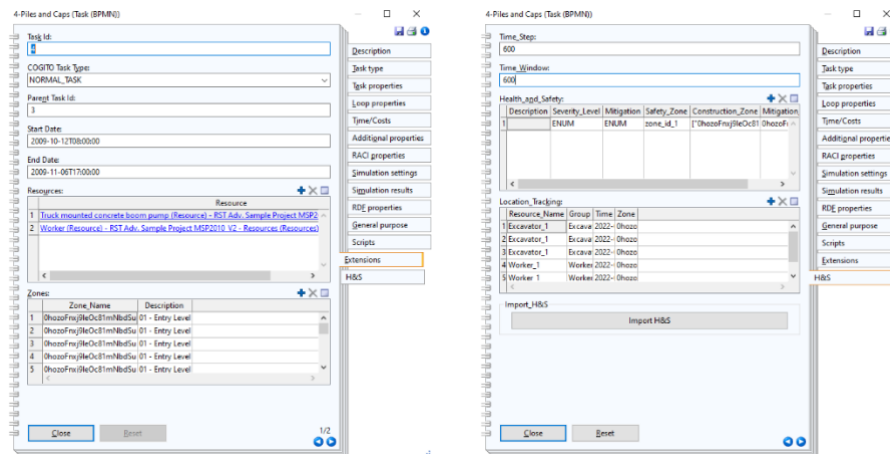


Figure 6 – Extension of the Task Object in BPMN

In addition to the extended attributes in BPMN tasks, two new model types, Resource and SLA model type, were introduced. The SLA model type is composed of one concept, the SLA object, that allows to model the SLAs defined for the different process tasks. For the SLA model type – see Figure 7 – following attributes of the **SLA object**, were introduced:

- Start_Time (date and time): SLA validity start time.
- End_Time (date and time): SLA validity end time.
- Parties (record table)
 - ID (integer): Id of the parties involved in the SLA.
 - Role (string): Role of the parties involved in the SLA.
- Tasks (record table)
 - Task (object reference to a BPMN task object): reference to the task involved in the SLA.
- Service_Level_Objectives (record table)
 - KPI (string): KPI measured in the SLA.
 - Target (string): target value of the KPI to fulfil the SLA.
 - Rule (enumeration: more than, less than, equal to): rule to compare the KPI value with the target.
- Equipment (record table)
 - ID (string): id of the equipment resource associated with the SLA.
 - Equipment_Name (string): name of the equipment resource associated with the SLA.

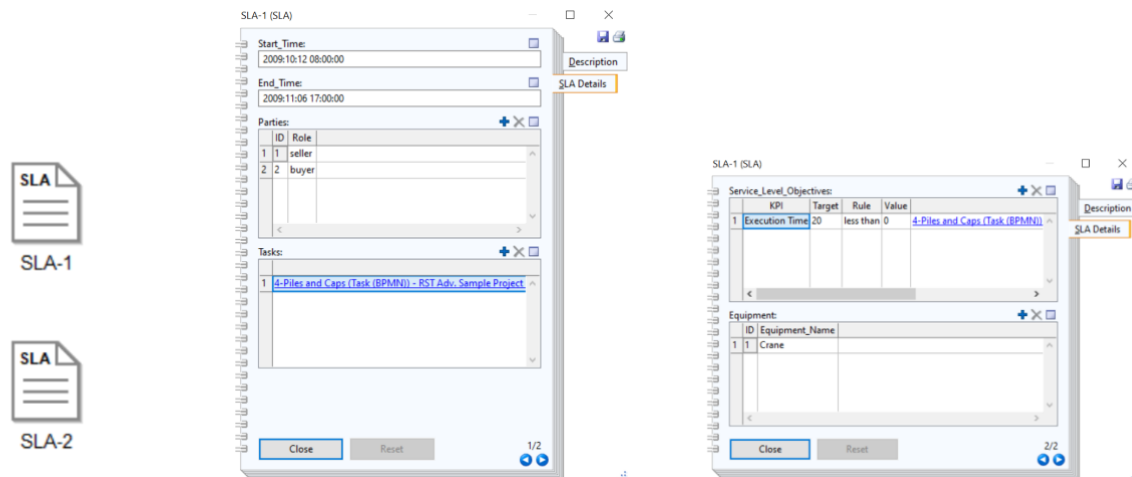


Figure 7 – SLA Model Type and SLA Objects

The Resources model type allows to represent all the human and machinery resources available and needed for each task. For the Resources model type, following attributes of the **Resource objects** were introduced – see Figure 8:

- ID (string): Id of the resource as provided in the COGITO DTP.
- Type (enumeration: Human, Equipment): type of resource, if human or equipment.
- Quantity (integer): Availability of the resource.
- Cost_per_Hour (integer): Cost per hour associated to the resource.

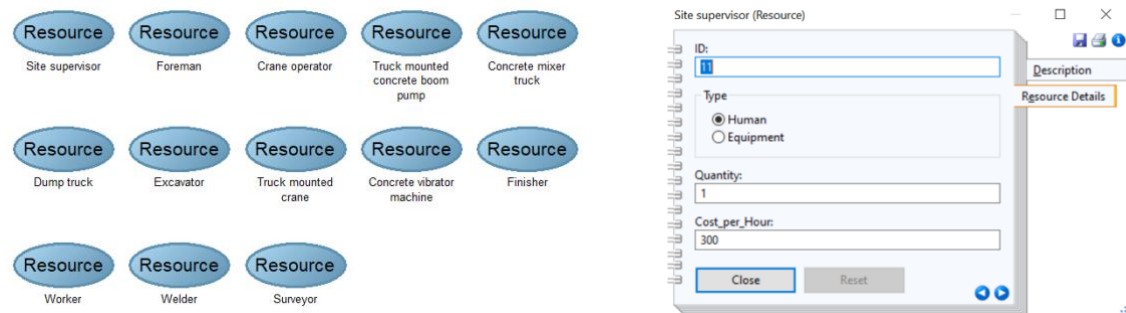


Figure 8 – Resource Model Type and Resource Objects

2.1.2 KPIs Modelling Environment

The KPI Modelling environment is a design tool created with the ADOxx meta-modelling platform that complement the Process Modelling Environment with KPIs support. It is based on the concepts of the balanced scorecard [1] extended with a data calculation model-type that allows to specify how the KPIs are retrieved and calculated. The KPI model type in particular allows to define KPIs and Goals with their dependencies and group them in specific perspectives. It is composed of the following elements (Figure 9):

- Perspectives shaped as BPMN Pools group similar KPIs, e.g., grouping all “Costs” indicators or all “Time”.
- Goals and sub-goals shaped as triangles describe the objective to be achieved.
- KPIs, shaped as circles, describe measurable data sets that in combination with the indicator context – plan value, real value, thresholds, type of thresholds and meta-data about the indicator – assess if the corresponding goal can be achieved or not.

The following relations, describing the dependencies between elements, are available:

- Goal-to-Goal: specify when a sub-goal influences the evaluation of another goal.

- KPI-to-Goal: used to represent the influence that a KPI has in the evaluation of a goal.

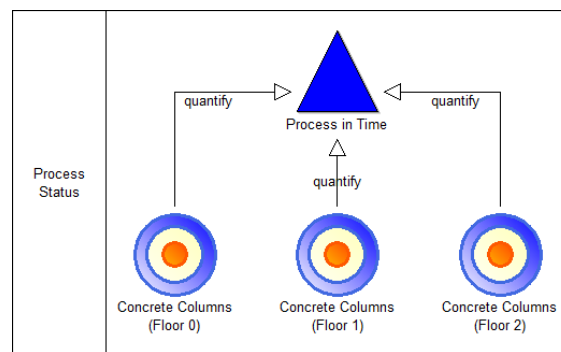


Figure 9 - KPI Model Sample

All objects in the model share a common set of attributes, i.e., a name and a description of the object plus a set of specific attributes that characterise it.

In the case of Goals and Sub-Goals, the specific attributes (Figure 10 Left) refer to the type of the goal, namely Strategic or Operational and on the data aggregation type, representing how often this goal is evaluated. Concerning the Goal elements, it is possible to specify the procedure used during their evaluation; the Goal can therefore succeed if all of its dependencies succeed, if at least one succeeds, or, in case of specific needs, if the evaluation of a customized function succeeds. In the case of KPI objects, the specific attributes are relevant to the fields of data available in the KPI along with information on the target and alert ranges of the KPI value (Figure 10 Right). The fields represent what kind of metrics are available in these KPIs. Usually there is a field (duration in the case of Figure 10 Right) that contains the value of the KPI and additional fields containing meta-information associated to the KPI value. Each field can also have a specific measurement unit; in this case, it is also possible to specify the data aggregation type representing how often the KPI is calculated.

KPIs, to be meaningful, must be associated with thresholds representing target and alert ranges. The target range must contain a formula (as a JavaScript expression) that uses the field name defined above and specifies the value range that is the target of our KPI (e.g., duration < 15). Using the same approach, it is possible to define one or many alert ranges that allow to specify when the KPI is approaching a risky value on the border of the target range. Multiple alert ranges are possible, e.g., if the process task duration exceeds 12 days, it should produce a yellow/moderate alert. Ranges allow to represent information on the threshold of a value (e.g., 12) combined with information on the expected tendency of the values (if the threshold is an upper or lower bound).

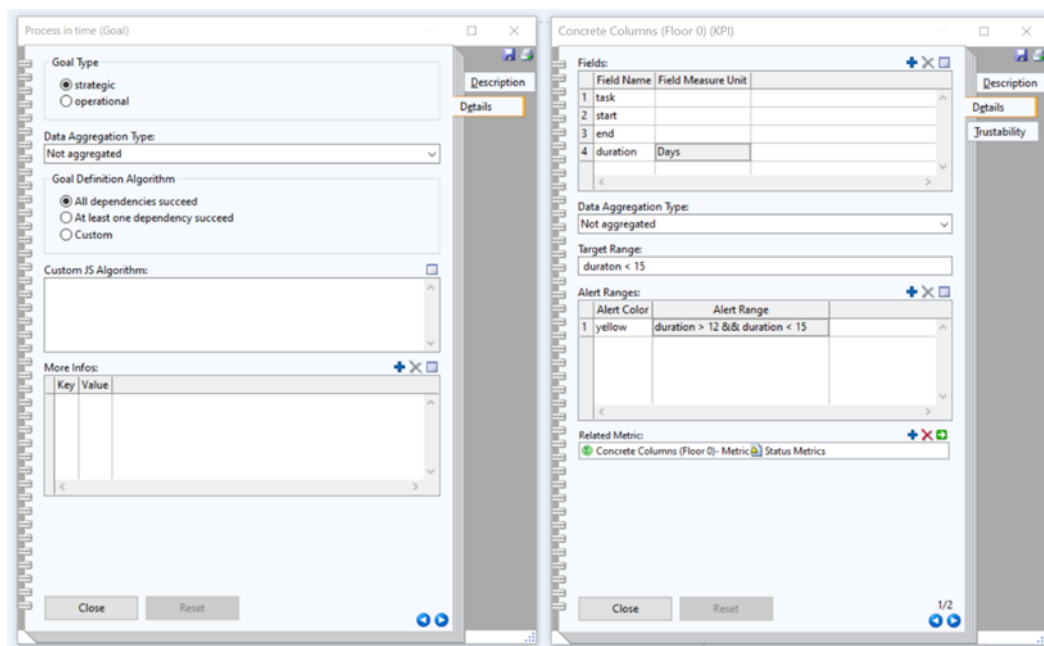


Figure 10 - Goal (Left) and KPI (right) attributes

Each KPI needs to be associated with a specific Metric object defined in a Data Calculation model that contains details about how the value of the KPI is calculated based on available metrics and where the latter's value can be retrieved from. The Data Calculation model type is composed of Metrics and Data items including dependencies between them. Metrics (depicted as green circles in Figure 11) represent data in a specific format containing information on how the value of these data must be calculated using as inputs sub-metrics and Data Items. The Data Items (shown as blue circles in Figure 11), on the other hand, describe how a data value is retrieved from an external system, e.g., a remote service such as the COGITO Digital Twin Platform, a database, or even an Excel sheet. In this context, the Data Item is strongly dependent on the Microservice Integration component underlying framework, OLIVE, responsible for providing the functionality (in form of microservice) for retrieving the needed data from external systems (described in Chapter 6).

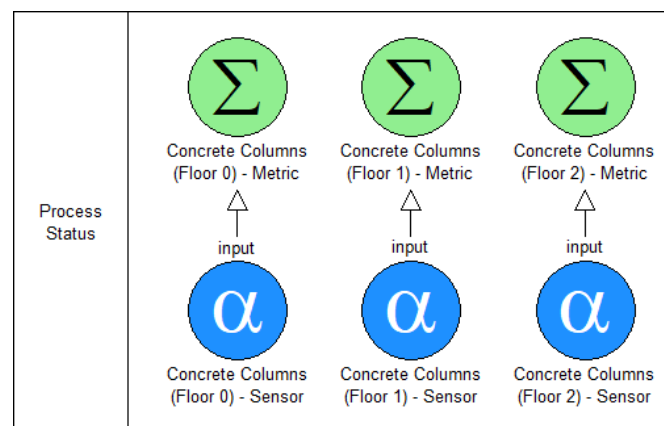


Figure 11 - KPI Data Calculation Model Sample

Moreover, every object in the Data Calculation model type has a common set of attributes, i.e., a name and a description of the object plus a set of specific attributes that characterise it.

Regarding the Data Item object's attributes, it is important to define the data fields that the microservice returns together with optional information on the measurement unit used for the specific value in each field. When the microservice returning the needed data is created, it is important to refer to it using its unique ID, providing the operation name and its required inputs (using their IDs) with the appropriate values. In

Figure 12 (left) the Data Item object representing the process duration on ground floor (floor 0) uses an existing OLIVE microservice named “estimateProcessStatus” that returns the status of the process in the specific area.

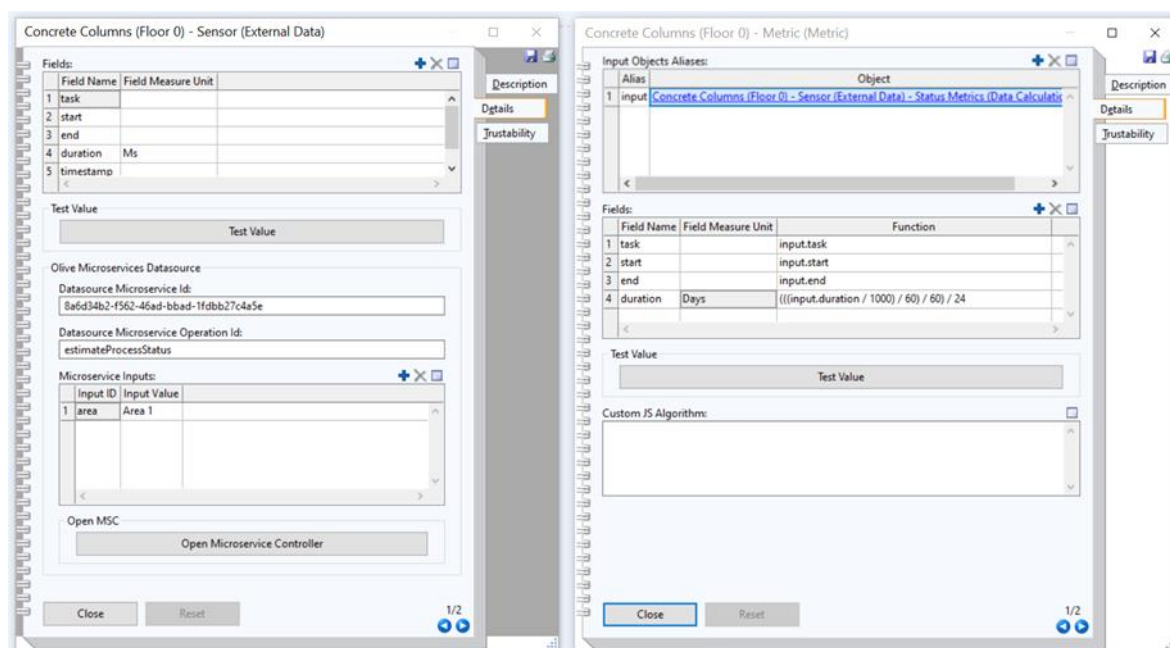


Figure 12 - Data (Left) and Metric (Right) attributes

Meanwhile, the attributes related to the Metric objects refer to the way the metric is calculated based on its dependencies. The Input Object Aliases attributes allow to specify an alias name (in the form of a string without spaces) for every dependency (reference to the dependent Metric or Data Item object) in order to use them in the calculation formula “Function” of every Fields attribute. The Fields represent what kind of data is available in this metric and as described in the KPI attributes, it usually contains a field about the metric value (in the Figure 12 sample it is “duration” for the “Concrete Columns (Floor 0)” Metric) and optionally additional meta-information fields (like the “start” and “end” times in the Figure 12 sample). Every field must specify the formula used to calculate the values and can optionally have a specific measurement unit. The formula can be described as a JavaScript expression and the defined aliases can be accessed and used throughout it. Fields of dependent objects can be accessed using the dot operator. In the sample in Figure 12 (right) the formula is used to convert the process duration returned by the microservice specified in the data object, from milliseconds to days.

As soon as the KPI model and the respective data model are ready, the KPI modelling component gives the option to export them in JSON format. This file will then be used by specific microservices defined in the integration framework (Chapter 6) that calculate each KPI value based on the information specified in the models.

2.2 Technology Stack and Implementation Tools

The Modelling component is built over the ADOxx meta-modelling platform. The created Process and KPIs library for the ADOxx platform has been developed using a Java-based creation environment provided by the ADOxx community named ADOxx JavaDSL. Finally, the deployment of the cloud version of the Modelling component for construction processes is based entirely on provided Docker images.

Table 1 - Libraries and Technologies used in the PMS Modelling component

Library/Technology Name	Version	License
ADOxx	1.5	Free for research purposes

ADOxx JavaDSL	1.5	GNU AGPL v3
Java OpenJDK	8	GNU GPL
Docker	1.8	Free

2.3 Input, Output and API Documentation

APIs are available in the Modelling component in order to programmatically interact with the platform, e.g., to get the processes in BPMN standard format, to retrieve process image and tasks and to update model details. The APIs are not publicly exposed to other COGITO modules in any way, but only to the Microservice Integration component described in Chapter 6, responsible for processing them and exposing only the needed functionality to the third-party requesting PMS components, with a uniform input and output structure.

In particular the following functions related to the Modelling component are exposed through the Microservice Integration component:

- **retrieveProcessList:**
Method: POST
Inputs: a query to filter the retrieved processes by name
Output: a JSON containing the list of construction processes IDs, including their names, descriptions, and image
- **exportProcess:**
Method: POST
Inputs: the process ID
Output: the BPMN file of the requested process
- **exportProcessJSON**
Method: POST
Inputs: the process ID
Output: the file of the requested process in JSON as accepted by COGITO DTP.
- **importProcessJSON**
Method: POST
Inputs: the process in JSON format as provided by the COGITO DTP.
Output: confirmation of successfully imported data.

2.4 Application Example

2.4.1 School Sample Use Case

As a common sample for demonstrating and testing out the interconnections of the different COGITO modules, a school construction case was used. This sample use case is conducted to bridge the time by enabling testing before the pre-validation site data (BIM models, schedules, resources, etc.) are available. The school sample model – see Figure 13 – was implemented using the local modelling environment with the extended BPMN version described in Section 2.1.1.2.

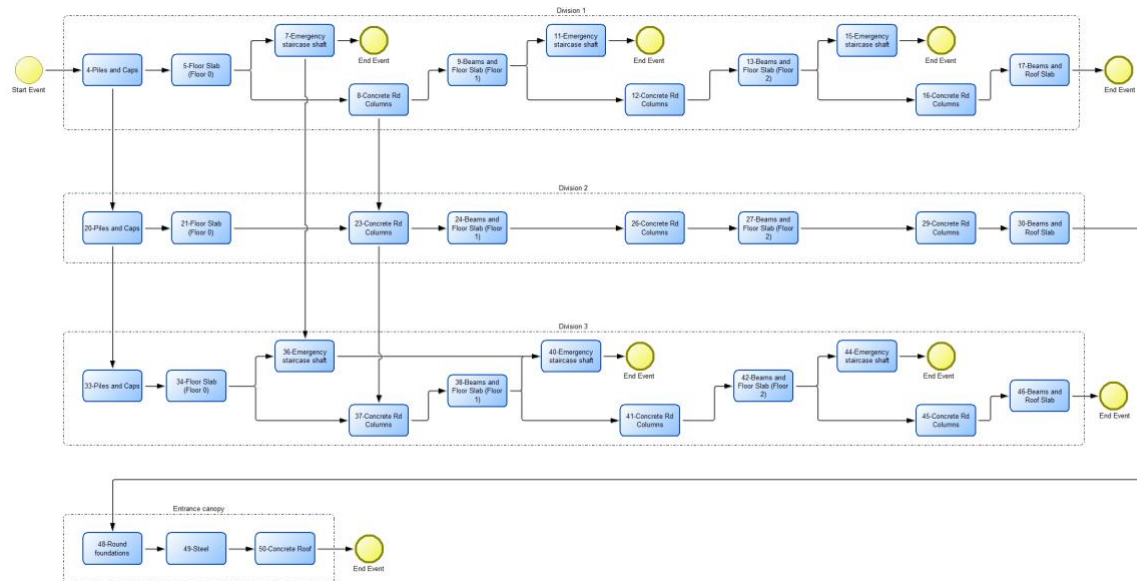


Figure 13 - BPMN Model for School Sample

The model is imported from the COGITO DTP using the COGITO import feature of the Modelling environment and refined adding template-based subtasks to each original task in order to add safety and quality control activities. Additionally, in the template, for each activity, the resources are assigned in terms of both workers and equipment needed, as well as the BIM elements specific for the safety activities. The final process (Figure 14) is then exported with the COGITO export feature of the Modelling environment and send back to the COGITO DTP for subsequent execution.

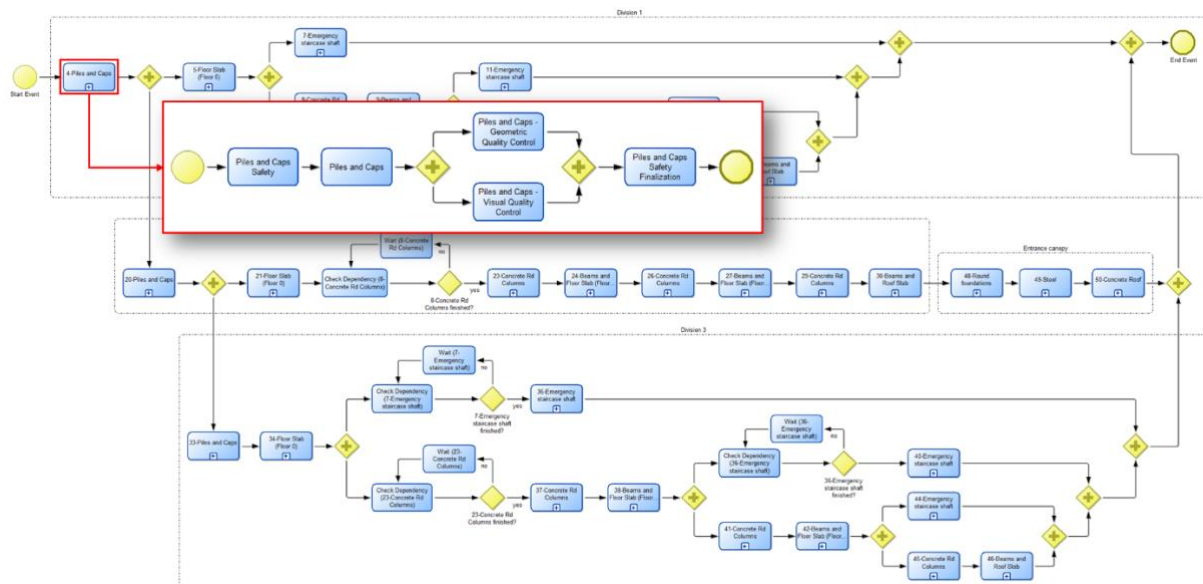


Figure 14 - Extended Process Model with Safety and Control Subprocesses

In order to ease the KPI identification based on the process import, a systematic approach is introduced. Here, rules can be seen as one way of systematically introducing KPIs based on process models. For this approach, modellers and domain experts can collaboratively set up rules that ease the KPI introduction. Such rules may be influenced by the task type itself (e.g. waiting or checking task) or by the sequence of tasks (e.g. parallel tasks). For instance, following rules may be reasonable in a construction scenario:

- If there is an activity related to checking other tasks and waiting for their finalization, the corresponding activity (predecessor) must be monitored.
 - For instance, when concrete is not applied and dried on the first floor, this task cannot be conducted on the second floor. Therefore, concreting in the individual floors requires monitoring to ensure that the tasks are conducted in line with the time planning and to avoid waiting time before proceeding with the process flow after concreting is finished.
- If in the process there is a BPMN merging path (e.g. after parallelism), the two (or more) merging paths must be monitored. This can be done either by monitoring all activities in the paths or only the final activity in the paths that are merging.
 - For instance, the emergency staircase can be installed in parallel to the concreting and the installation of beams & slabs at one floor (see Figure 15). Monitoring the emergency staircase installation as well as the beams & slabs installation is required to ensure that both paths are executed before merging them. In case, concreting is a critical task, monitoring it additionally may also be needed. The relevancy and criticality of tasks must be defined in collaboration with domain experts.
- If there is a task requiring several resources (e.g. human workers, equipment, materials, etc.) monitoring the predecessor task may be reasonable to ensure that the resources are available on time.
- If there are critical tasks with respect to the project progress or costs, monitoring these tasks is necessary to being able to interfere in case of problems.

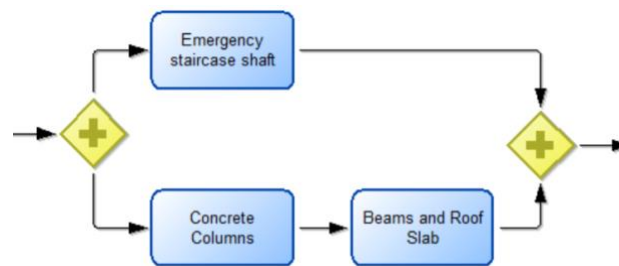


Figure 15 - Parallel Process Path

The Figure 16 shows KPIs that were identified based on Figure 14 following the above-described rules. For instance, KPI “Concrete Columns (Floor 0)” deems from a corresponding check and wait cycle for constructing the first floor. Another example would be the KPI for “44-Emergency staircase shaft”, which is used for monitoring activities conducted in parallel, as this is a task directly before merging two parallel paths.



Figure 16 - KPIs identified following the Rules

2.4.2 Railway Construction Use Case

Another scenario used for the demonstration of all the PMS components is related to the construction of a dewatering layer for a railway track bed. This use case has been decided as simple enough for showing the functionality of the components while still relevant for the future COGITO scenarios in the pre-validation sites.

A process template has been created in the form of a process general enough to be valid for different construction zones. Huge construction sites such as railway construction sites may be divided in construction sections of 300m; the construction process in every area is almost the same but may differ regarding some specific requirements and the execution time. An example is provided in the process template in Figure 17. The process starts by lifting the tracks into the correct position, which is done by the lifting team. Afterwards, welding and planking tasks can be taken over by a targeted team. When the first half of the construction section (0 to 150m) is welded, the concreting team can conduct the concreting. If the concreting is finished for the first half of the construction section (0 to 150m), the filling team can start filling the chamber. A final control task executed by a foreman may be introduced for controlling the process.

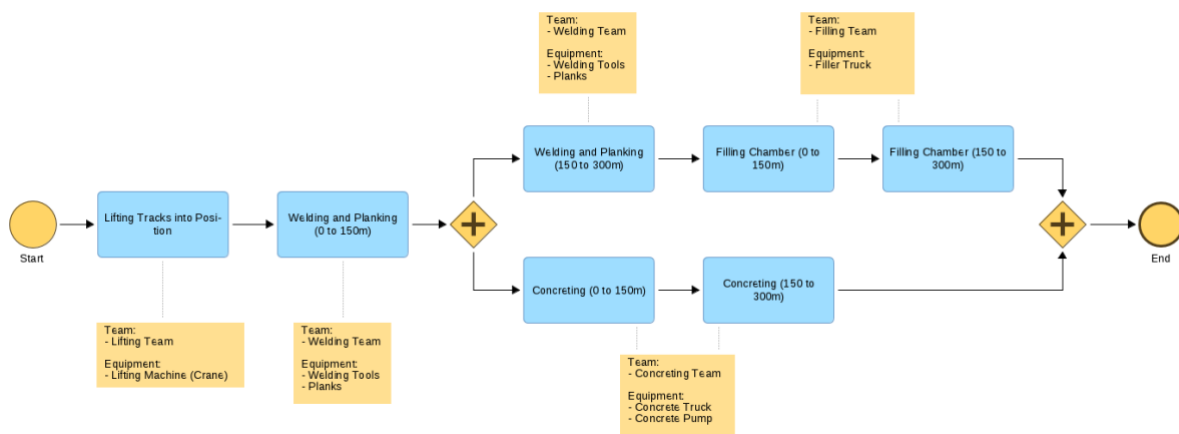


Figure 17 - Railway Track Construction Template Process

This process template is then instantiated in each construction section (each section is 300m long). The instantiated process for each construction section can be seen in Figure 18. In this sample process there are no decision to be taken. In case of decisions, each instantiated process for a construction section could look differently based on the decision path that is taken. However, although there is no visual difference in the instance process models in our sample, there will be differences when it comes to execution (like in terms of costs, times, or working team members).

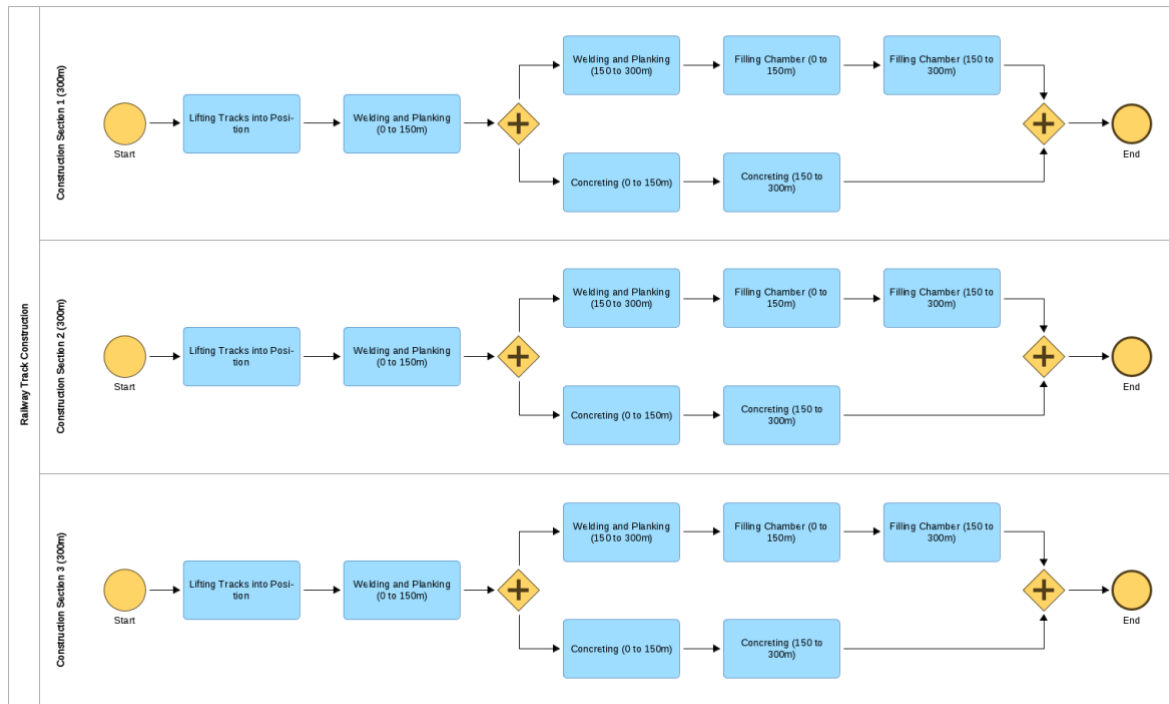


Figure 18 - Railway Track Construction Process Instances

Finally, for each process instance, roles and resources are assigned (Figure 19) in order to support the subsequent execution of the construction process in the workflow engine provided by the COGITO WODM module.

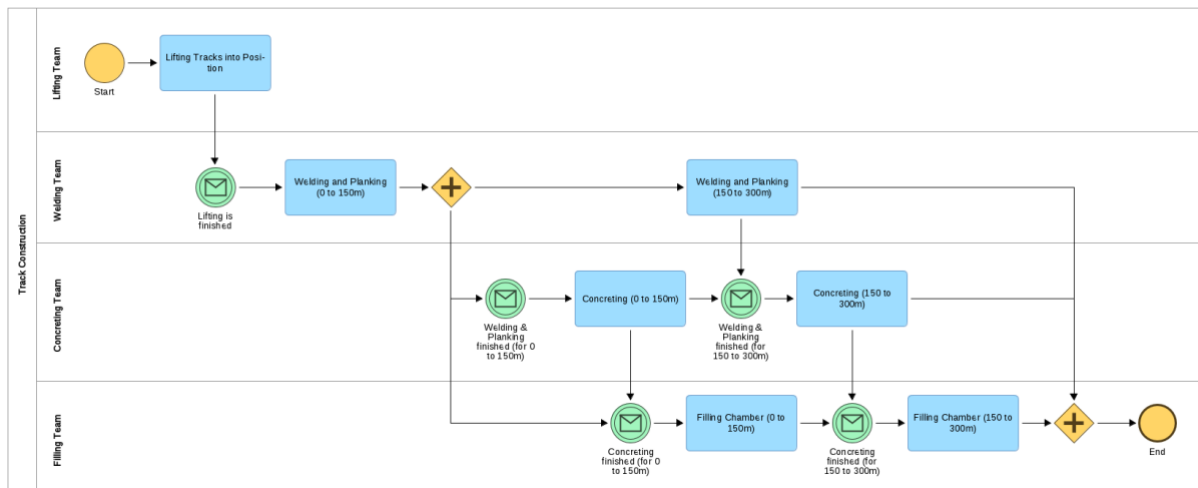


Figure 19 - Railway Track Construction Process Workflow

Based on the process capturing the construction scenario, various goals and related KPIs can be defined. Those are e.g., related to specific questions such as whether there is sufficient and correct equipment in the corresponding construction section or in what construction section the equipment is located at the moment. The equipment (crane, concrete truck, concrete pump, filling truck, etc.) as well as the manpower in the construction site sections are tracked in order to estimate the process status. This is done checking the equipment, and team for each activity and based on their location in the construction process area, it is estimated if the activity is proceeding on time or expecting delays.

A KPI model is created in particular in order to evaluate the status of the equipment in each construction section and the status of the process (Figure 20). Specific microservices have been created in the Microservice framework (Chapter 6) for estimating the status in the Sandbox Experiment (Chapter 3) and have been assigned to the right KPI data model objects (Section 2.1.2).

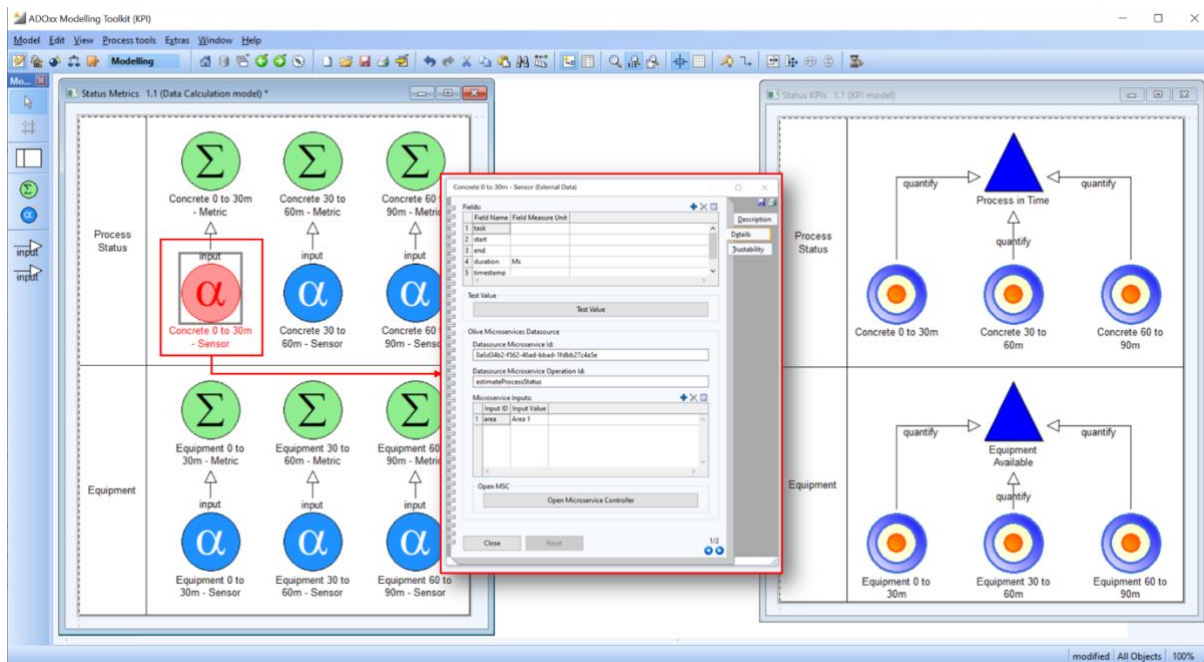


Figure 20 - Railway Track Construction KPI Models

2.5 Licensing

The Modelling component is released in two versions: a community edition for the Process and KPI modelling based on the desktop version of the ADOxx application that everyone can download for free, and a cloud version for the process modelling, deployed in the BOC cloud that can be downloaded and installed locally only with a license.

2.6 Installation Instructions

The cloud-based Process Modelling environment for construction processes is available at https://cogito.boc-group.eu/ADONISNP10_0_COGITO/ and the access is granted with the COGITO Identity provided.

The community version of the Process and KPI Modelling environment can be installed with the following instructions:

1. Download the ADOxx platform at <https://www.adoxx.org/live/download-guided>
2. Install it following the instructions provided in the page according to your operating system.
3. Download the Process library and models from here: <https://git.boc-group.eu/cogito/cogito-aio-standalone-package/-/tree/main/MODELS/BEEUP>
4. Download the KPI library and models from here: <https://git.boc-group.eu/cogito/cogito-aio-standalone-package/-/tree/main/MODELS/KPI>
5. Install the libraries in the ADOxx platform following the instruction provide in this video: <https://www.adoxx.org/live/import-new-application-library>

6. Import the downloaded sample models following the instruction provided in this video: https://www.adoxx.org/live/import_models_adl

2.7 Development and integration status

The following items were still in progress during the first publication of the deliverable (D6.3) and are released and described in this final version of the Modelling component (see particularly Sections 2.1.1 and 2.1.2):

- Extension of the process modelling component with attributes related to the scheduled start and end times for each activity.
- Extension of the process modelling component with attributes related to the location for each activity.
- Extension of the process modelling component with attributes related to the quality status and the H&S issues for each activity.
- Integration of services (described in Section 6.7) for obtaining 4D-BIM information from the COGITO DTP. In this case the functionality has been implemented in form of local file import while the connection with the COGITO DTP is still a work in progress.
- Integration of services (described in Section 6.7) for exporting the process, with resources and task detailed information in JSON format for the COGITO DTP. In this case the functionality has been implemented in form of local file export while the connection with the COGITO DTP is still a work in progress.
- Integration with the COGITO Identity Provided as centralized authentication mechanism.
- Integration with the PMS Optimisation module for calculate the optimised schedule and updating the models.
- Release of process models and KPIs related to the specific use case of school construction project used as sample for testing the interconnection between the different COGITO modules and initial release of the models for the pre-validation use case scenarios.

In addition, the following prototypes has been evaluated and tested in the modelling environment to improve its accessibility and completeness:

- Task suggestion prototype: This prototype is available for the cloud version of the process modelling environment and will help inexperienced users suggesting the next activity to perform in their process, applying a semantic inference over a catalogue of different construction models.
- Voice control prototype: This prototype is available for the cloud version of the process modelling environment and allows users to design and interact with the Modelling environment using voice commands through the Amazon Alexa service.
- Invite prototype: This prototype is available for the cloud version of the process modelling environment and allows users to collaborate on a specific model by inviting and granting access to external experts on the platform.

2.8 Requirements Coverage

In the following tables the technical, functional, non-functional requirements as defined in COGITO D2.4 and the stakeholder requirements as defined in COGITO D2.1 are reported with focus only on the support provided by the PMS Modelling component.

Table 2 - Requirements on Computing Systems for the PMS Modelling component

ID	Description: <i>The Computing solution...</i>	Type	Priority	Status
COGI-CS-1	[DCC, WODM, PMS, BC, SafetyConAI, VirtualSafety,	• Operational	Must	Supported for both process and KPI modelling

	gQC] runs on desktop or laptop PC			
COGI-CS-4	runs on Windows	• Operational	Must	Supported for both process and KPI modelling
COGI-CS-5	runs on Mac	• Operational	Could	Supported for both process and KPI modelling
COGI-CS-6	runs on Android	• Operational	Would	Support only for process modelling
COGI-CS-7	allows access to the whole data in one location	• Operation • Functional • Design constraint	Must	Supported: All the models are available in a single repository
COGI-CS-8	maintains communication and data security	• Legal • Design constraint	Must	Supported: All the created models are accessible only after authentication
COGI-CS-9	differentiates data and system access levels and modification rights	• Legal • Functional • Design constraint	Must	Supported: All the created models are accessible only to the authorized users

Table 3 - Requirements about Workflow Planning, Execution and Monitoring for the PMS Modelling component

ID	Description: <i>The Workflow Planning, Execution and Monitoring solution ...</i>	Type	Priority	Status
COGI-WF-6	[PMS, WODM, DCC] allows the PM and SM to efficiently detect and prioritise delays and cost escalation elements	• Functional	Should	Supported: The KPI models can be used for the definition of costs and delay indicators
COGI-WF-7	[PMS, WODM, DCC] allows the PM to extract reports about: project time performance, project cost performance, costs per unit, resource consumption	• Functional	Must	Supported: The KPI models are used for the definition of all the needed indicators
COGI-WF-8	[PMS, WODM] uses BIM models to support scheduling and budgeting	• Functional	Should	Supported: The 4D-BIM data is imported from the COGITO DTP
COGI-WF-16	[PMS] updates the project schedule at least monthly	• Operational • Functional	Must	Supported: The versioning support allows to keep multiple

		•Process		versions of the same process updated at users need
COGI-WF-21	[PMS, WODM] facilitates resource allocation during scheduling	•Functional	Should	Supported: The process modelling component will allow to associate resources to specific tasks
COGI-WF-23	[PMS, WODM] incorporates health and safety planning	•Functional •Design constraint	Should	Supported: The H&S issues are imported for each task from the COGITO DTP
COGI-WF-24	[PMS, WODM] incorporates quality control planning	•Functional •Design constraint	Should	Supported: The process modelling component allows to associate quality and safety related activity to specific tasks
COGI-WF-27	offers simple, easy to use, and intuitive interface to avoid workforce over-burdening	•Operational •Design constraint	Must	Supported: The modelling canvas provides drawing facilities to speed up the modelling process

Table 4 - Functional, Non-Functional Requirements and Interfaces for the PMS Modelling component

Functional and Non-Functional Requirements			Achievements
Functional	Req-1.1	Construct workflow and simulation model and populate with historical data	Supported
Non-Functional	Req-1.7	Web-based app	Supported
	Req-2.1	User-friendly	Supported: The solution provides modelling facilities for end users and compatibility with BPMN standard
	Req-2.2	Scalability	Supported: Thanks to the underlying ADOxx meta-modelling platform, the solution can be extended to support other modelling concepts in case of needs.
	Req-2.3	Security	Supported: The access to the models and features is possible only to authenticated and authorized users.

2.9 Assumptions and Restrictions

The following restriction applies and will be covered in the integration activities for the pre-validation and validation sites:

- The model import/export from the COGITO DTP (as well as the H&S import) are implemented in form of local file upload/download. This will be integrated with an automatic call to the COGITO DTP services as soon as the endpoints will be ready and will be reported as part of the integrated COGITO solution in the relevant deliverables of WP8.

3 PMS – Sandbox Experiment Component

This section demonstrates the Sandbox Experiment component, available in the form of a physical experiment. This experiment can be considered as an extension to the Modelling component presented in Section 2, where modellers and domain experts can brainstorm about the process using a design thinking approach in order to refine it. Afterwards, they have a safe environment to test the applicability of the digitisation approaches and finally they are able to generate sample log data emulating the process execution; the latter is useful to test corresponding services (like the Simulation and the Optimisation components) at an early stage of the project, when usually such data is not available from the pilot sites.

The tangible means of the physical experiment are one of the main concepts in the design thinking approach where stakeholders with different background and competences can brainstorm and collaborate to bring innovations and creativity in the modelled scenario. The abstracted construction environment gives the possibility to focus on specific issues and to evaluate the benefits of the introduction of digitisation patterns (like equipment tracking) in the digital twin solution. On the other hand, the introduction of technological devices supporting such digitisation patterns aids the identification of related constraints (like battery replacements on sensors) that should be taken into account during the process execution.

3.1 Prototype Overview

The Sandbox Experiment component helps identify and evaluate relevant digitisation patterns for the digital twin design and the features enabled with the use of digital technologies. Before going into more detail concerning the digital twin design, Figure 21 shows some real-world impressions from the track construction in the pre-validation site, where the rail track ballast is placed onto the track area before laying the track elements and inserting the rails.



Figure 21 - Impressions from a Real-World Track Construction Scenario at Pre-Validation Site

Based on such a real-world scenario, the design of the digital twin is built upon two main aspects, the monitoring of the construction site with respect to times and costs, and the management of logistics aspects. Part of the construction site monitoring is also the monitoring and simulation of the construction progress, consistency checks with respect to the process and regular assessments of aspects such as (but not limited to) times, costs, staff hours, or special equipment hours. In order to enable such progress monitoring towards a digital twin, some kind of digitisation is required to retrieve digital data about the equipment, the construction teams and the progress status. This digital information can further serve as input for managing the logistics; for instance, monitoring the equipment location, consistency checks of the logistics processes and simulating the logistics processes (e.g., for potential raw material shortage) are considered to be relevant when designing a construction digital twin.

The following major steps are conducted towards designing the digital twin for the construction process:

- Identification, abstraction and selection of use case requirements that can be addressed with the digital twin.

- Identification and evaluation of the appropriate digitisation design to support the use case requirements.
- Identification and evaluation of patterns (like the tracking of equipment at a specific location) that can be applied to the digital twin.
- Identification and evaluation of technologies that can be used to implement the suitable pattern (like markers code recognition).

The Open Model Initiative Laboratory (OMiLAB) Innovation Corner [2] is used as environment for the Sandbox Experiment component. It is based on three abstraction layers, namely the business, the conceptual, and the proof-of-concept layer – as can be seen in Figure 22(left side). This architecture provides different perspectives on layer-related questions and therefore allows for diverse considerations throughout the digitisation journey. Such questions could be: (a) should a new business model be developed or is optimizing the existing one appropriate, (b) what organisational structure is needed so that the processes can be managed by diverse stakeholders and (c) how could an innovation idea be implemented and operated within an industrial context. Within the scope of a construction experiment, among others, the following questions may be relevant: (a) what are the use case requirements that must be addressed with a digital twin (b) what is the appropriate digitisation design to address the use case needs, and (c) which pattern can be applied for digitization and technical implementation.

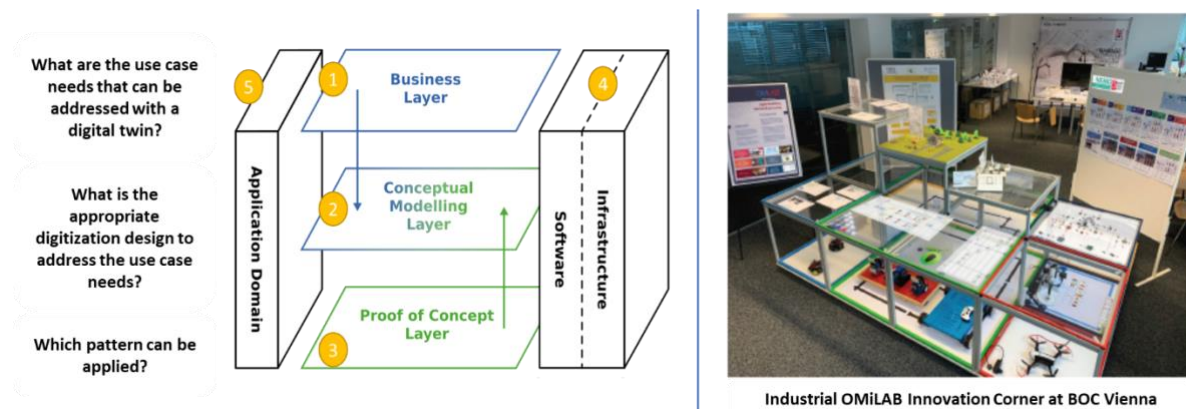


Figure 22 - OMiLAB Innovation Corner Conceptual Perspective (left) and Real Perspective (right)

The OMiLAB Innovation Corner offers a variety of methods and tools – such as modelling software, digitisation devices and physical experimentation spaces. Hence, the way is paved for (a) ideating digital innovation solutions in the construction domain, (b) formalising relevant knowledge assets by means of modelling and (c) testing within a secured physical environment. In general, guidance to lift conceptual digital transformation/innovations towards an industrial application is provided. Specifically, the modelling components are highly leveraged by the meta-modelling platform ADOxx which facilitates the development of domain-specific modelling languages thanks to the support of an open community of more than 5.000 contributors. The OMiLAB NPO⁸ is the underlying non-profit organization that powers the OMiLAB laboratory environments and provides community support in the form of training materials and books. In general, the model driven OMiLAB environment can be considered as a laboratory ecosystem for innovation workshops. Starting with physical infrastructure such as sensors and microcontrollers as well as digital methods and software, it allows for the creation of tangible experiments facilitating the transfer of conceptual innovation ideas and digital models towards industrial application cases.

For the COGITO construction Sandbox Experiment component, the following OMiLAB Innovation Corner instruments were used:

- **Physical Equipment** – The digital twin is designed in a sandbox environment building upon the design thinking modelling method Scene2Model⁹. Configuration towards the construction domain

⁸ OMiLAB NPO. OMiLAB Digital Innovation Environment. Retrieved from <https://www.omilab.org/>.

⁹ <https://austria.omilab.org/psm/content/scene2model/info>

was conducted by introducing appropriate paper figures and relating them to code markers. A Raspberry Pi connected to a Logitech C920 USB Webcam captures the status of the design thinking elements or the corresponding construction domain figures – such as the paper figures.

- **Tracing Software** – The Markers Tracing Tool builds upon the S2M tracking software and allows tracing the code markers in a specified area, generating data that emulate the equipment and workers movements in the working area.
- **Physical Space** – The physical experiment is composed in the OMiLAB laboratory space on a targeted experiment table located in reaching distance of an electric socket for the Raspberry Pi. A paper canvas specifies the experiment area, representing the construction site and its areas for the tracing software.
- **Additional** – Additional materials ranging from cables and power adaptors for the Raspberry Pi to scissors for cutting the paper figures are used.

The physical experiment design is conducted based on four main stages. An experimental prototype is created in several iterations; the iteration cycle is shown in Figure 23. First, the business scenario – in this case the construction process – is depicted using models; those models evolve throughout several development cycles. Particularly, the appropriate abstraction level will be refined throughout these iterations. For instance, in a first iteration a basic process consisting of preparatory work, actual construction work and finalisation can be used. However, as the construction sample aims at monitoring and tracking equipment, this basic process might be too generic. Therefore, in a next iteration round, additional tasks and also related KPIs targeting the different statues may be introduced. Second, paper figures are used to depict the scenario described in the models. The complexity of the scenario can be adapted by adjusting the number of concepts used for the paper figures that simulate the real-world scenario. As a basic level of granularity, three equipment entities (track lifting machine, concrete truck with pump and filling truck) and four human entities (lifting team, welding team, concreting team and filling team) are used. Third, various means of digitisation are introduced e.g., in the form of markers on the equipment and human worker figures in order to enable tracing them along different construction sections using a camera. Patterns related to the construction scenario described in the models can be identified – for instance equipment tracking. Such a pattern can then be implemented with different technologies – for instance using code markers, a monitoring webcam and a tracking software. Forth, the overall scenario is leveraged by introducing services – potentially enabled by the chosen digital technology – in order to facilitate digitisation and create additional value. In the construction experiment, a modelling service, a database service and a simulation service are introduced as a starting point.

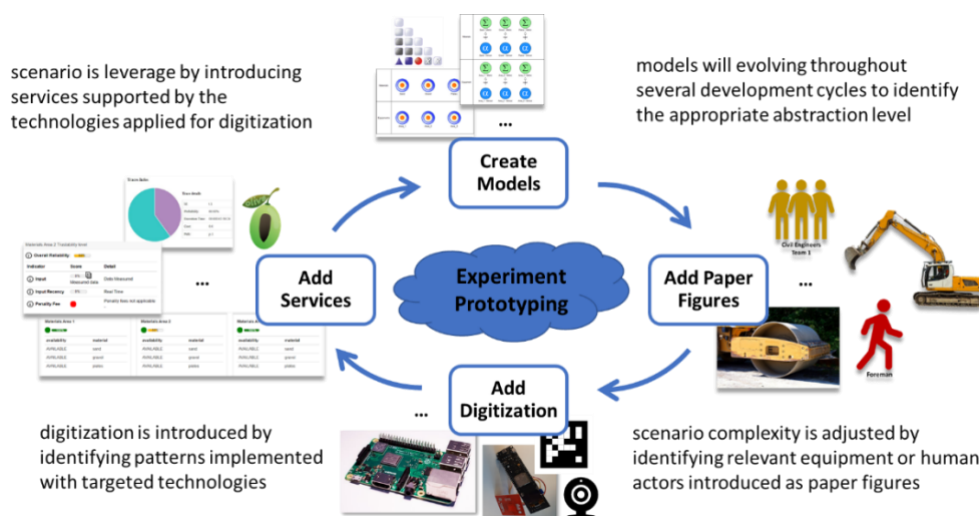


Figure 23 - Steps towards Designing the Physical Experiment

Several iterations of the experiment design stages are run through to adjust the models including the abstraction level, to adapt the complexity of the scenario by introducing appropriate concepts as paper

figures, to filter relevant characteristics for the identification of suitable patterns as well as digital technologies and to add value by adding digital services.

3.1.1 Architecture

The Figure 24 provides further insights on the Sandbox Experiment architecture. The process models and the related KPI models build upon the ADOxx meta-modelling platform that serves as a model repository for the OLIVE framework, responsible for providing different services for the KPIs evaluation, simulation, and optimisation, such as connections to the timeseries database, retrieving of the equipment position by reading a corresponding code marker or saving the current workflow stage. On an operational level the experiment introduces code markers and a webcam for recognition, as well as the related recognition software. Furthermore, the construction process can be emulated by means of virtual sensors allowing for the collection of sample construction process data by taking snapshots of the current construction process stage.

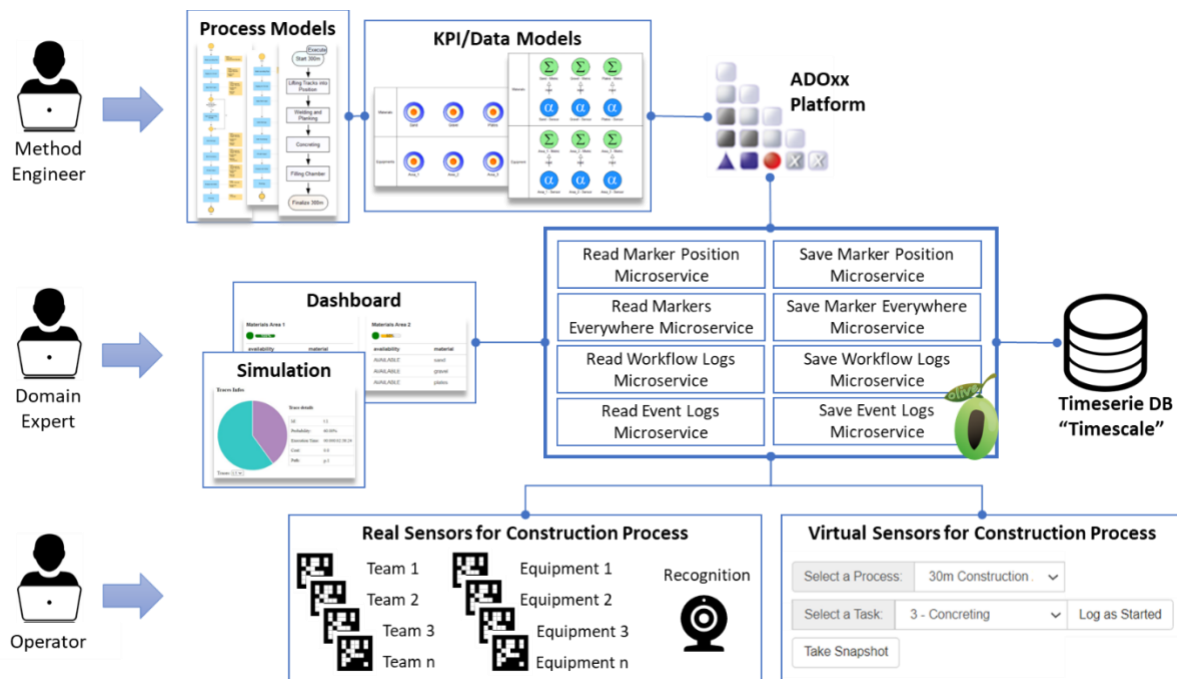


Figure 24 - Experiment Architecture

In the sample experiment, we use code markers to track the equipment and workers, as well as corresponding events (such as equipment movements). However, the sample can be extended with digital technologies and AI as required. For instance, image recognition may be introduced to check if safety installations (e.g. a fence around the construction site) are in place, in order to populate KPIs that guarantee the correct and safe execution of the task.

3.2 Technology Stack and Implementation Tools

The Sandbox Experiment used a NodeJS application to track markers associated with workers and equipment, from a video stream generated by a camera looking at the experiment table and connected to a Raspberry PI. The Raspberry used FFmpeg to process the camera input and stream it to the NodeJS application, deployed with Docker in an internal server. The application detects the markers and stores them in the Timescale database through a microservice as described in Chapter 6. The entry portal is a simple client-side HTML site hosted in the same server as the Node JS application. Additional services described in Chapter 6 are used to show the material, to process the status and retrieve the markers from the Timescale database.

Table 5 - Libraries and Technologies used in the *PMS Sandbox Experiment component*

Library/Technology Name	Version	License
TimescaleDB	14	Apache 2.0 License
FFmpeg	5.0	GNU LGPL 2.1
Node JS	17.5	MIT License
Docker	1.8	Free

3.3 Input, Output and API Documentation

The Sandbox Experiment exposes microservices related to the retrieval of material and process status along with the equipment and workers' location useful to emulate the IoT sensors data from the pilot site. Such services are not exposed to other COGITO modules but only to other PMS internal components. In particular the following services are available:

- **readMarkerPosition:**
Method: POST
Inputs: the marker ID
Output: a JSON containing the history of the marker locations, including timestamp, coordinates, and detected area
- **getProcessStatus:**
Method POST
Inputs: the area where the process is running
Output: a JSON containing the duration of each executed activity in the process

The services are created using the Microservice Integration component described in Chapter 6 and the endpoints with input and output samples can be visualized in the service test page described in Section 6.4.

3.4 Application Example

The Figure 26 shows a picture of a real-world track construction setting on the left and the laboratory setup on the right. In the experiment design phases, the real-world setting of the COGITO pre-validation site is simplified and abstracted to fit the laboratory setup. For instance, paper figures are introduced for the main human workers and for the major equipment.

The construction section is represented by a 300m section depicted in a suitable scale – subdivisions of 30m are marked. The construction process described in Section 2.4 was the foundation for the first prototypical experiment and was instantiated for each of those areas. After discussions with the domain experts from RSRG, another abstraction was chosen in the iterations of the physical experiment creation corresponding to the Karlsruhe pre-validation site. Therefore, the advanced Sandbox experiment focuses on 300m sections as well as on workers and equipment. Figure 25 shows the basic experiment workflow in form of a Flowchart. This workflow is instantiated for each of the 300m sections on a construction site. First, all of the tracks are lifted in the correct position on the construction site section. Afterwards, the welding and planking team fixes the position of the tracks (from 0 to 150m). When the 150m mark is reached, the concreting team can start at the beginning (0 to 150m) in

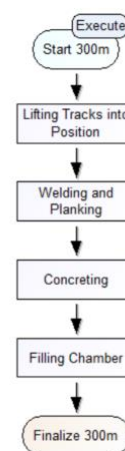


Figure 25 - Basic Workflow corresponding to the Experiment

parallel to the finalization of the welding. While the concreting is finalized (from 150 to 300m), the filling starts and is conducted for the whole construction section.

The modellers can manually emulate the execution of the defined experiment workflow using this setup by picking the cards related to the equipment, workers, and materials (material availability is assumed in this experiment) required to each task, placing them over the specific area, and then moving them around as soon as the workflow proceeds. The workflow in Figure 25 supports the user in this process by suggesting (once executed by clicking the "Execute" button on top of the "Start 300m" event) the next activity to be performed, providing information on the required resources and planned times, from the defined schedule. As an example, in the first task of the workflow the tracks are lifted in the correct position, therefore, cards of the lifting team and the lifting machine are picked and placed in the first subsection (0 to 30m) of the construction area. When the modeller decides that the task is completed, the cards are moved to the next subsection (30 to 60m) and so on. After finishing the track lifting task for the whole 300m, the welding team comes and continues at the first subsection. When the welding team has finished half of the 300m construction area (0 to 150m), the concreting team including the concreting machine can start at the beginning (0 to 30m) in parallel. Figure 27 shows a sample execution procedure in the Sandbox Experiment.

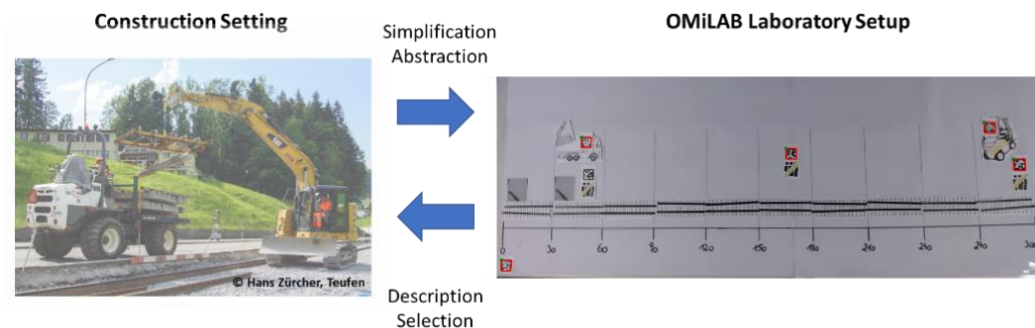


Figure 26 - Real World vs. Experiment Setting

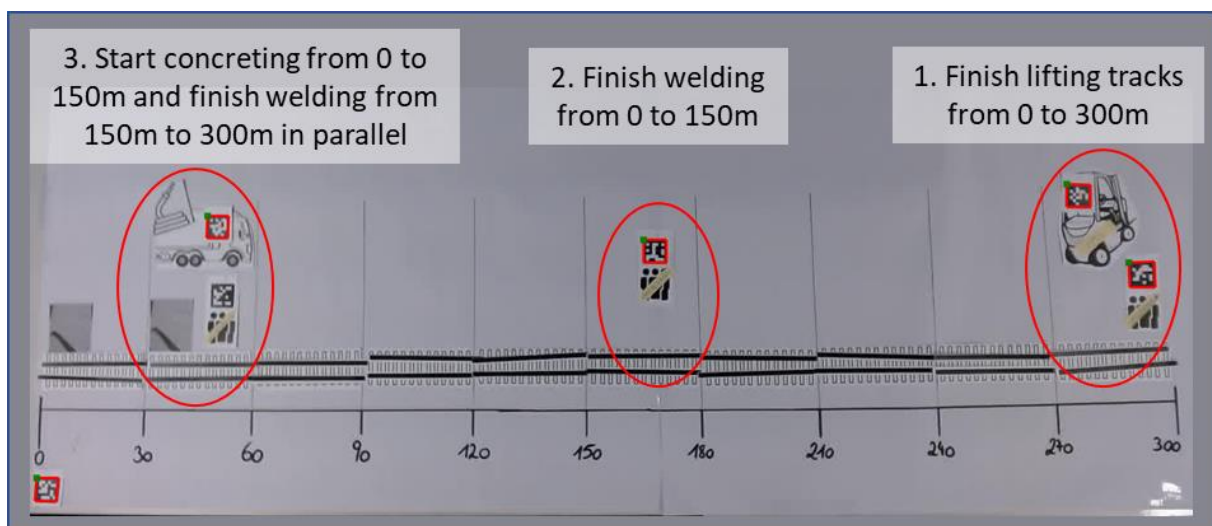


Figure 27 - Physical Experiment Execution Sample

The idea is to test and reason on the construction scenario in a secured physical environment. In addition, this allows to generate reasonable construction sample data at a very early stage and test the needed services. After testing and experimentation, further insights on potential digitisation technologies, as well as more detailed process descriptions can be handed over to the real world and be implemented directly at the construction site. A web page for the experiment is available in order to show the real-time view of the experiment, visualise the data generated, and access all the features and microservices related to the experiment (Figure 28).

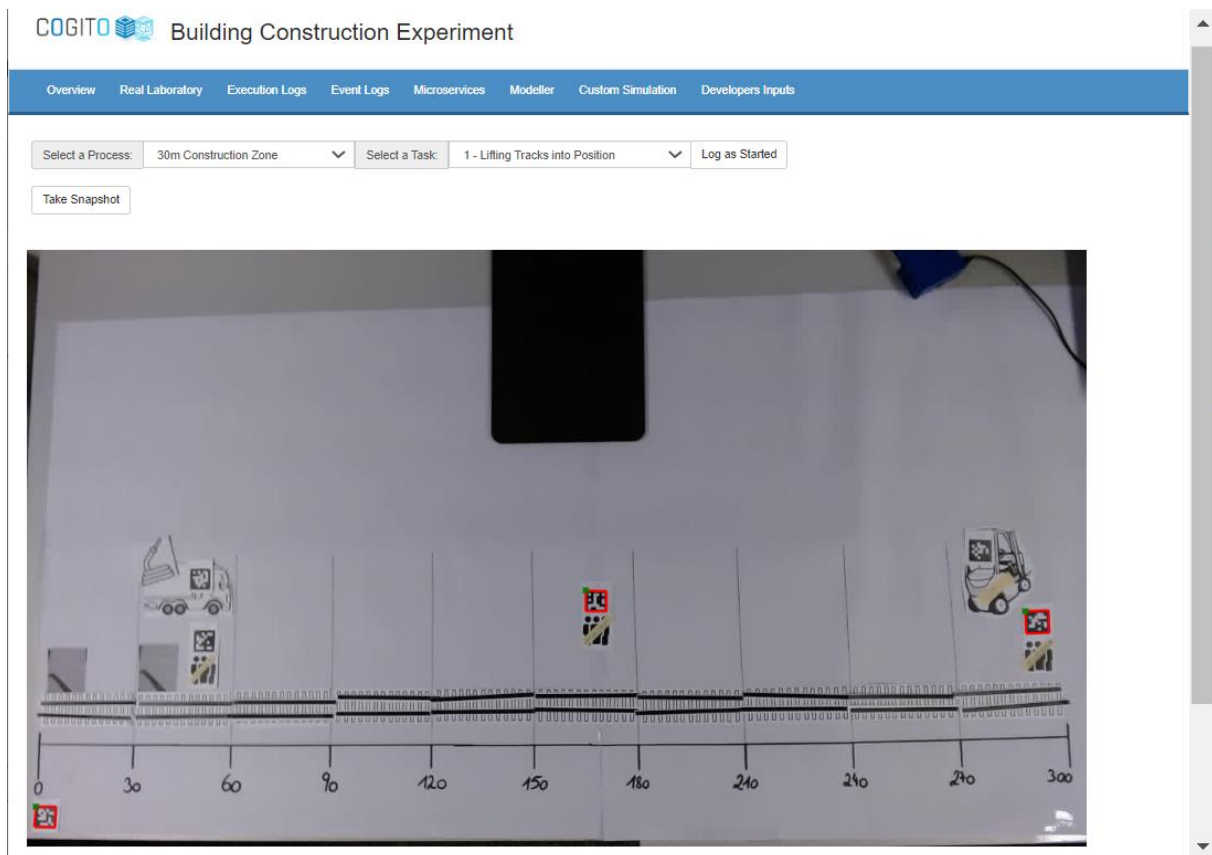


Figure 28 - Construction Experiment Entry Page

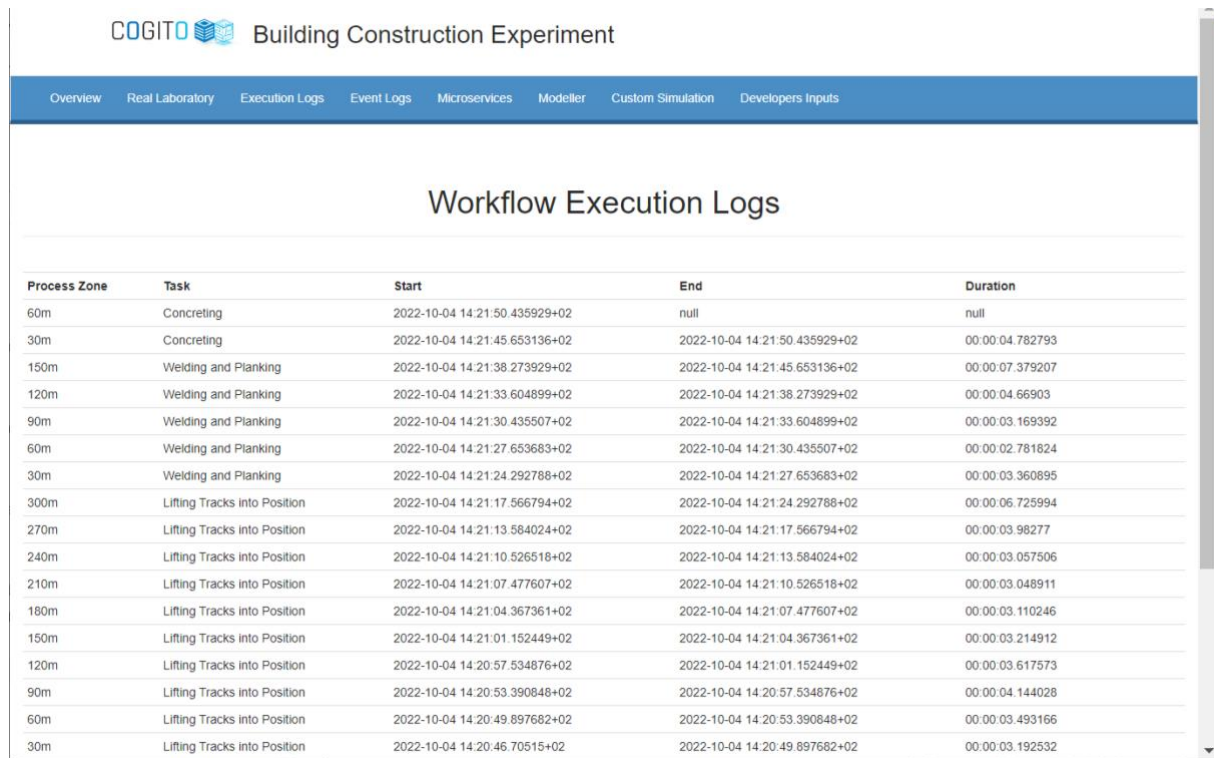
So-called “event logs” are introduced to depict the current situation at the construction site. By tracking the location of specific code markers attached to workers and equipment assets, the status in the construction areas is depicted. The position of the recognized markers, highlighted in red, is evaluated every time the snapshot button is pressed and information on their current zone is stored.

Through the event log page, the list of workers and equipment and their location is reported (Figure 29) with their most recent historical positions. At the time when the screenshot was taken, the concreting and the welding team were in the construction section, as well as the concrete mixing truck.

Event Logs				
Workers	Time	Latitude	Longitude	Zone
Concrete Workers Team	2022-10-04 14:54:42.884155+02	48.198504	16.365991	60m
Welding Workers Team	2022-10-04 14:54:42.867583+02	48.19847	16.366451	180m
Equipment				
Concrete Mixer Truck	2022-10-04 14:54:42.898453+02	48.198427	16.365978	60m
History				
Workers	Time	Latitude	Longitude	Zone
Concrete Workers Team	2022-10-04 14:54:42.884155+02	48.198504	16.365991	60m
Welding Workers Team	2022-10-04 14:54:42.867583+02	48.19847	16.366451	180m

Figure 29 - Equipment and Worker Logging

All of the event log data is captured in the form of timestamps in a timeseries database. Currently, the physical experiment automatically takes a snapshot of the actual status in the construction section to identify the workers and the equipment at a specific moment. Microservices are introduced to retrieve the data stored in the timeseries database and present the captured data on dashboards. Figure 30 presents a dashboard showing the execution logs of the workflow – e.g. used for estimating the status of the construction process in a specific construction section – simulated by taking snapshots of the construction section.



Process Zone	Task	Start	End	Duration
60m	Concreting	2022-10-04 14:21:50.435929+02	null	null
30m	Concreting	2022-10-04 14:21:45.653136+02	2022-10-04 14:21:50.435929+02	00:00:04.782793
150m	Welding and Planking	2022-10-04 14:21:38.273929+02	2022-10-04 14:21:45.653136+02	00:00:07.379207
120m	Welding and Planking	2022-10-04 14:21:33.604899+02	2022-10-04 14:21:38.273929+02	00:00:04.66903
90m	Welding and Planking	2022-10-04 14:21:30.435507+02	2022-10-04 14:21:33.604899+02	00:00:03.169392
60m	Welding and Planking	2022-10-04 14:21:27.653683+02	2022-10-04 14:21:30.435507+02	00:00:02.781824
30m	Welding and Planking	2022-10-04 14:21:24.292788+02	2022-10-04 14:21:27.653683+02	00:00:03.360895
300m	Lifting Tracks into Position	2022-10-04 14:21:17.566794+02	2022-10-04 14:21:24.292788+02	00:00:06.725994
270m	Lifting Tracks into Position	2022-10-04 14:21:13.584024+02	2022-10-04 14:21:17.566794+02	00:00:03.98277
240m	Lifting Tracks into Position	2022-10-04 14:21:10.526518+02	2022-10-04 14:21:13.584024+02	00:00:03.057506
210m	Lifting Tracks into Position	2022-10-04 14:21:07.477607+02	2022-10-04 14:21:10.526518+02	00:00:03.048911
180m	Lifting Tracks into Position	2022-10-04 14:21:04.367361+02	2022-10-04 14:21:07.477607+02	00:00:03.110246
150m	Lifting Tracks into Position	2022-10-04 14:21:01.152449+02	2022-10-04 14:21:04.367361+02	00:00:03.214912
120m	Lifting Tracks into Position	2022-10-04 14:20:57.534876+02	2022-10-04 14:21:01.152449+02	00:00:03.617573
90m	Lifting Tracks into Position	2022-10-04 14:20:53.390848+02	2022-10-04 14:20:57.534876+02	00:00:04.144028
60m	Lifting Tracks into Position	2022-10-04 14:20:49.897682+02	2022-10-04 14:20:53.390848+02	00:00:03.493166
30m	Lifting Tracks into Position	2022-10-04 14:20:46.70515+02	2022-10-04 14:20:49.897682+02	00:00:03.192532

Figure 30 - Execution Logs

3.5 Licensing

All software created for the Sandbox Experiment component is released as Open Source with MIT License. The source code of the experiment applications is available at <https://git.boc-group.eu/cogito/omitag-js-cogito> while all the materials for setting up the experiment, including figures and execution videos, are available at: <https://git.boc-group.eu/cogito/experiment-material>.

3.6 Installation Instructions

The first Sandbox Experiment component is publicly available and accessible under: <https://innovation-laboratory.org/experiments/building-construction/overview/>.

The advanced Sandbox Experiment component is publicly available and accessible under: <https://innovation-laboratory.org/experiments/building-construction/overview2/>.

Instructions to deploy the entry portal of the experiment are available in the BOC GitLab space: <https://git.boc-group.eu/cogito/building-construction-experiment-online>.

Instructions to deploy the docker image of the code markers recognition service are available at: <https://git.boc-group.eu/cogito/omitag-js-cogito-server-docker>.

3.7 Development and integration status

As planned in D6.3, in this second release of the PMS Sandbox Experiment, the construction processes used are aligned with the ones decided for the pre-validation pilot site. Further improvements are planned in future alignment of the PMS Sandbox Experiment with the second pre-validation use case and the pilot sites and will be reported in the relevant deliverables of WP8.

Additional prototypes have been evaluated in the context of PMS Sandbox Experiment component related to the management of material status in construction site. This mainly involve:

- Evaluation of image detection algorithms to visually identify the status and estimate the quantity of a specific material in the construction site.
- Evaluation of scale sensors to measure the quantity of a specific material in the PMS Sandbox Experiment.
- Introduction of safety elements in the PMS Sandbox Experiment (like barriers) and evaluation of their correct positioning before the task is executed.

3.8 Requirements Coverage

In the following tables the technical, functional, non-functional requirements as defined in COGITO D2.4 and the stakeholder requirements as defined in COGITO D2.1 are reported with focus only on the support provided by the PMS Sandbox Experiment component.

Table 6 - Requirements on Computing Systems for the PMS Sandbox Experiment component

ID	Description: <i>The Computing solution...</i>	Type	Priority	Status
COGI-CS-1	[DCC, WODM, PMS, BC, SafetyConAI, VirtualSafety, gQC] Runs on desktop or laptop PC	• Operational	Must	The service part of the experiment is accessible via browser on desktop or laptop PC
COGI-CS-4	runs on Windows	• Operational	Must	Supported as fully browser based
COGI-CS-5	runs on Mac	• Operational	Could	Supported as fully browser based
COGI-CS-6	runs on Android	• Operational	Would	Supported as fully browser based
COGI-CS-7	allows access to the whole data in one location	•Operation •Functional •Design constraint	Must	All the collected data are available in a single timeseries database

Table 7 - Requirements about Workflow Planning, Execution and Monitoring for the PMS Sandbox Experiment component

ID	Description: <i>The Workflow Planning, Execution and Monitoring solution ...</i>	Type	Priority	Status
----	---	------	----------	--------

COGI-WF-28	facilitates pre-construction training sessions (e.g., by using BIM models in augmented reality)	<ul style="list-style-type: none"> •Operational •Functional •Design constraint 	Would	The experiment is used to reason and explain the construction process
COGI-WF-29	facilitates swift tool adoption by easily available video tutorials and other learning materials online	<ul style="list-style-type: none"> •Operational 	Would	The experiment recorded videos can be uses as training material for the construction process

3.9 Assumptions and Restrictions

No assumptions or restrictions are applicable for the Sandbox Experiment component.

4 PMS – Simulation Component

This chapter demonstrates the Simulation component of the COGITO PMS module, used to provide a time and cost estimation of the construction process and generate data for the Optimisation component.

4.1 Prototype Overview

The Simulation component is a toolkit that allows the simulation of time and costs related to a construction process taking as input the process from the Modelling component and some model-specific simulation parameters from an Excel file. The simulation uses a Discrete Event Simulation approach, where the process is interpreted as a directed graph and the time that a token needs to pass the directed graph is measured [3]. For the explanation of the principle, we solely focus on the time, although other parameters can also be simulated [4]. Different types of distributions are used for (a) the generation of the execution time of each activity, combining multiple input factors and (b) the choice probability for each path. In particular normal distributions are commonly associated with execution times of activities, indicating a variability over an average value, while discrete distributions are associated with the choices indicating which path has to be followed. Based on the normal distribution of the activities' duration and the discrete distribution of the path selection, the execution time of the process could vary.

In our simulation a finer-grained calculation of probabilities and distributions is introduced through the use of a weighted sum of different parameters of interest, such as (but not limited to) the average execution time of previous executions, the expected weather conditions or an estimation from a domain expert based on the number of available workers (Figure 31).

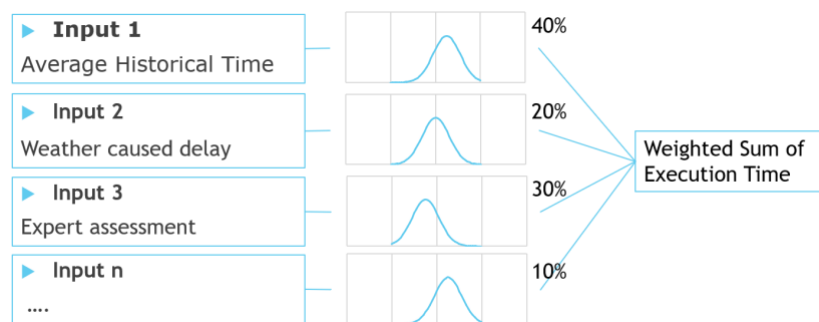


Figure 31 - Simulation Mechanisms for Multi-Distributions

Each of the input parameters is described using a mathematical distribution; the weighted sum is used to calculate to what extent the various effects are considered. Experts' (such as construction domain experts) input is hence needed to clarify (a) the **mathematical distribution** and (b) the different **weights** for the net sum. Both parameters are initially estimated and then continuously improved through observations during real execution and collaborations among experts.

The input parameters used for the simulation are provided in the form of an input Excel file aligned with the corresponding construction process model in BPMN format.

For each simulation run, a unique identifier with a corresponding start time (column B of Figure 32) is provided. Figure 32 introduces a simple simulation where a default deviation (column D) is applied to the provided time (column C), the latter being provided in milliseconds for each activity. A loose coupling concept is utilized here for the implicit mapping of semantics onto the tasks. In particular, the process tasks are mapped to the tab "C_TASK".

	A	B	C	D	E	F	G
1	4-Piles and Caps	2021-11-03T07:00:00	1248571	default			
2	5-Floor Slab (Floor 0)	2021-11-04T07:00:00	1443325	default			
3	7-Emergency staircase shaft	2021-11-05T07:00:00	1428403	default			
4	12-Concrete Rd Columns	2021-11-06T07:00:00	1263329	default			
5	13-Beams and Floor Slab (Floor 2)	2021-11-07T07:00:00	1195490	default			
6	15-Emergency staircase shaft	2021-11-08T07:00:00	1374937	default			
7	16-Concrete Rd Columns	2021-11-09T07:00:00	1340765	default			
8	17-Beams and Roof Slab	2021-11-10T07:00:00	1417374	default			
9	20-Piles and Caps	2021-11-11T07:00:00	1087054	default			
10	21-Floor Slab (Floor 0)	2021-11-12T07:00:00	1086269	default			
11	Check Dependency (8-Concrete Rd Columns)	2021-11-03T07:00:00	1318518	default			
12	Wait (8-Concrete Rd Columns)	2021-11-04T07:00:00	1322633	default			
13	23-Concrete Rd Columns	2021-11-05T07:00:00	1327836	default			
14	24-Beams and Floor Slab (Floor 1)	2021-11-06T07:00:00	1318873	default			
15	26-Concrete Rd Columns	2021-11-07T07:00:00	1068819	default			
16	27-Beams and Floor Slab (Floor 2)	2021-11-08T07:00:00	1063354	default			

Figure 32 - Process Task List for Simulation

The execution time for each activity is the result of the combination of different inputs. Figure 33 shows a realisation of the aforementioned weight-based distributions; it consists of several columns each representing the specific input combined to calculate a weighted sum of the expected deviation for each activity. Currently, Excel's implementation of the standard normal distribution is used to generate random probabilities and times, but more sophisticated tools can be used to fill the Excel sheet. Each value is multiplied by its corresponding weight to get the weighted value, which is then converted from milliseconds to the time unit appropriate for the use case.

		C		D				H				L			
		Weighted calculated time (ms)	Weighted calculated time (minutes)	Average Historical Time				Weather Related Delay Time				Expert Estimated Time			
				Calculated	Weight	Mean	Deviation	Calculated	Weight	Mean	Deviation	Calculated	Weight	Mean	Deviation
3	4-Piles and Caps	1632000	27.20	15	0.40	20	5	30	0.20	30	10	38	0.40	40	5

Figure 33 - Calculation of weighted Time considering Risks

As soon as the inputs are ready the simulation can start. Once completed, the results are visualised in two different forms:

- An overview of times and cost with path probabilities and generic information of the process (Figure 34).
- A detailed view in the form of an execution log that can also be used to perform a comparison with the real workflow execution (Figure 35).

Please select the file containing the model to simulate and press the Simulate button. Supported file format is BPMN.

General results

	Measure	Details
Average Cost:	0.00	
Max Cost:	0.0	Trace: t.1
Min Cost:	0.0	Trace: t.1
Total Costs:	0.00	
Average Executions Time:	00:000:02:48:20	
Max Executions Time:	00:000:03:21:23	Trace: t.1
Min Executions Time:	00:000:02:22:33	Trace: t.2
Total Executions Time:	00:001:04:03:28	
Total Runs:	10	
Total Traces:	2	
Total Paths:	2	

Paths Infos

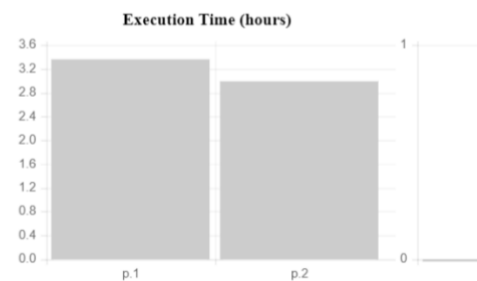
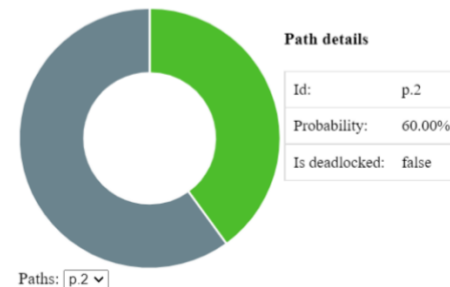


Figure 34 - Simulation General Results

The detailed result is provided in an Excel sheet in the same format as the Excel file used for the input data – same tabs and columns but also including details on the simulated starting and execution times. Having the same output format as the input format allows to run cascading simulations and use the results from one simulation to following simulations.

	A	B	C	D	E	F	G	H	I	J	K
1	Run-1	4-Piles and Caps	2021-11-03T07:21:55.598	1315598							
2	Run-1	5-Floor Slab (Floor 0)	2021-11-03T07:43:31.863	1296265							
3	Run-1	7-Emergency staircase shaft	2021-11-03T08:05:23.177	1311314							
4	Run-1	12-Concrete Rd Columns	2021-11-03T08:28:22.518	1379341							
5	Run-1	13-Beams and Floor Slab (Flo	2021-11-03T08:49:59.275	1396757							
6	Run-1	15-Emergency staircase shaft	2021-11-03T09:13:17.846	1398571							
7	Run-1	16-Concrete Rd Columns	2021-11-03T09:35:58.408	1358562							
8	Run-1	17-Beams and Roof Slab	2021-11-03T09:58:19.328	1342920							
9	Run-1	20-Piles and Caps	2021-11-03T10:21:23.353	1384025							
10	Run-1	21-Floor Slab (Floor 0)	2021-11-04T07:21:07.353	1267353							
11	Run-1	Check Dependency (8-Concre	2021-11-04T07:43:54.490	1366696							
12	Run-1	Wait (8-Concrete Rd Columns	2021-11-04T08:07:53.850	1439016							
13	Run-1	23-Concrete Rd Columns	2021-11-04T08:30:03.178	1330113							
14	Run-1	24-Beams and Floor Slab (Flo	2021-11-04T08:53:07.213	1384035							
15	Run-1	26-Concrete Rd Columns	2021-11-04T09:14:13.120	1265907							
16	Run-1	27-Beams and Floor Slab (Flo	2021-11-04T09:36:19.497	1326377							
17	Run-1	29-Concrete Rd Columns	2021-11-04T09:58:08.734	1309237							
18	Run-1	30-Beams and Roof Slab	2021-11-05T07:20:02.272	1202272							

Figure 35 - Simulation Detailed Results

4.1.1 Architecture

The core of the simulation uses Petri Net logics to simulate processes provided in BPMN2.0 formats and is flexible enough to support the simulation of other kind of models through the definition of appropriate mapping rules to Petri Net. The service is provided as REST API with a graphical HTML client that shows the results in a user-friendly way.

The service takes as input the BPMN process to simulate and an Excel file containing simulation inputs specific to the process and returns the results in the form of generic statistical indexes (i.e., average, minimum and maximum execution time) in the Web Client UI; detailed reports of each simulation run can be downloaded as an Excel file.

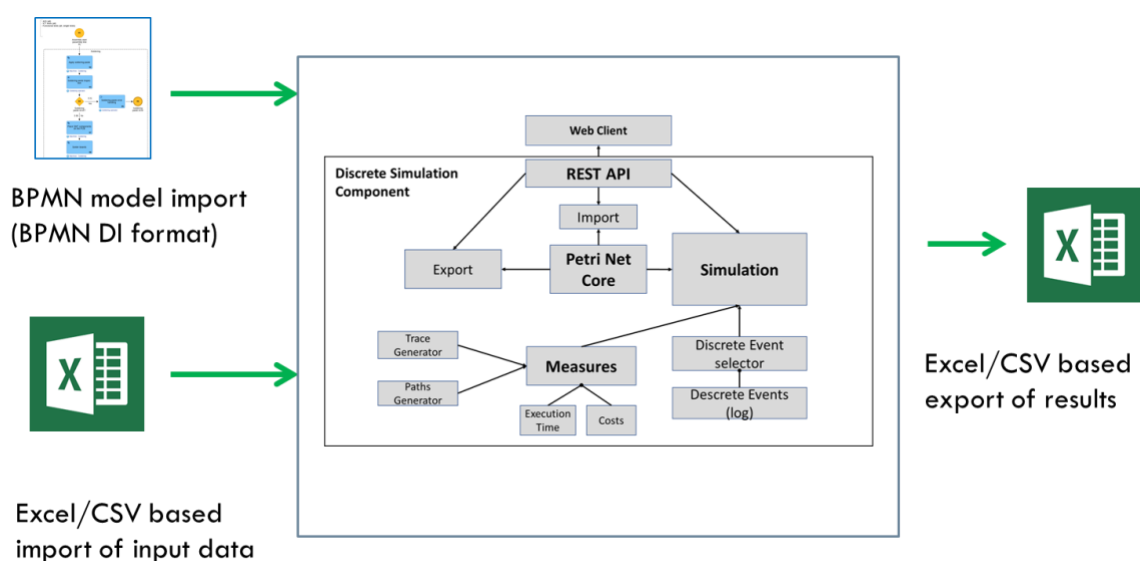


Figure 36 - Simulation Engine Architecture

The main components comprising the simulation tool are the following:

The **petri net core module** is the component that contains the main logic of a petri net and manages its semantics. The simulation service uses this component to evaluate at each step what transition can be enabled.

The **import module** is an easy-to-extend component that automatically recognizes the format of the provided model and converts it into the internal petri net structure. It manages document parsing and object mapping logic separately to reuse the same mapping logic for multiple file formats. It is also responsible for associating the input from the Excel sheet to the right BPMN object.

The **export module** is for diagnostic purposes only. It gives the possibility to export the internal petri net structure in Petri Net Modelling Language (PNML) standard format, to be visualised by any supported editor.

The **simulation measures module** is an easy-to-extend component that allows the definition of listeners for the simulation event. Each listener produces a measure or a result from a single simulation i.e., a trace, a path, the waiting times, or the execution costs. The resulting indexes can then be collected in a special container to calculate some final indexes (such as average values).

The **discrete event selector module** is the component that performs the selection of the transition to be executed among the available ones. The module provides a base mechanism that performs a fair choice between parallel transitions and a user-defined probabilistic choice between concurrent transitions. The base mechanism has also been extended to support dynamic probability evaluation using a scripting system.

The **simulation module** is the component that manages all the simulations, invoking the functionalities of the simulation measures module and the transition choice. It is also responsible for the generation of the simulation output in a structured format.

The **REST API** is the module responsible for exposing the features of the Simulation component to the external world. It is responsible for processing the input simulation parameters and generating the simulation results in the service specific output format.

The **Web Client** is a simple web interface that allows the user to simulate a BPMN model and visualise the simulation results in terms of charts and tables, communicating directly with the REST API.

4.2 Technology Stack and Implementation Tools

The Simulation component is a web-based application written in Java and deployed as a docker image.

Table 8 - Libraries and Technologies used in the PMS Simulation Component

Library/Technology Name	Version	License
Java OpenJDK	8	GNU GPL
Docker	1.8	Free

4.3 Input, Output and API Documentation

The Simulation component provides only one REST API endpoint that performs the simulation of the BPMN process provided as input with the respective Excel file for the configuration parameters:

- processSimulation
Method: POST.
Inputs: multipart composed of the BPMN and Excel file.
Output: multipart composed of an XML file containing general simulation results and an Excel file with details for each activity.

The service is not exposed to other COGITO modules but it is used internally by the PMS Simulation component.

4.4 Application Example

In order to demonstrate the PMS Simulation component, the school construction BPMN process described in Section 2.4 is used. An Excel file has been created for this process in order to provide the required simulation input parameters; it has been populated with the output of microservices described in Section 6.4. In particular, the average execution time of each process task has been calculated by a microservice extracting the historical data from the experiment logs as in Section 3.4, the weather prediction comes from a microservice retrieving data from a public forecast service and the expert assessment comes from a domain-expert interview.

In order to enable the processing of models for instance by running simulation algorithms, a manual transformation may be required as a pre-processing step. This manual transformation is intended to enable the semantic compliance of the model with the BPMN standard. Two main transformation steps are in particular conducted:

1. Check & Wait (see Figure 37) – Check and wait tasks including a decision are introduced to ensure BPMN compliance by modelling dependencies in a straightforward and easy understandable way. The complexity coming with dependencies and interrelationships is reduced following this approach. For instance, by getting rid of several interdependent arrows the visual appearance of the process schedule is clearer.
2. Parallelism (see Figure 38) – In case of parallel activities, the parallelism is introduced to mark the beginning and the end of the parallel tasks clearly. The visual representation using parallel

gateways is on the one hand clearer (as only using arrows without gateways) and provides on the other hand a formal structure allowing for the application of algorithms run on top of the BPMN model.

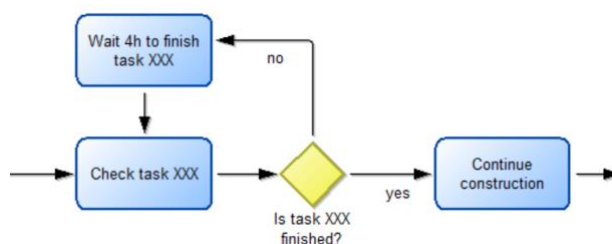


Figure 37 - Introduce Wait and Check Tasks including a Decision

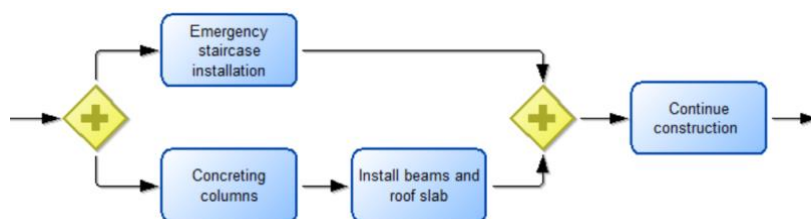


Figure 38 - Introduce Parallelism

In particular, the introduction of targeted formal modelling structures facilitates the systematic recognition of security issues and KPIs based on the models, as well as allows the application of advanced processing algorithms on top of the models such as the simulation described in this section. The following model (Figure 39) results after the transformation of the initial school sample process imported from the COGITO DTP and presented in Section 2.4.1.

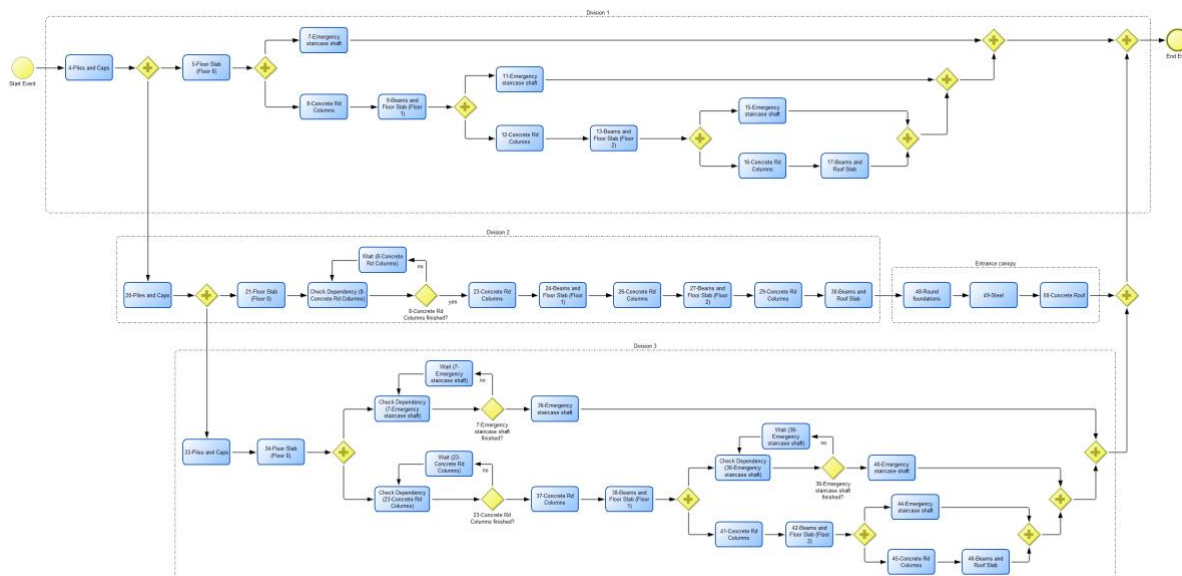


Figure 39 - Transformed Process Model compliant with BPMN

The discrete event simulation is applied by interpreting the process as a directed graph: the construction process BPMN is transformed into a petri net. Normal distributions are applied on the activities related to times and costs, while discrete distributions are applied on decisions such as the decision whether an additional layer with waterproofing sheets is required. Advanced probability and distribution calculations are introduced by applying the concept of a “weighted net sum”. Parameters of interest such as historical data providing the base distribution for the expected time, an estimation on the weather conditions causing

a delay in the construction and the expert assessment are each converted to a distribution assigned with a weight and used as input for the calculated net sum.

The simulation demonstration is accessible from the Sandbox experiment portal section related to the custom simulation menu (Figure 40), providing as input the school BPMN process model and the relative Excel file with the simulation parameter.

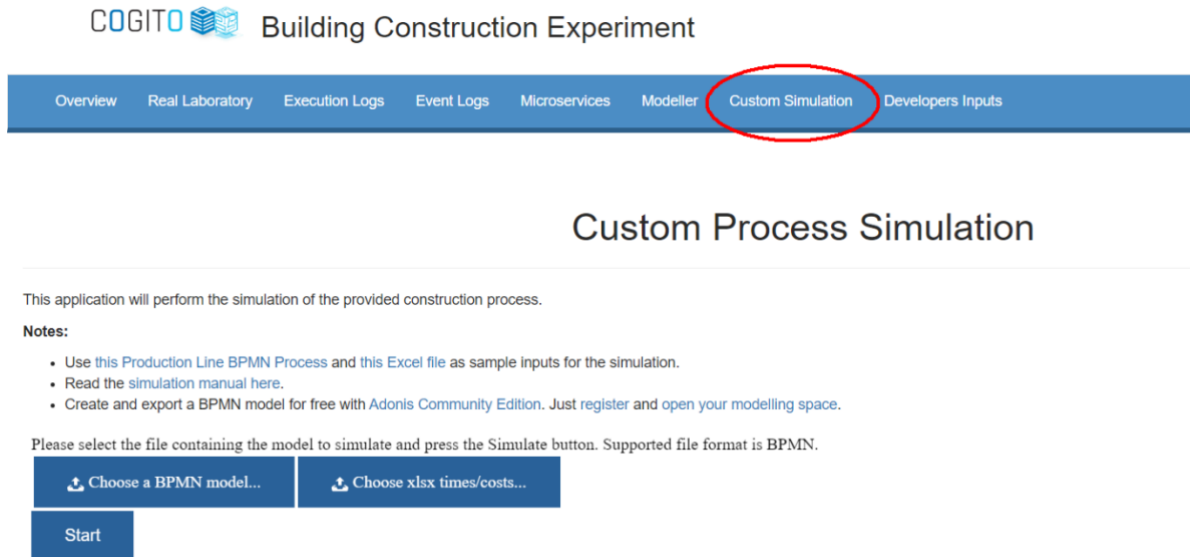


Figure 40 - Custom Simulation Page

4.5 Licensing

The Simulation component is provided as open source with MIT License. All the source code is available in the GitLab repository <https://git.boc-group.eu/adoxx/knowledge-based-model-simulation>.

4.6 Installation Instructions

The Simulation component is accessible from the Sandbox Experiment component entry page (<https://innovation-laboratory.org/experiments/building-construction/overview2/>) under the “Custom Simulation” menu item.

Instructions to deploy the Simulation component from the docker image are available here: <https://git.boc-group.eu/adoxx/knowledge-based-model-simulation-docker>.

4.7 Development and integration status

As mentioned in the first version of this deliverable (D6.3), in Chapter 6 a specific microservice has been introduced to support the collection of historical execution data from COGITO WODM module and provide as input for tuning the simulation parameters. Additionally, to support the simulation of construction processes imported from the COGITO DTP, some refinement steps have been evaluated and described in Section 4.4.

Finally, in order to improve the accessibility of the PMS Simulation component a prototype has been evaluated in form of an integrated interface with the PMS Modelling component and a configurable mobile interface. The services prototyped for this new functionality have been briefly introduced in Section 6.7.

4.8 Requirements Coverage

In the following tables the technical, functional and non-functional requirements as defined in COGITO D2.4 and the stakeholder requirements as defined in COGITO D2.1 are reported with focus only on the support provided by the PMS Simulation component.

Table 9 - Requirements on Computing Systems for the PMS Simulation component

ID	Description: <i>The Computing solution...</i>	Type	Priority	Status
COGI-CS-1	[DCC, WODM, PMS, BC, SafetyConAI, VirtualSafety, gQC] runs on desktop or laptop PC	• Operational	Must	Supported: The simulation service is accessible via browser on desktop or laptop PC
COGI-CS-4	runs on Windows	• Operational	Must	Supported as fully browser based
COGI-CS-5	runs on Mac	• Operational	Could	Supported as fully browser based
COGI-CS-6	runs on Android	• Operational	Would	Supported as fully browser based
COGI-CS-7	allows access to the whole data in one location	• Operation • Functional • Design constraint	Must	Supported: The simulation uses the models defined in the Modelling Component

Table 10 - Requirements about Workflow Planning, Execution and Monitoring for the PMS Simulation component

ID	Description: <i>The Workflow Planning, Execution and Monitoring solution ...</i>	Type	Priority	Status
COGI-WF-6	[PMS, WODM, DCC] allows the PM and SM to efficiently detect and prioritise delays and cost escalation elements	• Functional	Should	Supported
COGI-WF-7	[PMS, WODM, DCC] allows the PM to extract reports about: project time performance, project cost performance, costs per unit, resource consumption	• Functional	Must	Supported
COGI-WF-20	[PMS] incorporates risk prediction and critical path finding during scheduling	• Functional	Could	Supported
COGI-WF-22	[PMS] supports conflict predictions and solving during scheduling (e.g.,	• Functional	Could	Supported through the simulation of multiple to-be scenarios and the

	networks relocation before excavation work)			support of the PMS Sandbox experiment.
--	---	--	--	--

Table 11 - Functional, Non-Functional Requirements and Interfaces for the PMS Simulation component

Functional and Non-Functional Requirements			Achievements
Functional	Req-1.1	Construct workflow and simulation model and populate with historical data	Supported
	Req-1.2	Update simulation model using real-time data from WODM	Supported: the microservice can retrieve data from the COGITO WODM
Non-Functional	Req-1.7	Web-based app	Supported
	Req-2.2	Scalability	Supported via Docker Swarm. The simulation service does not share session information and can be scaled horizontally on heavy loads deploying it with Docker Swarm
	Req-2.3	Security	Supported: The simulation service is isolated from the system via docker containerization. Inputs are processed in order to avoid service exploitation

4.9 Assumptions and Restrictions

In the demonstration of the Simulation component the Excel input file creation is a semi-automatic process using data extracted from services, i.e., retrieving weather conditions and average execution times from historical execution.

5 PMS – Optimisation Component

In this chapter the PMS Optimisation component is described using the example described in Figure 18 of Section 2.4. This version of the Optimisation component aims to optimise the allocation of workers for the process execution.

5.1 Prototype Overview

In this section a resource allocation model for assigning workers and equipment to activities has been developed in order to support the PMS Modelling component in the refinement of the construction process. A construction project consists of a number of activities that need to be completed in a pre-specified order by a set of workers with particular skills using particular pieces of equipment. In any project there is a choice to be made about how many workers and how much equipment to assign to each activity. In general, the more workers/equipment assigned to a given activity the faster that activity can be completed, but there is a corresponding increase in cost. The PMS Optimisation component determines the number of each type of worker/equipment to assign to each activity to minimise a function of the cost and duration of the project.

To do this optimisation we need to know, for each activity:

- The types of workers/ equipment required to complete the activity.
- The cost of workers/ equipment of each type.
- The maximum available number of workers/ equipment of each type.
- The relationship between the number of workers/equipment assigned to a particular activity and the duration of that activity (for example, this may be a multivariate function which takes as input the number of workers/equipment of each type needed for the activity). We have currently assumed a specific function for this relationship (see Eq. 5.1 and associated text) but this will need to be defined by the user of the component. Options include the use of existing datasets, such as the RSMeans¹⁰ data, dynamic updating of the function as real-time data is collected and use of expert judgement, e.g., by eliciting appropriate relationships from those involved in the relevant activities.

The input data described above will be provided by the PMS Modelling component, including a workflow of activities that need to be carried out and the order of these activities.

In this section, we describe the current version of the PMS Optimisation component. This version only considers construction projects within which the activities occur in series. A specific function for the relationship between the number of workers and the duration of the activity is assumed based on prior discussion with the use case partners. For simplicity we will also describe the case where only the workforce is considered (i.e. no equipment). It is straightforward to extend this to include equipment – this requires a function to be specified that describes the relationship between the number of workers, the equipment and the duration of the task. Alternatively the workers and equipment can be combined together into a crew. One crew represents the minimum number of workers and pieces of equipment required to complete the task. The Optimisation component can then be used to optimise the number of crews assigned to each task.

The objective of the mathematical model developed in this module is to optimally allocate the required resources (in the simple case presented this is only the workforce) to each activity minimising a utility function considering the time (total duration of the project) and the total project cost.

This optimisation model is based on some assumptions which are listed below:

- Each worker only has one skill, but each activity may need several types of skill.
- We assume that if multiple workers are assigned to one activity, each worker spends an equal amount of time on that activity. We also assume that no activities occur in parallel (see discussion on extensions in Section 5.7), so there are no constraints associated with workers being busy with other tasks.

¹⁰ <https://www.rsmeans.com/>

- For each activity we are given the total work required from each type of worker (e.g., in m²) and the productivity of each type of worker (e.g., in m² per hour). The duration of the activity is assumed to be a function of the number of workers, the total work and the productivity.

Based on discussions with the use case partners our understanding is that when taking decisions about the planning of construction projects there is often a trade-off between the total duration and the cost of the project. The expectation is that assigning additional workers to a construction project will reduce the overall duration but will also increase the cost. Moreover, we do not expect the duration of all the activities to depend linearly on the number of workers (e.g., double the number of workers will not translate to half the time taken). To reflect this, we will use the following formula (Eq. 5.1) to relate the length of time a particular type of worker spends on an activity to the number of workers of that type assigned to the activity (using input data on productivity of different types of workers, total work required per activity, and number of available workers):

$$Duration = \frac{[Total\ work]}{[number\ of\ workers] \times [productivity]} \times \alpha^{[number\ of\ workers]-1} \quad (5.1)$$

This equation includes a multiplier term $\alpha^{[number\ of\ workers]-1}$ (in which α is a constant greater than one) which increases the duration when additional workers are added to the task. The constant α represents the diseconomy of scale in the activity, so that the duration does not decrease linearly with the number of workers. As described above, this constant could be estimated using historical data or expert judgement but we also expect that some sensitivity analysis will need to be carried out - an example of this is given in Section 5.4. For some tasks this relationship will not be appropriate, for example if additional workers do not decrease the time taken at all (e.g., in case a driver is needed to drive a single vehicle). The crucial input required by the PMS Optimisation component is the relationship between the number of workers and the duration of an activity so another specification that provides this relationship can be substituted in place of Eq. 5.1 if available. In future work such alternatives will be considered and ideally, duration functions for activities will be estimated from the available data. Eq. 5.1 applies only when the number of workers of a given type required to complete an activity is greater than zero. In case no workers of a given type are required, the duration is zero.

In the following, a mathematical model for the allocation of workforce is presented in which we seek to minimise a function of the project's cost and time.

• Nomenclature for model

Parameters

Pr_{jp}	Productivity of workforce with skill p for conducting activity j
CW_p	Cost for workforce with skill p (per person per hour)
TW_{jp}	Total amount of work for the task associated with worker type p in activity j
$NW_{j,p}^{Min}$	Minimum required number of workers with skill p for activity j
$NW_{j,p}^{Max}$	Maximum available number of workers with skill p for activity j
X_{jp}	Zero-one matrix indicating if activity j needs the skill p "1" otherwise "0"
w_1/w_2	Weights associated with each objective

Decision variables

TC	Total cost
Dr_{jp}	Duration of the job associated with the skill p for activity j
TDr_j	The duration of the longest job for activity j

NW_{jp} Number of workers with skill p for activity j

Indices

$j:\{1,...,J\}$ Activities

$p:\{1,...,P\}$ Groups of workers with different skills

$n:\{1,...,N\}$ Index associated with the piecewise linear approximation

The objectives in the proposed model are as follows:

(1) Minimization of the labour cost

$$\text{Total cost} = \sum_{p=1}^P \sum_{j=1}^J (NW_{jp} \times CW_p \times Dr_{jp}) \quad (5.2)$$

This labour cost is given by the number of each type of worker assigned to each activity multiplied by the amount of time worked on that activity and the cost per unit of time for that type of worker.

(2) Minimization of the total duration

In this version of the model, we have assumed that no activities occur in parallel. Given this assumption, the total duration of the construction project is given by (Eq. 5.3)

$$\text{Total Duration} = \sum_{j=1}^J TDr_j = \sum_{j=1}^J \text{Max}_p(Dr_{jp}) \quad (5.3)$$

i.e., the sum of the durations for each activity, where the duration for each activity is given by the maximum time needed for each type of worker to complete their given task.

Therefore, the objective function includes both the total duration and the total cost as follows (Eq. 5.4):

$$\text{Min} \left\{ w_1 \times \left[\sum_{j=1}^J TDr_j \right] + w_2 \times \left[\sum_{p=1}^P \sum_{j=1}^J (NW_{jp} \times CW_p \times Dr_{jp}) \right] \right\} \quad (5.4)$$

w_1 and w_2 are weights allowing the relative importance of time and cost to be varied. The weight w_1 is multiplied by the total duration given in (Eq. 5.4) and the weight w_2 is multiplied by the total cost given in (Eq. 5.3). Varying the values of w_1 and w_2 allows decision-makers to change the relative importance of time or cost in the equation, which may be especially useful when allocating resources in response to a delay in one or more activities. The aim of the Optimisation component is to allocate the workers to activities to minimise the weighted sum of the total duration and the total cost given in (Eq. 5.4).

• Duration constraints

The activity duration is calculated based on the following formula (Eq. 5.5), in which α is a constant. The duration of a type of worker's activity is set to zero if that type of worker is not required. More information on this constraint is given in (Eq. 5.1).

$$\begin{cases} Dr_{jp} = \frac{TW_{jp}}{NW_{jp} \times Pr_{jp}} \times \alpha^{NW_{jp}-1} & X_{jp} = 1 \\ Dr_{jp} = 0 & X_{jp} = 0 \end{cases} \quad (5.5)$$

The duration for each activity is equal to the duration of the longest job (considering that different jobs require different types of workers) for completing that activity, so that $TDr_j = \text{Max}_p(Dr_{jp})$. As the objective function minimises a function of time and cost, this is equivalent to the following set of inequalities (Eq. 5.6):

$$TDr_j \geq Dr_{jp} \text{ for all } j, p \quad (5.6)$$

- **Number of workers constraints**

The number of workers for each activity should be equal to or lower than the maximum available number of workers of different types for each activity. When a type of worker is not required for an activity, NW_{jp} should be zero. This information is represented in the equations below (Eq. 5.7).

$$\begin{cases} 1 \leq NW_{jp} \leq NW_{jp}^{Max} & X_{jp} = 1 \\ NW_{jp} = 0 & X_{jp} = 0 \end{cases} \quad (5.7)$$

To speed up the implementation of the PMS Optimisation component we developed a piecewise linear approximation of the mixed integer nonlinear program (MINLP) described above. It is usually computationally quicker to solve a linear approximation than a nonlinear program. The duration constraint in (Eq. 5.5) may be reformulated as a set of linear constraints, taking advantage of the convexity of the objective. Moreover, as we are only interested in integer solutions for the number of workers, if this piecewise linear approximation is exact at the integer solutions then it is exactly equivalent to the original MINLP described in Eq. (5.2)-(5.7) – except that we now have a mixed integer linear program (MILP) with a larger number of constraints. The description of this MILP is set out below.

We define two parameter arrays “h” and “g” that reformulate the duration constraint as linear constraints. These constraints are designed to encode the same information about the duration of the activities as (Eq. 5.5):

$$\begin{aligned} h_{jpn} &= \frac{TW_{jp}}{n_{jp} \times Pr_{jp}} \times \alpha^{n_{jp}-1} \\ g_{jpn} &= \frac{TW_{jp}}{Pr_{jp}} \times \alpha^{n_{jp}-1} \end{aligned} \quad (5.8)$$

We also define a new variable $Afunc_{jp}$ as

$$Afunc_{jp} = NW_{jp} \times Dr_{jp} = \frac{TW_{jp}}{Pr_{jp}} \times \alpha^{NW_{jp}-1} \quad (5.9)$$

Then the reformulated piecewise linear model is as follows:

$$\text{Min} \left\{ w_1 \times \left[\sum_{j=1}^J TDr_j \right] + w_2 \times \left[\sum_{p=1}^P \sum_{j=1}^J (CW_p \times Afunc_{jp}) \right] \right\} \quad (5.9)$$

$$Afunc_{jp} \geq g_{j,p,n} + (NW_{jp} - n_{jp})(g_{j,p,n+1} - g_{j,p,n}); \quad NW_{jp}^{Min} \leq n \leq NW_{jp}^{Max} - 1 \quad (5.10)$$

$$Dr_{jp} \geq h_{j,p,n} + (NW_{jp} - n_{jp})(h_{j,p,n+1} - h_{j,p,n}); \quad NW_{jp}^{Min} \leq n \leq NW_{jp}^{Max} - 1 \quad (5.11)$$

$$TDr_j \geq Dr_{jp}; \quad \forall j, p \quad (5.12)$$

$$NW_{jp}^{Min} \leq NW_{jp} \leq NW_{jp}^{Max} \quad (5.13)$$

These equations (5.8-5.13) are reformulations of equations (5.2-5.7) so the modelling assumptions are the same as described in the previous section. This MILP model is implemented on GAMS (General Algebraic Modelling System) [5] and solved through a BARON¹¹ Solver [6]. GAMS is a general solver and as such can be used for a wide variety of optimisation problems. The optimality of solutions in the simple version of the model (i.e., considering only the workforce) is checked by the total enumeration of all possible solutions implemented in R [7].

¹¹ Branch-And-Reduce Optimization Navigator (BARON)

5.2 Technology Stack and Implementation Tools

Currently the optimisation model has been specified in GAMS (General Algebraic Modelling System), a specialised high-level environment for specifying mathematical optimisation problems including linear, nonlinear, and mixed-integer programs. Such modelling languages allow the specification of the model on the computer in essentially the same form as the paper specification above – the modelling language then combines the model specification with the data for a specific instance and passes them to the chosen solver.

Unless there are very particular aspects of a problem's structure that one wishes to exploit (not the case here), it will rarely be more effective to write one's own solver code than to use a commercial or open solver code written by experts. Here we have used the *BARON* solver.

For the practical implementation of the PMS Optimisation component within the overall digital twin, it will be necessary to identify the most appropriate software environment for the optimisation model. The advantages of using a modelling language such as GAMS for specifying optimisation problems are overwhelming compared to hard-coding/combining model structure and data for a specific model. However, it would be easy to re-code the model in an equivalent environment other than GAMS including free options such as Pyomo (within Python) or Jump (within Julia) should this make the combination of code engineering and licensing easier.

Table 12 - Libraries and Technologies used in the PMS Optimisation component

Library/Technology Name	Version	License
GAMS	38.1.0	One-year free trial
R	4.0.5	Open source

5.3 Input, Output and API Documentation

The PMS Optimisation component is exposed in the form of a REST microservice using the PMS Integration component described in Section 6.4. The services will not be exposed publicly to other COGITO components but only internally to the PMS Modelling component. The input and output data fields in the current version of the Optimisation component are as follows:

Input

- Cost for workforce/ equipment per specialty (per person per unit of time)
- Total amount of work per activity (e.g., m²)
- Minimum required number of workers/equipment per specialty for conducting each activity
- Maximum available number of workers/equipment per specialty for conducting each activity
- Weights for objectives (w_1 and w_2) which represent the relative importance of time or cost in the objective function
- Duration function to relate the number of a given type of workers/equipment assigned to an activity to the length of that activity

Output

- Total duration
- Total cost
- Duration of the job associated with each skill for every activity
- The duration of the longest job for each activity
- Number of workers/ equipment with each skill for each activity

For the inputs, data for the model can be provided either from historical data or experts' opinions. For example, it may be possible to obtain the worker costs and the total amount of work for each activity from historical data of comparable construction projects. Where estimates are needed that are bespoke to the construction project being considered these may be set in consultation with the foreman of the project or others with specific knowledge of the workflow. Another possibility is to estimate the inputs dynamically – so that as the construction project proceeds estimates of input quantities can be updated. This may be

particularly of use when estimating the duration functions. The maximum/minimum required number of workers and weights for objectives can be determined based on decision makers' requirements.

5.4 Application Example

To test the optimisation algorithm, we used the sample construction project related to the pre-validation site about building of a railway line, for which example data and workflow schedule of activities are available (see Figure 18 of Section 2.4). The problem requires the optimal number of workers of different types to be allocated to different activities in order to minimise a function of cost and time. A workflow schedule is provided in the form of a BPMN process including the following activities:

- Marking of work areas according to plans by a foreman and measurement operator.
- Digging the ground and applying the sand layer by diggers under the supervision of a civil engineer.
- Application of waterproof sheaths by diggers under the supervision of a civil engineer.
- Construction of drainage and soakaway by diggers under the supervision of a civil engineer.
- Filling of soakaway with gravel by diggers under the supervision of a civil engineer.
- Compacting of gravel with roller by a roller operator.
- Finishing confirmed by foreman.

The input data for this use case provided in the form of an Excel file is presented below. Table 13 displays data on the available workforce and their cost. Table 14 shows which workers with different skills are required for each activity. Table 15 displays the productivity of different types of workers for each activity. Table 16 shows the maximum available workers of each type for each activity. The total work required for all activities is equal to 100 m².

Table 13 – Available workforce and their cost

Teams	Maximum available number of workers	Cost per Hour (£)
Foreman	1	50
Measurement Operator	3	20
Civil Engineers	1	30
Digger Operator	10	20
Roller Operator	5	20

Table 14 – Type of workers required for each activity

TASK	Team				
	Foreman	Measurement Operator	Civil Engineers	Digger Operator	Roller Operator
Marks according Plan	✓	✓	-	-	-
Digging the Ground	-	-	✓	✓	-
Apply Sand Layer	-	-	✓	✓	-

Build Drainage	-	-	✓	✓	-
Build Soakaway	-	-	✓	✓	-
Fill with Gravel	-	-	✓	✓	-
Compact with Roller	-	-	-	-	✓
Finishing	✓	-	-	-	-

Table 15 – Productivity of workers for each activity (m² per h)

TASK	Team				
	Foreman	Measurement Operator	Civil Engineers	Digger Operator	Roller Operator
Marks according Plan	50	30	-	-	-
Digging the Ground	-	-	10	10	-
Apply Sand Layer	-	-	10	10	-
Build Drainage	-	-	10	10	-
Build Soakaway	-	-	10	10	-
Fill with Gravel	-	-	10	10	-
Compact with Roller	-	-	-	-	10
Finishing	10	-	-	-	-

Table 16 - Maximum available workers per type for each activity

	Team				
	Foreman	Measurement Operator	Civil Engineers	Digger Operator	Roller Operator
Marks according Plan	1	3	-	-	-
Digging the Ground	-	-	1	10	-
Apply Sand Layer	-	-	1	10	-
Build Drainage	-	-	1	10	-
Build Soakaway	-	-	1	10	-
Fill with Gravel	-	-	1	10	-
Compact with Roller	-	-	-	-	5
Finishing	1	-	-	-	-

Table 17 and Table 18 show the results obtained from running our optimisation model using GAMS, with weights $w_1=0.8$ and $w_2=0.2$ (where w_1 and w_2 represent the relative importance of time or cost respectively). The weights should ideally be determined through interviews with the decision-makers associated with a project. In this demonstration of our initial optimisation model our choice of values is arbitrary. To determine how the weights affect our results, we carried out a small-scale sensitivity analysis using different values for w_1 and w_2 . The results of this analysis are displayed below (see Table 17 and Table 18 for $w_1=0.8$ and $w_2=0.2$ and Table 19 and Table 20 for $w_1=0.2$ and $w_2=0.8$). These results illustrate that when a higher weight is considered for cost (and consequently a lower weight for duration), the number of roller operators decreases to 1 and the duration increases to 10 (from 5.5). This is as expected – as the relative importance of time decreases, it is more cost-effective to reduce the number of workers.

Table 17 - Total duration (h) for each activity carried out by different types of workers (where $w_1=0.8$ and $w_2=0.2$)

Total duration (h) for each activity	Team					Total Duration [h]
	Foreman	Measurement Operator	Civil Engineers	Digger Operator	Roller Operator	
Marks according Plan	2	3.333	-	-	-	3.333
Digging the Ground	-	-	10	10	-	10
Apply Sand Layer	-	-	10	10	-	10

Build Drainage	-	-	10	10	-	10
Build Soakaway	-	-	10	10	-	10
Fill with Gravel	-	-	10	10	-	10
Compact with Roller	-	-	-	-	5.5	5.5
Finishing	10	-	-	-	-	10

Table 18 - Number of workers per type for each activity (where $w_1=0.8$ and $w_2=0.2$)

Number of workers	Team				
	Foreman	Measurement Operator	Civil Engineers	Digger Operator	Roller Operator
Marks according Plan	1	1	-	-	-
Digging the Ground	-	-	1	1	-
Apply Sand Layer	-	-	1	1	-
Build Drainage	-	-	1	1	-
Build Soakaway	-	-	1	1	-
Fill with Gravel	-	-	1	1	-
Compact with Roller	-	-	-	-	2
Finishing	1	-	-	-	-

Table 19 - Total duration (h) for each activity carried out by different types of workers (where $w_1=0.2$ and $w_2=0.8$)

Total duration (h) for each activity	Team					Total Duration [h]
	Foreman	Measurement Operator	Civil Engineers	Digger Operator	Roller Operator	
Marks according Plan	2	3.333	-	-	-	3.333
Digging the Ground	-	-	10	10	-	10
Apply Sand Layer	-	-	10	10	-	10
Build Drainage	-	-	10	10	-	10

Build Soakaway	-	-	10	10	-	10
Fill with Gravel	-	-	10	10	-	10
Compact with Roller	-	-	-	-	10	10
Finishing	10	-	-	-	-	10

Table 20 - Number of workers per type for each activity (where $w_1=0.2$ and $w_2=0.8$)

Number of workers	Team				
	Foreman	Measurement Operator	Civil Engineers	Digger Operator	Roller Operator
Marks according Plan	1	1	-	-	-
Digging the Ground	-	-	1	1	-
Apply Sand Layer	-	-	1	1	-
Build Drainage	-	-	1	1	-
Build Soakaway	-	-	1	1	-
Fill with Gravel	-	-	1	1	-
Compact with Roller	-	-	-	-	1
Finishing	1	-	-	-	-

We also carried out a sensitivity analysis for different weights in the objective functions and different values of α (originally set to 1.1 in the objective function). The effect that varying α has on the total duration and cost of the project are reported in Figure 41 and Figure 42. To examine the effect of varying w_1 , w_2 and α together, Figure 41 and Figure 42 show the relationship between values of α and total duration and cost for different weights.

We can see from Figure 41 and Figure 42 that the results are sensitive to both the weights and the choice of α . As α increases we would expect both the total duration and the total cost to increase as the number of workers on each activity is held constant, as increasing the value of α reduces the contribution an extra worker makes in reducing the duration of an activity and hence increases the cost. The jump observed for $w_1=0.8$ and $w_2=0.2$ corresponds to an increase in the number of workers assigned to the activities – this has the effect of reducing the duration but increasing the cost. This jump is not observed when $w_1=0.2$ and $w_2=0.8$ because the increased weight assigned to the cost means that increasing the number of workers assigned to activities is not optimal. Note that for $w_1=0.2$ and $w_2=0.8$ the model assigns the minimum number of workers to each activity for α greater than 1.05, hence there is no variation in the results beyond this point. These results highlight the importance of carefully choosing the weights to represent the relative valuation of time and cost.



Figure 41 - Total Duration for different values of α (for TW=100 m²)

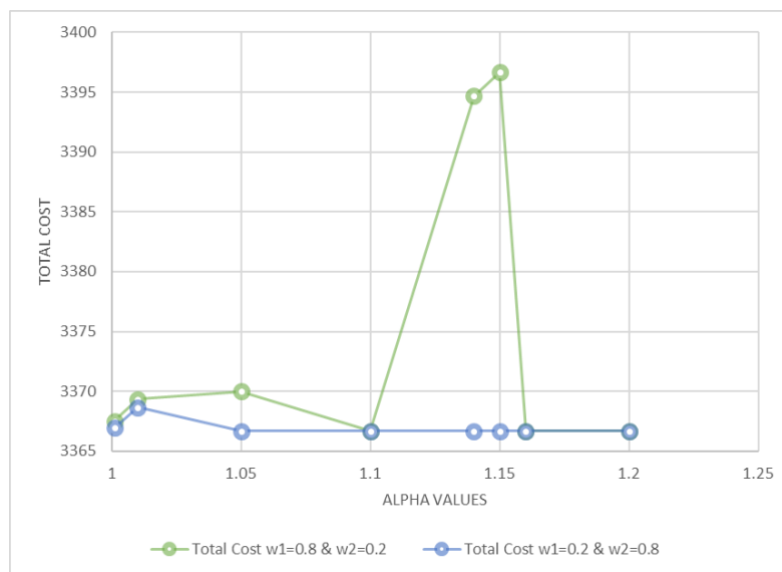


Figure 42 - Total Cost for different values of α (for TW=100 m²)

5.5 Licensing

The current version of the PMS Optimisation component is released as an open-source solution.

5.6 Installation Instructions

A standalone version of the microservice for the PMS Optimization component is available at the following url: <https://git.boc-group.eu/cogito/olive-aio-uedin-gams>.

In order to run the service the following steps are needed:

- 1) Install GAMS from https://www.gams.com/try_gams/ being sure to have it available in the folder "C:\GAMS\39\gams.exe".

- 2) Download and unpack the zip archive of the PMS Optimization component microservice from <https://git.boc-group.eu/cogito/olive-aio-uedin-gams/-/tags/v1.0>.
- 3) Execute the "1-Start Tomcat.bat" file.
- 4) Test the microservice by open the link "2-Open Olive Microservice Controller Management UI".

5.7 Development and integration status

A version of the Optimisation component that can be implemented with parallel tasks is currently in progress as part of the integrated COGITO solution that will be reported in the relevant deliverables of WP8. In addition to the inputs to the current version this will use the precedence relations of different activities to ensure that activities are completed in the correct order. Once this version is completed it will be demonstrated on the School Project use case.

In addition to the next steps identified by the assumptions as described in Section 5.9, the following points are possible extensions to the current optimisation model for future releases. We aim to identify which of these extensions are most critical in collaboration with the use case partners:

- Refining the form of the task duration function according to available data and expert opinions from the COGITO partners.
- Including constraints or additional costs that arise as a result of working patterns. Currently the Optimisation model assumes that workers can be deployed or demobilised as necessary without any additional costs incurred (other than the direct cost of their time). We can investigate methods that better reflect the realities of workforce mobilisation on a construction site, e.g. where employing workers on a short timescale may be more expensive.
- Other resources (e.g., material): as materials are a consumable, adding material to the model will require additional constraints and information on consumption rates and replenishment policies.
- Other objectives: indirect costs (as a multiplication of the project-duration by the indirect cost per day) can be added to the objective function.
- Possible uncertainties in productivity rates: productivity measurements that consider the effects of different driver variables are crucial to the successful completion of a construction project and so we aim to include uncertainty in these measurements in the optimisation model.

We will soon be able to take a view on whether consideration of parallel tasks combined with more detailed consideration of costs and availabilities of workers can be done as a direct extension of the compact optimization formulation used so far, or whether it will be necessary to use an approach requiring a larger number of variables (for instance having binary flags indicating whether each task starts at each possible time).

5.8 Requirements Coverage

In the following tables the technical, functional, non-functional requirements as defined in COGITO D2.4 and the stakeholder requirements as defined in COGITO D2.1 are reported with focus only on the support provided by the PMS Optimisation component.

Table 21 - Requirements on Computing Systems for PMS Optimisation component

ID	Description: <i>The Computing solution...</i>	Type	Priorit y	Status
COGI-CS-1	[DCC, WODM, PMS, BC, SafetyConAI, VirtualSafety, gQC] Runs on desktop or laptop PC	• Operational	Must	Supported: The optimisation service is released as REST service accessible via browser

				or client on desktop or laptop PC
COGI-CS-4	runs on Windows	• Operational	Must	Supported as fully REST based
COGI-CS-5	runs on Mac	• Operational	Could	Supported as fully REST based
COGI-CS-6	runs on Android	• Operational	Would	Supported as fully REST based
COGI-CS-7	allows access to the whole data in one location	•Operation •Functional •Design constraint	Must	Supported: The Optimisation component uses models stored in the Modelling component
COGI-CS-8	maintains communication and data security	•Legal •Design constraint	Must	Supported: The access to the PMS Optimization component is allowed only internally by the Modelling component
COGI-CS-9	differentiates data and system access levels and modification rights	•Legal •Functional •Design constraint	Must	Supported: All the created models are accessible only to the authorized users in the Modelling component

Table 22 - Requirements about Workflow Planning, Execution and Monitoring for the PMS Optimisation component

ID	Description: <i>The Workflow Planning, Execution and Monitoring solution ...</i>	Type	Priority	Status
COGI-WF-6	[PMS, WODM, DCC] allows the PM and SM to efficiently detect and prioritise delays and cost escalation elements	•Functional	Should	In progress: the schedule analysis detects possible costs optimisation
COGI-WF-22	[PMS] supports conflict predictions and solving during scheduling (e.g., networks relocation before excavation work)	•Functional	Could	In progress: the schedule analysis in the optimisation helps in the detection of invalid scheduling

Table 23 - Functional, Non-Functional Requirements and Interfaces for the PMS Optimisation component

Functional and Non-Functional Requirements			Achievements
	Req-1.6	Output updated estimates of project progress to WODM according to real-time data and performed optimisation	In progress: The integration with the modelling environment (responsible to communicate with WODM) is in progress
Non-Functional	Req-1.7	Web-based app	Supported: the component is released as REST service

	Req-2.1	User-friendly	Supported: the user does not interface directly to the PMS Optimization component but call it from the PMS Modelling component
	Req-2.2	Scalability	Supported: the REST service is deployed in a Docker Swarm cluster for automatic scalability on load
	Req-2.3	Security	Supported: the access to the optimisation REST service is granted only to the modelling component

5.9 Assumptions and Restrictions

Related to the Optimisation component, the following assumptions are in place:

- In this report we have provided an Optimisation model with a structure based on discussions with use case partners regarding how cost and duration depend on the resources deployed. Next steps will include elicitation from domain experts in more detail of how cost and duration depend on resources, including efficiency gains/losses from additional resources, how jobs can be carried out in series or parallel, and the extent to which one can estimate parameters in the model from external sources (e.g. RSMeans data).
- The assumptions made in the proof-of-concept model are set out in the model specification provided above.

6 PMS – Microservice Integration Component

This chapter demonstrates the Microservice Integration component of the PMS, provided in the form of a microservice framework used to create all the services and integration endpoints for the PMS components described in this deliverable and their communication with other COGITO modules, in particular with the WODM and DTP.

6.1 Prototype Overview

The Microservice Integration component of the PMS relies on the microservice framework OLIVE¹², used as the basis for the development of all PMS services and functionalities related to the integration between its internal and external components as well as its integration with the COGITO Digital Twin Platform. The OLIVE framework allows to create model-aware web applications through configuration of existing components both for the back-end and the front-end side. For the back-end side such components are named Connectors and their configuration results in ready to use REST microservices. For the front-end side such components are named Widgets and their configuration results in a multi-channel, ready to use, user interface. Both the Connectors and the Widgets are part of the OLIVE platform that can be extended using plug-ins depending on the needs. In the Microservice Integration component of the PMS, only the back-end functionalities of the OLIVE platform are used for the generation of the microservices needed by the different PMS components.

Connectors provide the functionality of back-end services enabling the connection to external systems like databases, the COGITO WODM module, and the COGITO DTP. The framework provides out-of-the-box connection to more than twenty services and data storage systems. As soon as a Connector is configured a new microservice instance is created, executed and exposed through a REST interface with standardized input and output format. The lifecycle of the service can then be controlled by the framework. Instantiated services can be orchestrated in case more complex business logic is involved. This configuration approach enables the definition of specific model types reflecting the microservice definition and the integration with the ADOxx platform, used as a base for the PMS Modelling component; it offers the possibility to have a complete environment where (a) the microservices and UI elements could be created starting from models and (b) models could be used by the microservices as data with recognised semantics, such as in the case of the PMS Simulation component.

Moreover, this integration allows ADOxx based modellers like the PMS Modelling component to communicate with the external world through a common interface in a bi-directional way. It is therefore possible to use the features of existing microservices to enrich the modelling platform and use models as data for microservices like in the PMS Simulation component.

6.1.1 Architecture

The Microservice Integration component uses the OLIVE Microservice framework as back-end component that allows to define and manage microservices following the configuration approach. A microservice in OLIVE is defined merely through the configuration of an existing platform component named Connector (Figure 43).

A Connector is a component developed in the form of an OSGi plug-in that provides a specific functionality, e.g., performs a query on a timeseries database or delivers a workflow model to the COGITO WODM module. The name Connector is derived from the fact that usually such functionalities depend on external systems (e.g., a database) and the Connector is responsible for connecting to such systems to exploit their features. As introduced in Figure 1, the communications with the COGITO WODM and with the COGITO DTP are managed by the PMS Microservice Integration component. For this purpose, two specific connectors have been created handling the connection to the WODM module for the upload of workflow/retrieval of the execution status and the connection and authentication to the DTP for project specific data retrieval respectively. An additional connector as also been introduced to support the communication with the PMS Optimization component described in Chapter 5. The ADOxx interface as also be updated with a specific

¹² <https://www.adoxx.org/live/olive>

connector for the community version of the ADOxx platform in order to support the integration with ADOxx based PMS Modelling environment as described in Section 2.1.1.2.

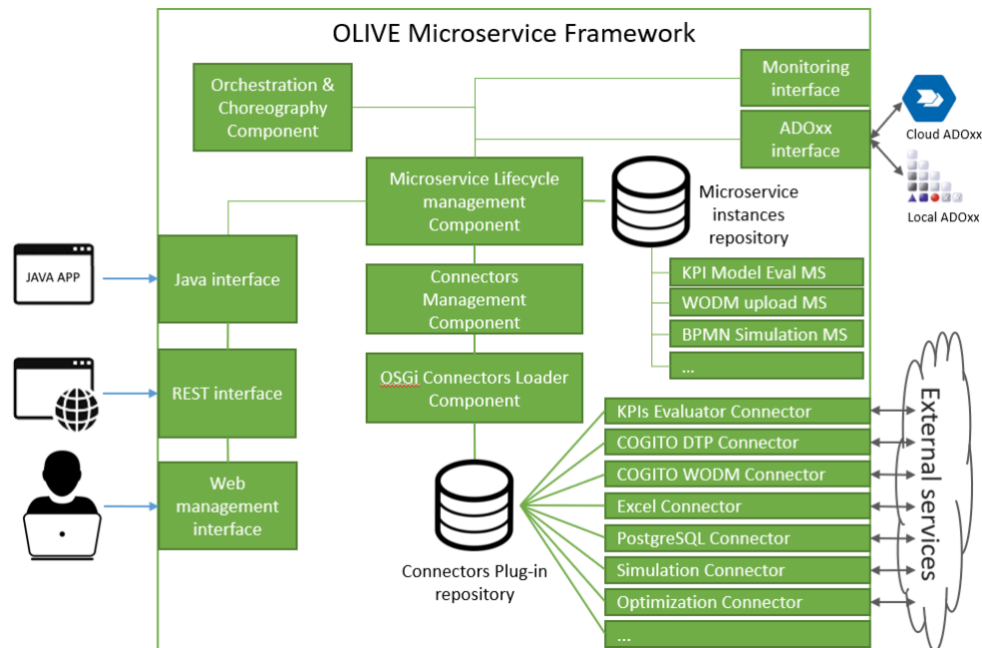


Figure 43 - OLIVE Microservice Controller Architecture

The OLIVE Microservice framework allows to manage the configuration of such Connectors, giving the possibility to create Microservices and control their whole lifecycle. It is the responsibility of the lifecycle management component to (1) generate an instance of the REST microservice from the configuration, (2) allow to start the microservice, (3) keep the microservice running in an isolated environment, (4) allow to stop the microservice and (5) allow to dismiss it.

The OSGi Connectors Loader component is responsible for loading all the Connectors and making them available to the platform. It is built on the OSGi framework Apache Felix¹³ and dynamically checks the presence of the OSGi bundles (plug-ins) defining Connectors, loading, and unloading them on request.

As soon as the microservices have been defined, they can be combined to achieve the business logic task thanks to the Orchestrator component. This component is in charge of combining existing microservices using the Enterprise Integration Pattern¹⁴ notation or alternatively, following a more programmatic approach, using the JavaScript scripting language.

The OLIVE Microservice framework exposes all this functionality both with Java and REST APIs. The former is used to integrate the OLIVE platform within a local desktop application. The latter is used to integrate the OLIVE platform with remote applications. Concerning the REST APIs, a management web user interface has been made available allowing to exploit all the features of the OLIVE Microservice framework through the web browser.

6.2 Technology Stack and Implementation Tools

The Microservice Integration component is a Java-based REST service built over the JAX-RS library. The component uses the Apache Felix framework for loading the Connectors; each Connector may rely on specific libraries to provide its own functionality. More particularly, the Connectors used for the definition of services described in Section 6.4 rely on Apache POI for services interacting with Excel files, on the PostgreSQL JDBC for services communicating with the Timescale database and on the Nashorn JS engine for

¹³ <https://felix.apache.org/>

¹⁴ <https://www.enterpriseintegrationpatterns.com/>

the services requiring data manipulation. Finally, the deployment of the solution is entirely based on provided Docker images.

Table 24 - Libraries and Technologies used in *PMS Microservice Integration component*

Library/Technology Name	Version	License
Java OpenJDK	8	GNU GPL
JAX-RS	2.35	GNU GPL v2
Apache Felix	6.0.3	Apache 2.0 License
Apache POI	5.1	Apache 2.0 License
PostgreSQL JDBC	42.3.1	Apache 2.0 License
Nashorn JS	15.3	GNU GPL

6.3 Input, Output and API Documentation

The Microservice Integration component provides REST APIs to create and control the lifecycle of each defined microservice. Such APIs are not exposed publicly but instead are used only by the OLIVE Microservice framework UI to allow the management of the services from the web management interface. Details on the APIs are available in the OLIVE Javadoc available at <https://git.boc-group.eu/adoxx/microservice-controller/-/tags/v1.0>.

6.4 Application Example

The Microservice Integration component is employed to create the services used by the PMS Modelling component to import and export the construction schedule (including BIM related data and resources) from and to the COGITO DTP and exchange the process model information internally with the PMS Simulation and PMS Optimization components. The PMS Simulation component uses services for the retrieval of the simulation input parameters and the generation of data for the PMS Optimisation component. The PMS Optimization component use services as input interface and for the management of the optimization execution lifecycle, while the PMS Sandbox Experiment component uses services for storing and retrieving tracked workers and equipment as well as for evaluating the process status. In particular the following services have been created:

- Authentication service: the microservice allows the SAML based authentication of the cloud version of the PMS Modelling component in the COGITO DTP.
- Process import service: the microservice uses the OLIVE REST and ADOxx Connectors to retrieve the process schedule from the COGITO DTP and generate the model in the PMS Modelling component.
- Process retrieval service: the microservice uses the OLIVE REST Connector to retrieve the list of all available processes from the PMS Modelling component.
- BPMN export service: the microservice uses the OLIVE REST Connector to communicate with the PMS Modelling component and returns the specified process in BPMN standard format.
- Process export service: the microservice uses the OLIVE REST and ADOxx Connectors to export, from the PMS Modelling component, the construction process model in the JSON format required by the COGITO DTP.
- H&S import service: the microservice use the OLIVE REST and ADOxx Connectors to retrieve from the COGITO DTP H&S issues and assign them to the specific task in the construction process.
- Equipment and Workers tracker service: the microservice uses the OLIVE PostgreSQL Connector to perform an insert query in a Timescale database for every recognized marker id in the experiment sandbox along with its location and timestamp.

- Equipment and Workers location retrieval service: the microservice uses the OLIVE PostgreSQL Connector to perform a select query in a Timescale database to retrieve all the markers in a specified area of the sandbox experiment.
- Process status retrieval service: the microservice returns the status of the process executed in the sandbox experiment by looking at the generated event logs.
- Execution status retrieval service: the microservice use the OLIVE REST Connector to retrieve from the COGITO WODM the status of running tasks in the construction process.
- Status Validation service: the microservice, using the results of the “process status service” and of “Equipment and Workers location retrieval service”, evaluate if the resources assigned to a specific task have been effectively used at task execution.
- KPI calculation service: the microservice uses the OLIVE Dashboard Connector to process a specific KPI model and evaluate all the KPIs defined.
- Weather forecast service: the microservice uses the OLIVE REST Connector to retrieve weather forecast data used by the simulation component in order to estimate a possible delay in the process execution.
- Average execution time service: the microservice calculates the average execution time of each process task based on the sandbox Experiment event logs that will then be used by the Simulation component as input parameters.
- Historical execution time service: the microservice use the OLIVE REST Connector to retrieve from the COGITO WODM the execution times of passed activities and export in CSV format to support the input in the simulation.
- Optimization execution service: the microservice use the OLIVE Shell Connector to execute and operate the PMS Optimization component scripts through the GAMS engine.
- Optimization input service: the microservice use the OLIVE ADOxx Connector to export from the PMS Modelling component, all the data needed by the PMS Optimization component in the required format.

All services can be visualized from the PMS Sandbox Experiment entry page under the “Microservices” menu (Figure 44). Here, the management page of the Microservice Integration component is available where each defined microservice can be controlled and also executed using the integrated test page.

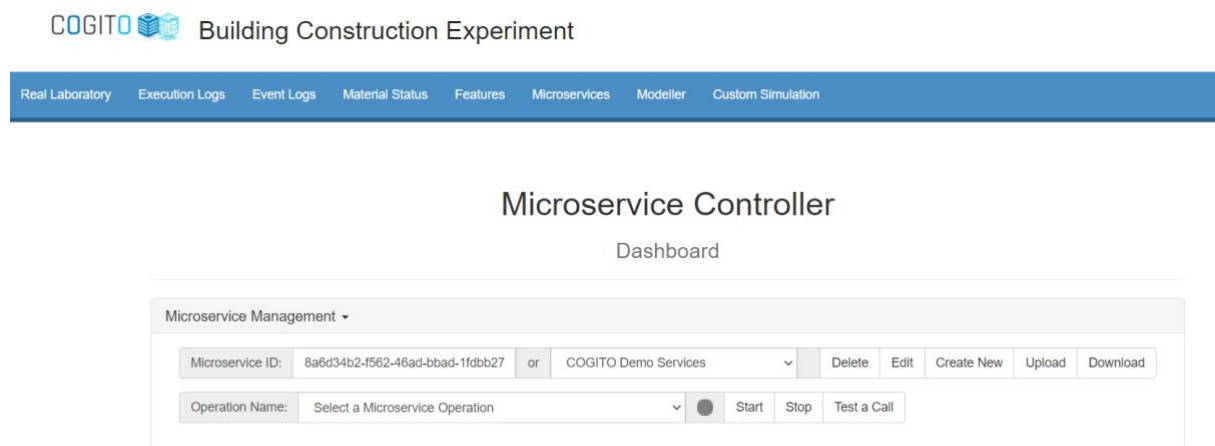


Figure 44 - Microservice Management Page

6.5 Licensing

The microservice framework OLIVE used in the Microservice Integration component is released as open source with MIT license. All the source code is available at <https://git.boc-group.eu/adoxx/microservice-controller-rest>.

6.6 Installation Instructions

The deployed instance of the Microservice Integration component is accessible from the experiment main page <https://innovation-laboratory.org/experiments/building-construction/overview/> (first version) or <https://innovation-laboratory.org/experiments/building-construction/overview2/> (advanced version) under the “Microservices” menu item.

Detailed instructions on how to deploy the component through a docker image are available in at: <https://git.boc-group.eu/adoxx/microservice-controller-docker/>.

6.7 Development and integration status

As introduced in D6.3, the microservices in this deliverable focus on the integration between the PMS components and the other COGITO modules, in particular the DTP. The PMS Integration component has been also extended in order to interface with the community version of the PMS Modelling environment and to connect with the community version of the Optimization engine GAMS giving an added value to the ADOxx.org¹⁵ community.

In addition, the following prototypes has been evaluated and tested as extension of the PMS Integration component to improve its accessibility and connection capabilities:

- Android APK generation connector: This connector support the configurable generation of accessibility UIs of the PMS Modelling and Simulation components, specific for Android devices, in form of ready to install APK files.
- Amazon Alexa connectors: This connector is used to support the PMS Modelling component in interacting with the users via vocal inputs through the Amazon Alexa services.

6.8 Requirements Coverage

In the following tables the technical, functional, non-functional requirements as defined in COGITO D2.4 and the stakeholder requirements as defined in COGITO D2.1 are reported with focus only on the support provided by the PMS Microservice Integration Component.

Table 25 - Requirements on Computing Systems for the PMS Microservice Integration component

ID	Description: <i>The Computing solution...</i>	Type	Priority	Status
COGI-CS-1	[DCC, WODM, PMS, BC, SafetyConAI, VirtualSafety, gQC] Runs on desktop or laptop PC	• Operational	Must	Supported: The service interface is accessible via browser on desktop or laptop PC
COGI-CS-4	runs on Windows	• Operational	Must	Supported as fully browser based
COGI-CS-5	runs on Mac	• Operational	Could	Supported as fully browser based
COGI-CS-6	runs on Android	• Operational	Would	Supported as fully browser based
COGI-CS-7	allows access to the whole data in one location	• Operation • Functional	Must	Supported: the microservices will not

¹⁵ <https://www.adoxx.org/live/home>

		•Design constraint		store data but retrieve from the COGITO DTP
COGI-CS-8	maintains communication and data security	•Legal •Design constraint	Must	Supported: authentication through COGITO DTP available
COGI-CS-9	differentiates data and system access levels and modification rights	•Legal •Functional •Design constraint	Must	Supported: authorization through COGITO DTP available

Table 26 - Requirements about Workflow Planning, Execution and Monitoring for the PMS Microservice Integration component

ID	Description: <i>The Workflow Planning, Execution and Monitoring solution ...</i>	Type	Priority	Status
COGI-WF-8	[PMS, WODM] uses BIM models to support scheduling and budgeting	•Functional	Should	Supported: The microservices for retrieving data from the COGITO DTP are available
COGI-WF-22	[PMS] supports conflict predictions and solving during scheduling (e.g., networks relocation before excavation work)	•Functional	Could	Supported: The microservices for the connection to the Optimisation component have been released
COGI-WF-23	[PMS, WODM] incorporates health and safety planning	•Functional •Design constraint	Should	Supported: The microservices for the retrieval of H&S issues from the COGITO DTP have been implemented

Table 27 –Functional, Non-Functional Requirements and Interfaces for the PMS Microservice Integration component

Functional and Non-Functional Requirements			Achievements
Functional	Req-1.1	Construct workflow and simulation model and populate with historical data	Supported: the microservice can retrieve data from both the Sandbox experiment and the COGITO WODM
	Req-1.2	Update simulation model using real-time data from WODM	Supported: the microservice can retrieve data from the COGITO WODM

	Req-1.3	Connect and Authenticate to the DT platform	Supported: the authentication through COGITO DTP is available
	Req-1.4	Estimates project progress	Supported: microservices use location information to evaluate the tasks process status.
	Req-1.5	Receive task progress from WODM	Supported: the microservice can retrieve data from the COGITO WODM
	Req-1.6	Output updated estimates of project progress to WODM according to real-time data and performed optimisation	Supported: microservices sending data to the COGITO DTP for later use by the WODM are available
Non-Functional	Req-1.7	Web-based app	Supported
	Req-2.1	User-friendly	Supported: The integration framework is a low-code platform where users can define microservices configuring existing component in a UI
	Req-2.2	Scalability	Supported: the services can be scaled horizontally on heavy loads thanks to the deployment in Docker Swarm clusters
	Req-2.3	Security	Supported: the authentication with the COGITO DTP is available.

6.9 Assumptions and Restrictions

No assumptions are applicable for the demonstration of the Microservice Integration component.

7 Conclusions

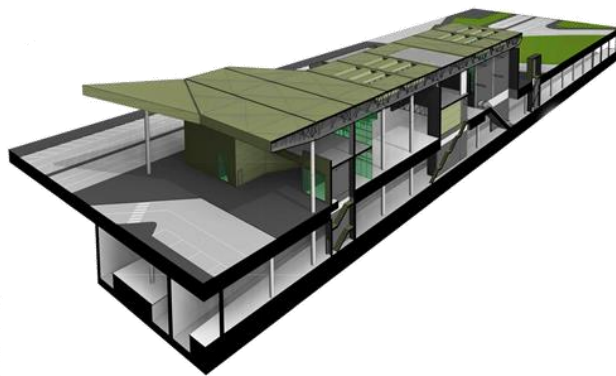
In this document, the final release of the COGITO PMS module is demonstrated focusing on the features of each component. In particular the following components have been demonstrated using a sample common for all the COGITO modules:

- Modelling component: import and generation of the construction processes, workflows, and KPI models.
- Sandbox Experiment component: collaborative reasoning and testing of the created models' features.
- Simulation component: time and cost simulation of the process.
- Optimisation component: evaluation of schedule alternatives and reorganizations that will optimise the process in terms of costs and times.
- Microservice Integration component: definition of all the microservices required by each component for integration with other PMS components and COGITO modules.

More details on finalized and on-going activities as well as potential next steps, are provided for each component in the Section related to “Development and Integration Status” (for Chapters 2, 3, 4, 5, and 6).

References

- [1] D. Norton and R. Kapla, "The Balanced Scorecard: Measures that Drive Performance," *Harvard Business Review* 70, no. 1, 1992.
- [2] R. Woitsch, "Industrial Digital Environments in Action: The OMiLAB Innovation Corner," *In Grabis J., Bork D. (Eds), The Practice of Enterprise Modelling*, vol. 13th IFIP Working Conference PeEM 2020, pp. 8-22, 2020.
- [3] G. S. Fishman, "Principles of discrete event simulation," John Wiley & Sons, Inc., USA, 1978.
- [4] K. Tumay, "Business process simulation," in *Proceedings Winter Simulation Conference*, 1996.
- [5] G. D. Corporation, "General Algebraic Modeling System (GAMS) Release 38.1.0," [Online]. Available: <https://www.gams.com/download/>. [Accessed 2021].
- [6] M. Tawarmalani and N. Sahinidis, "A polyhedral branch-and-cut approach to global optimization," *Mathematical Programming*, vol. 103(2), pp. 225-249, 2005.
- [7] R. C. Team, "R: A language and environment for statistical computing," R Foundation for Statistical Computing, 2021. [Online]. Available: <https://www.R-project.org/>.
- [8] ISO 16739, "Industry Foundation Classes (IFC) for data sharing in the construction and facility management industries (ISO 16739:2013)," CEN, 2016.
- [9] ISO 13790, "Energy performance of buildings — Calculation of energy use for space heating and cooling," 2008.



COGITO

CONSTRUCTION PHASE
DIGITAL TWIN MODEL

cogito-project.eu



This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 958310