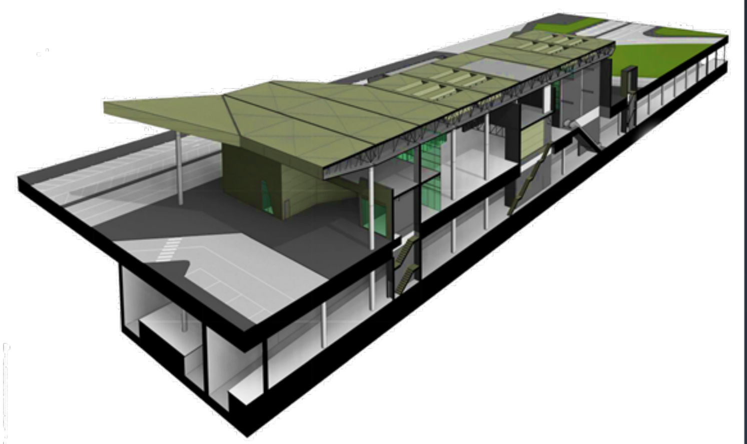
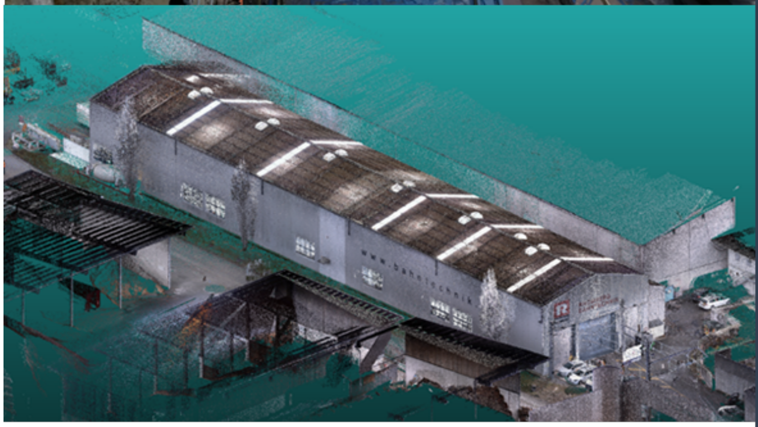




COGITO

CONSTRUCTION PHASE DIGITAL TWIN MODEL

cogito-project.eu

Adaptive Processes / Workflow Modelling and Simulation-based Optimisation Module v1

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

D6.3 – Adaptive Processes/Workflow Modelling and Simulation-based Optimisation Module v1

Dissemination Level:	Public
Deliverable Type:	Demonstrator
Lead Partner:	BOC-AG
Contributing Partners:	UEDIN, QUE, NT
Due date:	31-03-2022
Actual submission date:	31-03-2022

Authors

Name	Beneficiary	Email
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

Reviewers

Name	Beneficiary	Email
Apostolos Papafragkakis	Hypertech	a.papafragkakis@hypertech.gr
Georgios Lilis	UCL	g.lilis@ucl.ac.uk
Dimitrios Rovas	UCL	d.rovas@ucl.ac.uk

Version History

Version	Editors	Date	Comment
0.1	BOC-AG	01.12.2021	ToC
0.2	BOC-AG	21.02.2022	Demonstration of PMS Components
0.3	BOC-AG, UEDIN	24.02.2022	Integrated contribution from UEDIN
0.4	BOC-AG	10.03.2022	Final draft for internal review
0.5	BOC-AG	29.03.2022	Review comments addressed
1.0	BOC-AG	31.03.2022	Submission to the EC Portal

Disclaimer

Executive Summary

The current document demonstrates the first 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 while at the same time facilitating their continuous optimisation and refinement based on forward-looking simulation and real-time execution monitoring. The COGITO PMS module provides its spectrum of features through different components.

The Modelling component besides allowing the definition of a particular construction process, offers an environment where all related resources and metrics can be represented, namely the associated Key Performance Indicators (KPIs), the needed resources in the form of equipment and workers as well as the linkage to information available from Building Information Models (BIM) data as 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 modelling of construction-related processes is supported by a physical Sandbox Experiment component where the stakeholders can (a) collaboratively reason over the process via design thinking[1], (b) test the required digitisation strategies in an isolated environment and (c) execute the process in an abstracted version of the construction site that can generate sample log data; the latter could be 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, the Simulation component is used 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 are used at first, eventually followed by ones from the actual process execution.

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 is later going to 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 were demonstrated in this document using one particular use case, relevant also within the scope of the final pilot sites, i.e., related to the construction of a water drainage system for a railway track. The 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

List of Figures

List of Tables

List of Acronyms

Term	Description
BIM	Building Information Model
BPMN	Business Process Modelling Notation
CMS	Content Management System
COGITO	Construction Phase diGItal Twin mOdel
DES	Discrete Event Simulation
DTP	Digital Twin Platform
EIP	Enterprise Integration Pattern
GAMS	General Algebraic Modelling System
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
WODM	Workflow Management Automation Tools

1 Introduction

1.1 Scope and Objectives of the Deliverable

This deliverable demonstrates the first 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 and provide optimisation capabilities 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 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 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 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.

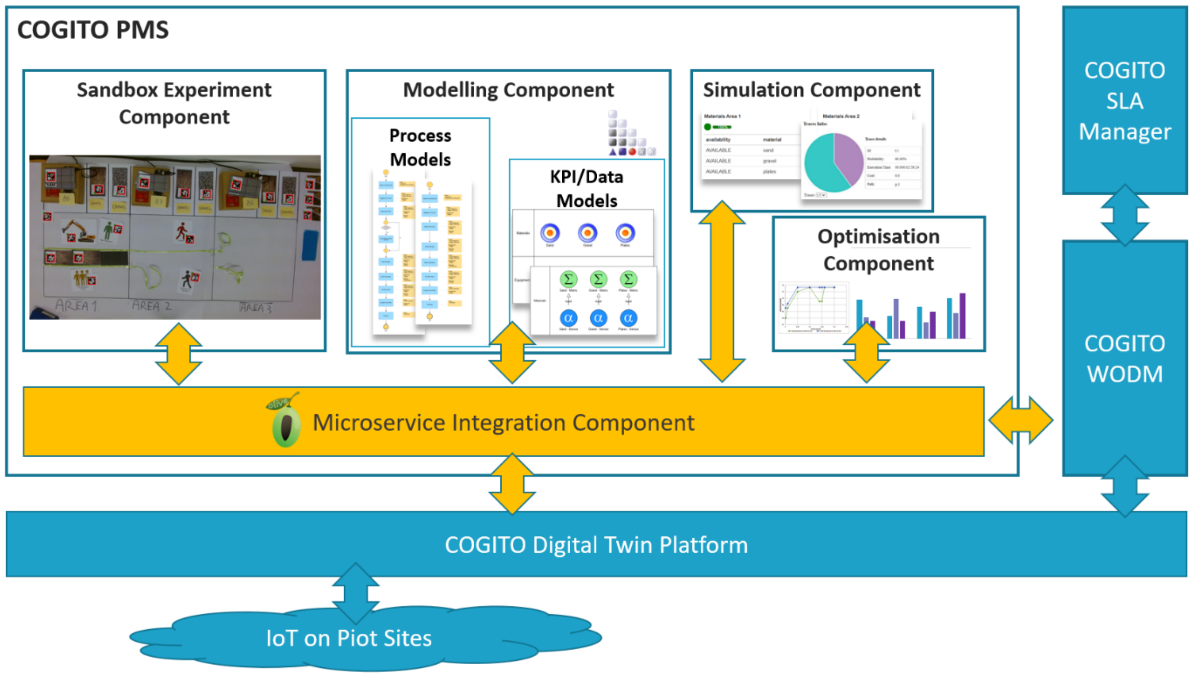


Figure 1 - PMS Experiment Architecture

The Modelling component of the PMS introduces the meta-models and the tools for modelling the construction processes; it is created around the ADOxx, a meta-modelling platform freely available for academic use via the world-wide community ADOxx.org[2]. As such, most of the COGITO prototypes introduced in this deliverable are therefore released as open source, free to 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 the defined meta-models. In our case we adapted the meta-model of the BPMN

[2] https://www.adoxx.org/

(Business Process Modelling Notation) modelling language to also include Key Performance Indicators (KPIs) and entities relevant to the construction domain, i.e., equipment, teams and Building Information Model (BIM) data. Models span across different abstraction layers – ranging from template processes over 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 the 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 approach[3]. 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 OLIVE[4] based on which all the services required by each demonstrated component have been defined. Currently, such services involve the export of process and KPI details from the Modelling component, 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. In the second version of the PMS module, the Microservice Integration component will be responsible for the communication 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 and a common use case, related to the construction of a drainage system for a railway track, is used as demonstration scenario.

1.2 Relation to other Tasks and Deliverables

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 Task 6.3 and the generated log data will be used to evaluate KPIs.

The interactions between the components described in this deliverable depend on the COGITO use cases defined in Task 2.4 and described in the first version of the COGITO architecture in D2.4. Additionally, the

[3] https://www.omilab.org/nodes/design-thinking.html
[4] https://www.adoxx.org/live/olive

stakeholder requirements described in Task 2.1 are considered for the creation of each component besides the functional and non-functional requirements described in D2.4.

Finally, the exchanged data between the PMS and other COGITO modules is aligned with the COGITO ontology as defined in 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.

2 PMS – Modelling Component

This chapter demonstrates the first version of the Modelling component part of the PMS module, used to support the creation of process models specific to the construction domain. In particular, a process model allows the definition of 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 data retrieved from the COGITO DTP. 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; at that point, real-time data and additional feedback from workers are collected and used to continuously improve the construction process model.

2.1 Prototype Overview

BOC's ADOxx meta-modelling platform[5] 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 modelling environment is based on the ADONIS Community Edition and follows the OMG[6] standard BPMN. 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.

- **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.

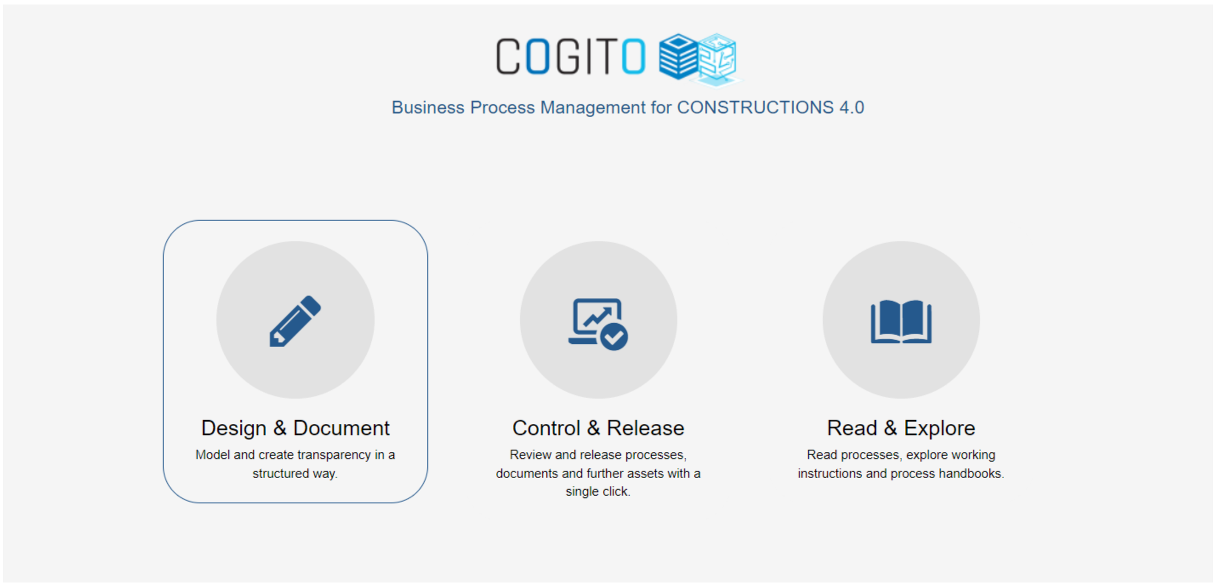
In the following, more details for each modelling environment are provided.

2.1.1 Process Modelling Environment

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[7] 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 in every automatic task and the KPIs associated with each activity.

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 renovation process templates, instances and workflow models in the BPMN2.0 standard model-type (Design & Document" section in Figure 2).

[5] https://www.adoxx.org/live/home
[6] https://www.omg.org/bpmn/
[7] https://www.bpmn.org/


COGITO

Business Process Management for CONSTRUCTIONS 4.0

Design & Document

Model and create transparency in a structured way.

Control & Release

Review and release processes, documents and further assets with a single click.

Read & Explore

Read processes, explore working instructions and process handbooks.

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 objects supported by the BPMN2.0 standard, 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.

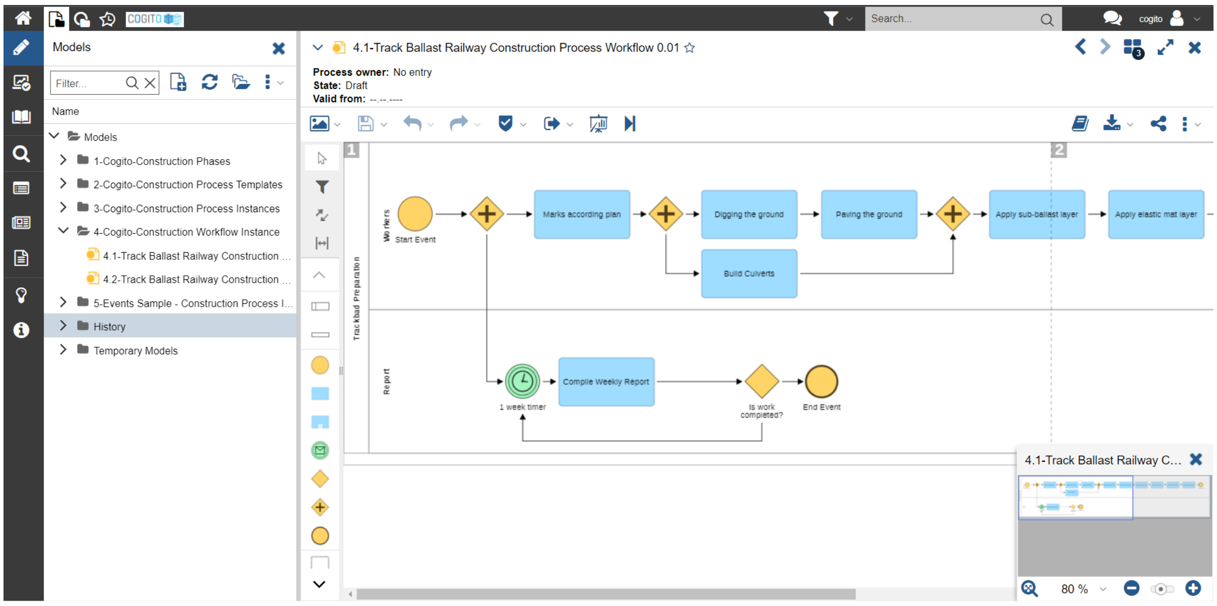

COGITO
Search...
cogito
Models
Filter...
Name
Models
1-Cogito-Construction Phases
2-Cogito-Construction Process Templates
3-Cogito-Construction Process Instances
4-Cogito-Construction Workflow Instance
4.1-Track Ballast Railway Construction ...
4.2-Track Ballast Railway Construction ...
5-Events Sample - Construction Process I...
History
Temporary Models
4.1-Track Ballast Railway Construction Process Workflow 0.01
Process owner: No entry
State: Draft
Valid from: --.--.----
Trackbed Preparation
Workers
Report
Start Event
Marks according plan
Digging the ground
Paving the ground
Build Culverts
Apply sub-ballast layer
Apply elastic mat layer
1 week timer
Compile Weekly Report
Is work completed?
End Event
4.1-Track Ballast Railway C...
80 %

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.

2.1.2 KPIs Modelling Environment

The KPI Modelling environment is a design tool created with the ADOxx meta-modelling platform 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 4):

- 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.

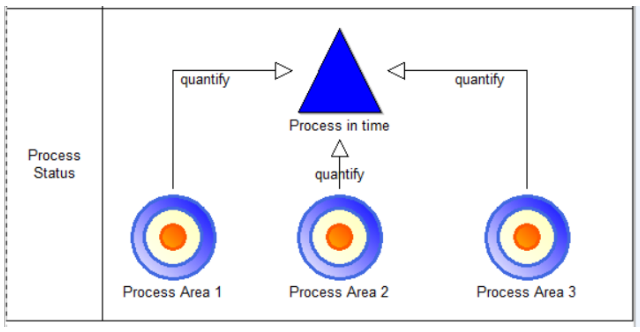
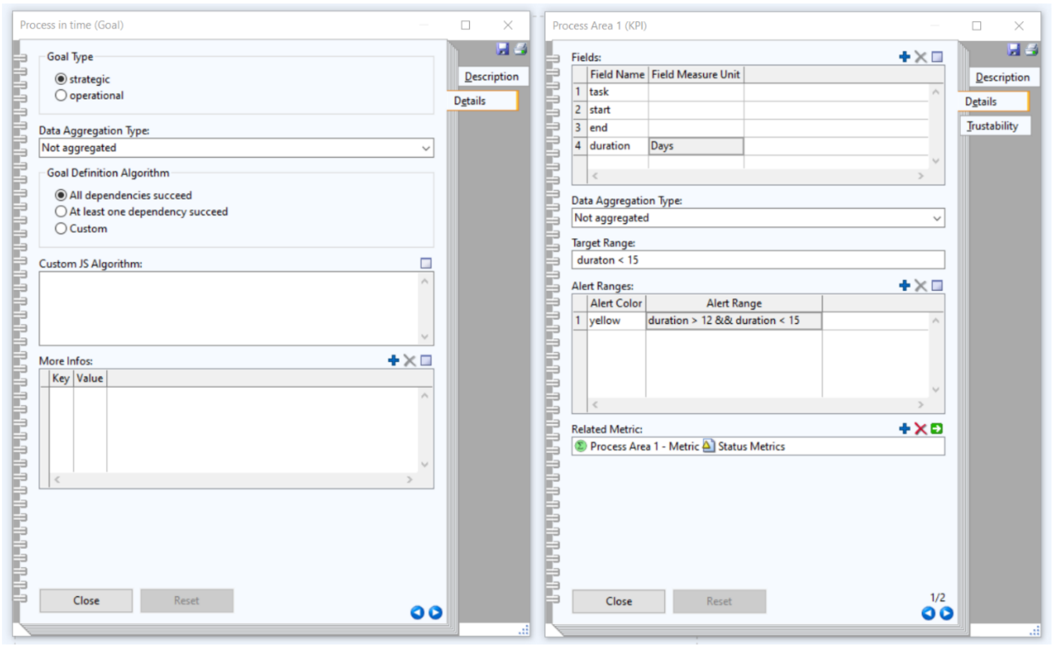


Figure 4 - KPI Model Sample

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 5 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 5 Right). The fields represent what kind of metrics are available in these KPIs. Usually there is a field (duration in the case of Figure 5 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).



Process in time (Goal)

Goal Type

[✓ strategic] operational

Data Aggregation Type: Not aggregated

Goal Definition Algorithm

[✓ All dependencies succeed] At least one dependency succeed Custom

Custom JS Algorithm:

More Infos:

	Key	Value

Process Area 1 (KPI)

Fields:

	Field Name	Field Measure Unit
1	task	
2	start	
3	end	
4	duration	Days

Data Aggregation Type: Not aggregated

Target Range: duraton < 15

Alert Ranges:

	Alert Color	Alert Range
1	yellow	duration > 12 && duration < 15

Related Metric: Process Area 1 - Metric Status Metrics

Figure 5 - 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 6) 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 6), 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).

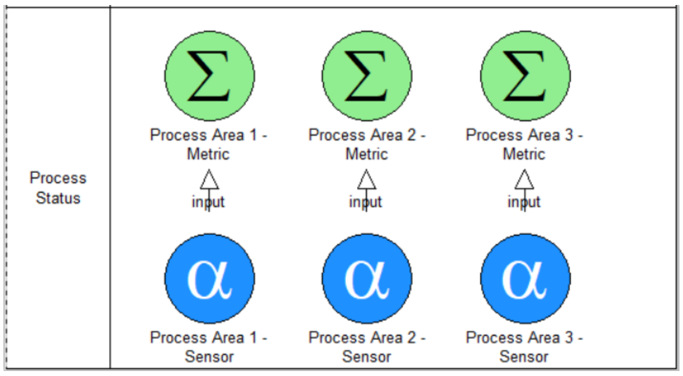
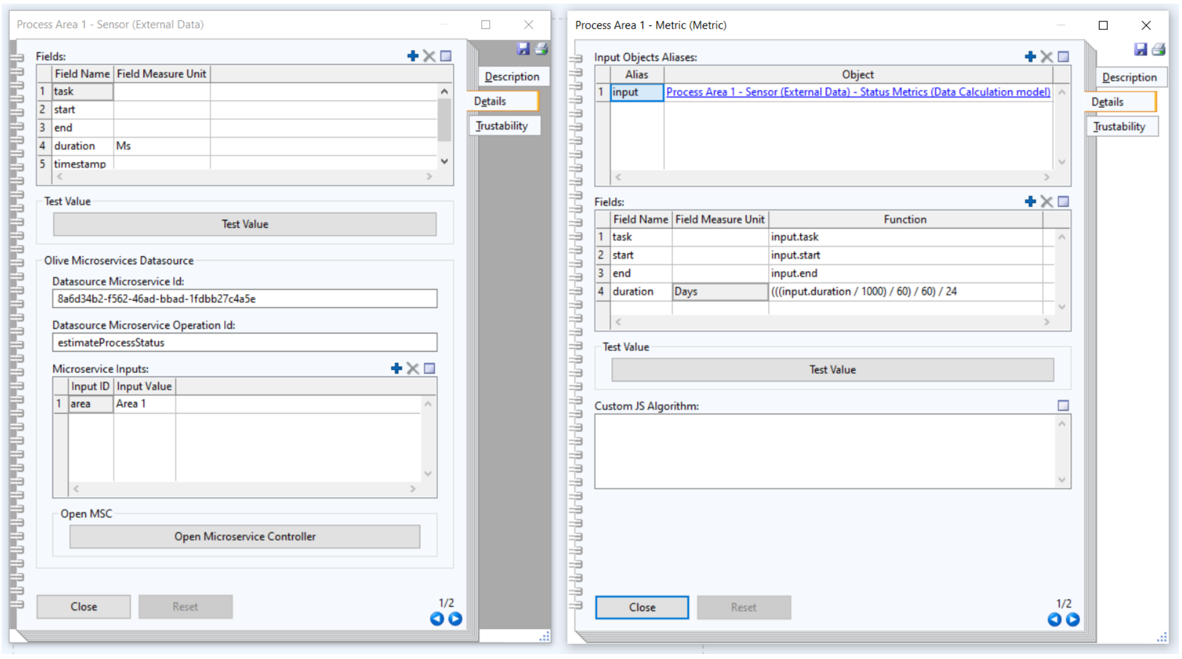


Figure 6 - KPI Data Calculation Model Sample

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 7 (left) the Data Item object representing the process duration in Area 1 uses an existing OLIVE microservice named "estimateProcessStatus" that returns the status of the process in the specific area.


Process Area 1 - Sensor (External Data)

Fields:

	Field Name	Field Measure Unit
1	task	
2	start	
3	end	
4	duration	Ms
5	timestamp	

Test Value

Olive Microservices Datasource

Datasource Microservice Id: 8a6d34b2-f562-46ad-bbad-1fdbb27c4a5e

Datasource Microservice Operation Id: estimateProcessStatus

Microservice Inputs:

	Input ID	Input Value
1	area	Area 1

Open MSC

Process Area 1 - Metric (Metric)

Input Objects Aliases:

	Alias	Object
1	input	Process Area 1 - Sensor (External Data) - Status Metrics (Data Calculation model)

Fields:

	Field Name	Field Measure Unit	Function
1	task		input.task
2	start		input.start
3	end		input.end
4	duration	Days	(((input.duration / 1000) / 60) / 60) / 24

Custom JS Algorithm:

Figure 7 - 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 7 sample it is “duration” for the “Process Area 1” Metric) and optionally additional meta-information fields (like the “start” and “end” times in the Figure 7 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 7 (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

2.4 Application Example

The 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 pilot 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 areas; the construction process in every area is almost the same but may differ regarding some specific requirements. An example is provided in the process template in Figure 8. The process starts by marking the construction area according to a plan, which is done by the foreman and a specialized worker for measurement operations. Then, a digger is used to dig the ground accordingly; this involves a digger operator and a team of civil engineers. Afterwards, a layer of sand is introduced. In case special impermeabilisation measures are required in the area, waterproofing sheaths are introduced, otherwise this task is omitted. Plates for building the drainage layer are implemented; a soakaway is installed and then the surrounding area is filled with gravel. After this step, a roller – operated by a roller operator – is used to compact the gravel layer. Finally, there are some finishing tasks conducted by the foreman controlling the work.

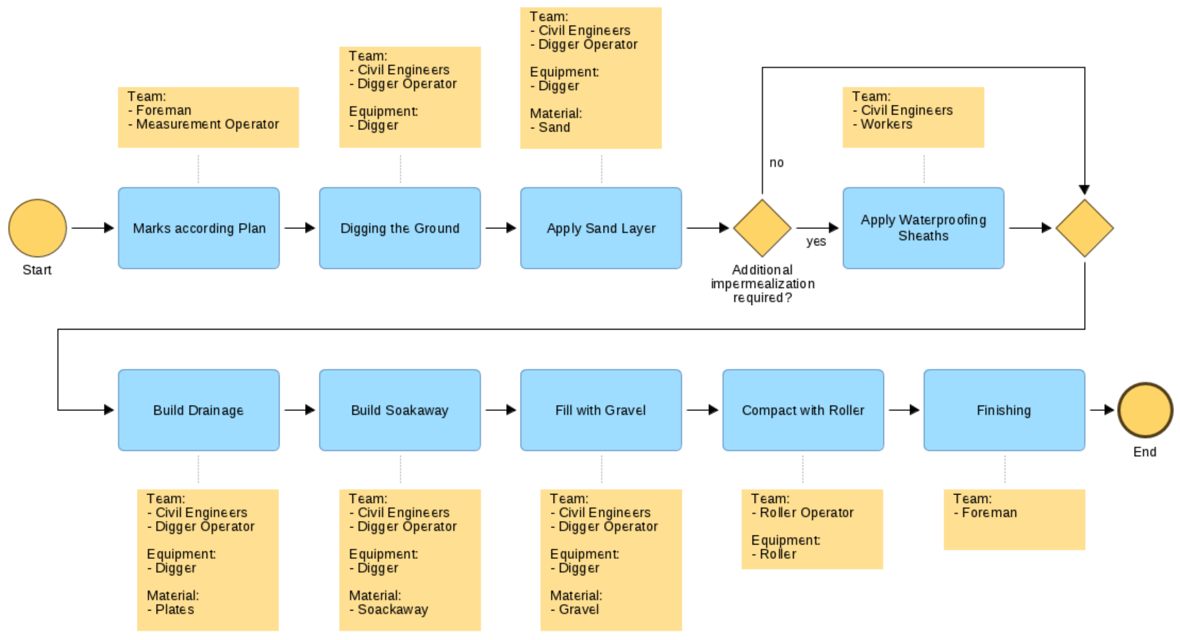


Figure 8 - Railway Track Drainage Construction Template Process

This process template is then instantiated in each construction area. At this stage each decision that can be evaluated at design time is applied. In our case the decision to apply a special waterproofing layer is evaluated assuming three working areas. We can see the instantiated process for each area in Figure 9 where, in particular, the additional activity of the waterproofing layer application is present in Area 2 (hypothetically required due to a specific geological constrain in this area).

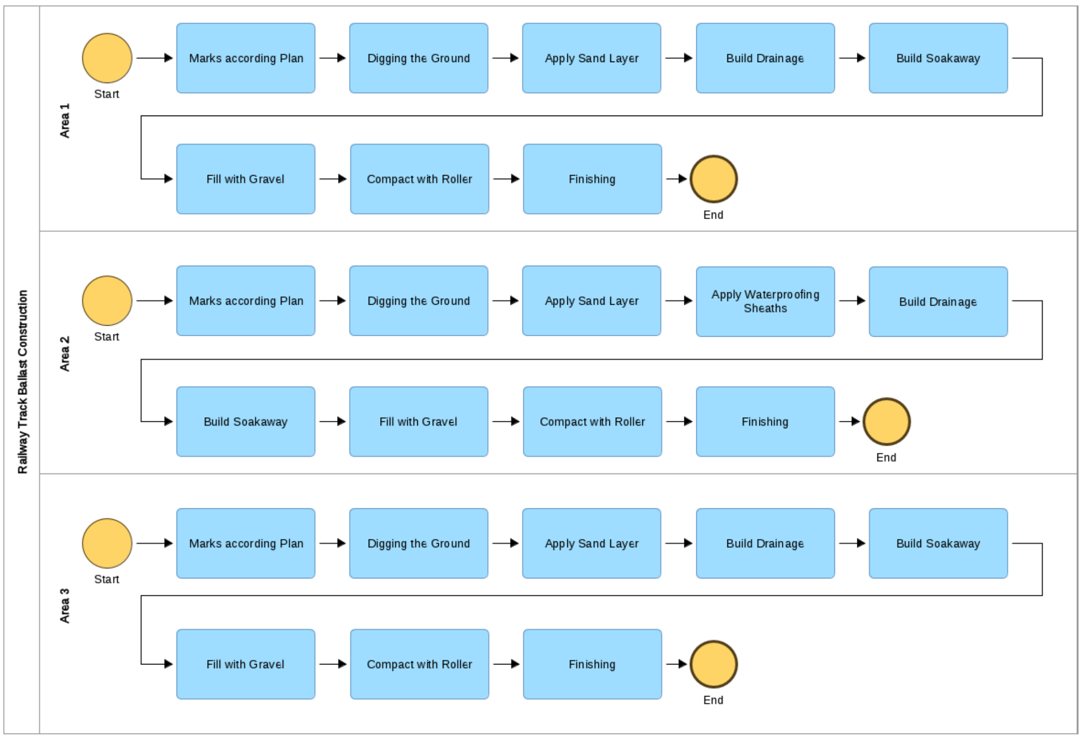


Figure 9 - Railway Track Drainage Construction Process Instances

Finally, for each process instance, workflows are defined (Figure 10) in order to provide technical details and interfaces for the actual execution of the construction process in the COGITO workflow engine provided by the WODM component.

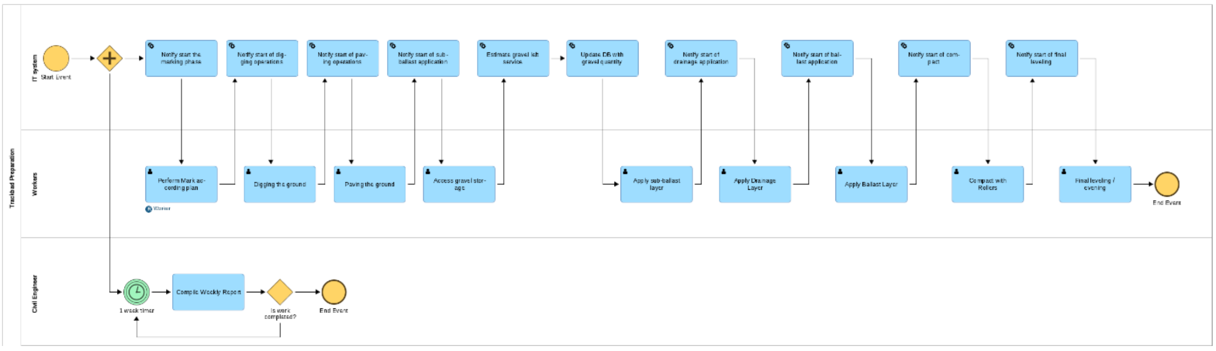
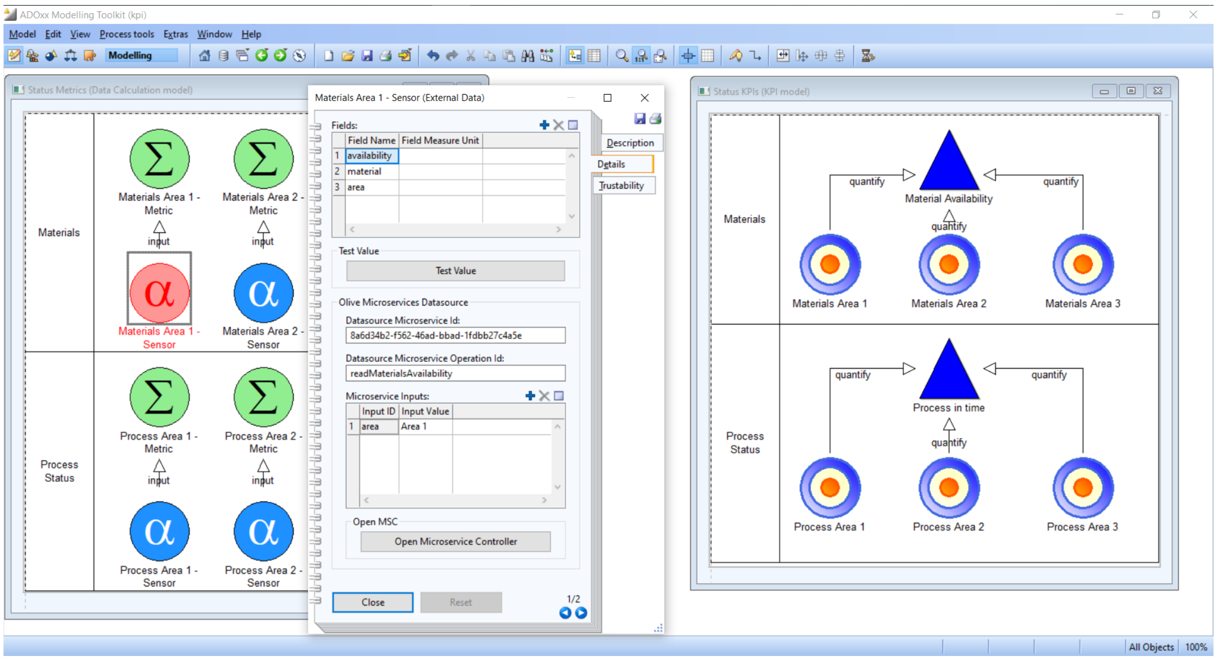

Trackbed Preparation
IT system
Start Event
Notify start the marking phase
Notify start of digging operations
Notify start of paving operations
Notify start of sub-ballast application
Estimate gravel left service
Update DB with gravel quantity
Notify start of drainage application
Notify start of ballast application
Notify start of compact
Notify start of final leveling
Workers
Perform Mark according plan
Worker
Digging the ground
Paving the ground
Access gravel storage
Apply sub-ballast layer
Apply Drainage Layer
Apply Ballast Layer
Compact with Rollers
Final leveling / evening
End Event
Civil Engineer
1 week timer
Compile Weekly Report
Is work completed?
End Event

Figure 10 - Railway Track Drainage 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 sand in a specific area or in what area the digger is located at the moment. In the track dewatering construction process described above, the status of three material types is captured: sand, plates and gravel. Moreover, the equipment (digger and roller) as well as the manpower in the construction site areas are tracked in order to estimate the process status. This is done checking the required material, 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 material in each construction area and the status of the process (Figure 11). Specific microservices have been created in the microservice framework (Chapter 6) for estimating the materials status in the Sandbox Experiment (Chapter 3) and have been assigned to the right KPI data model objects (Section 2.1.2).



Materials Area 1 - Sensor (External Data)

Fields:

	Field Name	Field Measure Unit
1	availability	
2	material	
3	area	

Description
Details
Trustability

Test Value
Test Value

Olive Microservices Datasource

Datasource Microservice Id:
8a6d34b2-f562-46ad-bbad-1fdbb27c4a5e

Datasource Microservice Operation Id:
readMaterialsAvailability

Microservice Inputs:

	Input ID	Input Value
1	area	Area 1

Open MSC
Open Microservice Controller

Close
Reset
1/2

Figure 11 - Railway Track Drainage Construction KPI models

2.5 Licensing

The Modelling component is released in two versions: a community edition for the 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 credentials can be freely requested on faq@adoxx.org.

The 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 KPI library from here: https://git.boc-group.eu/cogito/building-construction-experiment-online/-/blob/master/DASHBOARDS/KPI%20v1.0.0.abl
4. Install the KPI library in the ADOxx platform following the instruction provide in this video: https://www.adoxx.org/live/import_new_application_library
5. Download the sample KPI model from here: https://git.boc-group.eu/cogito/building-construction-experiment-online/-/blob/master/DASHBOARDS/Building%20Construction%20Experiment_v2%20-%20KPI%20Models.adl
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 are still in progress and are planned to be released in the second version of the Modelling component:

- 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 Digital Twin platform.
- Integration of services (described in section 6.7) for exporting the equipment and resources information in JSON format for the PMS Optimisation and the COGITO WODM components.
- Integration with the PMS Optimisation module for visualizing the optimised schedule and updating the models.
- Release of process models templates, KPIs, and workflows related to the specific COGITO pilot use case scenarios.

In addition, the following modelling environments could be considered useful for the creation of the construction experiment described in Chapter 3 and may be used for further advancing the construction experiments towards a construction digital twin:

- **CoChaCo** Meta-Modelling Environment: This modelling environment focuses on the modelling of meta-models. The short form CoChaCo stands for Concepts, Characteristics and Connectors. Those elements are used for describing meta-models. Basically, the modelling environment can be used for (a) meta-model improvement for existing modelling environments such as the KPI modelling environment, and (b) meta-model development in terms of designing new modelling environments supporting domain-specific as well as overlapping challenges such as a potential digitisation modelling environment.

- **Digitization** Modelling Environment: Currently, research in the direction of a targeted digitisation modelling environment is conducted to ease the process of digitization in different industry domains by using physical experiments. It is expected to (a) be an enabler for the establishment of a common understanding among diverse stakeholders, (b) facilitate the creation of physical models and related challenges, (c) serve as a foundation for applying advanced mechanisms supported by AI and (d) ease the management of integrating physical and virtual aspects.

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: *The Computing solution...*	Type	Priority	Status
COGI-CS-1	**[DCC, WODM, PMS, BC, SafetyConAI, VirtualSafety, gQC] runs on desktop or laptop PC**	• **Operational**	**Must**	Supported for both process and KPI modelling
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: *The Workflow Planning,*	Type	Priority	Status

	Execution and Monitoring solution …			
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**	In progress: The integration with Digital Twin platform is in progress
COGI-WF-16	**[PMS] updates the project schedule at least monthly**	**•Operational •Functional •Process**	**Must**	Supported: The versioning support allows to keep multiple 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 allows to associate resources to specific tasks
COGI-WF-23	**[PMS, WODM] incorporates health and safety planning**	**•Functional •Design constraint**	**Should**	In progress: The process modelling component will allows to associate H&S issues to specific tasks
COGI-WF-24	**[PMS, WODM] incorporates quality control planning**	**•Functional •Design constraint**	**Should**	In progress: The process modelling component will allows to associate quality status 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 provide 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

In the first release of the Modelling component the following restriction applies and will be covered in the second release:

- Schedule information for the process is maintained separately from the model in an Excel file that is distributed between the different PMS components.
- Location information, equipment and roles for the process tasks are exported manually in an Excel file that is distributed between the different PMS components.

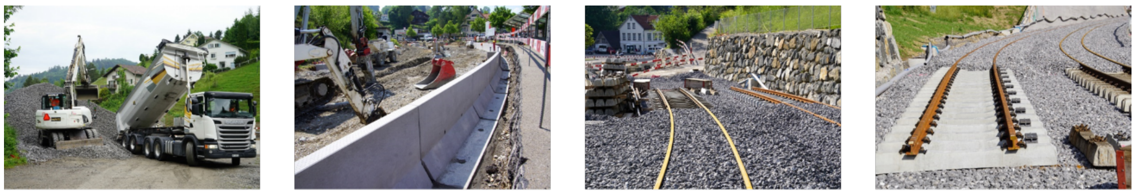
3 PMS – Sandbox Experiment Component

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 can firstly brainstorm about the process using a design thinking approach in order to refine it, then have a safe environment to test the applicable digitisation approaches and finally be able to generate sample log data emulating the process execution; the latter is useful to test dependent 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 the 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 take 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 12 shows some real-world impressions from a track construction in Teufen, where the rail track ballast is placed onto the track area before laying the track elements and inserting the rails.



Source: © **Hans Zürcher**, Teufen https://zukunft-teufen.ch/bauprojekte/gleisverlegung-wie-funktioniert-das

Figure 12 - Impressions from a Real-World Track Construction Scenario[8]

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, materials, 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 material status or 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 location) that can be applied to the digital twin.

[8] https://zukunft-teufen.ch/bauprojekte/gleisverlegung-wie-funktioniert-das

- 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 13 (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.

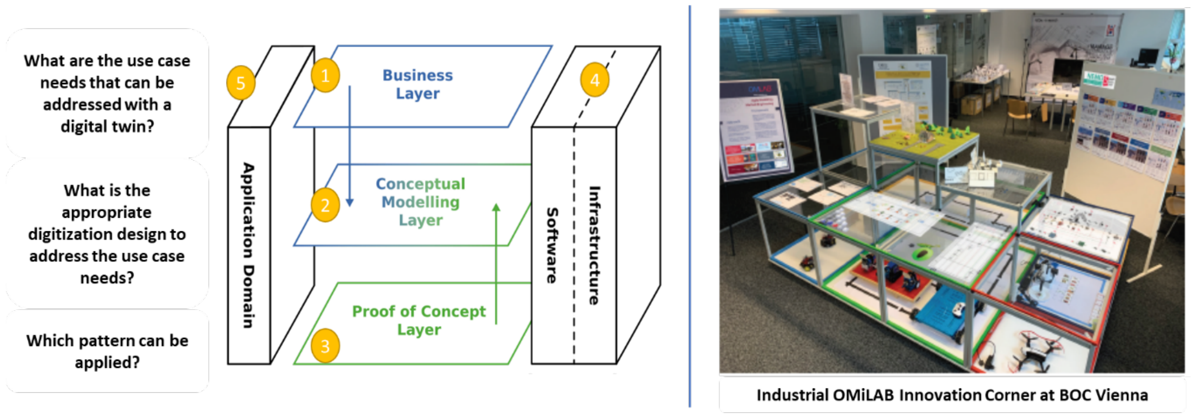

What are the use case needs that can be addressed with a digital twin?
What is the appropriate digitization design to address the use case needs?
Which pattern can be applied?
5
1
Business Layer
4
Application Domain
2
Conceptual Modelling Layer
Software
Infrastructure
Proof of Concept Layer
3
Industrial OMiLAB Innovation Corner at BOC Vienna

Figure 13 - 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 NPO[9] 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 Scene2Model[10]. Configuration towards the construction domain 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.

[9] OMiLAB NPO. OMiLAB Digital Innovation Environment. Retrieved from https://www.omilab.org/.
[10] https://austria.omilab.org/psm/content/scene2model/info

- **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 14. 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 raw materials as well as equipment, this basic process might be too generic. Therefore, in a next iteration round, additional tasks and also related KPIs targeting the different material layers on the construction site 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, two equipment entities (digger and roller) and five human entities (foreman, measurement operator, roller operator, digger operator and civil engineering 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. 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.

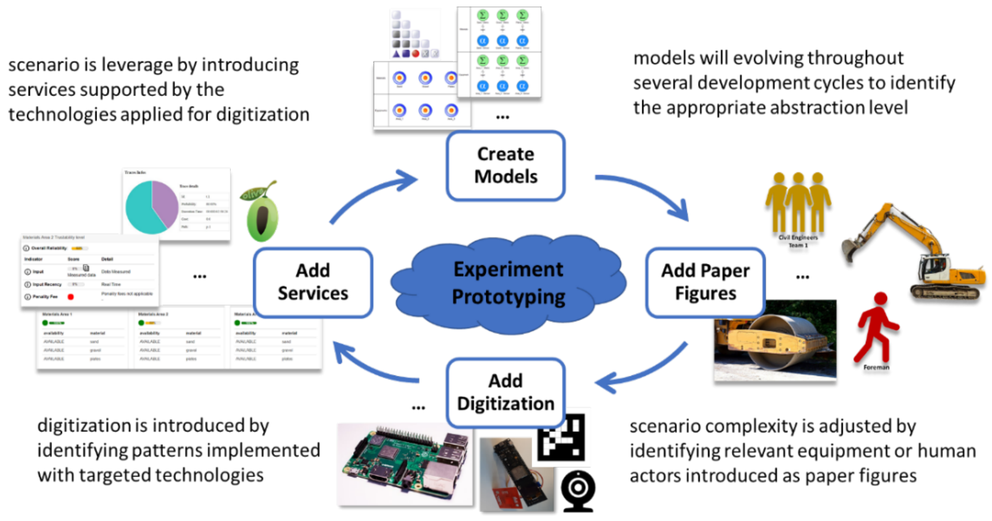

scenario is leverage by introducing services supported by the technologies applied for digitization
models will evolving throughout several development cycles to identify the appropriate abstraction level
Create Models
Add Services
Experiment Prototyping
Add Paper Figures
Add Digitization
digitization is introduced by identifying patterns implemented with targeted technologies
scenario complexity is adjusted by identifying relevant equipment or human actors introduced as paper figures
...
Civil Engineers Team 1
Foreman

Figure 14 - Steps towards Designing the Physical Experiment

3.1.1 Architecture

Figure 15 provides further insights on the Sandbox Experiment architecture. The process models and the related KPI models build upon the ADOxx meta-modelling platform. ADOxx serves as a model repository for the OLIVE framework that provides 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.

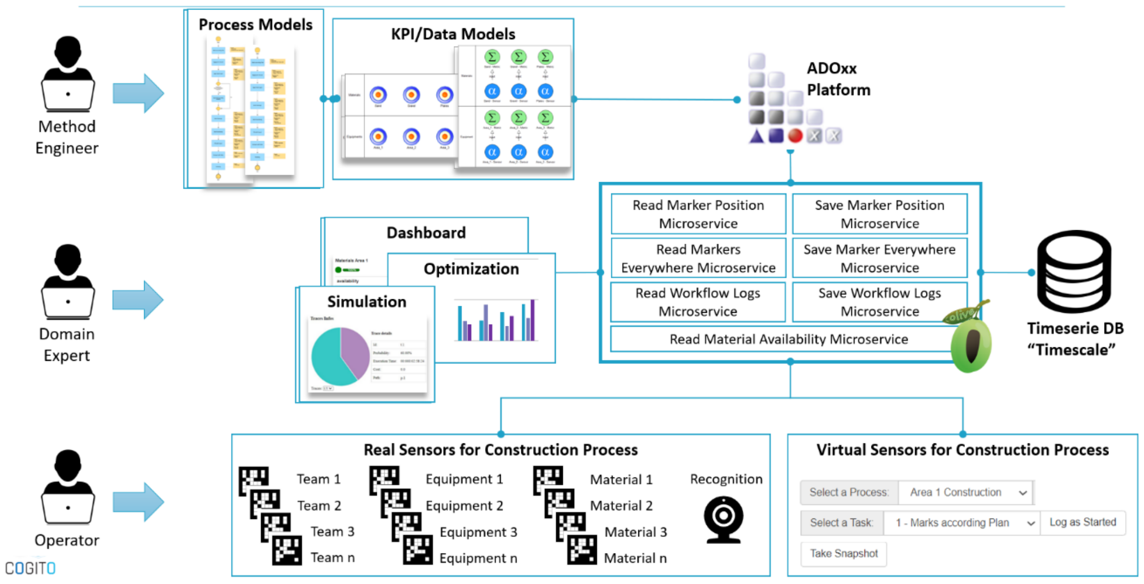


Figure 15 - Experiment Architecture

In the sample experiment, we use code markers to track the materials, equipment and workers. However, the sample can be extended with digital technologies and AI as required. For instance, image recognition may be introduced to capture the height of the sand heap and estimate its quantity.

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
- readMaterialStatus:
  Method: POST
  Inputs: the area where the materials are stored
  Output: a JSON containing the status of availability for each type of material in the sandbox experiment
- 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

Figure 16 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 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 sections are represented by three construction areas 1, 2 and 3 that are marked. The construction process described in Section 2.4 is instantiated for each of those areas. On the top, the raw materials – sand, plates and gravel – for each area are stored.

The modellers can manually emulate the execution of the defined workflow explained in Section 2.4 using this setup, picking the cards related to the equipment, workers, and materials required to each task, placing them over the specific area, and then moving them around as soon as the workflow proceeds. As an example, in the first task of the workflow, related to the marking of the working area, only the cards of a foreman and of a worker with "Measurement Operator" role are picked and placed in the Area 1. When the modeller decides that the task is completed, such cards are moved to Area 2 to let the workflow in this area start. In the meanwhile, in Area 1 the task "Digging the Ground" starts and the cards of the digger and of the digger operator are placed and moved in the working area.

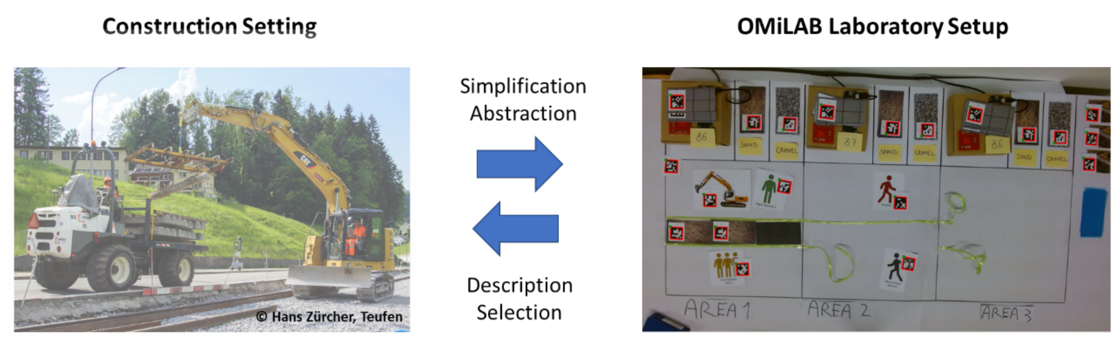


Figure 16 - Real World vs. Experiment Setting

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 by the experiment and access all the features and microservices related to the experiment (Figure 17).

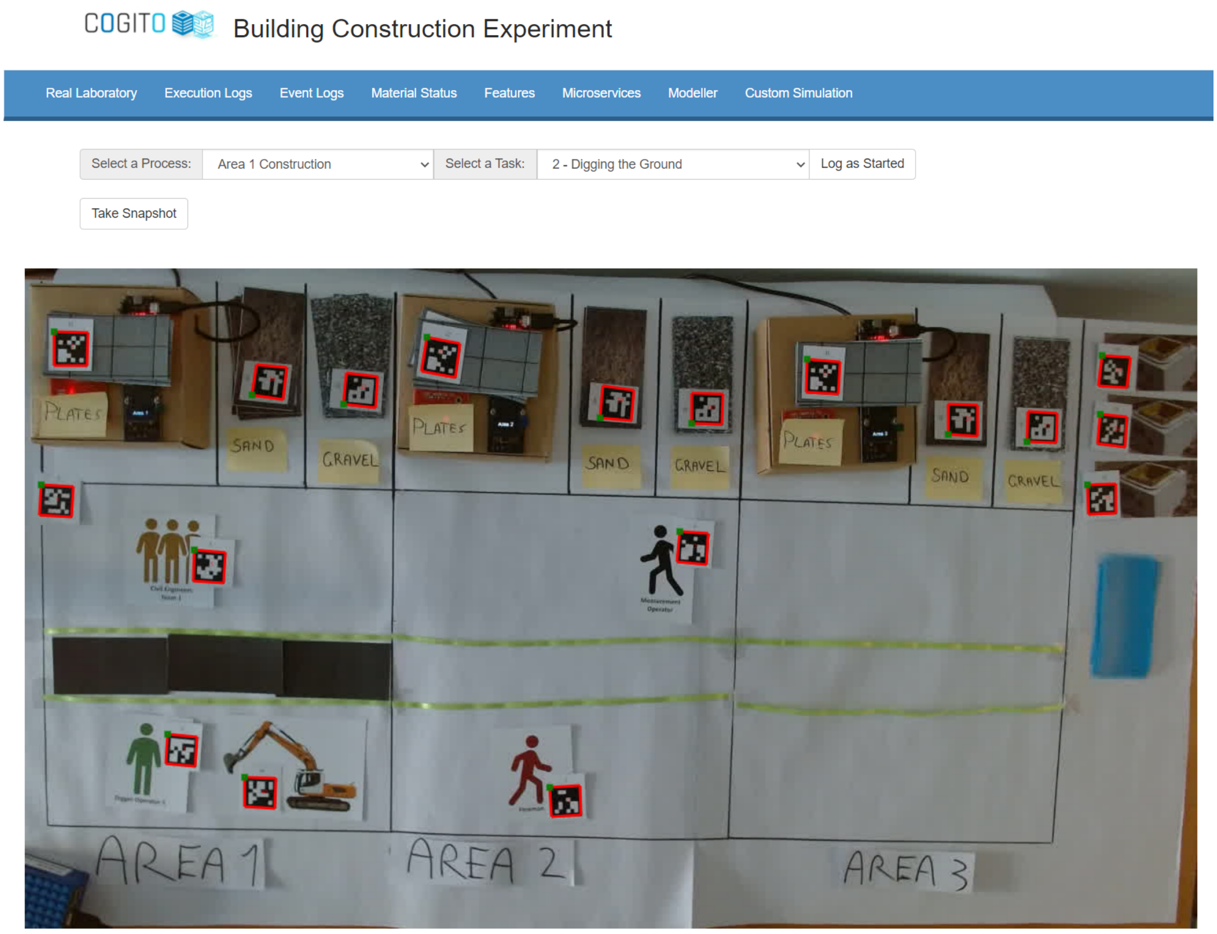
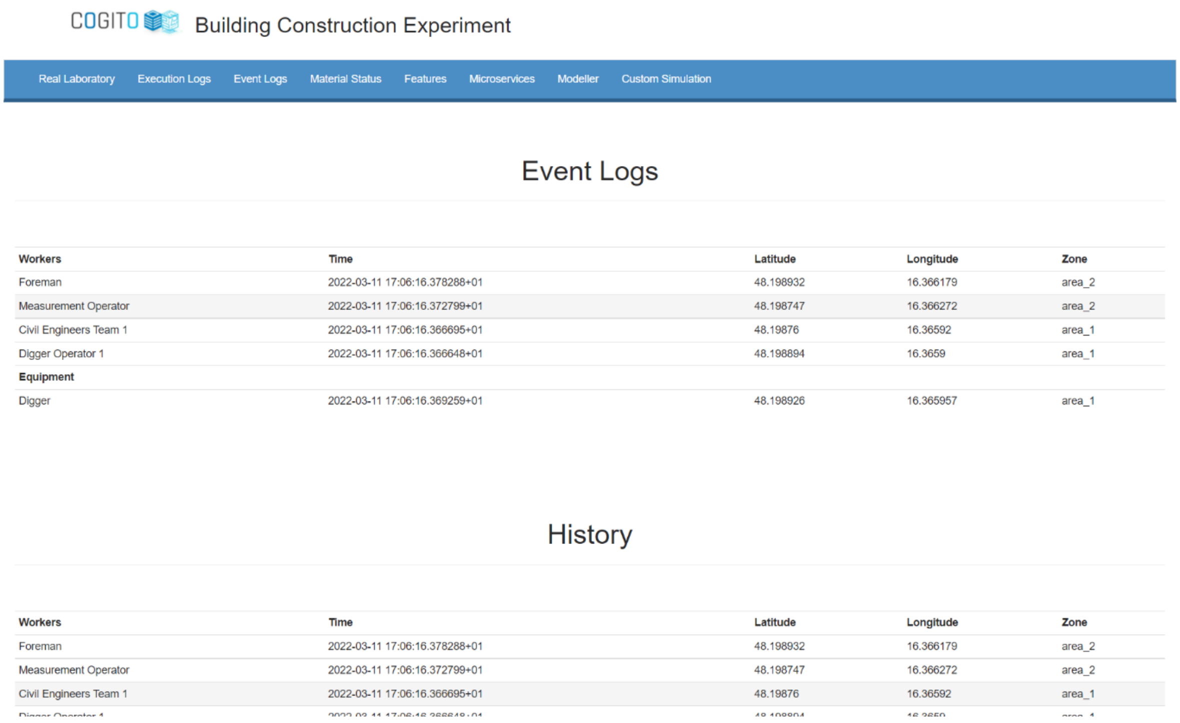


Figure 17 - Construction Experiment entry page

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 18) with their most recent historical positions. At the time when the screenshot was taken, the foreman and the measurement operator were in Area 2, while the digger was in Area 1; that allows for conclusions such as that the measuring operations are already finished in Area 1.



Event Logs

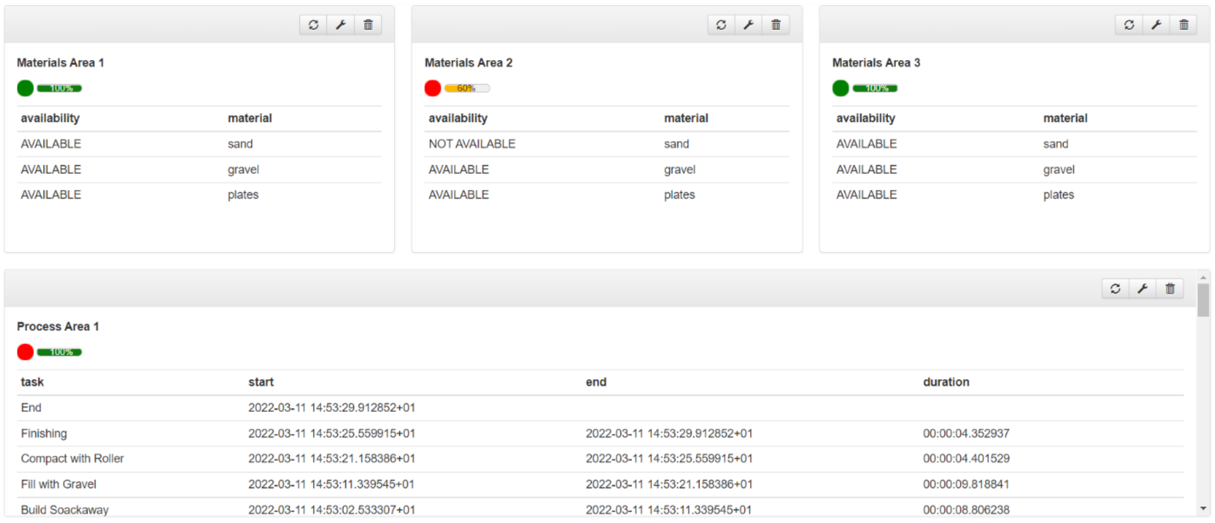
Workers	Time	Latitude	Longitude	Zone
Foreman	2022-03-11 17:06:16.378288+01	48.198932	16.366179	area_2
Measurement Operator	2022-03-11 17:06:16.372799+01	48.198747	16.366272	area_2
Civil Engineers Team 1	2022-03-11 17:06:16.366695+01	48.19876	16.36592	area_1
Digger Operator 1	2022-03-11 17:06:16.366648+01	48.198894	16.3659	area_1
Equipment				
Digger	2022-03-11 17:06:16.369259+01	48.198926	16.365957	area_1

History

Workers	Time	Latitude	Longitude	Zone
Foreman	2022-03-11 17:06:16.378288+01	48.198932	16.366179	area_2
Measurement Operator	2022-03-11 17:06:16.372799+01	48.198747	16.366272	area_2
Civil Engineers Team 1	2022-03-11 17:06:16.366695+01	48.19876	16.36592	area_1

Figure 18 - 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 areas as well as the raw material areas to identify where workers and equipment are at that moment. Microservices are introduced to retrieve the data stored in the timeseries database and present the captured data on dashboards. Figure 19 presents a dashboard showing the availability of the raw materials in the individual construction areas and the estimated status of the process.



Materials Area 1

● 100%

availability	material
AVAILABLE	sand
AVAILABLE	gravel
AVAILABLE	plates

Materials Area 2

● 60%

availability	material
NOT AVAILABLE	sand
AVAILABLE	gravel
AVAILABLE	plates

Materials Area 3

● 100%

availability	material
AVAILABLE	sand
AVAILABLE	gravel
AVAILABLE	plates

Process Area 1

● 100%

task	start	end	duration
End	2022-03-11 14:53:29.912852+01		
Finishing	2022-03-11 14:53:25.559915+01	2022-03-11 14:53:29.912852+01	00:00:04.352937
Compact with Roller	2022-03-11 14:53:21.158386+01	2022-03-11 14:53:25.559915+01	00:00:04.401529
Fill with Gravel	2022-03-11 14:53:11.339545+01	2022-03-11 14:53:21.158386+01	00:00:09.818841
Build Soackaway	2022-03-11 14:53:02.533307+01	2022-03-11 14:53:11.339545+01	00:00:08.806238

Figure 19 - Material Availability Dashboard

The green dots indicated that material is available, while the percentage bar next to the dot displays the reliability of the measurement. In particular, in case different sensors are used, the reliability of each of them may vary. For instance, when using image recognition for measuring the height of the sand heap, the light conditions in the evening or in a tunnel may be a problem with respect to accuracy. Also, estimating the weight of the sand heap or manually entering the amount of sand based on an expert estimation could be possible. In general, finding the appropriate digital technologies is not trivial and therefore may require several refinements throughout the experiment prototyping phases.

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

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

The second version of the Sandbox Experiment component will demonstrate the integration with the WODM module where the construction workflow of the experiment will be executed.

Additionally, in the second version, the construction processes used in the sandbox experiment will be aligned with the ones decided for the pilot sites.

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: *The Computing solution…*	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: *The Workflow Planning, Execution and Monitoring solution …*	Type	Priority	Status
COGI-WF-28	facilitates pre-construction training sessions (e.g., by using BIM models in augmented reality)	•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	•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 (DES) 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 20).

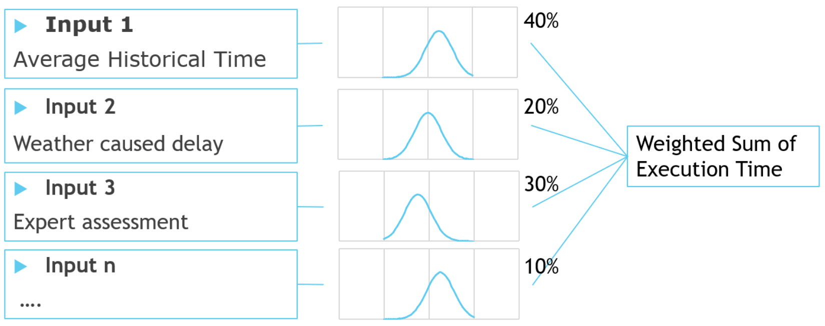
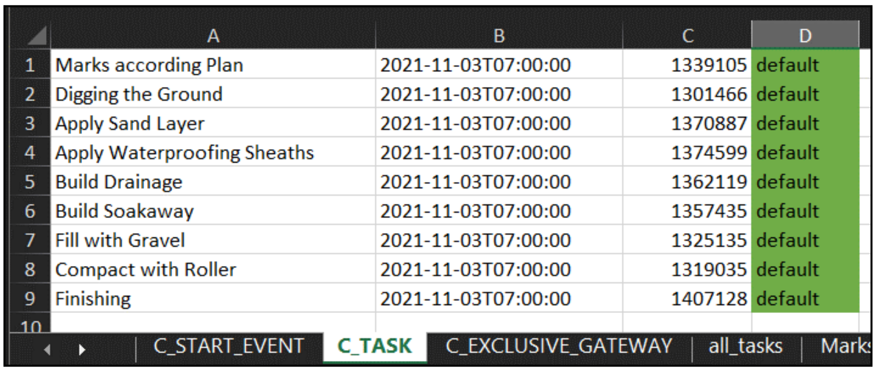


Figure 20 - Simulation mechanisms for multi-distributions

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' 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.

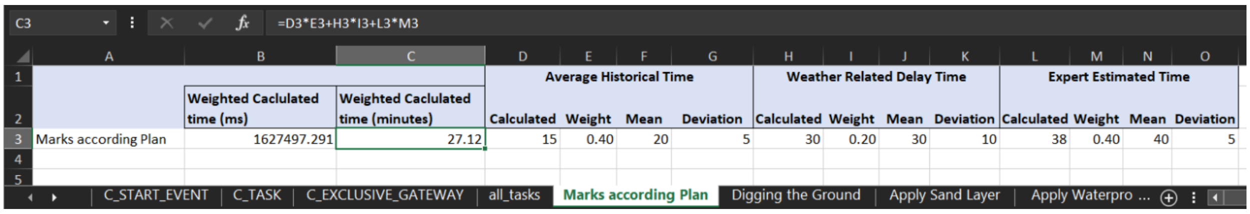
For each simulation run, a unique identifier with a corresponding start time (column B of Figure 21) is provided. Figure 21 introduces a simple simulation where a default deviation (column D) is applied to the provided time (column C), the latter being provided in milliseconds for each activity. A loose 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
1	Marks according Plan	2021-11-03T07:00:00	1339105	default
2	Digging the Ground	2021-11-03T07:00:00	1301466	default
3	Apply Sand Layer	2021-11-03T07:00:00	1370887	default
4	Apply Waterproofing Sheaths	2021-11-03T07:00:00	1374599	default
5	Build Drainage	2021-11-03T07:00:00	1362119	default
6	Build Soakaway	2021-11-03T07:00:00	1357435	default
7	Fill with Gravel	2021-11-03T07:00:00	1325135	default
8	Compact with Roller	2021-11-03T07:00:00	1319035	default
9	Finishing	2021-11-03T07:00:00	1407128	default
10				

Figure 21 - Process Task List for Simulation

The execution time for each activity is the result of the combination of different inputs. Figure 22 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.



C3 =D3*E3+H3*I3+L3*M3

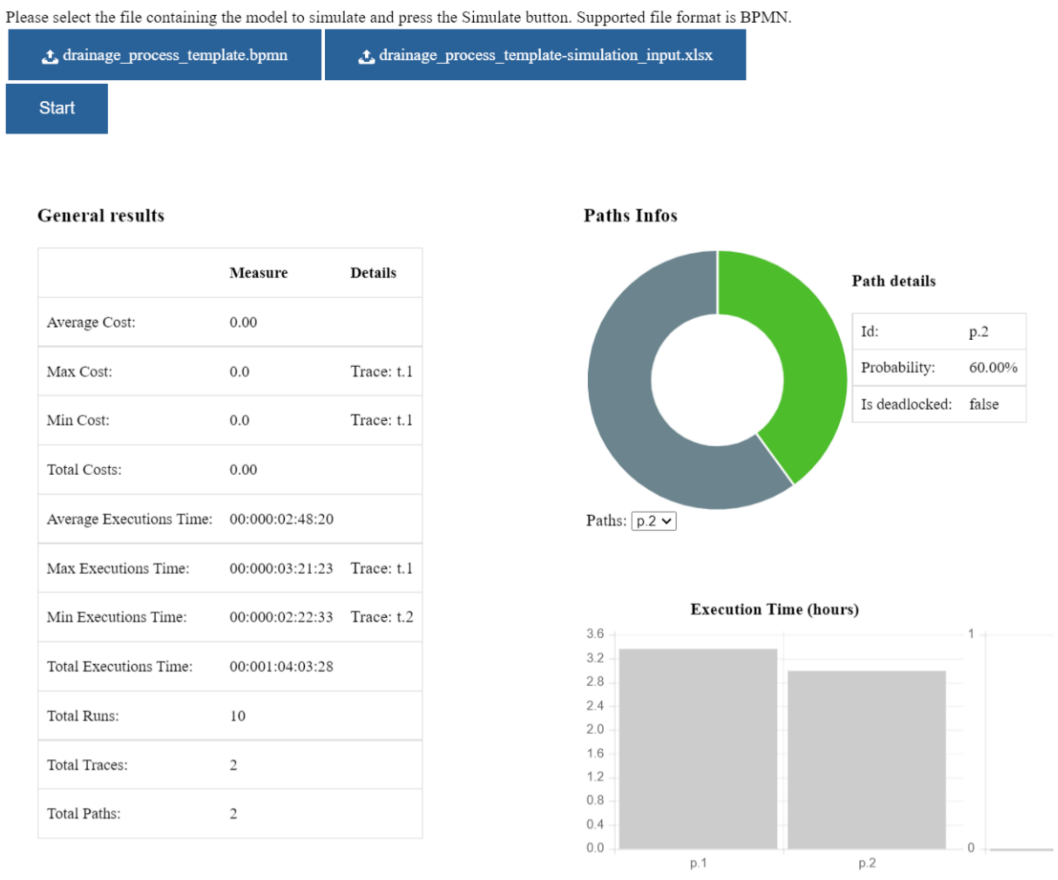
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1				Average Historical Time				Weather Related Delay Time				Expert Estimated Time			
2		Weighted Caclulated time (ms)	Weighted Caclulated time (minutes)	Calculated	Weight	Mean	Deviation	Calculated	Weight	Mean	Deviation	Calculated	Weight	Mean	Deviation
3	Marks according Plan	1627497.291	27.12	15	0.40	20	5	30	0.20	30	10	38	0.40	40	5
4															
5															

C_START_EVENT
C_TASK
C_EXCLUSIVE_GATEWAY
all_tasks
Marks according Plan
Digging the Ground
Apply Sand Layer
Apply Waterpro ...

Figure 22 - 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 23)
- 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 24).



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

drainage_process_template.bpmn
drainage_process_template-simulation_input.xlsx
Start

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

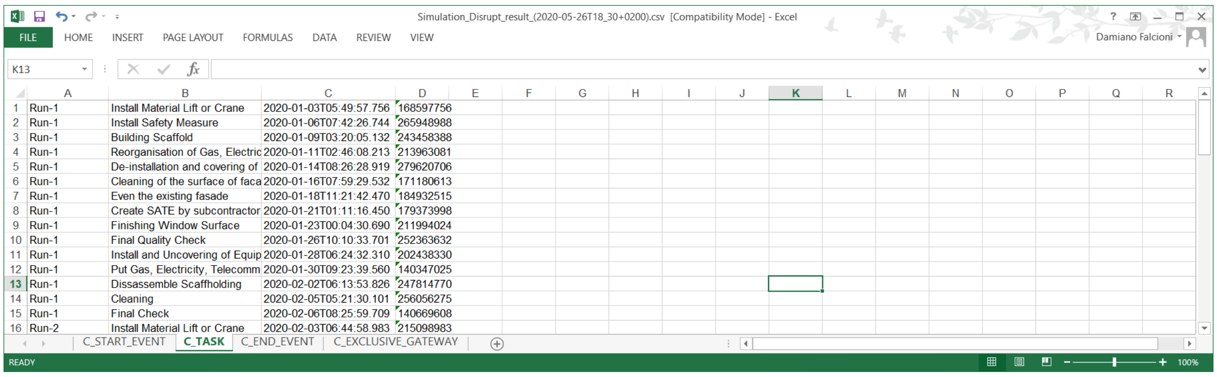
Path details

Id:	p.2
Probability:	60.00%
Is deadlocked:	false

Paths: p.2

Figure 23 - 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
1	Run-1	Install Material Lift or Crane	2020-01-03T05:49:57.756	168597756
2	Run-1	Install Safety Measure	2020-01-06T07:42:26.744	265948988
3	Run-1	Building Scaffold	2020-01-09T03:20:05.132	243458388
4	Run-1	Reorganisation of Gas, Electric	2020-01-11T02:46:08.213	213963081
5	Run-1	De-installation and covering of	2020-01-14T08:26:28.919	279620706
6	Run-1	Cleaning of the surface of faca	2020-01-16T07:59:29.532	171180613
7	Run-1	Even the existing fasade	2020-01-18T11:21:42.470	184932515
8	Run-1	Create SATE by subcontractor	2020-01-21T01:11:16.450	179373998
9	Run-1	Finishing Window Surface	2020-01-23T00:04:30.690	211994024
10	Run-1	Final Quality Check	2020-01-26T10:10:33.701	252363632
11	Run-1	Install and Uncovering of Equip	2020-01-28T06:24:32.310	202438330
12	Run-1	Put Gas, Electricity, Telecomm	2020-01-30T09:23:39.560	140347025
13	Run-1	Dissassemble Scaffholding	2020-02-02T06:13:53.826	247814770
14	Run-1	Cleaning	2020-02-05T05:21:30.101	256056275
15	Run-1	Final Check	2020-02-06T08:25:59.709	140669608
16	Run-2	Install Material Lift or Crane	2020-02-03T06:44:58.983	215098983

Figure 24 - 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.

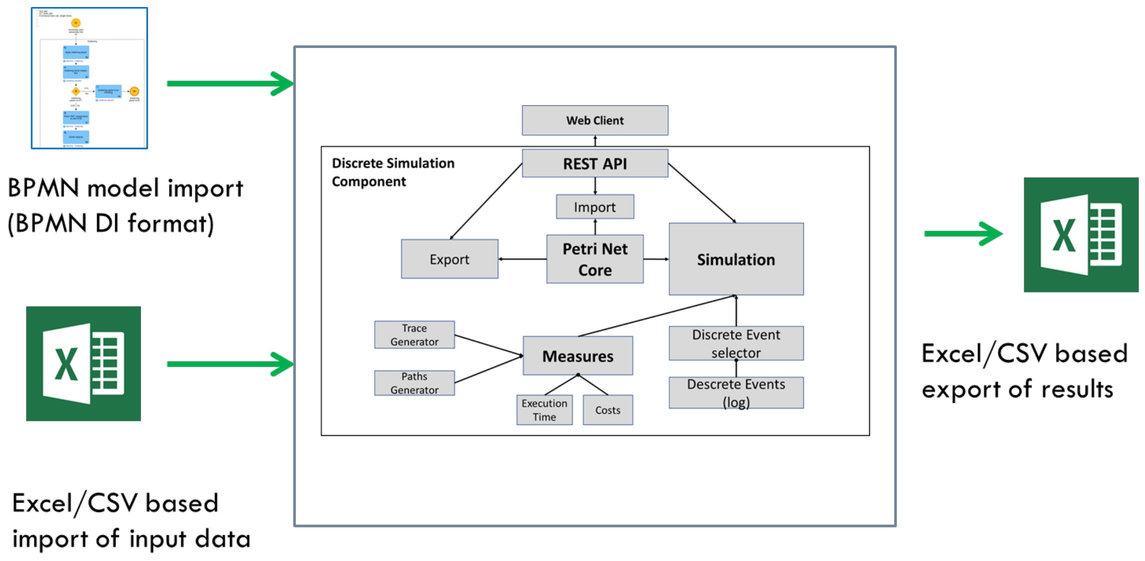


Figure 25 - Simulation Engine Architecture

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.

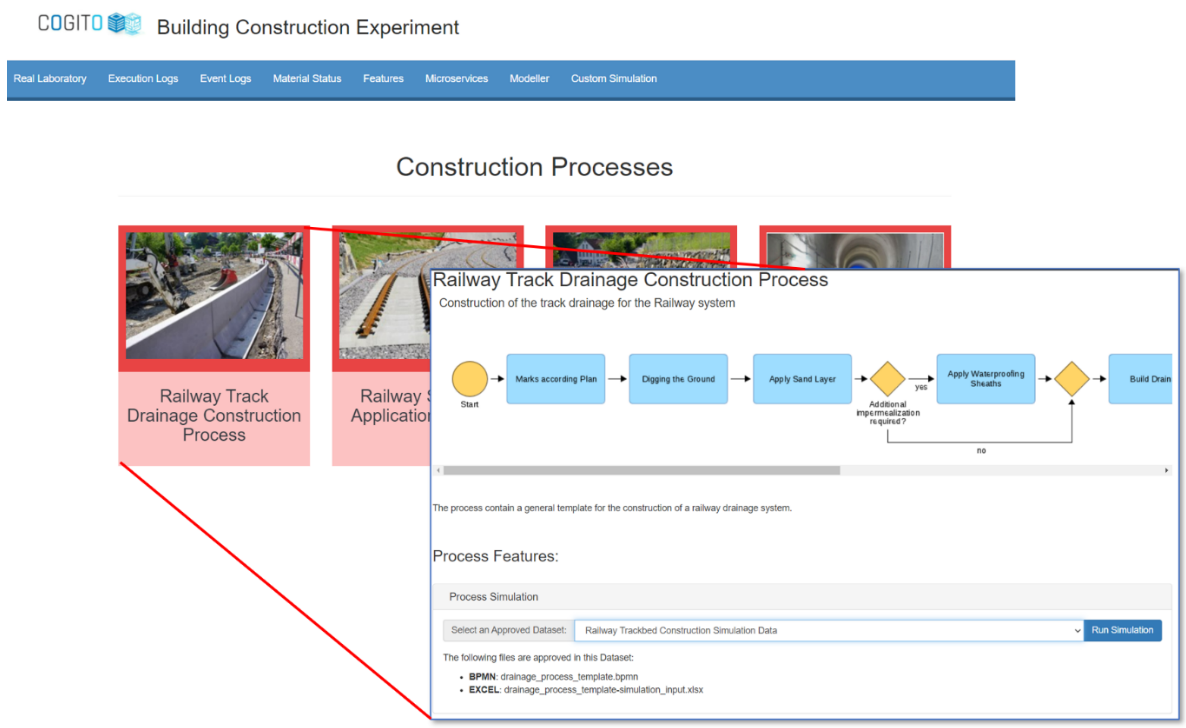
The service is not exposed to other COGITO modules but it is used internally by the PMS Modelling component.

4.4 Application Example

In order to demonstrate the Simulation component, the railway track drainage 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 some 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.

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 experiment portal section related to the features (Features menu in Figure 26) by selecting the Railway track drainage construction process and then executing the Simulation feature.



Railway Track Drainage Construction Process

Construction of the track drainage for the Railway system

The process contain a general template for the construction of a railway drainage system.

Select an Approved Dataset: Railway Trackbed Construction Simulation Data

The following files are approved in this Dataset:

- **BPMN**: drainage_process_template.bpmn
- **EXCEL**: drainage_process_template-simulation_input.xlsx

Figure 26 - Simulation Feature 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/overview/) under the "Custom Simulation" menu item, or alternatively under the Features menu item, preloaded with process and Excel inputs related to the specific experiment process.

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

In the second release the microservices required to retrieve real-time data from the COGITO WODM module will be integrated and used to update the simulation input.

Additionally, the simulation engine will be extended in order to explicitly consider the resources (workers and equipment associated to each task) as well, during the time and costs simulations.

Finally, the input and output interfaces of the service will be improved in order avoid the dependency from the Excel file format.

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: *The Computing solution…*	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: *The Workflow Planning, Execution and Monitoring solution …*	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., networks relocation before excavation work)	•Functional	Could	In progress

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 by the experiment. In progress for the WODM connection
	Req-1.3	Connect and Authenticate to the DT platform	In progress
	Req-1.4	Estimates project progress	In progress
	Req-1.5	Receive task progress from WODM	In progress
	Req-1.6	Output updated estimates of project progress to WODM according to real-time data and performed optimisation	In progress
Non-Functional	Req-1.7	Web-based app	Supported
	Req-2.1	User-friendly	In progress: The simulation inputs are provided in form of Excel file. A specific user interface will be released in the second version
	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 currently a manual process using data extracted from services, i.e., retrieving weather conditions and average execution times from historical execution. In the second release, the Simulation component will extract the needed input data directly from the microservices (the Excel file will not be required).

5 PMS – Optimisation Component

In this chapter the Optimisation component is described using the example described in Figure 9 of Section 2.4 (for one area). This initial 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 to activities has been developed. 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. In any project there is a choice to be made about how many workers of each skill to assign to each activity. In general, the more workers assigned to a given activity the faster that activity can be completed, but there is a corresponding increase in cost. The Optimisation component determines the number of each type (skill) of worker 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 required to complete the activity.
- The cost of workers of each type.
- The maximum available number of workers of each type.
- The relationship between the number of workers 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 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 in future work alternatives in discussion with project partners will be investigated. Options include the use of existing datasets, such as the RSMeans[11] 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.

Before implementing the Optimisation component, input data will be received from the Modelling component (including the productivity for each type of worker and the total work needed to complete each activity – see details in Section 2.4) including a workflow of activities that need to be carried out and the order of these activities. Type and minimum number of workers needed to complete each activity will also be provided. For some activities, some types of workers will not be required and this will also be included in the input data.

In this section, we describe an initial version of the Optimisation component that only considers construction projects within which the activities occur in series and 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. The objective of the mathematical model developed in this module is to optimally allocate the required resources (in the simple case only the workforce is considered) 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, so there are no constraints associated with workers being busy with other tasks.
- For each activity we are given the total work required from each type of worker (e.g., in m^2) and the productivity of each type of worker (e.g., in m^2 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

[11] https://www.rsmeans.com/

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):

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

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). In future work 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

The objectives in the proposed model are as follows:

(1) Minimization of the labour cost:

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

(2) Minimization of the total duration

In this first 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)

$$\text{Total Duration} = \sum_{j=1}^{J} TDr_j = \sum_{j=1}^{J} Max_p\left(Dr_{jp}\right) \tag{5.3}$$

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):

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

w_1 and w_2 are weights allowing the relative importance of time and cost to be varied. 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.

- **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.

$$\begin{cases} Dr_{jp} = \dfrac{TW_{jp}}{NW_{jp} \times \Pr_{jp}} \times \alpha^{NW_{jp}-1} & X_{jp} = 1 \\ Dr_{jp} = 0 & X_{jp} = 0 \end{cases} \tag{5.5}$$

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 that $TDr_j = Max_p\left(Dr_{jp}\right)$. As the objective function minimises a function of time and cost, this is equivalent to the following set of inequalities (Eq. 5.6):

$$TDr_j^D \geq Dr_{jp}^D ; \forall j, p \tag{5.6}$$

- **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, as shown below (Eq. 5.7).

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

This Mixed Integer Nonlinear Programming (MINLP) model is implemented on *GAMS (*General Algebraic Modelling System*)* [5] and solved through a *BARON*[12] 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

[12] Branch-And-Reduce Optimization Navigator (BARON)

model (i.e., considering only the workforce) is checked by the total enumeration of all possible solutions implemented in R [7]).

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 implementation 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 Optimisation component will be exposed in the form of a REST microservice accepting a JSON input file and a BPMN process model in a POST method and producing a JSON file as output. 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 initial version of the Optimisation component are as follows:

Input

- Productivity of workforce per specialty for conducting each activity (e.g., m^2 per h)
- Cost for workforce per specialty (per person per hour)
- Total amount of work per activity (e.g., m^2)
- Minimum required number of workers per specialty for conducting each activity
- Maximum available number of workers per specialty for conducting each activity
- Zero-one matrix indicating if activity j needs the skill p ("1" otherwise "0")
- 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 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 with each skill for each activity

For the inputs, data for the initial model can be provided either from historical data or experts' opinions. For example, it may be possible to obtain the productivity of different types of workers, 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 building of a drainage system for a railway line, for which example data and workflow schedule of activities are available (see Figure 9 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 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 m^2.

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 (m^2 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					
	Foreman	**Measurement Operator**	**Civil Engineers**	**Digger Operator**	**Roller Operator**	**Total Duration [h]**
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					
	Foreman	Measurement Operator	Civil Engineers	Digger Operator	Roller Operator	Total Duration [h]
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 27 and Figure 28. To examine the effect of varying w_1, w_2 and α together, Figure 27 and Figure 28 show the relationship between values of α and total duration and cost for different weights.

We can see from Figure 27 and Figure 28 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.

Our results demonstrate that the proposed optimisation model can be used to delegate workers to tasks to minimise a function of cost and time. We also observed that varying the weights affects the results, highlighting the importance of carefully choosing these weights to represent their relative valuation of time and cost that the decision-makers have to consider.

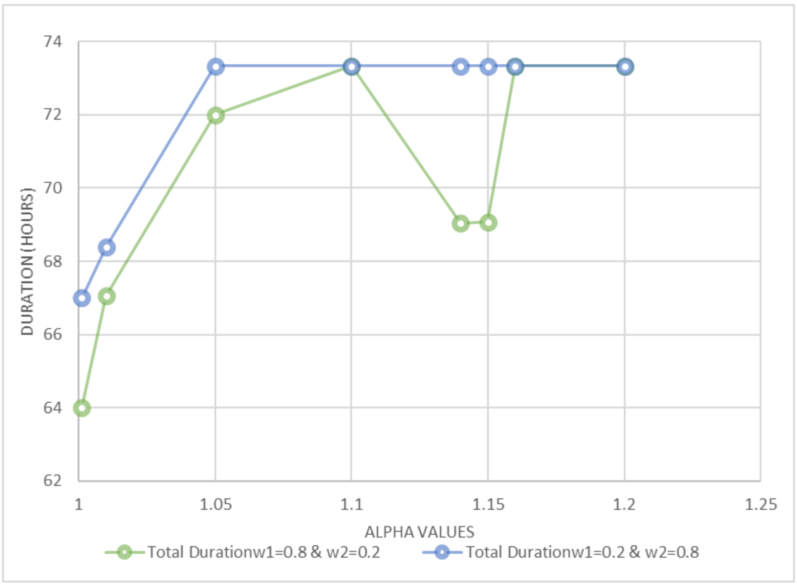


Figure 27 - Total Duration for different values of α (for TW=100 m^2)

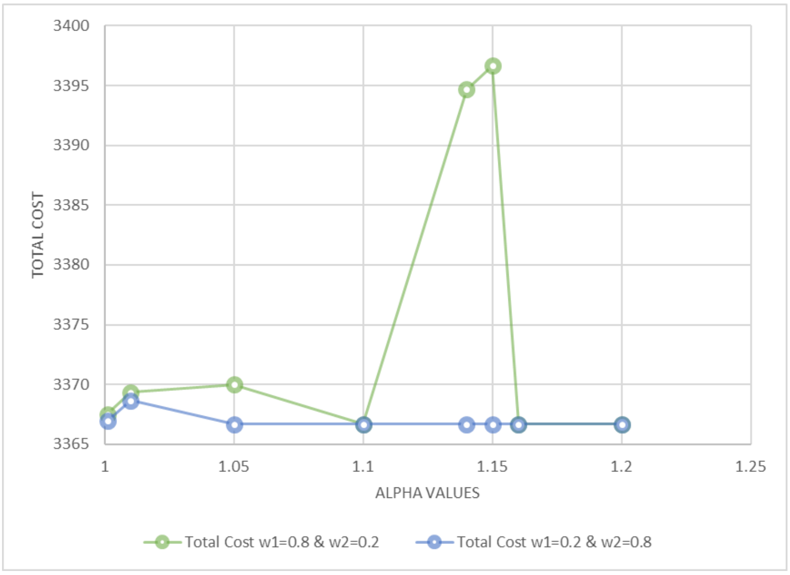


Figure 28 - Total Cost for different values of α (for TW=100 m^2)

5.5 Licensing

The current version of the Optimisation component is released as a closed-source solution. An open-source release will be evaluated for the second version.

5.6 Installation Instructions

In this first version the installation instructions are not available due to the Optimisation component being closed-source. In the second release, the component will be accessed as a REST service and as such no installation instructions will be required.

5.7 Development and integration status

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 proof-of-concept optimisation model for future releases. We aim to identify which of these extensions are most critical in collaboration with the use case partners:

- Overlapping and parallel activities: currently the model only considers activities that occur in series. One possible extension is to add the option for activities that can overlap/occur in parallel, subject to a set of constraints concerning the precedence relations of different activities. This will require changes to the constraints on the maximum available workers to account for the option of them carrying out alternating tasks.

- Including start and finish times as variables: currently do temporal information is not included in the model. In order to include overlapping activities, the start and finish times of the tasks will need to be modelled, which will also require information on how the number of available workers varies with time.

- Refining the form of the task duration function according to available data and expert opinions from the COGITO partners.

- Other resources (e.g., equipment and material): the inclusion of equipment in the model is likely to follow the same formulation as the proof-of-concept model shown for the workforce, except changes needed to accommodate the duration function. However, as material is not a renewable resource, 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 effect 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.

- Solution of the optimisation model: currently the formulation is a mixed integer nonlinear program, which in general is a class of optimisation problems hard to solve. We will explore opportunities to reformulate it as a mixed integer linear program, taking advantage of how a piecewise linear approximation to the nonlinear functions in the constraints, which is exact at integer numbers of workers, gives the same solution as the current nonlinear formulation to the integer optimisation problem. The effect this has on robustness and speed of the solution will be a matter for experimentation on specific examples.

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: *The Computing solution…*	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 will be 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**	In progress: the communication between the Modelling and the Optimisation component will be secured by 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 in the Modelling component

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

ID	Description: *The Workflow Planning, Execution and Monitoring solution ...*	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-	The integration with the modelling environment (responsible to

		time data and performed optimisation	communicate with WODM) is in progress
Non-Functional	Req-1.7	Web-based app	In progress: the component will be integrated as REST service
	Req-2.1	User-friendly	In progress: the UI for optimisation will be defined in the modelling component
	Req-2.2	Scalability	In progress: the REST service will be deployed in a Docker Swarm cluster for automatic scalability on load
	Req-2.3	Security	In progress: the access to the optimisation REST service will be 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 a proof-of-concept optimisation model with a structure based on the discussion 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 OLIVE[13], 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 29).

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 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.

13 https://www.adoxx.org/live/olive

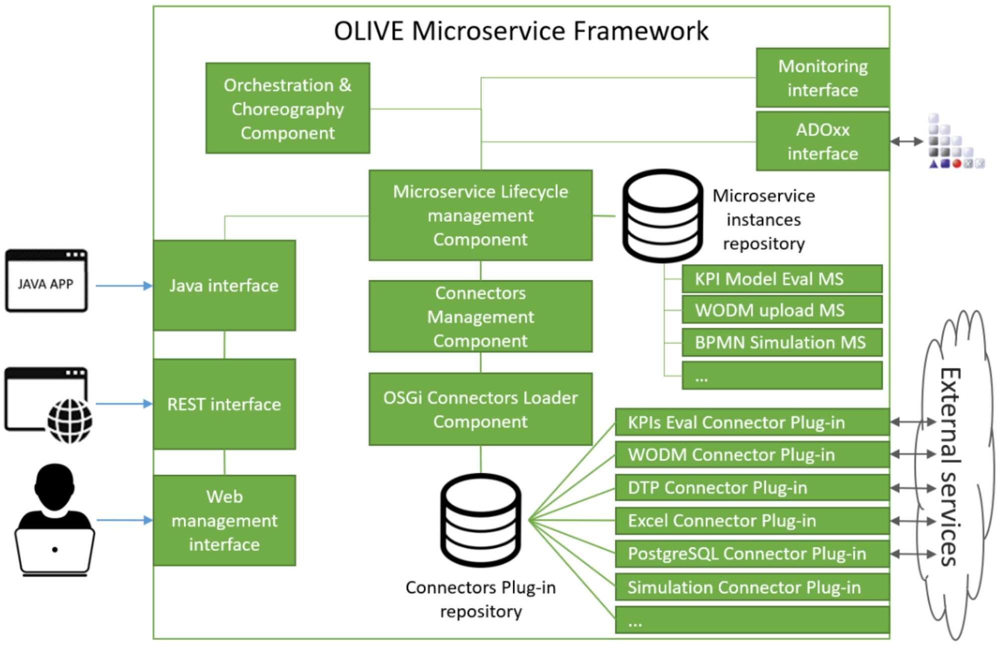

OLIVE Microservice Framework
Orchestration & Choreography Component
Monitoring interface
ADOxx interface
Microservice Lifecycle management Component
Microservice instances repository
JAVA APP
Java interface
KPI Model Eval MS
WODM upload MS
BPMN Simulation MS
...
Connectors Management Component
REST interface
OSGi Connectors Loader Component
KPIs Eval Connector Plug-in
WODM Connector Plug-in
DTP Connector Plug-in
Excel Connector Plug-in
PostgreSQL Connector Plug-in
Simulation Connector Plug-in
...
Web management interface
Connectors Plug-in repository
External services

Figure 29 - 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 Felix[14] 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 Pattern[15] (EIP) 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 Nashon JS engine for the services requiring data manipulation. Finally, the deployment of the solution is entirely based on provided Docker images.

[14] https://felix.apache.org/
[15] https://www.enterpriseintegrationpatterns.com/

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 was employed to create the services used by the PMS Modelling component to exchange the process model with the PMS Simulation and PMS Optimization components. The Simulation component uses services for the retrieval of the simulation input parameters and the generation of data for the Optimisation component, while the Sandbox Experiment component uses services for storing and retrieving tracked workers and equipment as well as for evaluating the material and process status. In particular the following services have been created:

- Process retrieval service: the microservice uses the OLIVE REST Connector to retrieve the list of all available processes from the Modelling component.
- BPMN export service: the microservice uses the OLIVE REST Connector to communicate with the Modelling component and returns the specified process in BPMN standard format.
- 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.
- Material status retrieval service: the microservice uses an image recognition technique to identify the availability of each material in 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.
- 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.

In the second release of the PMS toolkit, microservices for the communication with the COGITO Digital Twin Platform and with the COGITO WODM component will be defined.

All services can be visualized from the PMS Sandbox Experiment entry page under the “Microservices” menu (Figure 30). 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.

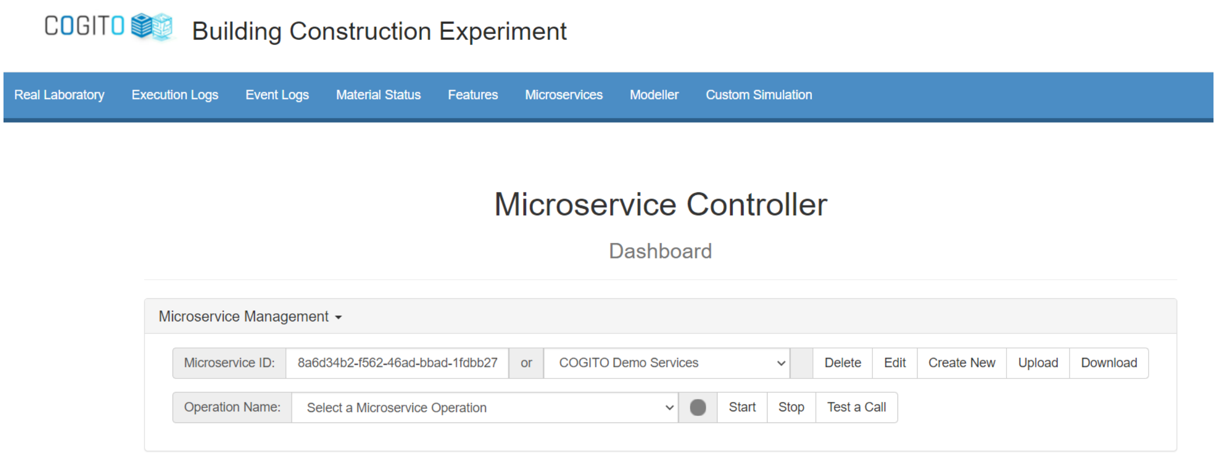


Figure 30 - Microservice management page

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/ 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

The following microservices are planned to be introduced in the second release of the component:

- Microservice to authenticate users through the COGITO Digital Twin platform.
- Microservice to retrieve the 4D BIM data from the COGITO Digital Twin platform. This microservice will be used by the PMS Modelling component to update the construction process models.
- Microservice to retrieve the workflows from the PMS Modelling component and send them to the COGITO WODM module for execution, including information on equipment, location, and scheduled times collected by the second release of the PMS Modelling component.
- Microservice for retrieval of real-time data from the COGITO WODM module.
- Microservice for the evaluation of process status, using the results of the "Equipment and Workers location retrieval service" and give an estimation of expected delay.

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: *The Computing solution...*	Type	Priorit y	Status

COGI-CS-1	**[DCC, WODM, PMS, BC, SafetyConAI, VirtualSafety, gQC] Runs on desktop or laptop PC**	• **Operation al**	**Must**	Supported: The service interface is accessible via browser on desktop or laptop PC
COGI-CS-4	**runs on Windows**	• **Operation al**	**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**	In progress: the microservices will not store data but retrieve from the Digital Twin platform
COGI-CS-8	**maintains communication and data security**	**•Legal •Design constraint**	**Must**	In progress: the microservices will authenticate the users using the Digital Twin platform authentication mechanism
COGI-CS-9	**differentiates data and system access levels and modification rights**	**•Legal •Functional •Design constraint**	**Must**	In progress: the microservices will be usable only to authorised users.

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

ID	Description: *The Workflow Planning, Execution and Monitoring solution ...*	Type	Priority	Status
COGI-WF-8	**[PMS, WODM] uses BIM models to support scheduling and budgeting**	**•Functional**	**Should**	In progress: The microservices for connection to the Digital Twin platform will be released in the second version of the component
COGI-WF-22	[PMS] supports conflict predictions and solving during scheduling (e.g., networks relocation before excavation work)	•Functional	Could	In progress: The microservices for the connection to the Optimisation component will be released in the second version of the component
COGI-WF-23	**[PMS, WODM] incorporates health and safety planning**	**•Functional •Design constraint**	**Should**	In progress: The microservices for the connection to the Digital Twin platform will be released in the

				second version of the component
COGI-WF-24	**[PMS, WODM] incorporates quality control planning**	**•Functional •Design constraint**	**Should**	In progress: The microservices for the connection to the Digital Twin platform will be released in the second version of the component

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	In progress: the microservice currently uses data from the experiment. Connection to the WODM is in progress
	Req-1.2	Update simulation model using real-time data from WODM	In progress: microservices retrieving data from the WODM will be released in the second version of the component
	Req-1.3	Connect and Authenticate to the DT platform	In progress: microservices connecting to the Digital Twin Platform will be released in the second version of the component
	Req-1.4	Estimates project progress	
	Req-1.5	Receive task progress from WODM	In progress: microservices retrieving data from the WODM will be released in the second version of the component
	Req-1.6	Output updated estimates of project progress to WODM according to real-time data and performed optimisation	In progress: microservices sending data to the WODM will be released in the second version of the component
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 configurating 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	In progress: the microservices will

			authenticate users through the Digital Twin Platform.

6.9 Assumptions and Restrictions

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

7 Conclusions

In this document, the first release of the COGITO PMS module is demonstrated focusing on the features of each component. In particular the following components have been demonstrated:

- Modelling component: generation of the construction processes, workflows, and KPIs 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, PMS integration with the other COGITO modules.

In the second release the focus will be on the interactions with the other COGITO modules in order to support the use cases that will be finalized in D2.5 (related to the second version of the COGITO system architecture). More details on on-going tasks and next steps that will be demonstrated in the second version of the PMS, 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.

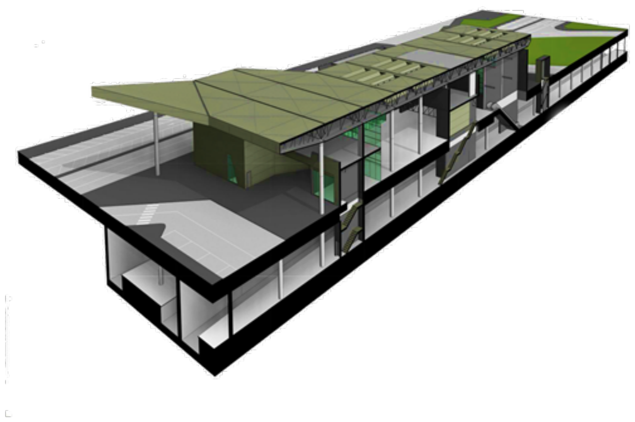
[5] G. D. Corporation, “General Algebraic Modeling System (GAMS) Release 38.1.0,” [Online]. Available: https://www.gams.com/download/. [Accessed 2021].

[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