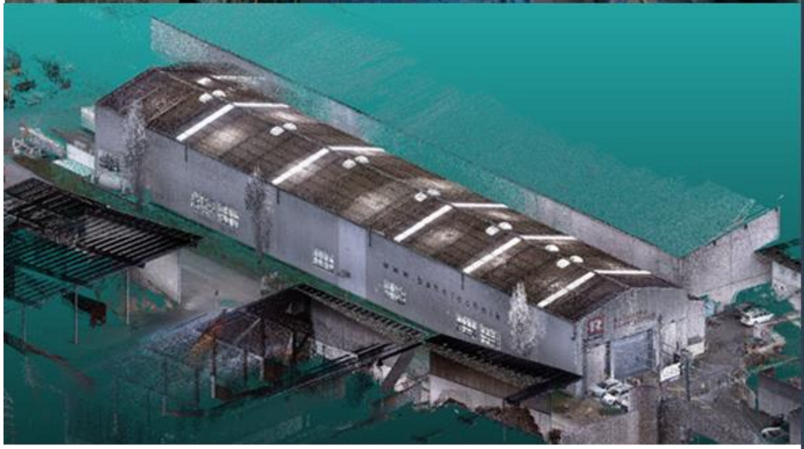


COGITO

CONSTRUCTION PHASE
DIGITAL TWIN MODEL

cogito-project.eu

D6.6 – Adaptive Workflow Management and Automation Tool v2

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

D6.6 – Adaptive Workflow Management and Automation Tool v2

Dissemination Level:	Public
Deliverable Type:	Demonstrator
Lead Partner:	NT
Contributing Partners:	UEDIN, BOC-AG, QUE
Due date:	31-10-2022
Actual submission date:	28-11-2022

Authors

Name	Beneficiary	Email
Ján Varga	NT	varga@novitechgroup.sk
Martin Straka	NT	straka@novitechgroup.sk
Bohuš Belej	NT	belej@novitechgroup.sk

Reviewers

Name	Beneficiary	Email
Giorgos Giannakis	Hypertech	g.giannakis@hypertech.gr
Konstantinos Papakostas	OLOD	kpapakostas@olympiaodos.gr

Version History

Version	Editors	Date	Comment
0.1	NT	01.09.2022	Table of Contents updated
0.3	NT	10.11.2022	Sections 1 and 2 updated
0.5	NT	15.11.2022	Draft version ready for internal review
0.7	Hypertech, OLOD	18.11.2022	First draft internal review
0.9	NT	23.11.2022	Comments addressed
0.8	NT, Hypertech	28.11.2022	Final version
1.0	NT, Hypertech	28.11.2022	Submission to the EC portal

Disclaimer

Executive Summary

The document describes the second version of the Adaptive Workflow Management and Automation Tool, which will orchestrate the construction process tasks and facilitate information exchange between them. It will further enable stakeholders to keep track of the entire process in real time as well as automated reporting and adaptation of the workflow in case of planned or unplanned developments during the construction phase.

The provided tool is a standalone toolset based on the I3D platform adjusted for the needs of COGITO, which provides the set of tools covering the whole life cycle of the management and utilisation of the know-how. In this version, the following main functionalities are provided:

- Importing the workflow created in the Process Modelling and Simulation tool (PMS);
- Management of the workflows;
- Creation of Work Orders that are issued when executing the workflows;
- Assignment of workorders and tasks to specific human resources;
- Monitoring and management of workorders and workflow tasks; and
- Exporting data of workorders to the Digital Twin Platform.

This first version focused on the definition of internal data structure and adjusting the existing I3D ecosystem to meet the COGITO requirements. The second release implements all the functionalities required for interfacing with other COGITO tools as defined in the use cases “UC1.1-Efficient and detailed project workflow planning using the project’s construction schedule and as-planned BIM model” and “UC1.2-Systematic and secure execution, monitoring and updating of the project workflow”.

The Adaptive Workflow Management and Automation tool development has been driven by the stakeholder’s requirements identified in “D2.1 – Stakeholder requirements for the COGITO system” and the tool specifications that have been documented in “D2.5 – COGITO system architecture v2”.

Table of contents

List of Figures

List of Tables

List of Acronyms

Term	Description
BCSC	BlockChain network Smart Contracts
BIM	Building Information Model
BPMN	Business Process Model and Notation
COGITO	Construction Phase diGItal Twin mOdel
DTP	Digital Twin Platform
I3D	I3D industrial services
IoT	Internet of Things
KPI	Key Performance Indicator
PMS	Process Modelling and Simulation
RAMS	Risk Assessment (and) Method Statement
SaaS	Software as a Service
SLAM	Service Level Agreements Manager
WO	Work Order
WODM	Workflow Management and Automation
WOEA	Work Order Execution Assistance

1 Introduction

1.1 Scope and Objectives of the Deliverable

The purpose of this deliverable is to document the final release of an Adaptive Workflow Management tool, named hereafter Work Order Definition and Management tool (WODM), that orchestrates the construction process tasks. It enables stakeholders to keep track of the entire process in real time as well as timely report and adapt the workflow in case of planned or unforeseen changes during the construction phase. The first activity of this task is the definition and implementation of the concrete, executable workflow based on the stakeholders' requirements collected and documented in the deliverable "D2.1 – Stakeholder requirements for the COGITO system" as well as the outcomes of "T6.2 – Adaptive Processes/Workflow Modelling and Simulation-based Optimization" in terms of construction process analysis.

Built upon an existing solution that is briefly described in the following sections, extensions and adjustments are being implemented to meet the COGITO requirements for workorder modelling and monitoring, and interactions with other components of the COGITO ecosystem. In parallel, interfaces to applications developed in T6.4 are also being modified to enable an automated monitoring and reporting of the construction works. In this version automatic data exchange has been implemented, using as input the School project data available in the Digital Twin Platform (DTP). However, it remains subject for further investigation how the implemented functionalities behave on real data and with real end-users engagement.

1.2 Relation to other Tasks and Deliverables

This deliverable is closely related to other WP6 tasks, namely "T6.1-Blockchain & Smart Contracts on the Workflow Modelling and Management", "T6.2-Adaptive Processes/Workflow Modelling and Simulation-based Optimization" and "T6.4-Personalized On-site Works Support and Relevant Apps Development" and their main outcomes (corresponding deliverables). It is also related to task "T7.1-Digital Twin Platform Design & Interface Specification" and its deliverables. The end-user requirements for WODM were gathered and described in "D2.1-Stakeholder requirements for the COGITO system". Furthermore, the specifications, the functional and non-functional requirements, as well as the interactions of WOEA (which is a frontend of WODM used on mobile devices by on-field stakeholders like workers, inspectors etc.) with other components of the COGITO ecosystem are presented in "D2.5-System Architecture v2".

1.3 Structure of the Deliverable

This deliverable contains following sections:

- prototype overview;
- technology stack and implementation tools used;
- API documentation;
- licensing information;
- installation instructions;
- development and integration status overview;
- requirements coverage; and
- assumptions and restrictions.

1.4 Updates to the first version of WODM

The second version of deliverable presents new features and improvements in comparison with the first version of this deliverable, namely "D6.5 – Adaptive Workflow Management and Automation Tool v1".

First, from the technology stack viewpoint, the version of Angular framework has been updated to accommodate the implementation of Swagger service which is used for management and documentation of REST endpoints within the WODM server structure. From a server infrastructure perspective, a backup server has been installed to ensure separation of development environment and implementation environment, including improved reliability of the system. The WODM's internal database (and data model) has been updated to follow to the extent possible the latest version of the COGITO ontology and data model. Additional data structures to manage the SLAM and KPIs data are supported, which are separated from original data structure of I3D to ensure better transparency of code.

As far as the WODM's interaction with other COGITO components is concerned, all data exchange formats have been refined to reflect the current status of the integrated COGITO ecosystem, and consequently, all the input and output data illustrated in this document have been updated to present data exchange results created based on the School project data that have been accessible through the DTP endpoints. An ETL service to import JSON from DTP has been created for manual and automatic upload from DTP. REST endpoints for communication with PMS, SLAM and BCSC have been implemented. The WODM's integration with the Identity Provider of the DTP has been successfully completed, providing a webservice used for authentication of WOEA (described in D6.8).

2 WODM – Work Order Definition and Management tool

The development and delivery of WODM has been based on an existing workflow execution engine, called I3D Platform, provided by NT. The following key sub-components of the I3D Platform are kept, refined, and repurposed to meet the tool requirements that have been detailed in D2.1 and D2.5:

- services to define and manage the templates of work processes, enriched with an Extract Transform Load (ETL) tool to import work processes in Business Process Model and Notation (BPMN) format and automatically translate its content to populate the WODM's data model; this update will streamline the interactions and data exchange with the Process Modelling and Simulation tool (PMS). This ETL service has been changed to accommodate JSON format instead BPMN as it was agreed during project kick off.
- services to issue executable work processes – work orders based on templates. Once the process template is defined, it can be used to generate running instances.
- sevices for assignation of resources to task
- services for the execution of work processes in semi-automatic way with the options to enable user interactions with running workorders and tasks.

The I3D Platform functionalities and services that are not relevant to the scope of COGITO have been removed, i.e., location manager and editor. Other functionalities are considered as temporary solutions, for instance the internal I3D Identity Management, these will be used as development option even though they are not necessary for COGITO project. WODM use original internal database structure from I3D updated by necessary data structures based on COGITO ontology model defined in D3.4 – COGITO Data Model and Ontology Definition and Interoperability Design v3.

2.1 Prototype Overview

2.1.1 I3D overview

The data structure of the I3D platform is depicted in Figure 1. A workflow, as well as its executable instance, i.e., workorder, consists of a set of steps. Every step consists of several actions (typically working instructions). Step are connected to a workspace. The predefined way of execution of a work process is sequential execution of every action, step by step. This sequence can be adjusted with events and preconditions, which allow to skip the execution of any of the actions.

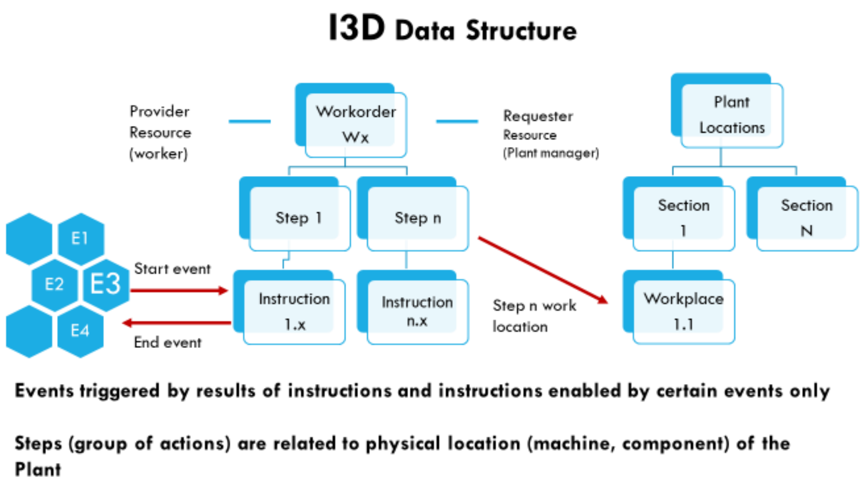
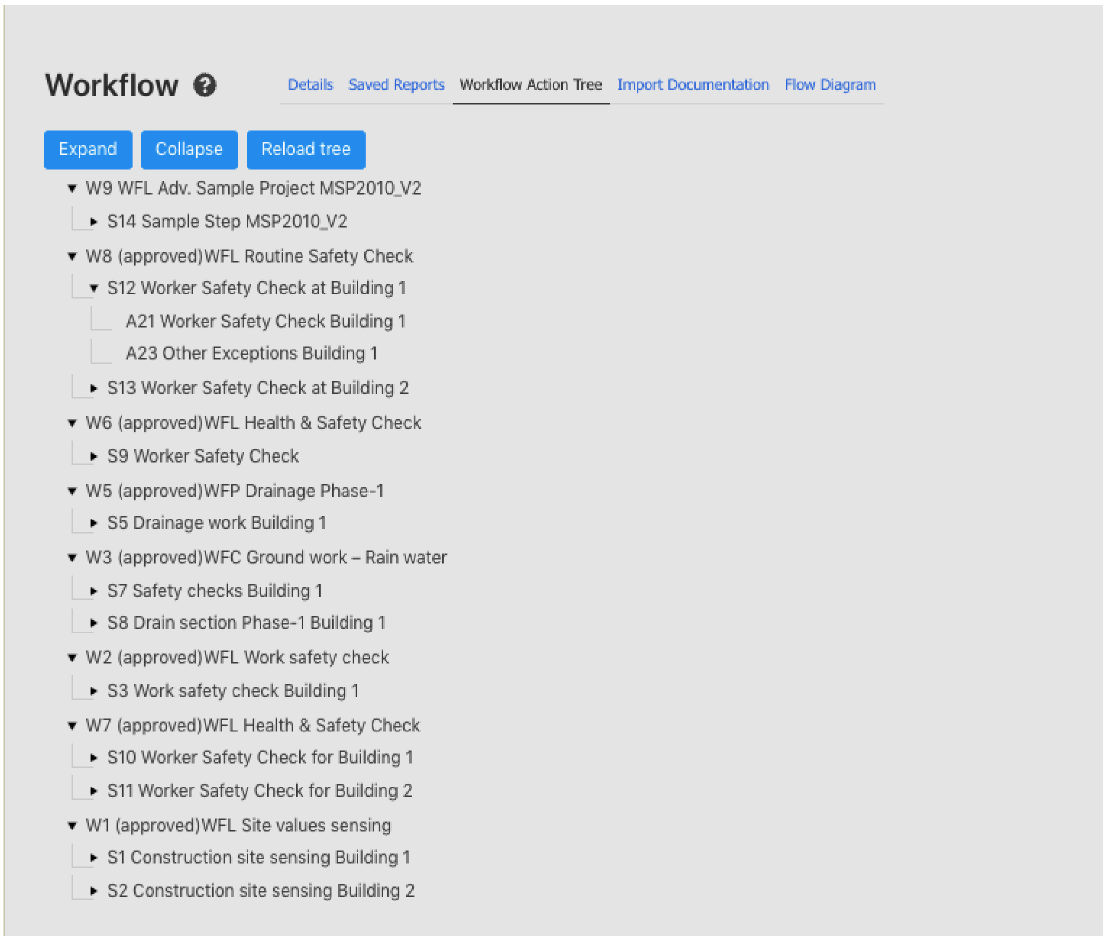

I3D Data Structure
Provider Resource (worker)
Workorder Wx
Requester Resource (Plant manager)
Plant Locations
Step 1
Step n
Section 1
Section N
E1
E2
E3
E4
Start event
End event
Instruction 1.x
Instruction n.x
Step n work location
Workplace 1.1
Events triggered by results of instructions and instructions enabled by certain events only
Steps (group of actions) are related to physical location (machine, component) of the Plant

Figure 1 – Data structure of the I3D platform

Since the WODM tool constitutes a revised and updated version of I3D platform, it follows its own terminology, not fully aligned with the COGITO's terminology. Therefore, an abridged description of some key terms (workflow, workorder, step, action) is given below.

In I3D, the term **Workflow** is used for a work process definition, a template, which contains a step-by-step definition of a work process defining the “where, how and what” that needs to be done and “with what” resources. In terms of COGITO, this is the process model generated by the PMS. Workflows (with prefix W), and their relations with **Steps** (with prefix S) and **Actions** (with prefix A), as they visualised in the WODM UI, are illustrated in Figure 2. Tasks defined in COGITO are related with Steps and actions in I3D platform.



- W9 WFL Adv. Sample Project MSP2010_V2
  - S14 Sample Step MSP2010_V2
- W8 (approved)WFL Routine Safety Check
  - S12 Worker Safety Check at Building 1
    - A21 Worker Safety Check Building 1
    - A23 Other Exceptions Building 1
  - S13 Worker Safety Check at Building 2
- W6 (approved)WFL Health & Safety Check
  - S9 Worker Safety Check
- W5 (approved)WFP Drainage Phase-1
  - S5 Drainage work Building 1
- W3 (approved)WFC Ground work – Rain water
  - S7 Safety checks Building 1
  - S8 Drain section Phase-1 Building 1
- W2 (approved)WFL Work safety check
  - S3 Work safety check Building 1
- W7 (approved)WFL Health & Safety Check
  - S10 Worker Safety Check for Building 1
  - S11 Worker Safety Check for Building 2
- W1 (approved)WFL Site values sensing
  - S1 Construction site sensing Building 1
  - S2 Construction site sensing Building 2

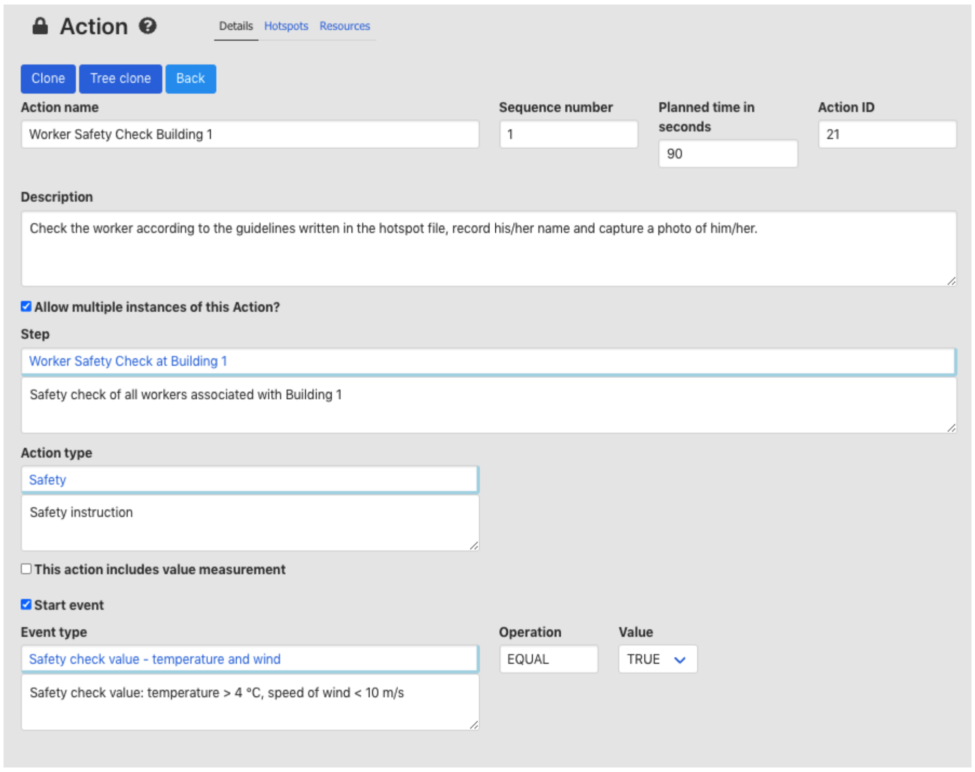
Figure 2 – Workflow tree example

The status of a workflow can be draft, approved or expired. A workflow is closed only if its status is set to “approved”, meaning that it is protected against any unauthorised changes. Only an approved workflow can be used to generate a workorder. Expiration of workflow is useful when external optimisations of a process is ongoing. In that case the old workflow is expired and replaced by a new, optimized instance.

Every workflow has one or more **Steps**. Each step is linked to a specific location. Every step has its own sequence number which defines sequence of the steps inside workflow. Where needed, steps can be used to group actions that are executed at the same place.

Every step consists of one or more **Actions** (instructions). An action represents the exact work that must be done. An example of an action is depicted in Figure 3. The action has a short name and long description, and an expected duration which expresses the time required to complete the action. Specific human and equipment **Resources** are assigned to each action that will be used to execute it. Furthermore, each action can have events. **Events** are rules that are defined and assigned to an action to evaluate its results.

Workorder is the executable instance of workflow. In other words, it it’s the actual implementation (or instantiation) of a plan of work that includes information about the actions that need to be performed by specific workers within the predefined timeframe. The aim of the workorders management in COGITO is to have exact information of their statuses and continuously optimising all workorders according to changing conditions.



Action name	Sequence number	Planned time in seconds	Action ID
Worker Safety Check Building 1	1	90	21

Description

Check the worker according to the guidelines written in the hotspot file, record his/her name and capture a photo of him/her.

☑ Allow multiple instances of this Action?

Step

Worker Safety Check at Building 1

Safety check of all workers associated with Building 1

Action type

Safety

Safety instruction

☐ This action includes value measurement

☑ Start event

Event type	Operation	Value
Safety check value - temperature and wind	EQUAL	TRUE

Safety check value: temperature > 4 °C, speed of wind < 10 m/s

Figure 3 – An Action instance

2.1.2 Data model

The data model of WODM is slightly different from COGITO data model described in “D3.4 - COGITO Data Model and Ontology Definition and Interoperability Design v3”. The original data model of WODM is shown in (Figure 4).

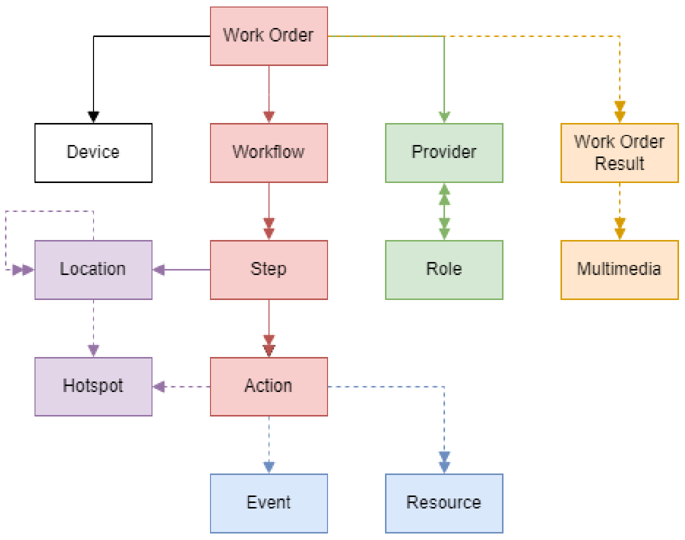


Figure 4 – Original WODM data model

To ease the data processing the communication with the DTP updates have been performed in the original WODM data model. The resulted data model is depicted in Figure 5.

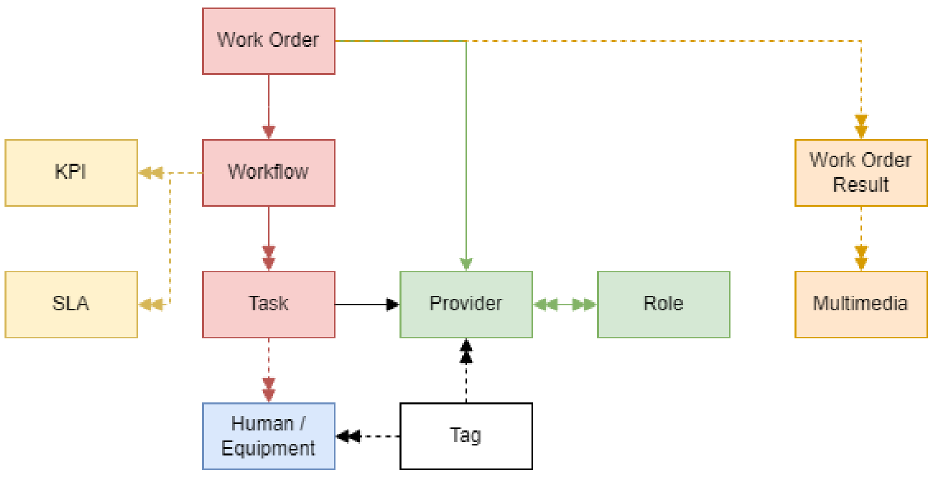


Figure 5 – Updated I3D data model

The main difference is the definition of **Task**. In the original WODM data model, each task consists of **Steps** and **Actions**, as described in chapter 2.1.1. To avoid radical changes in the WODM database, we decided to join **Step** and **Action** to COGITO **Task**. In other words, each COGITO task consists of 1 Step and "n" actions. This concept will be hidden for the user and only the term "**Task**" will be used in UI.

2.2 Technology Stack and Implementation tools

WODM is a backend solution assisted by two frontend components: (1) the WOEA application, documented in "D6.8-Personalized On-site Works Support App v2"; and (2) the WODM UI which is described in "D6.10-Workflow User Interface for Project Managers v2". The WODM's deployment architecture has been based on state-of-the-art technologies, as Figure 6 depicts. The data storage layer utilizes PostgreSQL to store project specific data provided by other COGITO components or generated by WODM backend. As for the business logic layer, to host all the ETL tools developed to interface with other COGITO components, a JAVA application server and a JAVA web server are deployed. HIBERNATE is used to achieve the connection with the Data Storage Layer. Finally, a webservice enables the data exchange with wearable devices, where WOEA is installed.

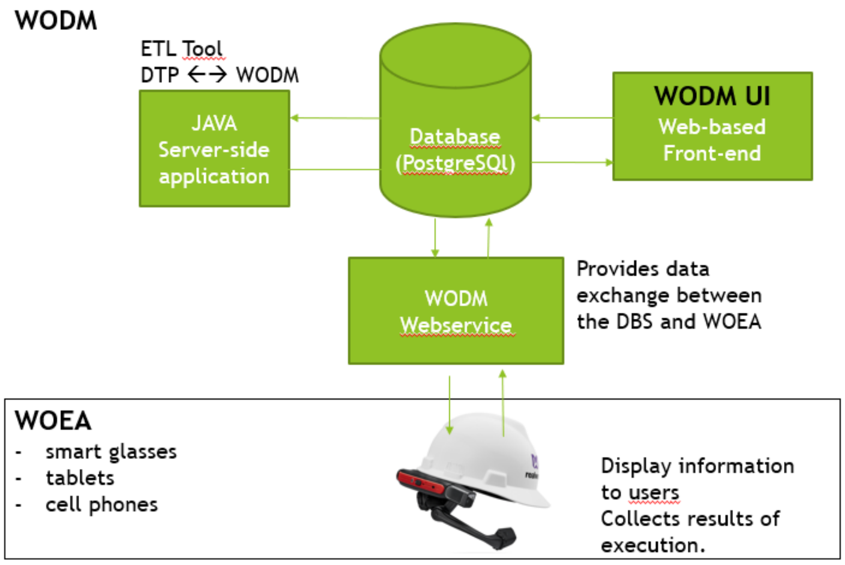

WODM
ETL Tool
DTP <-> WODM
JAVA Server-side application
Database (PostgreSQL)
WODM UI
Web-based Front-end
WODM Webservice
Provides data exchange between the DBS and WOEA
WOEA
- smart glasses
- tablets
- cell phones
Display information to users
Collects results of execution.

Figure 6 – WODM deployment architecture

All the libraries and frameworks used for developing WODM are summarised in Table 1.

Table 1 – Libraries and Frameworks used in WODM

Library/Technology Name	Version	License
Angular 7.2	7.2	MIT licence
Ng2-charts	4.0.0	ISC licence
Angular-tree-component	8.2.0	MIT licence
@ngx-translate/i18-polyfill	1.0.0	MIT licence
Ng2-cytoscape	0.4.0	MIT licence
Angular2-query-builder	0.5.0	MIT licence
jquery	3.6.0	MIT licence
PHPMailer	6.0.7	LGPL 2.1

Angular is a development platform, built on TypeScript. As a platform, Angular includes: (1) a component-based framework for building scalable web applications, (2) a collection of well-integrated libraries that cover a wide variety of features, including routing, forms management, client-server communication, and more; and (3) a suite of developer tools to help develop, build, test, and update the code

Ng2-charts is a library to create charts based on Chart.js

Angular-tree-component allows users to represent hierarchical data in a tree-view structure, maintaining parent-child relationships, as well as to define static tree-view structure without a corresponding data mode.

@ngx-translate/i18-polyfill is an extraction tool and service to add support for code translations in Angular.

Ng2-cytoscape is an Angular wrapper around the CytoscapeJS 3 module. This library enables to add Cytoscape-based graph visualizations into Angular application, with values supplied by instance variables.

Angular2-query-builder - A modernized Angular 4+ query builder based on jQuery QueryBuilder. It provides support for heavy customization with Angular components and a flexible way to handle custom data types.

Jquery is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers.

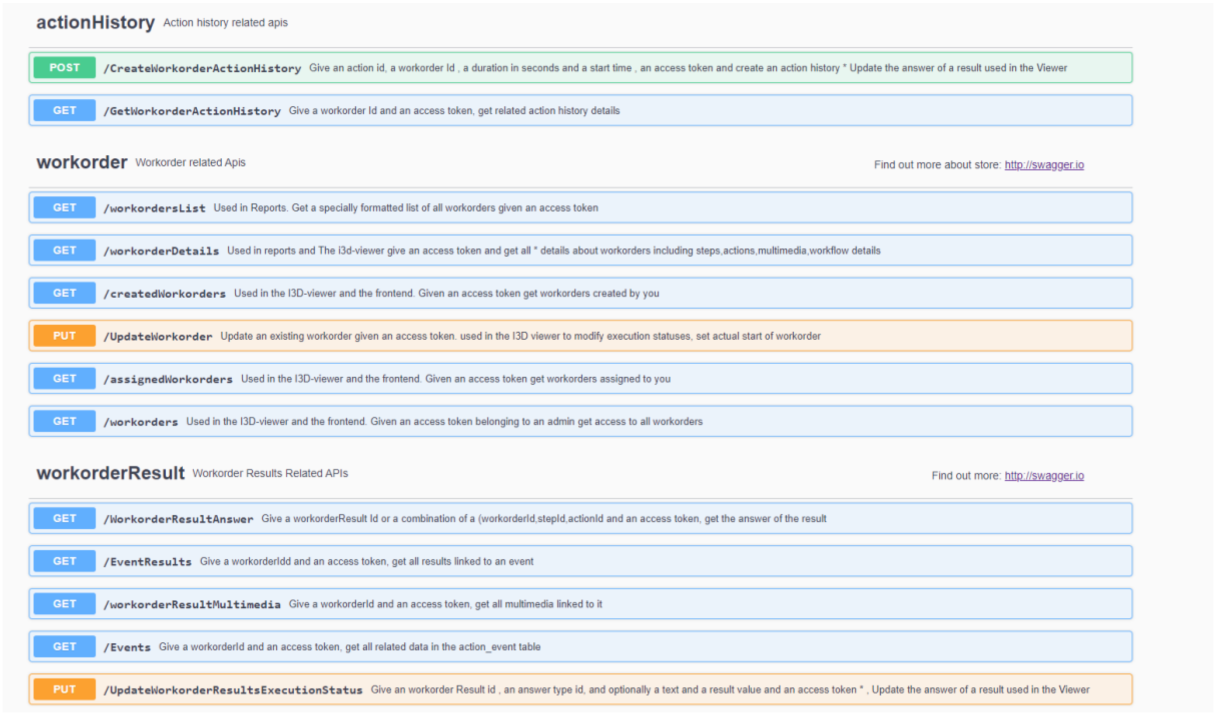
PHPMailer is a code library to send emails safely and easily via PHP code from a web server.

2.3 Input, Output and API Documentation

2.3.1 REST API documentation

The documentation of the WODM APIs is presented in Swagger (Figure 7). The WODM (originally I3D) API documentation is accessible through the following link:

[REST API documentation](https://i3d.econtent.lu/gl/webservices-core/swagger.json) (https://i3d.econtent.lu/gl/webservices-core/swagger.json).



actionHistory Action history related apis

POST /CreateWorkorderActionHistory Give an action id, a workorder Id , a duration in seconds and a start time , an access token and create an action history * Update the answer of a result used in the Viewer

GET /GetWorkorderActionHistory Give a workorder Id and an access token, get related action history details

workorder Workorder related Apis — Find out more about store: http://swagger.io

GET /workordersList Used in Reports. Get a specially formatted list of all workorders given an access token

GET /workorderDetails Used in reports and The i3d-viewer give an access token and get all * details about workorders including steps,actions,multimedia,workflow details

GET /createdWorkorders Used in the I3D-viewer and the frontend. Given an access token get workorders created by you

PUT /UpdateWorkorder Update an existing workorder given an access token. used in the I3D viewer to modify execution statuses, set actual start of workorder

GET /assignedWorkorders Used in the I3D-viewer and the frontend. Given an access token get workorders assigned to you

GET /workorders Used in the I3D-viewer and the frontend. Given an access token belonging to an admin get access to all workorders

workorderResult Workorder Results Related APIs — Find out more: http://swagger.io

GET /WorkorderResultAnswer Give a workorderResult Id or a combination of a (workorderId,stepId,actionId and an access token, get the answer of the result

GET /EventResults Give a workorderIdd and an access token, get all results linked to an event

GET /workorderResultMultimedia Give a workorderId and an access token, get all multimedia linked to it

GET /Events Give a workorderId and an access token, get all related data in the action_event table

PUT /UpdateWorkorderResultsExecutionStatus Give an workorder Result id , an answer type id, and optionally a text and a result value and an access token * , Update the answer of a result used in the Viewer

Figure 7 - REST API documentation

2.3.2 Data interaction

WODM, as an orchestrator of the workflow management and execution, interacts with all relevant services or tools.

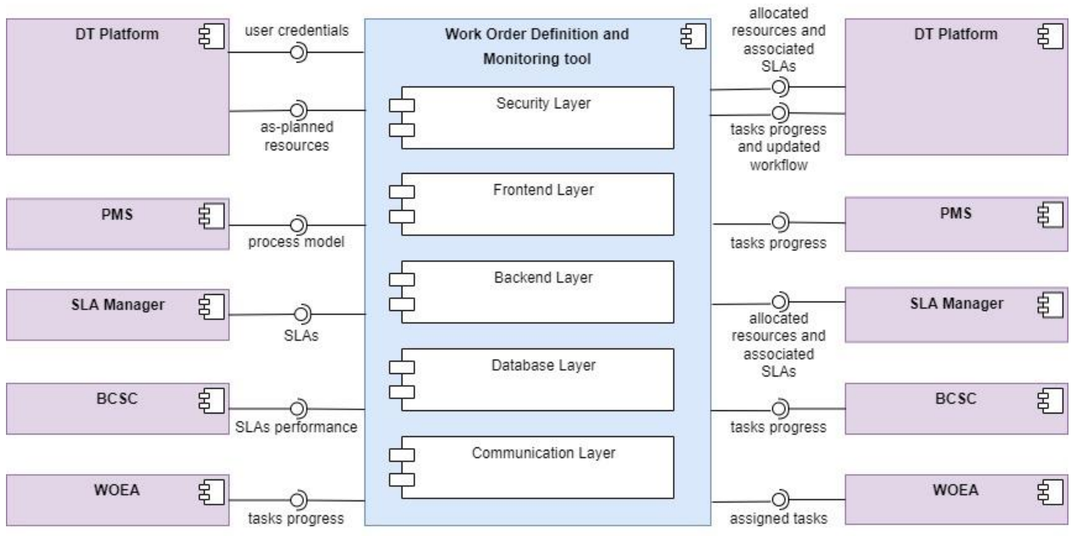


Figure 8 – WODM component diagram

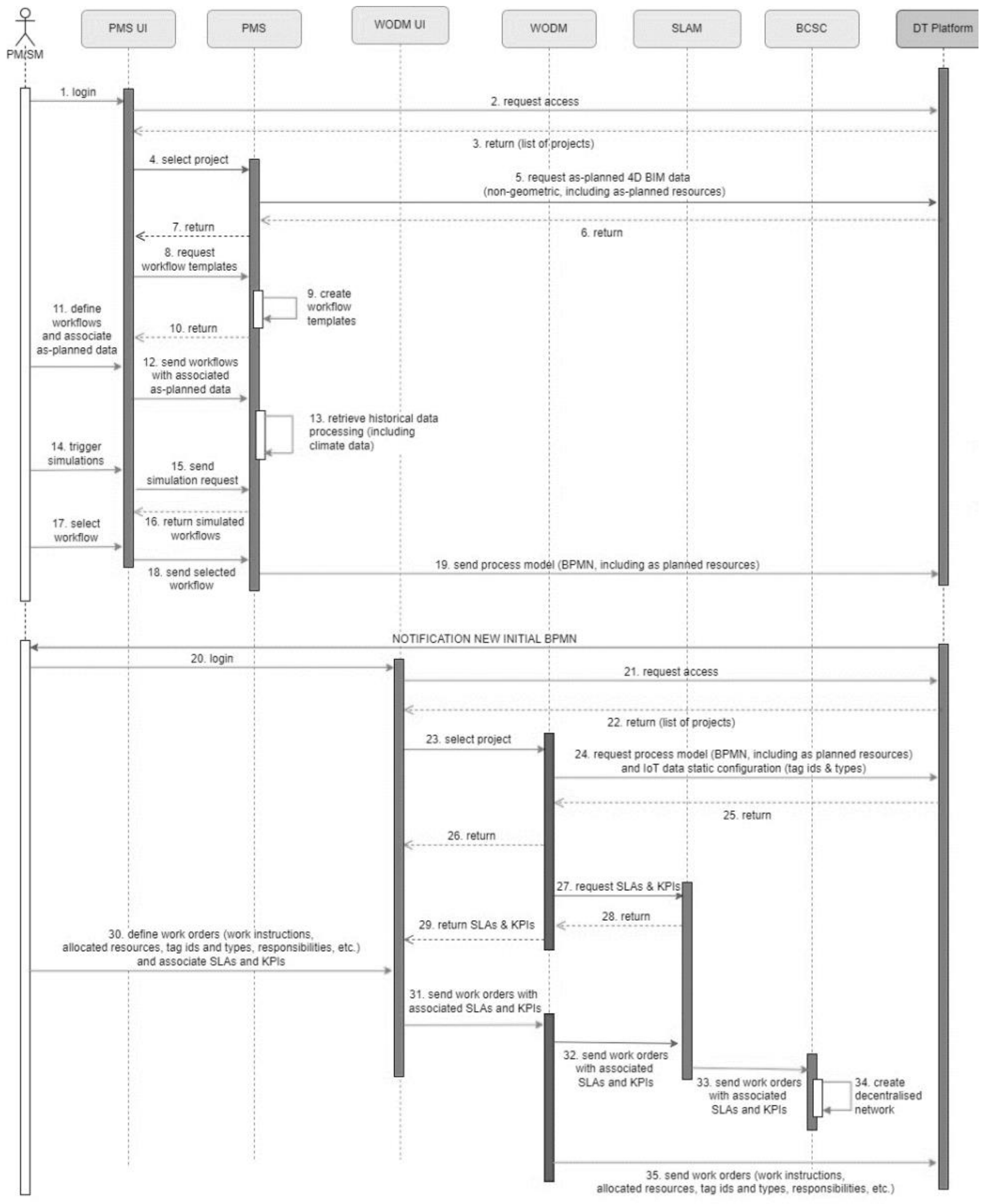

PM/SM
PMS UI
PMS
WODM UI
WODM
SLAM
BCSC
DT Platform
1. login
2. request access
3. return (list of projects)
4. select project
5. request as-planned 4D BIM data (non-geometric, including as-planned resources)
6. return
7. return
8. request workflow templates
9. create workflow templates
10. return
11. define workflows and associate as-planned data
12. send workflows with associated as-planned data
13. retrieve historical data processing (including climate data)
14. trigger simulations
15. send simulation request
16. return simulated workflows
17. select workflow
18. send selected workflow
19. send process model (BPMN, including as planned resources)
NOTIFICATION NEW INITIAL BPMN
20. login
21. request access
22. return (list of projects)
23. select project
24. request process model (BPMN, including as planned resources) and IoT data static configuration (tag ids & types)
25. return
26. return
27. request SLAs & KPIs
28. return
29. return SLAs & KPIs
30. define work orders (work instructions, allocated resources, tag ids and types, responsibilities, etc.) and associate SLAs and KPIs
31. send work orders with associated SLAs and KPIs
32. send work orders with associated SLAs and KPIs
33. send work orders with associated SLAs and KPIs
34. create decentralised network
35. send work orders (work instructions, allocated resources, tag ids and types, responsibilities, etc.)

Figure 9 – Sequence diagram of UC-1.1

Namely they are:

- DTP (Digital Twin Platform): DTP interacts with WODM as Identity Provider and As-planned resources provider. WODM is sending task progress and information of allocated resources with associated SLAs.
- PMS (Process Model and Simulation): PMS is providing Process Model and WODM sends Task Progress update to PMS.
- SLAM (Service Level Agreement Manager): SLAM is providing KPI and SLA definition, WODM sends workorders with associated SLAs and KPIs.
- BCSC (Blockchain - Smart Contract): BCSC sends SLAs performance and WODM sends updated Work Orders with associated SLAs and KPIs.
- WOEA (Work Order Execution Assistance): This is internal communication within WODM tool, on-field device with WOEA tool receiving assigned task to be executed by user and sends task status to WODM.

These interactions, introduced in "D2.5-COGITO System Architecture v2", are depicted in Figure 8. In this section, examples of the payload formats that have been defined and agreed among the involved partners for each data interaction with WODM are provided.

The sequence diagrams of UC 1.1 and UC1.2, thoroughly described in "D2.5-System Architecture v2" and updated during Working Groups meetings (WG1 and WG2) are illustrated in Figure 9 and Figure 10, respectively. Each arrow represents a data flow (exchange) and has a number. Those numbers will be referenced in the following data exchange descriptions.

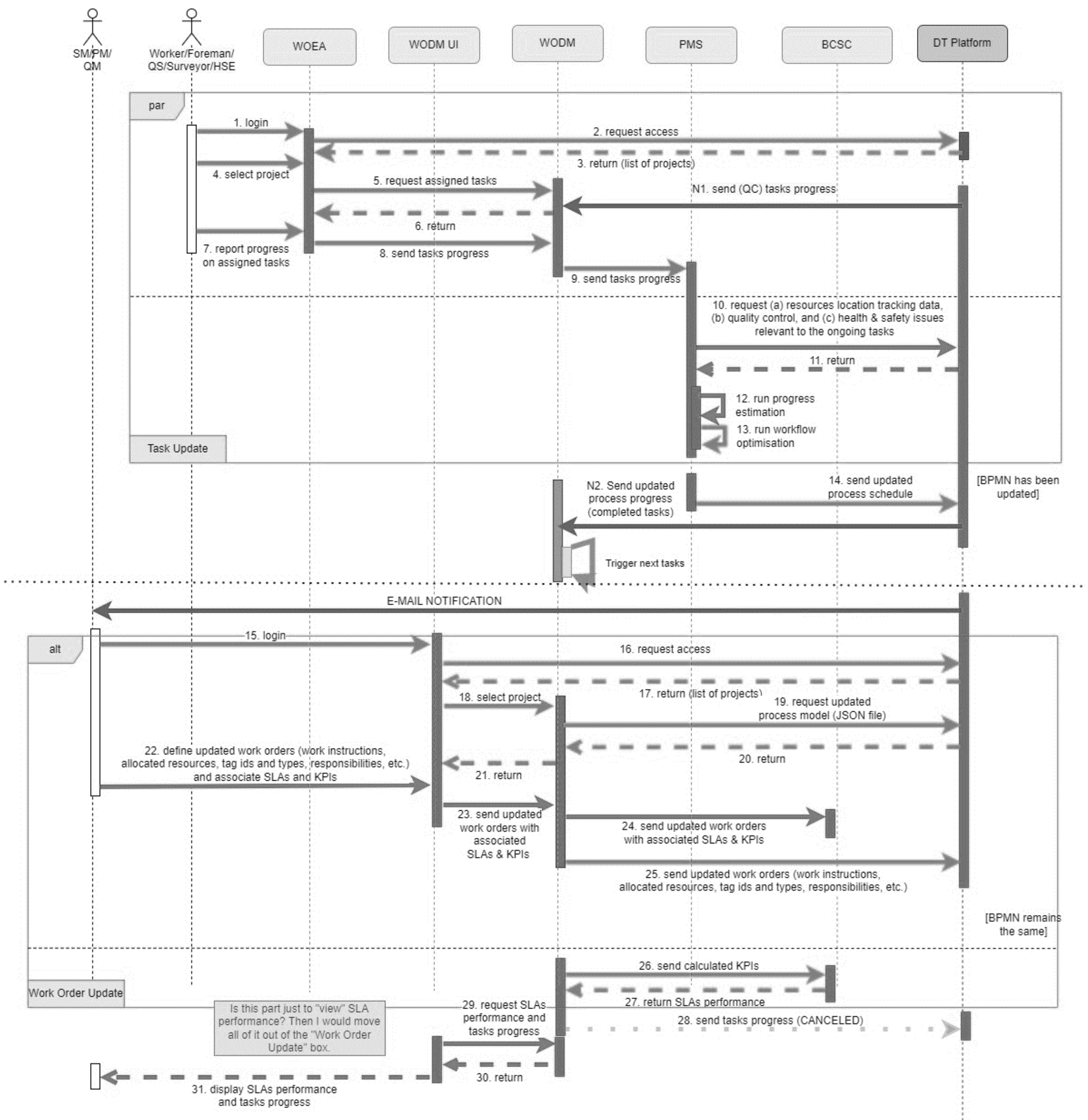
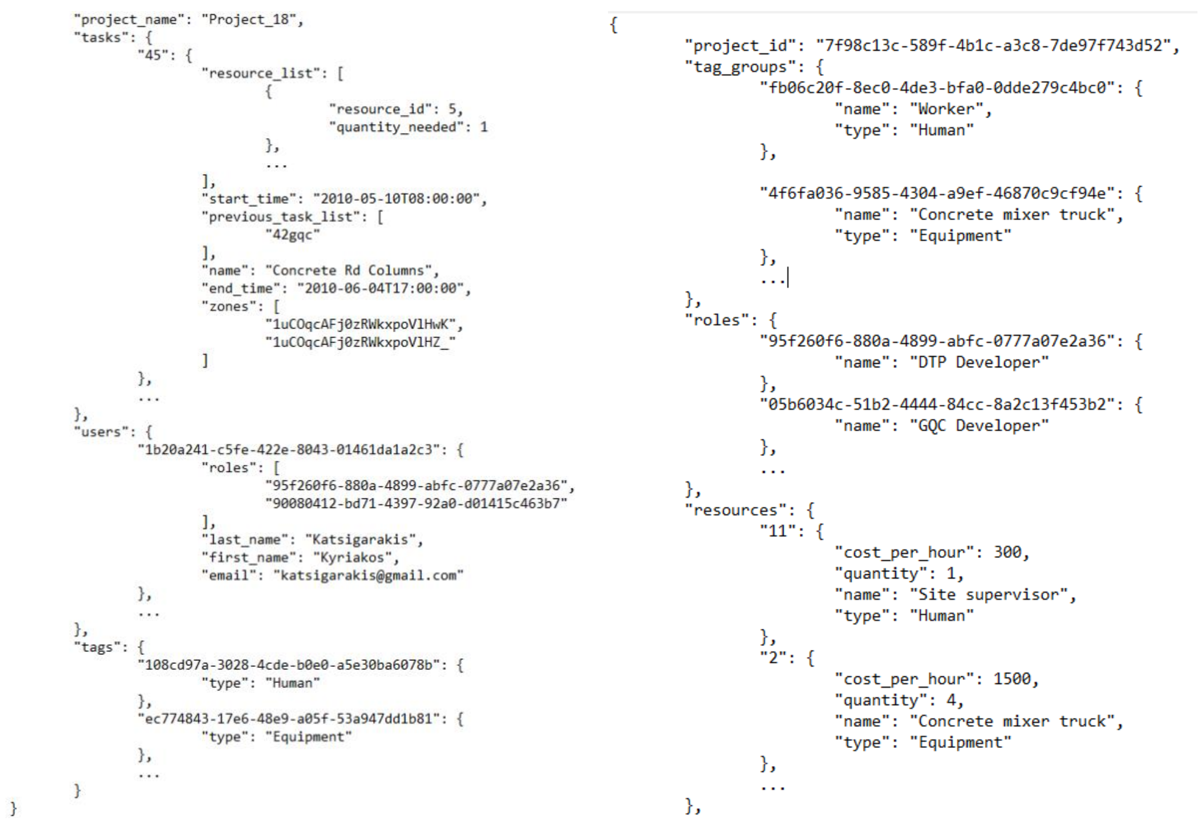

SM/PM/
QM
Worker/Foreman/
QS/Surveyor/HSE
WOEA
WODM UI
WODM
PMS
BCSC
DT Platform
par
1. login
2. request access
3. return (list of projects)
4. select project
5. request assigned tasks
N1. send (QC) tasks progress
6. return
7. report progress
on assigned tasks
8. send tasks progress
9. send tasks progress
10. request (a) resources location tracking data,
(b) quality control, and (c) health & safety issues
relevant to the ongoing tasks
11. return
12. run progress
estimation
13. run workflow
optimisation
Task Update
N2. Send updated
process progress
(completed tasks)
14. send updated
process schedule
[BPMN has been
updated]
Trigger next tasks
E-MAIL NOTIFICATION
alt
15. login
16. request access
17. return (list of projects)
18. select project
19. request updated
process model (JSON file)
20. return
22. define updated work orders (work instructions,
allocated resources, tag ids and types, responsibilities, etc.)
and associate SLAs and KPIs
21. return
23. send updated
work orders with
associated
SLAs & KPIs
24. send updated work orders
with associated SLAs & KPIs
25. send updated work orders (work instructions,
allocated resources, tag ids and types, responsibilities, etc.)
[BPMN remains
the same]
26. send calculated KPIs
Work Order Update
27. return SLAs performance
Is this part just to "view" SLA
performance? Then I would move
all of it out of the "Work Order
Update" box.
29. request SLAs
performance and
tasks progress
28. send tasks progress (CANCELED)
30. return
31. display SLAs performance
and tasks progress

Figure 10 – Sequence diagram of UC-1.2

From DTP to WODM – planning phase (UC1.1 exchange 24, 25): The main input for WODM is the process model designed in the PMS. To adapt this information to data model of WODM respecting original data model of I3D we use an ETL service for importing various JSON files automatically, but for development purposes also manually. Since DTP acts as the central repository of COGITO, the involved partners aggreged that the PMS-to-WODM interaction should be realised through DTP. The JSON format that is used for that data exchange is shown in Figure 11.



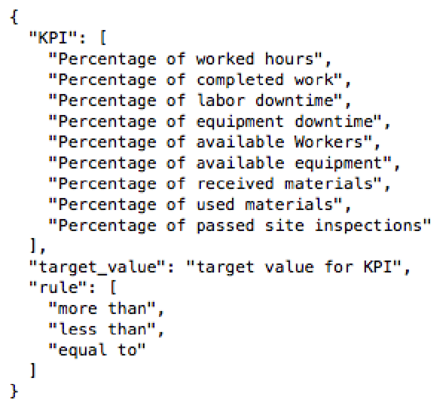
```
        "project_name": "Project_18",
        "tasks": {
                "45": {
                        "resource_list": [
                                {
                                        "resource_id": 5,
                                        "quantity_needed": 1
                                },
                                ...
                        ],
                        "start_time": "2010-05-10T08:00:00",
                        "previous_task_list": [
                                "42gqc"
                        ],
                        "name": "Concrete Rd Columns",
                        "end_time": "2010-06-04T17:00:00",
                        "zones": [
                                "1uCOqcAFj0zRWkxpoV1HwK",
                                "1uCOqcAFj0zRWkxpoV1HZ_"
                        ]
                },
                ...
        },
        "users": {
                "1b20a241-c5fe-422e-8043-01461da1a2c3": {
                        "roles": [
                                "95f260f6-880a-4899-abfc-0777a07e2a36",
                                "90080412-bd71-4397-92a0-d01415c463b7"
                        ],
                        "last_name": "Katsigarakis",
                        "first_name": "Kyriakos",
                        "email": "katsigarakis@gmail.com"
                },
                ...
        },
        "tags": {
                "108cd97a-3028-4cde-b0e0-a5e30ba6078b": {
                        "type": "Human"
                },
                "ec774843-17e6-48e9-a05f-53a947dd1b81": {
                        "type": "Equipment"
                },
                ...
        }
}
```

```
{
        "project_id": "7f98c13c-589f-4b1c-a3c8-7de97f743d52",
        "tag_groups": {
                "fb06c20f-8ec0-4de3-bfa0-0dde279c4bc0": {
                        "name": "Worker",
                        "type": "Human"
                },

                "4f6fa036-9585-4304-a9ef-46870c9cf94e": {
                        "name": "Concrete mixer truck",
                        "type": "Equipment"
                },
                ...
        },
        "roles": {
                "95f260f6-880a-4899-abfc-0777a07e2a36": {
                        "name": "DTP Developer"
                },
                "05b6034c-51b2-4444-84cc-8a2c13f453b2": {
                        "name": "GQC Developer"
                },
                ...
        },
        "resources": {
                "11": {
                        "cost_per_hour": 300,
                        "quantity": 1,
                        "name": "Site supervisor",
                        "type": "Human"
                },
                "2": {
                        "cost_per_hour": 1500,
                        "quantity": 4,
                        "name": "Concrete mixer truck",
                        "type": "Equipment"
                },
                ...
        },
```

Figure 11 – DTP to WODM JSON: planning phase

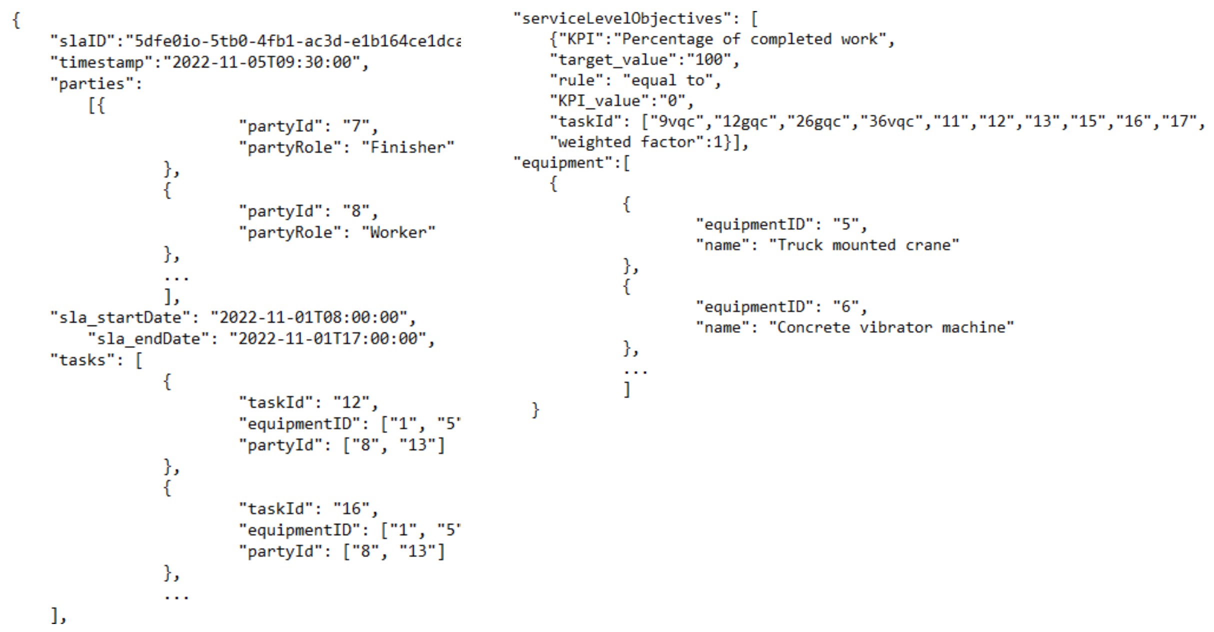
From SLAM to WODM (UC1.1 exchange 27,28): SLAM provides the WODM with the KPIs definition. These KPIs represent measurable performance of construction tasks. This exchange is implemented and manually tested. In the phase of ICT integration, automatic data exchange will be validated with real project data. The structure of the JSON file is presented in Figure 12.



```
{
  "KPI": [
    "Percentage of worked hours",
    "Percentage of completed work",
    "Percentage of labor downtime",
    "Percentage of equipment downtime",
    "Percentage of available Workers",
    "Percentage of available equipment",
    "Percentage of received materials",
    "Percentage of used materials",
    "Percentage of passed site inspections"
  ],
  "target_value": "target value for KPI",
  "rule": [
    "more than",
    "less than",
    "equal to"
  ]
}
```

Figure 12 – SLAM to WODM JSON

From WODM to SLAM (UC1.1 exchange 32): This data exchange provides the SLAM with the WODM's workorders with their directly associated KPIs and SLAs. The associations are made on the side of WODM after it receives the list of KPIs and SLAs from the SLAM sooner (*UC1.1 exchange 27,28*). Associations between the workorders and the received KPIs and SLAs on the side of WODM have been reflected in data model and database configuration. The structure of the JSON file is presented in Figure 13.

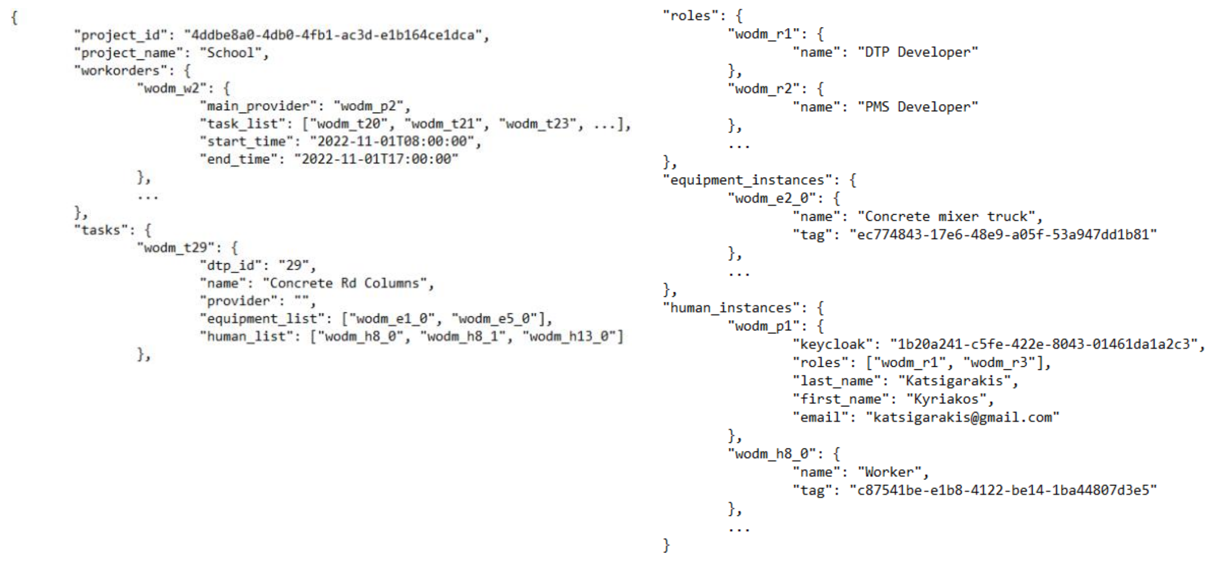


```
{
    "slaID":"5dfe0io-5tb0-4fb1-ac3d-e1b164ce1dcа
    "timestamp":"2022-11-05T09:30:00",
    "parties":
        [{
                    "partyId": "7",
                    "partyRole": "Finisher"
            },
            {
                    "partyId": "8",
                    "partyRole": "Worker"
            },
            ...
            ],
    "sla_startDate": "2022-11-01T08:00:00",
        "sla_endDate": "2022-11-01T17:00:00",
    "tasks": [
            {
                    "taskId": "12",
                    "equipmentID": ["1", "5'
                    "partyId": ["8", "13"]
            },
            {
                    "taskId": "16",
                    "equipmentID": ["1", "5'
                    "partyId": ["8", "13"]
            },
            ...
    ],
```

```
"serviceLevelObjectives": [
    {"KPI":"Percentage of completed work",
    "target_value":"100",
    "rule": "equal to",
    "KPI_value":"0",
    "taskId": ["9vqc","12gqc","26gqc","36vqc","11","12","13","15","16","17",
    "weighted factor":1}],
"equipment":[
    {
            {
                    "equipmentID": "5",
                    "name": "Truck mounted crane"
            },
            {
                    "equipmentID": "6",
                    "name": "Concrete vibrator machine"
            },
            ...
            ]
  }
```

Figure 13 – WODM to SLAM JSON

From WODM to DTP – planning phase (UC1.1 exchange 35): In this data exchange, the WODM sends to the DTP all workorders and associated data, including: the instructions, all allocated resources, linked tag ids and types, all worker responsibilities, and further details. These would usually be either automatically linked to the workorder, or manually set by the site manager (through WODM UI). The structure of the JSON file is presented in Figure 14.



```
{
        "project_id": "4ddbe8a0-4db0-4fb1-ac3d-e1b164ce1dca",
        "project_name": "School",
        "workorders": {
                "wodm_w2": {
                        "main_provider": "wodm_p2",
                        "task_list": ["wodm_t20", "wodm_t21", "wodm_t23", ...],
                        "start_time": "2022-11-01T08:00:00",
                        "end_time": "2022-11-01T17:00:00"
                },
                ...
        },
        "tasks": {
                "wodm_t29": {
                        "dtp_id": "29",
                        "name": "Concrete Rd Columns",
                        "provider": "",
                        "equipment_list": ["wodm_e1_0", "wodm_e5_0"],
                        "human_list": ["wodm_h8_0", "wodm_h8_1", "wodm_h13_0"]
                },
```

```
"roles": {
        "wodm_r1": {
                "name": "DTP Developer"
        },
        "wodm_r2": {
                "name": "PMS Developer"
        },
        ...
},
"equipment_instances": {
        "wodm_e2_0": {
                "name": "Concrete mixer truck",
                "tag": "ec774843-17e6-48e9-a05f-53a947dd1b81"
        },
        ...
},
"human_instances": {
        "wodm_p1": {
                "keycloak": "1b20a241-c5fe-422e-8043-01461da1a2c3",
                "roles": ["wodm_r1", "wodm_r3"],
                "last_name": "Katsigarakis",
                "first_name": "Kyriakos",
                "email": "katsigarakis@gmail.com"
        },
        "wodm_h8_0": {
                "name": "Worker",
                "tag": "c87541be-e1b8-4122-be14-1ba44807d3e5"
        },
        ...
}
```

Figure 14 – WODM to DTP JSON: planning phase

From BCSC to WODM (UC-1.2 exchange 27): This data exchange provides SLA performance for WODM. The BCSC to WODM data exchange has been reflected in WODM data model and new instances in WODM internal database have been created. Manual testing has been performed. In the phase of ICT integration automatic import will be validated. The structure of the JSON file is presented in Figure 15.

```
{
    "slaID":"SLA ID",
    "parties": [{
        "partyId": "default partyId",
        "partyRole": "default partyRole"
     }],
    "sla_startDate": "default start date",
        "sla_endDate": "default end date",
        "sla_description": "default description",
    "task": {
                "taskId": "default task ID",
                "taskDescription": "default task description"
    },
    "serviceLevelObjectives": [{
        "KPI":"selected kpi",
        "target_value":"selected target value",
        "rule": "more than / less than / equal to",
        "KPI_value":"current value",
        "taskId": "default task ID",
        "weighted factor":1
        }],
    "equipment":[{
        "equipmentID":"equipment ID",
        "name": "resource name"
          }],
      "slaPerformance": "default performance"
}
```

Figure 15 – BCSC to WODM JSON

<u>From WODM to WOEA (UC1.2 exchange 5,6):</u> The internal communication between WODM and WOEA aims to provide the workers with the workorders and their data. The WOEA receives all relevant workorders assigned to the specific worker who has logged into the application.

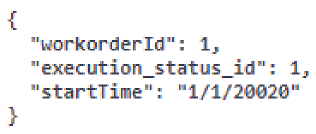
```
{
   "process_id":null,
   "i3d_id":"i3d-wo-proj_dev-1111",
   "execution_status":"Completed",
   "name":"Site values sensing",
   "locations":[

   ],
   "storey":"",
   "apartment":null,
   "plannedStartDateTime":"2022-01-21 09:52:00",
   "plannedFinishDateTime":"2022-01-22 09:52:00",
   "actualStartDateTime":"2022-01-21 09:54:35",
   "actualFinishDateTime":"2022-01-21 09:56:19",
   "related_project":null,
   "creationDate":"2022-11-16 15:58:32",
   "version":1,
   "space_id":null,
   "space_name":" ",
   "provider_id":"i3d-prov-proj_dev-30",
   "manager_login":"i3d-prov-proj_dev-29",
   "tasks":[
      {
         "i3d_result_id":"i3d-task-proj_dev-4198",
         "id":null,
         "name":"Take a photo of location Buiding 1",
         "result":"YES",
         "provider_id":"i3d-prov-proj_dev-30",
         "start":"2022-01-21 09:55:20",
         "duration":12,
         "planned_start":null,
         "planned_end":null,
         "real_start":"2022-01-21 10:55:31",
         "real_end":"2022-01-21 10:55:31"
      },
      {
         "i3d_result_id":"i3d-task-proj_dev-4201",
         "id":null,
         "name":"Take a photo of location Buiding 2",
         "result":"YES",
         "provider_id":"i3d-prov-proj_dev-30",
         "start":"2022-01-21 09:56:06",
         "duration":12,
         "planned_start":null,
         "planned_end":null,
         "real_start":"2022-01-21 10:56:17",
         "real_end":"2022-01-21 10:56:17"
      }
   ]
}
```

Figure 16 – WODM to WOEA JSON

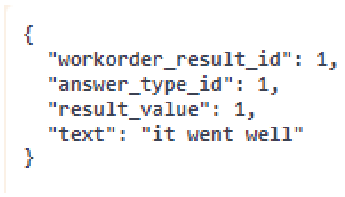
<u>From WOEA to WODM (UC1.2 exchange 8):</u> The WOEA sends two kinds of updates to the WODM. The first type of update (Figure 17) reflects the workorder's status getting changed by the workers (ready, started, completed,

aborted…). The second type of update (Figure 18) occurs at every task update, so any time the worker finishes the task with an optional message and value, the workorder gets updated with this information. The WODM then calculates all KPIs and handles the data automatically.



```
{
  "workorderId": 1,
  "execution_status_id": 1,
  "startTime": "1/1/20020"
}
```

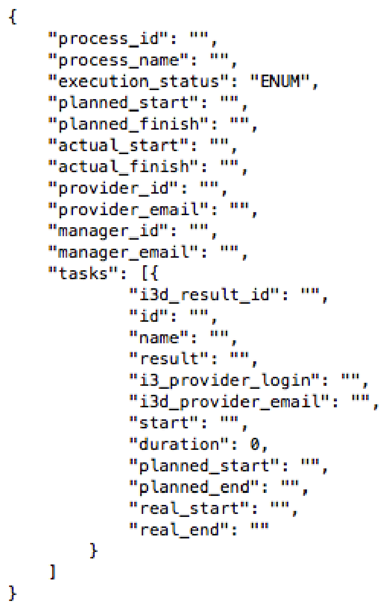
Figure 17 – WOEA to WODM JSON: Workorder update



```
{
  "workorder_result_id": 1,
  "answer_type_id": 1,
  "result_value": 1,
  "text": "it went well"
}
```

Figure 18 – WOEA to WODM JSON: Task update

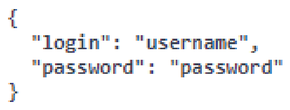
<u>From WODM to PMS UC1.2 exchange 9):</u> After the workorder's progress is reported through WOEA, it is automatically reflected in the WODM via an internal data exchange. After this, the WODM sends the progress to the PMS. The structure of the JSON file is presented in Figure 19



```
{
    "process_id": "",
    "process_name": "",
    "execution_status": "ENUM",
    "planned_start": "",
    "planned_finish": "",
    "actual_start": "",
    "actual_finish": "",
    "provider_id": "",
    "provider_email": "",
    "manager_id": "",
    "manager_email": "",
    "tasks": [{
            "i3d_result_id": "",
            "id": "",
            "name": "",
            "result": "",
            "i3_provider_login": "",
            "i3d_provider_email": "",
            "start": "",
            "duration": 0,
            "planned_start": "",
            "planned_end": "",
            "real_start": "",
            "real_end": ""
        }
    ]
}
```

Figure 19 – WODM to PMS JSON

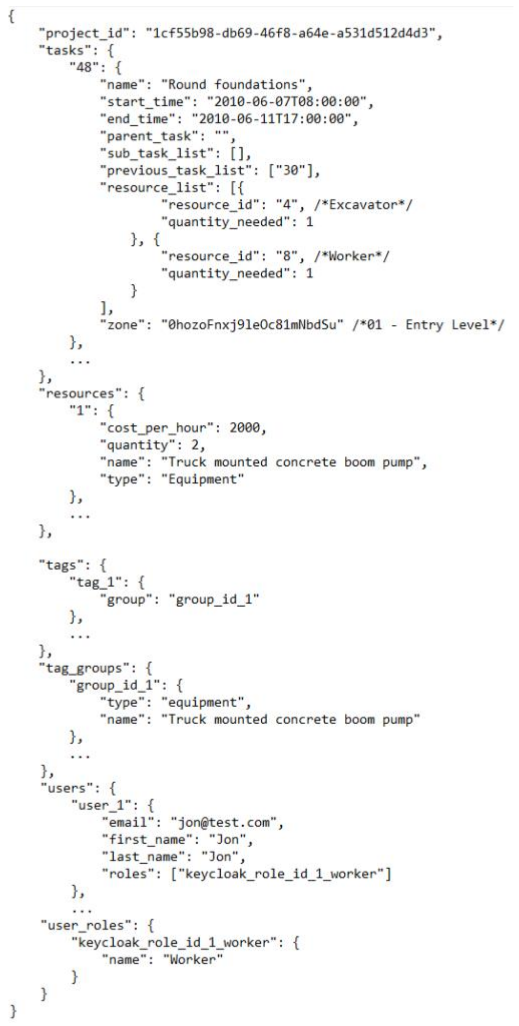
<u>From DTP to WODM (user authorisation) (UC1.2 exchange 16,17):</u> This data exchange provides the WODM with an option to log in the users of the WODM and WOEA (Figure 20). Once they are successfully verified and logged in, the WODM receives the user roles and groups linked with the specific users.



```
{
  "login": "username",
  "password": "password"
}
```

Figure 20 – WODM to DTP JSON

<u>*From DTP to WODM – construction phase (UC1.2 exchange 19,20)*</u>*:* The WODM request the process model including as planned resources, and IoT data static configuration (tag ids & types) from the DTP, which automatically sends the data back. The structure of the JSON file is presented in Figure 21.

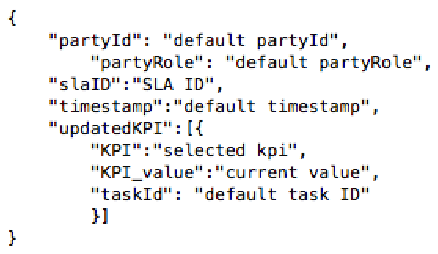


```
{
    "project_id": "1cf55b98-db69-46f8-a64e-a531d512d4d3",
    "tasks": {
        "48": {
            "name": "Round foundations",
            "start_time": "2010-06-07T08:00:00",
            "end_time": "2010-06-11T17:00:00",
            "parent_task": "",
            "sub_task_list": [],
            "previous_task_list": ["30"],
            "resource_list": [{
                    "resource_id": "4", /*Excavator*/
                    "quantity_needed": 1
                }, {
                    "resource_id": "8", /*Worker*/
                    "quantity_needed": 1
                }
            ],
            "zone": "0hozoFnxj9le0c81mNbdSu" /*01 - Entry Level*/
        },
        ...
    },
    "resources": {
        "1": {
            "cost_per_hour": 2000,
            "quantity": 2,
            "name": "Truck mounted concrete boom pump",
            "type": "Equipment"
        },
        ...
    },

    "tags": {
        "tag_1": {
            "group": "group_id_1"
        },
        ...
    },
    "tag_groups": {
        "group_id_1": {
            "type": "equipment",
            "name": "Truck mounted concrete boom pump"
        },
        ...
    },
    "users": {
        "user_1": {
            "email": "jon@test.com",
            "first_name": "Jon",
            "last_name": "Jon",
            "roles": ["keycloak_role_id_1_worker"]
        },
        ...
    "user_roles": {
        "keycloak_role_id_1_worker": {
            "name": "Worker"
        }
    }
}
```

Figure 21 – DTP to WODM JSON: construction phase

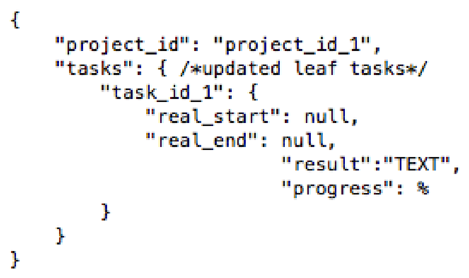
<u>*From WODM to BCSC (UC1.2 exchange 24):*</u> This data exchange allows the WODM to send the updated workorders along with their linked SLAs and KPIs directly to the BCSC. The exchange has been in WODM internal database and reflected in WODM data model. The structure of the JSON file is presented in Figure 22.



```
{
    "partyId": "default partyId",
        "partyRole": "default partyRole",
    "slaID":"SLA ID",
    "timestamp":"default timestamp",
    "updatedKPI":[{
        "KPI":"selected kpi",
        "KPI_value":"current value",
        "taskId": "default task ID"
        }]
}
```

Figure 22 – WODM to BCSC JSON

<u>*From WODM to DTP (task progress) – construction phase (UC1.2 exchange 26):*</u> This data exchange works in a comparable way to the one described in previous section. While the previous one was more focused on the initial data transfer during the planning phase, this one will run periodically any time there is a progress in any of the workorder's tasks. The structure of the JSON file is presented in Figure 23.



```
{
    "project_id": "project_id_1",
    "tasks": { /*updated leaf tasks*/
        "task_id_1": {
            "real_start": null,
            "real_end": null,
                        "result":"TEXT",
                        "progress": %
        }
    }
}
```

Figure 23 – WODM to DTP (task progress) JSON: construction phase

2.4 Licensing

WODM is a closed source component.

2.5 Installation Instructions

WODM is provided as a SaaS (Software as a Service), thus installation or downloading of any component is not required. To access WODM and test the already implemented functionalities, please navigate its main UI, accessible through the following link: https://I3D.econtent.lu/i3d2/I3D-frontend/I3D-en-cogito/.

There are two possibilities to test WODM (WODM UI):

Developer access:

Demo user credentials are provided below:

- **Name**: CogitoUser
- **Password**: rdh486o38qw9sf4jz

After login, choose "Construction prototype". The specific project uses as input data (School project) manually imported from the DTP.

Identity Provider access:

Should you have a valid COGITO project assigned to your account, use your credentials from DTP to sign in. Demo account is also available:

- Login: info@novitechgroup.sk
- Password: qB6USqBGJ8Wed4A

The application is tested on all relevant web browsers, Chrome, Edge and Firefox. WODM UI has not been widely tested on mobile devices. Android and Apple tablets with large screens are working well.

2.6 Development and integration status

The current version of WODM implements functionalities related to workflow management and workorder creation and monitoring described in UC1.1 and UC1.2. The integration with other components of the COGITO ecosystem has been tested, based on manual data importing and exporting tests. During ICT integration in WP8 the automated data exchange with other tools will be validated and tested with real project data. WODM interacts with the identity provider of DTP to ensure secure access to data by authorised users, based on their roles, and authenticated clients. ETL service to import workflows from DTP is finalized and tested with data from School project. Finally, all the necessary REST communications with other COGITO tools have been implemented and manually tested. In parallel, WODM UI has been updated to accommodate all the changes on the WODM backend.

2.7 Requirements Coverage

The WODM tool is designed as a back-end COGITO solution, but it covers several of the requirements defined in D2.1 and D2.4. Table 14 presents the Stakeholders Requirements documented in D2.1 which are relevant for WODM. COGI-CS-1, COGI-CS-4 and COGI-CS5 are covered simply by the fact that UI is web based, COGI-CS-6 has not been tested on all Android devices, but it is working well on larger tablets. All other requirements are maintained by the principle of programming language of the application.

Table 2 – Stakeholders' requirements – computing systems

ID	Solution	Priority	Status
COGI-CS-1	runs on desktop or laptop PC	Must	Achieved Via WODM UI
COGI-CS-4	runs on Windows	Must	Achieved Via WODM UI
COGI-CS-5	runs on Mac	Could	Achieved Via WODM UI
COGI-CS-6	runs on Android	Would	Achieved Via WODM UI
COGI-CS-7	allows access to the whole data in one location	Must	Achieved
COGI-CS-8	maintains communication and data security	Must	Achieved
COGI-CS-9	differentiates data and system access levels and modification rights	Must	Achieved

Stakeholders' requirements about Workflow planning, Execution and Monitoring are defined in D2.1 and presented in Table 15. COGI-WF-5 is partially achieved as material usage and cost have not been tested yet. COGI-WF-6 is partially achieved due missing cost escalation. COGI-WF-28 and 29 requirements will be achieved during implementation phase of the project as original functionalities of I3D allowing these functions.

Table 3 – Workflow planning, execution and monitoring requirements

ID	Solution	Priority	Status
COGI-WF-1	allows the PM and Client to share information (design data, photos, videos, schedules, design issues, cost)	Could	Achieved
COGI-WF-2	allows the PM and SM to share information (design data, photos, RAMs, design issues, schedules, work orders, work reports, materials, schedule and usage, equipment usage, costs)	Must	Achieved
COGI-WF-3	allows the PM and the QS to share information (design data, photos, RAMS, design issues, schedules, work orders, materials schedule and usage, equipment usage, costs)	Should	Achieved
COGI-WF-4	allows the PM and QM to share information (design data, photos, design issues, schedules, work orders, materials schedule and usage, costs)	Should	Achieved
COGI-WF-5	allows the SM to share information with Subcontractors, Foreman, and Workers (design data, photos, RAMs, design issues, schedules, work orders, equipment usage)	Should	Partially achieved
COGI-WF-6	allows the PM and SM to efficiently detect and prioritise delays and cost escalation elements	Should	Partially achieved
COGI-WF-7	allows the PM to extract reports about project time performance, project cost performance, costs per unit, resource consumption	Must	Achieved
COGI-WF-9	issues work orders that include detailed method statements	Should	Achieved
COGI-WF-10	allows work orders assignment to specific workers/crews.	Should	Achieved
COGI-WF-11	updates the activity status during work execution and monitoring	Should	Achieved
COGI-WF-12	allows display of activity description, planned duration and activities relationship	Should	Achieved
COGI-WF-13	defines work orders containing work description, location, start and end, construction drawings or BIM, safety measures, materials and equipment needed, quality measures, etc.	Must	Achieved
COGI-WF-14	allows work progress reports	Must	Achieved
COGI-WF-15	updates work progress weekly	Must	Achieved
COGI-WF-16	updates the project schedule at least monthly	Must	Achieved

COGI-WF-18	displays only current information or document versions, related to the project, to all stakeholders	Must	Achieved
COGI-WF-19	enables quick and easy reporting.	Must	Achieved
COGI-WF-21	facilitates resource allocation during scheduling	Should	Achieved
COGI-WF-23	incorporates health and safety planning	Should	Achieved
COGI-WF-24	incorporates quality control planning	Should	Achieved
COGI-WF-25	accesses the project execution monitoring off site	Must	Achieved
COGI-WF-26	allows efficient reporting of work completion (using sensor data or simple app interface)	Could	Achieved via WOEA
COGI-WF-27	offers simple, easy to use, and intuitive interface to avoid workforce over-burdening	Must	Achieved via WODM UI
COGI-WF-28	facilitates pre-construction training sessions (e.g., by using BIM models in augmented reality)	Would	Not implemented yet
COGI-WF-29	facilitates swift tool adoption by easily available video tutorials and other learning materials online	Would	Not implemented yet

The functional and non-functional requirements, which are relevant to the WODM component, were documented in D2.5 and are presented in Table 16. Req-2.1 is achieved by WODM UI as WODM is only backend application.

Table 4 – Functional and Non-Functional Requirements coverage

ID	Description	Type	Status
Req-1.1	Ability to create work orders from workflows	Functional	Achieved
Req-1.2	Assigning multiple workers to work orders	Functional	Achieved
Req-1.3	Monitoring the work order and seeing the report	Functional	Achieved
Req-1.4	Editing the work order details during the runtime	Functional	Achieved by WODM UI
Req-2.1	User-friendly UI	Non-Functional	Achieved by WODM UI
Req-2.2	Scalability	Non-Functional	Achieved
Req-2.3	Stability	Non-Functional	Achieved

3 Conclusions

This demonstrator deliverable presents the current status of development of the WODM component which plays a key role in the workflow and workorder management. The component is an extended and repurposed version of an existing commercial platform, I3D, which provides functionalities tailored to COGITO requirements that have been elicited from the use-cases UC1.1 and UC1.2 sequence diagrams. In its final version, WODM's data exchange with other COGITO tools is covered and all necessary changes in data model have been implemented. In the phase of ICT integration, all interaction with COGITO tools will be validated and tested with real project data. WODM database has been rewritten and prepared for necessary changes which could appears during implementation phase of COGITO project.

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