Knowledge-based services for and optimization of machines

Deliverable D1.5a

Description of demonstrator implemented in crane domain

Deliverable type:	Document
Deliverable reference number:	ITEA 18030 | D1.5a
Related Work Package:	WP1: Use Cases, Requirements & Evaluation
Due date:	M38 (30 November 2022)
Actual submission date:	7.12.2022
Author(s)	Pauli SALMINEN / Aalto University
Confidentiality	Public
Version	v 1.1

Contributors:

Name	Company
Chao YANG	Aalto University
Riku ALA-LAURINAHO	Aalto University
Juuso AUTIOSALO	Aalto University
Markus RANTA	IDEAL GRP
Fernando GARCIA	IDEAL GRP
Thomas WIDMAIER	RollResearch International
Miika OKKO	Remion
Valtteri PELTORANTA	Konecranes
Kimmo RANTALA	Konecranes

Revision History

Version	**Date**	**Description**
v 0.0	1/10/2022	Initial draft
v 1.0	29/11/2022	Final edit
v 1.1	5/11/2022	Final edit continues

Table of Contents

1. Abstract

This document describes the Finnish use case demonstration activities for a Machinaide project. Finnish demonstration consists of five partners, Konecranes, Aalto University, Remion, RollResearch International and Ideal. Use case for Finnish consortium is an industrial smart crane, Ilmatar (Autiosalo, 2018), together with an intelligent grinding machine in a Smart Factory Ecosystem. The Ecosystem, including different machines, represents central platform and research environment when developing new technologies for the machines of the Ecosystem and ways to integrate them seamlessly to operate in smart and connected factories of the future.

Machinaide project started actively it the beginning of 2020. However, there have been two events that have had an effect on the demonstration work and forced us to adapt to the situation. Covid-19 had major influence when combined with renovation work at the Aalto laboratory. These two together caused the loss of accessibility to the Aalto Industrial Internet Campus (AIIC) for c. two years. Now at the end of 2022 we have gained access back to the lab. During these two years we changed original plans more towards to the digital platform instead of physical platform. In practice, we have developed Virtual Model around Ilmatar crane, using VR as a Human Machine Interface (HMI). In the model, there are Ilmatar Crane from Konecranes and Virtual Grinding Machine from RollResearch International. Together with this, Digital Twin Web server, Twinbase, is used to host Digital Twin Documents (DTD). This has been used as a platform that hosts DTD of our international partners as well. Ideal has been working with Smart Factory app that is connected with MES and ERP, Remion has done data processing and aggregation. One of Smart factory app's features is a Worklist view for the crane operator. Worklist lists MES/ERP tasks for the operator to be carried out. Each task contains the needed information and instructions for safe, efficient and timely material moves. Responsibilities of various partners are listed in Table 1.

As a result of demonstrations, Finnish consortium has been able to enhance the communication and connectivity with machines and systems. For example, Twinbase now hosts DT-documents that have been used for various purposes, for example to control machines in a smart factory (Mattila, Ala-Laurinaho, Autiosalo, Salminen, & Tammi, 2022). Twinbase has also a link to a Twinaide, that is an open-source platform fo managing and creating interoperable digital twins. It can be used also to access, analyze and search DT data. On top of these, virtual models have been build and these have been used for HMI tests and demonstrations. Software and concepts have been tested and developed, including program updates over system lifetime. At the time of writing, the many connections to the physical machines are under construction. Demonstrations with industrial machines have been delayed for two reasons: Laboratory renovation and pandemia that caused delays for face-to-face work and that caused worldwide delay for access to the critical hardware components. However, situation has now eased up and consortium is building new final demonstrations for the spring season.

Table 1 Finnish use case partners and responsibilities

Host for demonstrator activities:	Aalto Industrial Internet Campus (AIIC)
Partner contribution:	• ALL PARTNERS: Use case description, Requirements description, Platform creation, • AALTO: Virtual model and VR HMI, Twinbase Digital Twin Web platform • IDEAL: Digital Twin model for the production process (Tecnomatix), Digital Twin model for the factory automation, Smart Factory app • KONECRANES: Technology provider, Worklist app, HMI • REMION: Data modeling and processing pipeline • ROLLRESEARCH INTERNATIONAL: Virtual grinding machine
Use cases:	• System interfaces • Collect and Process Data • Digital ecosystem (Twinbase, Virtual model, Smart Factory app)
What is going to be validated with this demonstrator?	• WP1 – Use Cases, requirements and evaluation • WP2 – Interoperability of Digital Twin eco-systems • WP3 – Processing of multiple Digital Twin's data • WP4 – Creating innovative HMIs for Digital Twin based services • WP5 – Information usage across the machine lifecycle
Start of project:	• 10/2019

2. Demonstrator in crane domain

Machinaide demonstration in crane domain is part of the cyber-physical system. At the highest level we have a Digital Twin Web (DTW) which is built by a web of Digital Twin Documents (DTD). These documents host metadata and can be used for example for the factory control. Second level is a software layer with virtual- and simulation models, data processing and software update management over whole system lifecycle. These are then linked closely with third layer, physical machines, such as Ilmatar crane and (virtual) grinding machine.

2.1. Twinbase & factory control using DT documents

Digital twins are expected to form a network, a "Digital Twin Web," in the future. Digital Twin Web follows a similar structure to the World Wide Web and consists of meta-level digital twins that are described as digital twin description documents and distributed via Digital Twin Web servers. Standards must be established before the Digital Twin Web can be used efficiently and having an easily accessible server implementation can foster the development of those standards. Twinbase is an open-source, Git-based Digital Twin Web server developed with user-friendliness in mind. Twinbase stores digital twin documents in a Git repository, modifies them with Git workflows, and distributes them to users via a static web server, from which the documents can be accessed via a client library or a regular web browser. A demo server is available at https://dtw.twinbase.org and new server instances can be initialized free-of-charge at GitHub via its browser interface. Twinbase is built with GitHub repository, Pages, and Actions but can be extended to support other providers or self-hosting.

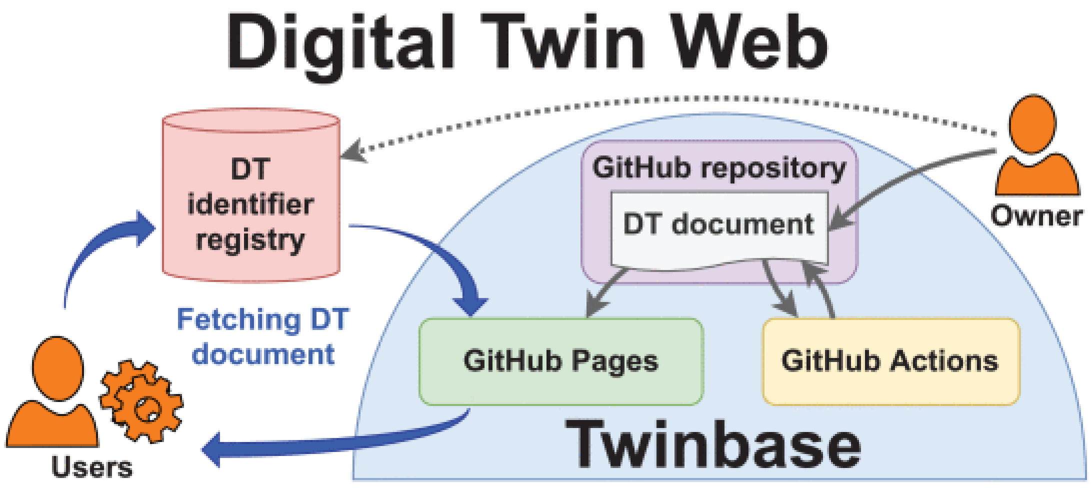


Figure 1 The overall workflow for distributing digital twin documents from owners to users with Twinbase (Autiosalo, Siegel, & Tammi, Twinbase: Open-source server software for the Digital Twin Web, 2021)(Autiosalo, Siegel, & Tammi, 2021)￼ https://ieeexplore.ieee.org/document/9568895

To evaluate the basic functionality of Digital Twin Web and Twinbase, we conducted two types of performance measurements. First, we compared the DT document fetch times of eight DTID registries (Figure 2), and second, we measured the fetch times of a series of six networked twin documents for three registries (Figure 3). Hosting servers were not compared because Twinbase currently supports only GitHub Pages, and initial tests showed consistent fetch times at approximately 0.1 seconds also for other hosting providers (Domainhotelli and users.aalto.fi were tested). The python scripts used for the measurements are available from GitHub at https://github.com/juusoautiosalo/dtweb-measurements.

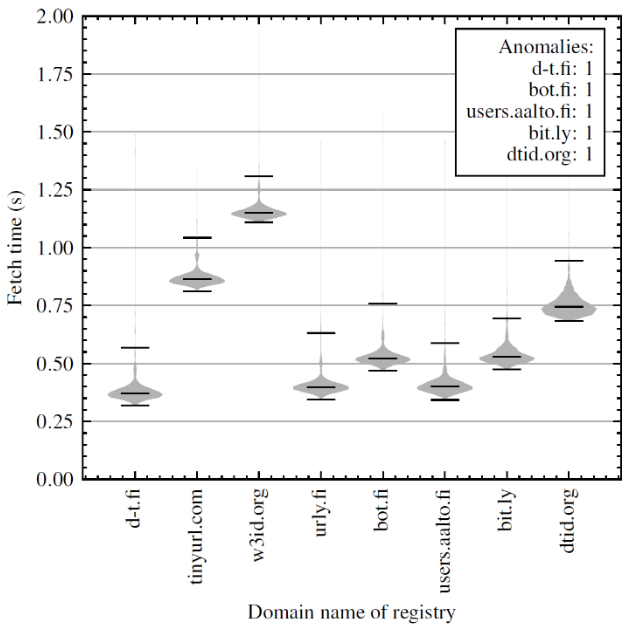


Figure 2 Total times for fetching DT documents with DTIDs at different DTID registries. The times include both fetching the hosting URL and the DT document. The gray areas represent the distribution of samples with the violin plot method, and the black horizontal lines from lower to higher are quantiles 0.00 (minimum), 0.50 (median), and 0.99. The sample size was 1000 for each registry, constituting a total of 8 000 fetches for the measurement. Samples over 2 seconds were excluded from the plot and reported as anomalies. All documents were hosted at the same Twinbase hosted by GitHub Pages, giving consistent fetch times of about 0.1 seconds.

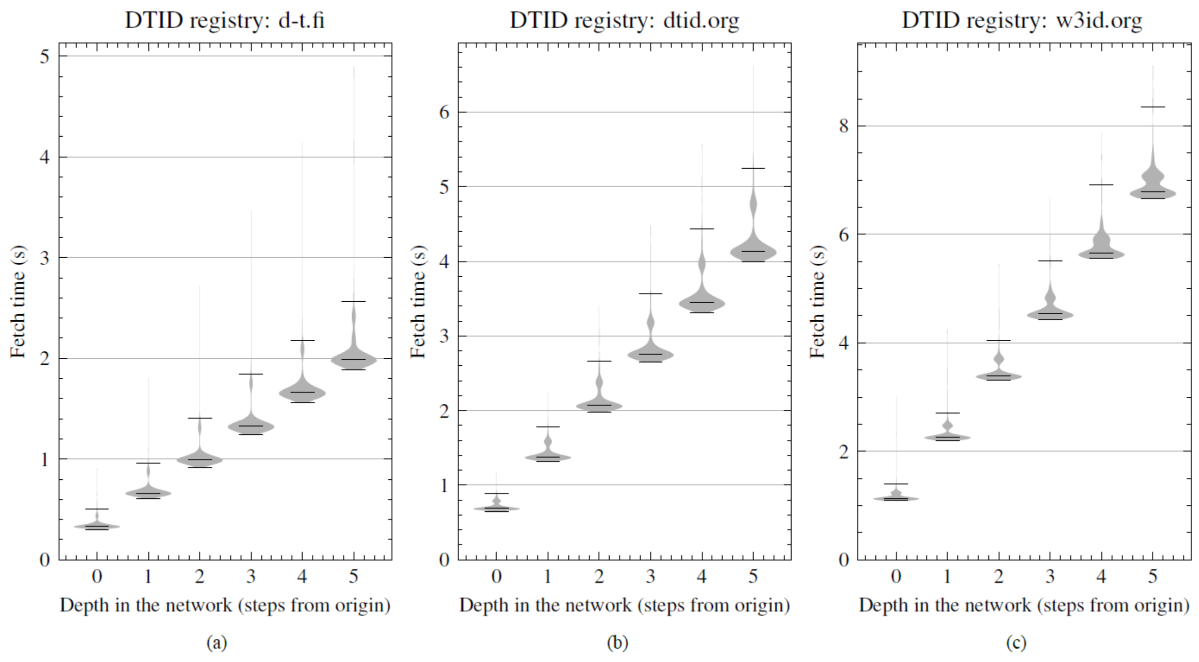


Figure 3 Total times for fetching DT documents referenced in other DT documents. The gray areas represent the distribution of samples with the violin plot method, and the black horizontal lines from lower to higher are quantiles 0.00 (minimum), 0.50 (median), and 0.99. The measurement started by fetching the DT document of the origin DT (depth 0) and continued to its child (depth 1) and so on until the last child (depth 5) had been fetched. The sample size is 1000 for each DT in each registry, constituting a total of 18 000 fetches for the three measurements. All documents were hosted at the same Twinbase hosted by GitHub.

To test the utility of the Digital Twin Web paradigm and Twinbase software in a practical use case, we used digital twin documents as a middleware for machine-to-machine communication in a smart factory (https://doi.org/10.3390/machines10040225). We implemented a proof-of-concept simulation model of a smart factory that allowed simulating three different control methods: centralized client-server, decentralized client-server, and decentralized peer-to-peer. Digital twin documents were used to store the necessary information for these control methods, so that local information could be minimized. We used Twinbase to host the digital twin documents (Figure 4). Our analysis showed that decentralized peer-to-peer control was most suitable for a smart factory because it allowed implementing the most advanced cooperation between machines while still being scalable. The utilization of Twinbase allowed straightforward removal, addition, and modification of entities in the factory.

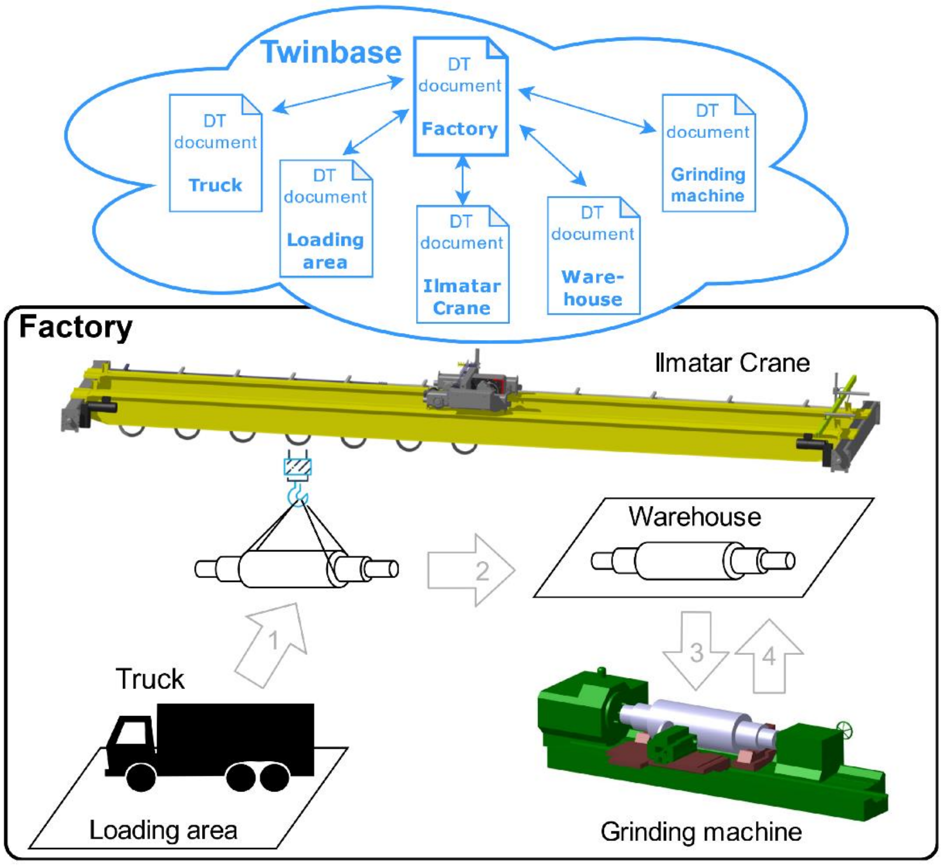


Figure 4 Overview of the factory operation and its DT documents. The gray arrows 1–4 show the operating logic of the factory, i.e., unloading rolls, storing them in a warehouse, and maintaining them on a grinding machine. Each object and the factory itself have corresponding DT documents that are stored in Twinbase. The DT documents have parent–child relationships to enable the discovery of other machines located in the factory.

Figure 5 demonstrates the usages of Twinbase in practice. The first client requests the DT document for the loading area from Twinbase. After that, the client gets the loading area's location from DT documents and sends this as a target value for the truck's drive function. After the truck arrives at the loading area client requests from Twinbase the DT document for the parent of the loading area using the parent's DTID in the loading area's DT document. Using this parent's DT document, which is the factory, the client gets all of its childrens' DTIDs and requests DT documents for them. From these DT documents, the client looks for a transport type of machine, and when it finds one, it commands it to drive to the loading area and pick up the roll from the truck. In this description, it was assumed that the client knew the information from the DT document of the truck beforehand.

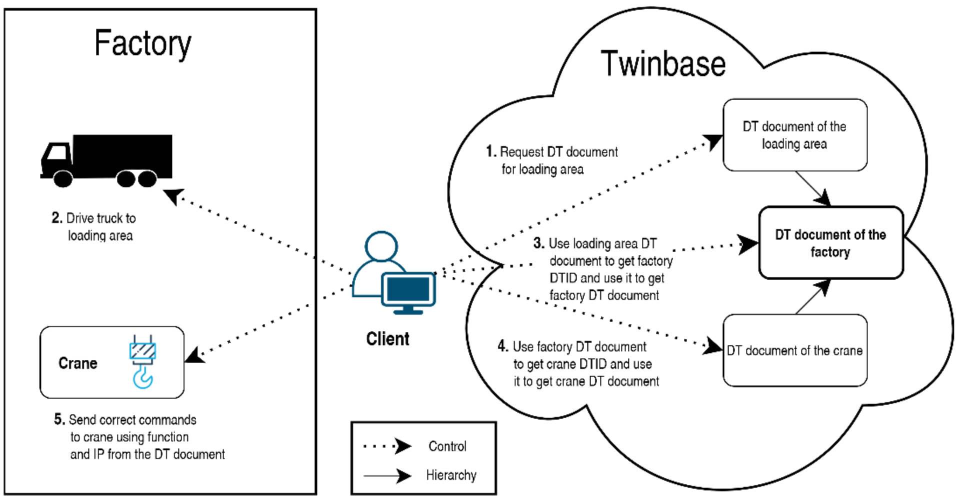

Factory
2. Drive truck to loading area
Crane
5. Send correct commands to crane using function and IP from the DT document
Client
Twinbase
1. Request DT document for loading area
DT document of the loading area
3. Use loading area DT document to get factory DTID and use it to get factory DT document
DT document of the factory
4. Use factory DT document to get crane DTID and use it to get crane DT document
DT document of the crane
Control
Hierarchy

Figure 5 Practical example of applying Twinbase to find entities in the same operating area (Mattila, Ala-Laurinaho, Autiosalo, Salminen, & Tammi, 2022).

The experiments show that creating a network of DTs—called Digital Twin Web or DTW—is beneficial for the creation of self-adapting smart factories. DTW allows distribution and discoverability of DT documents analogously to the early World Wide Web that allowed accessing (hyper-)text documents. Storing DT information in an easily accessible form is a necessity for the wide adoption of digital twins in smart factories and enables the control of machines.

After developing and testing Twinbase, it has been used to host DT documents from Machinaide project partners (Figure 6). Documents and metadata can be transferred between Twinbase and Twinaide (Deliverable D1.5b Description of demonstrator for production process optimization).

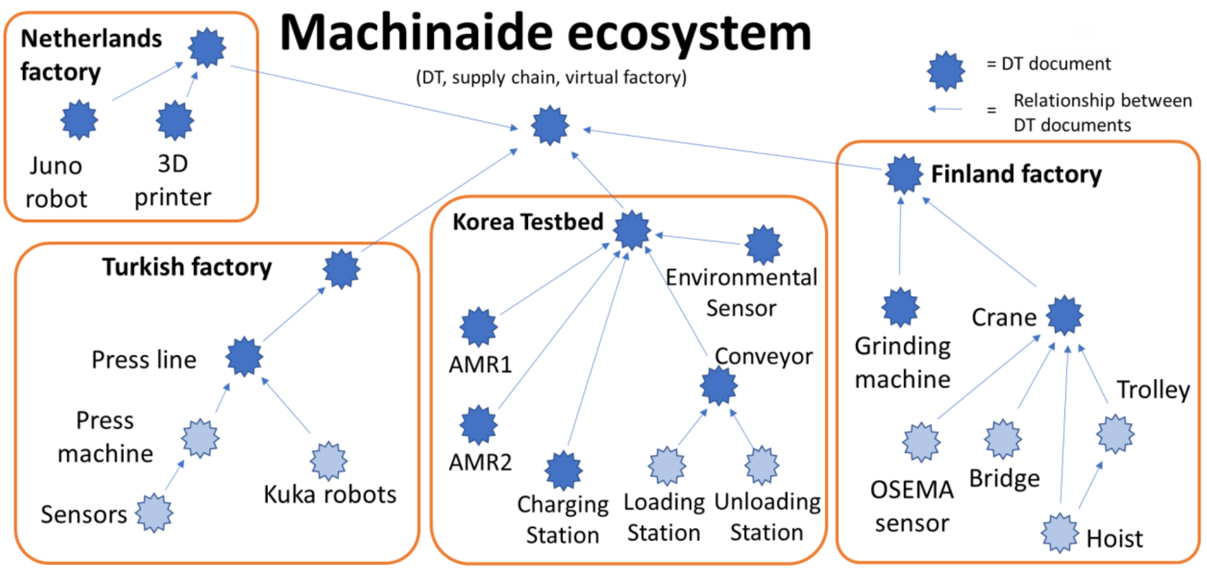


Figure 6 Machinaide eco-system described as a graph of DT documents that represent the nodes of Digital Twin Web as it was at the end of 2021.

2.2. Connecting physical and digital world

Modern digital tools allow building different kinds of use cases connecting physical and digital worlds. During the Machinaide project, IDEAL GRP has demonstrated this connectivity in three different demonstrators. Those are shown in the Figure 7 with balloons connecting to the most relevant data sources for each of the demonstrator:

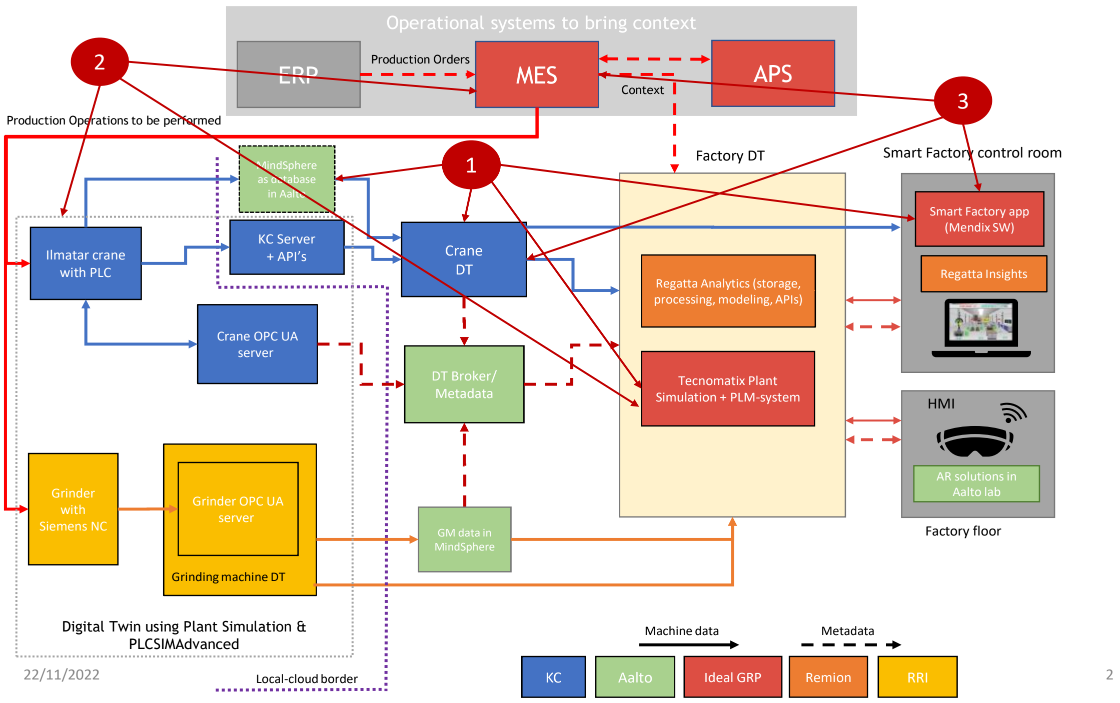

Operational systems to bring context
ERP
Production Orders
MES
Context
APS
2
3
1
Production Operations to be performed
Factory DT
Smart Factory control room
MindSphere as database in Aalto
Ilmatar crane with PLC
KC Server + API's
Crane DT
Smart Factory app (Mendix SW)
Regatta Insights
Regatta Analytics (storage, processing, modeling, APIs)
Crane OPC UA server
DT Broker/ Metadata
Tecnomatix Plant Simulation + PLM-system
HMI
AR solutions in Aalto lab
Grinder with Siemens NC
Grinder OPC UA server
Grinding machine DT
GM data in MindSphere
Factory floor
Digital Twin using Plant Simulation & PLCSIMAdvanced
22/11/2022
Local-cloud border
Machine data
Metadata
KC
Aalto
Ideal GRP
Remion
RRI
2

Figure 7 – IDEAL GRP demonstrators in the Crane domain – See next chapters for description on the balloons

Each of the above demonstrators are described in more detail below

2.2.1. A time machine for the digital twin in process simulation (1)

In the first demonstrator the main target was to build a simulation model for the crane in Tecnomatix Plant Simulation and connect it to real life measured data from the crane. This enables replaying the manufacturing process in the simulation model using actual production data to dig into any issues that have occurred at specific moment in time (Figure 8).

Simulation model was built using standard capabilities in Tecnomatix. It includes not only the crane but also the grinding machine, surroundings and the logic how these production machines work together. This would allow for simulating the material flow through the process, but that was not the topic for this demonstrator.

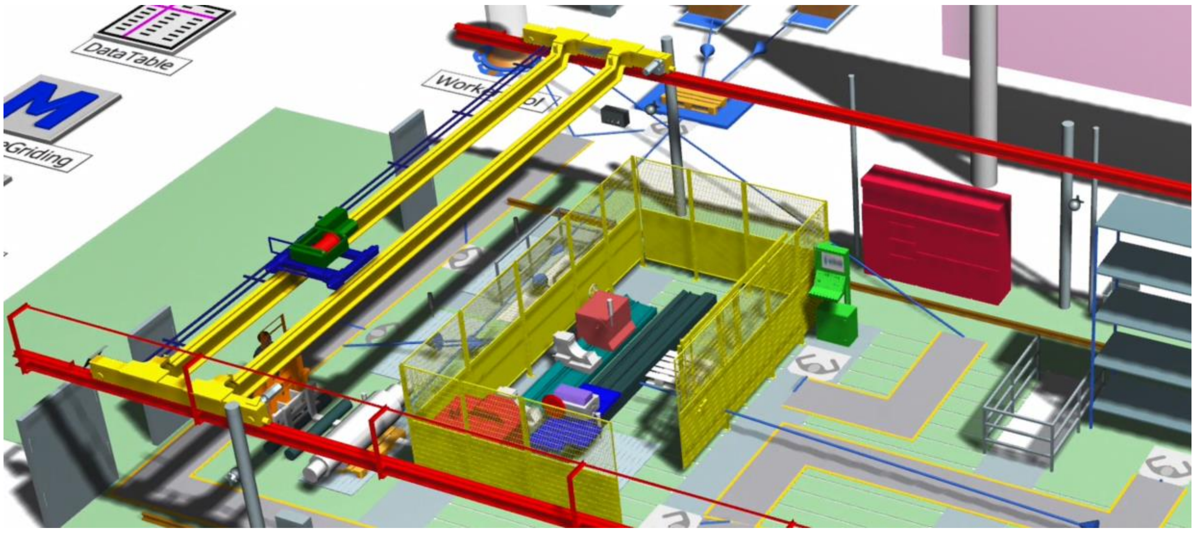
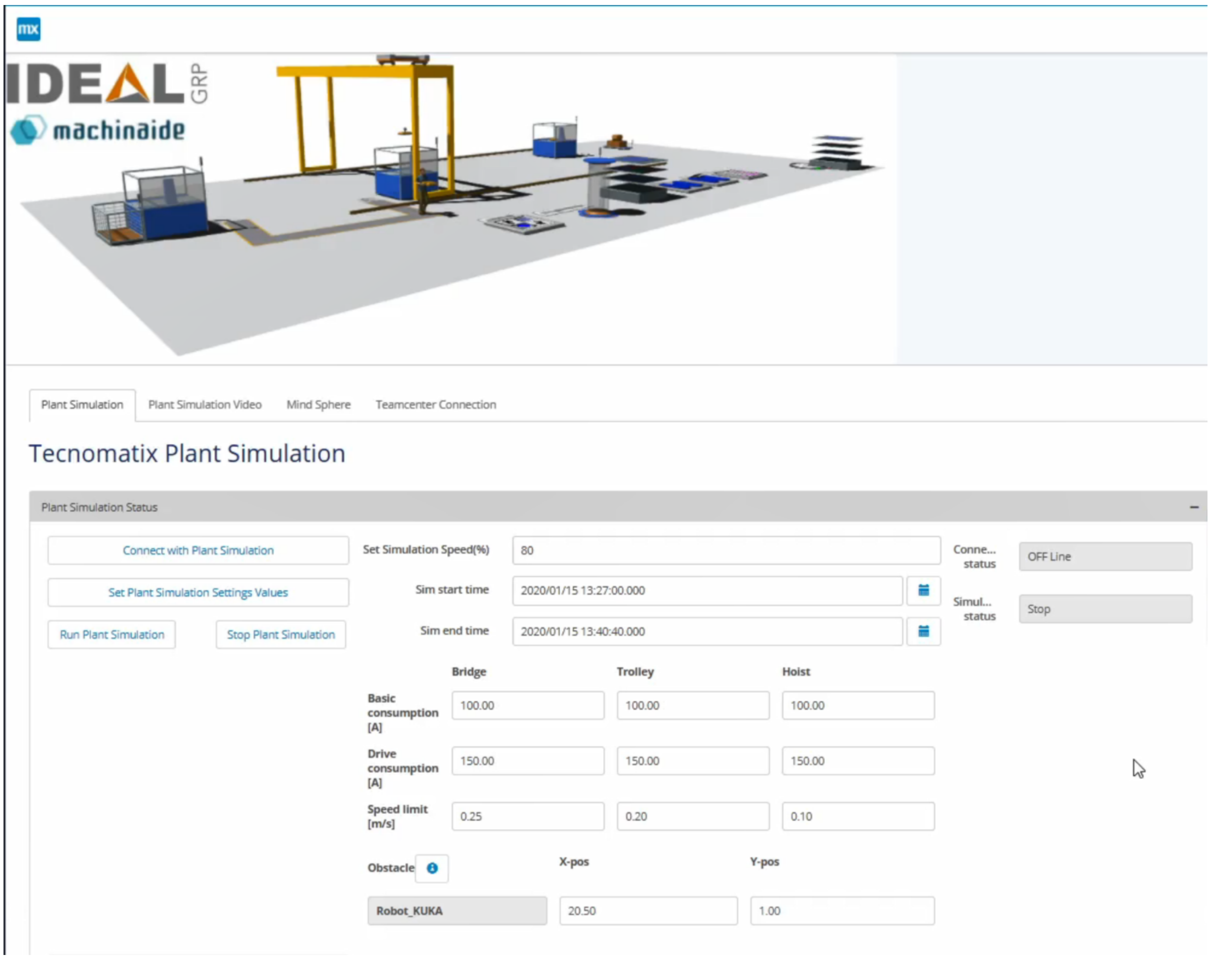


Figure 8 - Plant Simulation model for the Crane domain

A connection was made from the Plant Simulation model to time series data from the crane movements. Plant Simulation can replay the model against this time series data to support going back in history to see what exactly happened at a specific point of time. To facilitate the process, a low-code app made in Mendix was created to specify different parameters for the simulation model as well as to specify a location where a robot was located at a specific moment in time. This was used to run a collision test between the crane hook and the robot (Figure 9).



Tecnomatix Plant Simulation

Plant Simulation Status

Set Simulation Speed(%)	80	
Sim start time	2020/01/15 13:27:00.000	
Sim end time	2020/01/15 13:40:40.000	
Conne... status	OFF Line	
Simul... status	Stop	

	Bridge	Trolley	Hoist
Basic consumption [A]	100.00	100.00	100.00
Drive consumption [A]	150.00	150.00	150.00
Speed limit [m/s]	0.25	0.20	0.10

Obstacle	X-pos	Y-pos
Robot_KUKA	20.50	1.00

Figure 9 Mendix app for controlling the simulation model

Outcome from this demonstrator was that we were able to demonstrate how a time machine for a digital twin can be achieved. We were able to visually see how the process was working at a specific point in time and run a virtual collision test with a robot.

2.2.2. Virtual validation of a Smart Factory (2)

In the second demonstrator (balloon 2) we concentrated on building a digital twin from the automation, connect that to the MES system and use the same simulation model that was created already for the first demonstrator to visually see what is happening in our production process. Used connectivity can be seen below (Figure 10):

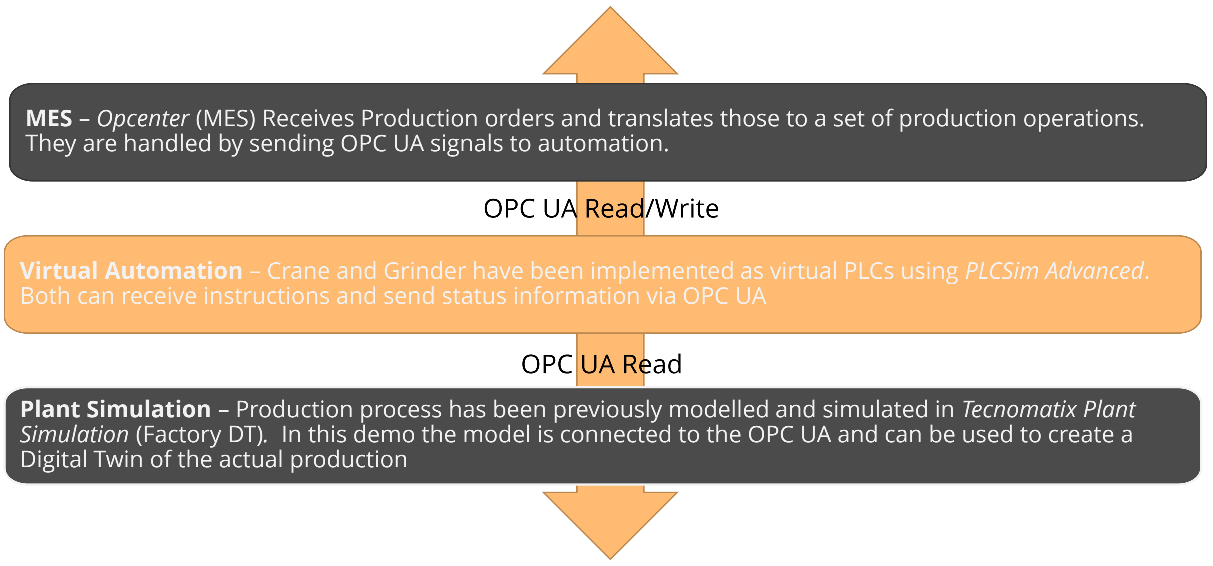

MES – *Opcenter* (MES) Receives Production orders and translates those to a set of production operations. They are handled by sending OPC UA signals to automation.
OPC UA Read/Write
Virtual Automation – Crane and Grinder have been implemented as virtual PLCs using *PLCSim Advanced*. Both can receive instructions and send status information via OPC UA
OPC UA Read
Plant Simulation – Production process has been previously modelled and simulated in *Tecnomatix Plant Simulation* (Factory DT). In this demo the model is connected to the OPC UA and can be used to create a Digital Twin of the actual production

Figure 10 Connectivity between OPC UA, MES, PLCSim Advanced and Plant Simulation.

The MES system, which in this case was Opcenter from Siemens, has different methods of connecting to outside world. In this case we used OPC UA to both send information to the virtual automation but also to read the status information. A process was modelled in the MES to define different steps in the production process which then are used to process the work orders accordingly. Each operation in the process contained also OPC UA writes and reads to drive the automation.

Siemens PLC Sim Advanced was used to virtually run the created automation for the crane and grinder. This allows for testing the actual automation code using a Software-in-a-loop approach. Communication to and from the virtual automation was done through OPC UA.

Simulation model in Plant Simulation was connected to the same virtual automation as the MES. That enables the visualization of the production process in 3D/2D according to the actual situation as published through the OPC UA.

As a result, in this demonstrator, it was possible to showcase a complete production process simulated within the digital tools. This enables the development of the MES, automation and the connectivity between those systems using digital twins (Figure 11, Figure 12).

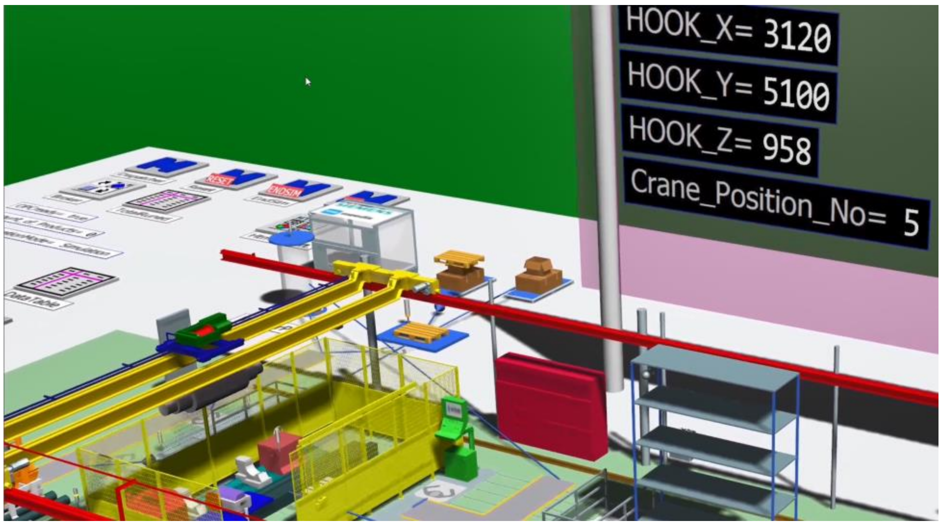


Figure 11 Simulation model controlled by virtual automation

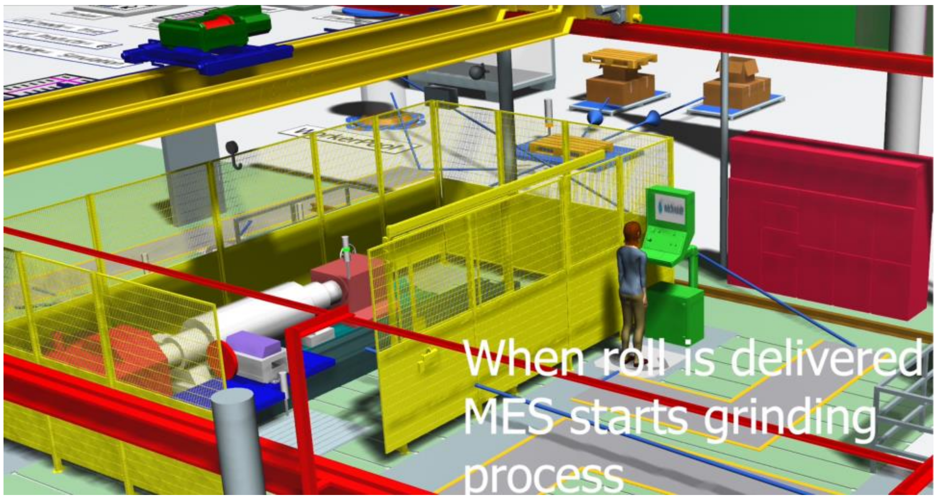


Figure 12 Roll is delivered by the crane. MES starts the grinding process.

2.2.3. Smart Factory worklist app (3)

Final demonstrator (balloon 3) is still being worked on during the writing of this document so only targets for that will be described here. The Smart Factory worklist app will be made in Mendix low-code platform and will connect to Opcenter MES as well as Crane DT. Main target for this demonstrator is to showcase how a low-code platform can be used to quickly develop different kinds of valuable apps for different purposes while getting the data from different sources. Opcenter MES will be used to provide the crane operator a list of next operations to perform. Each task contains the needed information for successful task execution; current and target location of the load, lifting accessories to be used and instructions for safe lifting. Additionally, the crane operator will be given a possibility to record details about the performed operation such as recording the serial number of the moved part. This serial number will be stored in the MES as part of the performed operation. Smart Factory app will also be connected to the crane DT to provide current status of the crane within the app.

2.3. Ilmatar Smart Crane platform

Ilmatar smart crane platform is physically located at the Aalto Industrial Internet Campus (AIIC) and it was a challenge as the whole building went under renovation between 2021-2022 and our access to the physical environment was limited and for example Ilmatar crane was out of power and connections for long periods of time. Thus, more focus was set to build up virtual environment that was used to develop various control methods using XR (Hololens, Varjo, Unity), DT Documents and external HMI´s. While Ilmatar is complete physical crane, the Virtual Grinding Machine from RollResearch International has physical HMI hardware for Roll Grinder, but the grinding process and machine itself are virtualized.

2.3.1. Remote control and monitor applications for Crane

A technological framework was developed to improve the efficiency of the XR application development and the usability of the XR-based HMI systems. Framework consists of four layers shown in Figure 13. The perception layer involved the DT-based "Ilmatar" crane and its corresponding simulation platform. The machine communication layer mainly introduced the communication bridge between the simulation platform ROS 2.0 and the OPC UA server. For the network layer, three communication middlewares were developed aiming for different application scenarios, from the lab environment to the actual industrial environment. Additionally, the service layer brought the procedures of hardware and software integration and XR environment setup.

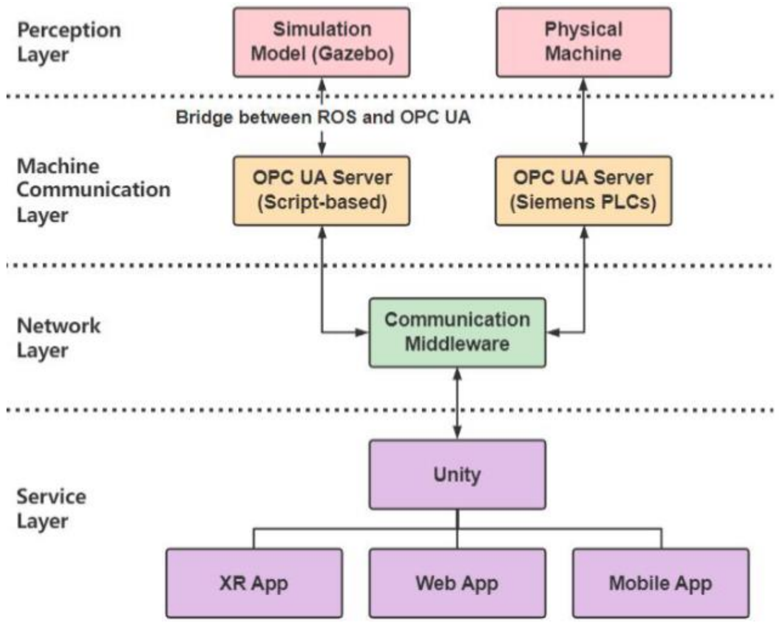

Perception Layer
Simulation Model (Gazebo)
Physical Machine
Bridge between ROS and OPC UA
Machine Communication Layer
OPC UA Server (Script-based)
OPC UA Server (Siemens PLCs)
Network Layer
Communication Middleware
Service Layer
Unity
XR App
Web App
Mobile App

Figure 13 Framework and workflow of the applications for a DT-based crane (Yang, et al., 2022).

Physical-field-based simulation model represents the complicated working condition of a machine, providing an overall picture of performance even in the face of unprecedented conditions, thus realizing the potential of DTs. DT always contains a simulation model that not only reflects how the machine works but also predicts how the machine will perform. Furthermore, the simulation model enables the reflection of the real-time states of the operating machine. Additionally, compared to numerical and analytical models, simulation models are most often applied in Industry 4.0. Hence, the

Perception layer and machine communication layer are the digital representation of the physical machine.

To establish bidirectional stable communication, three kinds of communication middlewares were developed in the network layer, shown in Figure 14. The OPC UA-GraphQL wrapper targeting the remote-control application, the OPC UA-MQTT wrapper focusing on the remote monitor application, and the general OPC-Unity client.

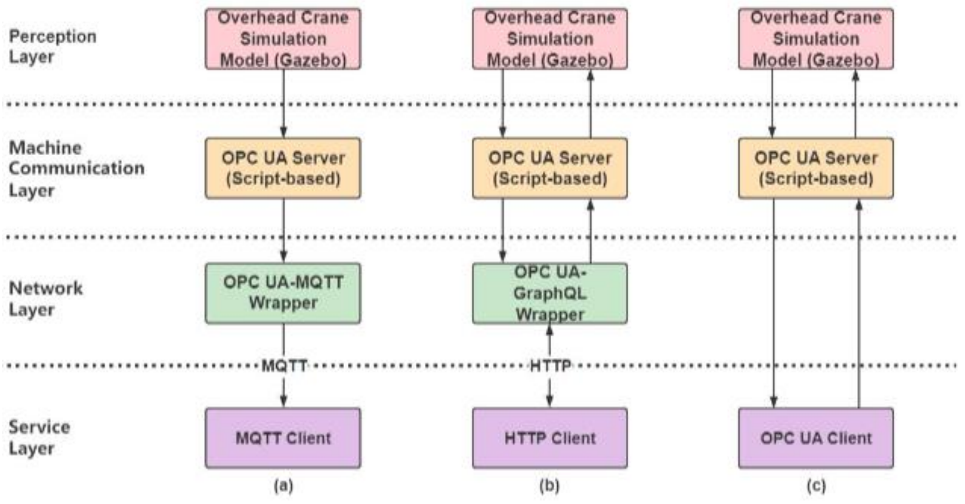

Perception Layer
Machine Communication Layer
Network Layer
Service Layer
Overhead Crane Simulation Model (Gazebo)
Overhead Crane Simulation Model (Gazebo)
Overhead Crane Simulation Model (Gazebo)
OPC UA Server (Script-based)
OPC UA Server (Script-based)
OPC UA Server (Script-based)
OPC UA-MQTT Wrapper
OPC UA-GraphQL Wrapper
MQTT
HTTP
MQTT Client
HTTP Client
OPC UA Client
(a)
(b)
(c)

Figure 14 The architectures of three kinds of communication middleware: (a) OPC UA-MQTT wrapper. (b) OPC UA-GraphQL wrapper. (c) OPC UA-Unity Client.

OPC UA-MQTT Wrapper

MQTT (Message Queuing Telemetry Transport) is a publish/subscribe messaging transport designed for remote devices, especially machine-to-machine (M2M) communication with poor network conditions and limited network bandwidth. Since there is no direct connection between publisher and subscriber under MQTT communication, it is necessary to set up a broker to decouple the message. OPC UA is a robust, secure, and scalable industrial standard communication protocol. MQTT is lightweight and therefore suitable for remote data transmission. Hence, an OPC UA-MQTT wrapper was developed in our research, which can transmit the data from the OPC UA server to the MQTT broker. The OPC UA-MQTT wrapper contains an OPC UA client, an OPC UA-MQTT gateway that transfers data read from the server to the MQTT message format, an MQTT client that sends the data, and an MQTT broker deployed in the open network. The network layer of the monitoring application was realized by the OPC UA-MQTT wrapper.

OPC UA-GraphQL Wrapper

OPC UA-GraphQL Wrapper is the middleware between the client and OPC UA server, which connects to an OPC UA interface to provide it with a GraphQL interface. OPC UA is one of the most important industrial communication protocols in the automation field. The OPC UA information model is a structured graph composed of nodes and references. Meanwhile, GraphQL is a query language that is advantageous for querying the data of the Graph structure. The combination of OPC UA and GraphQL can provide an understandable and friendly API for the developers. In this work, the OPC UA-GraphQL was used for the remote control use case.

Figure 15 depicts the structure of the service layer. The XR environment development contains three sub-modules, the model module, the function module, and the communication module. The model module determines the generally used models that keep consistency with the DT-based crane. The function module realizes the crane features and AR or VR functions related to the corresponding device. The communication module is responsible for interaction with the network layer on the Unity side. Figure 16 displays the "Ilmatar" crane model in Unity.

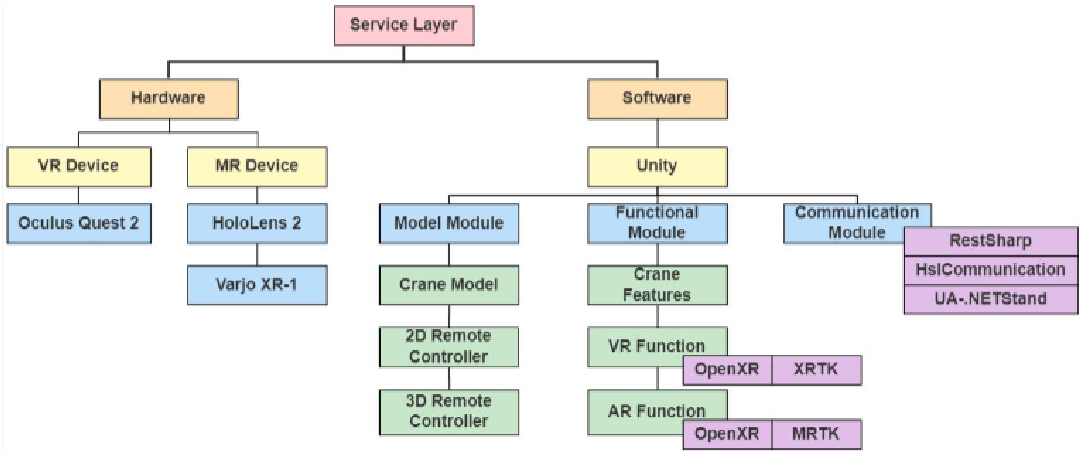


Figure 15 Overview of the service layer. It involved two subsections, namely hardware setup and software development.

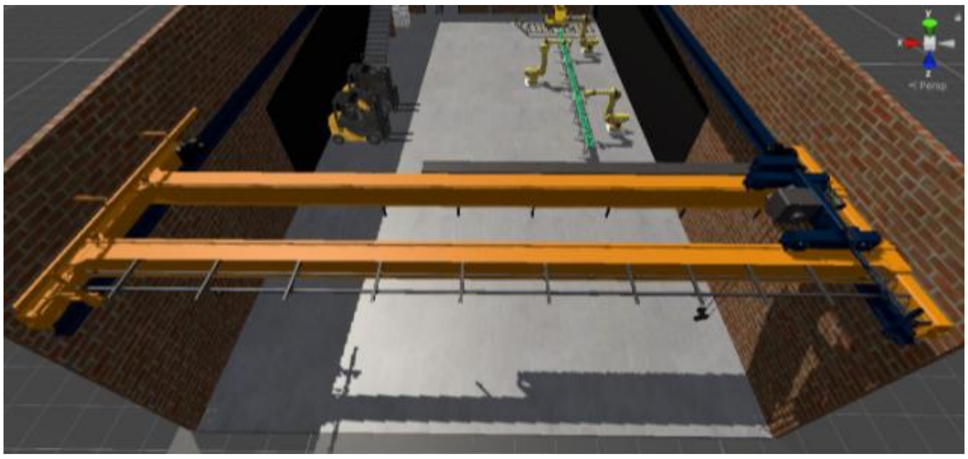


Figure 16 "Ilmatar" crane model in Unity.

To guarantee stability of communication, it is necessary to define the standard data format for the whole target machine. Table 2 illustrates the definition of data in the XR environment, including the design parameters of the crane, the status data of the running crane, and the data concerning the controller. The design data describe the constant design parameters. The data is stored as a component in Unity as the constant private value. Later, all the data will be accessed from the DT (Digital Twin) Document. The DT document describes the metadata and features of a single digital twin, which is under development. The document is designed to be used along with a Data Link that connects the features of the digital twin behind a single access point. Then, the status data shows the real-time location of the crane accessed from the network layer. Next, the control data represents the status of the controller, namely the smart feature buttons and the joysticks, used for the data flow between the network layer and the service layer. In the end, for the middleware of the OPC UA-GraphQL wrapper, the communication module established the HTTP connection through the RestSharp API to receive the corresponding data predefined in Table 2.

Table 2 Definition of data in the XR environment: The research defined the data class, name, basic format, reference type, source, and description to keep consistency with the internal and external interface. The Format defined the data basic type, such as int, float, char, etc. The reference type was categorized as constant and variable. The data source represented the source of the data. The Description depicted the definition of the data.

Class	Name	Format	Ref_Type	Data Source	Description
Design	Hoist_height_max	float	constant	Unity	Hoist maximum height
	Hoist_height_min	float	constant	Unity	Hoist minimum height
	Hoist_speed_max	float	constant	Unity	Hoist maximum speed
	Hoist_capacity	float	constant	Unity	Hoist maximum capacity
	Trolley_range_max	float	constant	Unity	Trolley maximum range
	Trolley_range_min	float	constant	Unity	Trolley minimum range
	Trolley_speed_max	float	constant	Unity	Trolley maximum speed
	Bridge_range_max	float	constant	Unity	Bridge maximum location
	Bridge_range_min	float	constant	Unity	Bridge minimum range
	Bridge_speed_max	float	constant	Unity	Bridge maximum speed
	Coefficient_microspeed	float	constant	Unity	Microspeed scale factor
	Coefficient_inching_speed	float	constant	Unity	Predefined speed for inching
	Coefficient_inching_distance	float	constant	Unity	Predefined distance for inching
Status	HoistPosition	float	variable	Middleware	The height of the hook block
	TrolleyPosition	float	variable	Middleware	The location of the trolley
	BridgePosition	float	variable	Middleware	The location of the bridge
Control	Inching	Boolean	variable	Unity/Middleware	Inching button status
	MicroSpeed	Boolean	variable	Unity/Middleware	Microspeed button status
	SwayControl	Boolean	variable	Unity/Middleware	Swaycontrol button status
	RopeAngleFeatureBypass	Boolean	variable	Unity/Middleware	Rope angle button status
	SwayControl_SlingLength_mm	int	variable	Unity/Middleware	The sling length
	Hoist.Up	Boolean	variable	Unity	Whether hoist moves up
	Hoist.Down	Boolean	variable	Unity	Whether hoist moves down
	Hoist.Speed	float	variable	Unity	The speed of the hoist
	Trolley.Forward	Boolean	variable	Unity	Whether trolley moves forward
	Trolley.Backward	Boolean	variable	Unity	Whether trolley moves backward
	Trolley.Speed	float	variable	Unity	The speed of trolley
	Bridge.Forward	Boolean	variable	Unity	Whether bridge moves forward
	Bridge.Backward	Boolean	variable	Unity	Whether bridge moves backward
	Bridge.Speed	float	variable	Unity	The speed of bridge

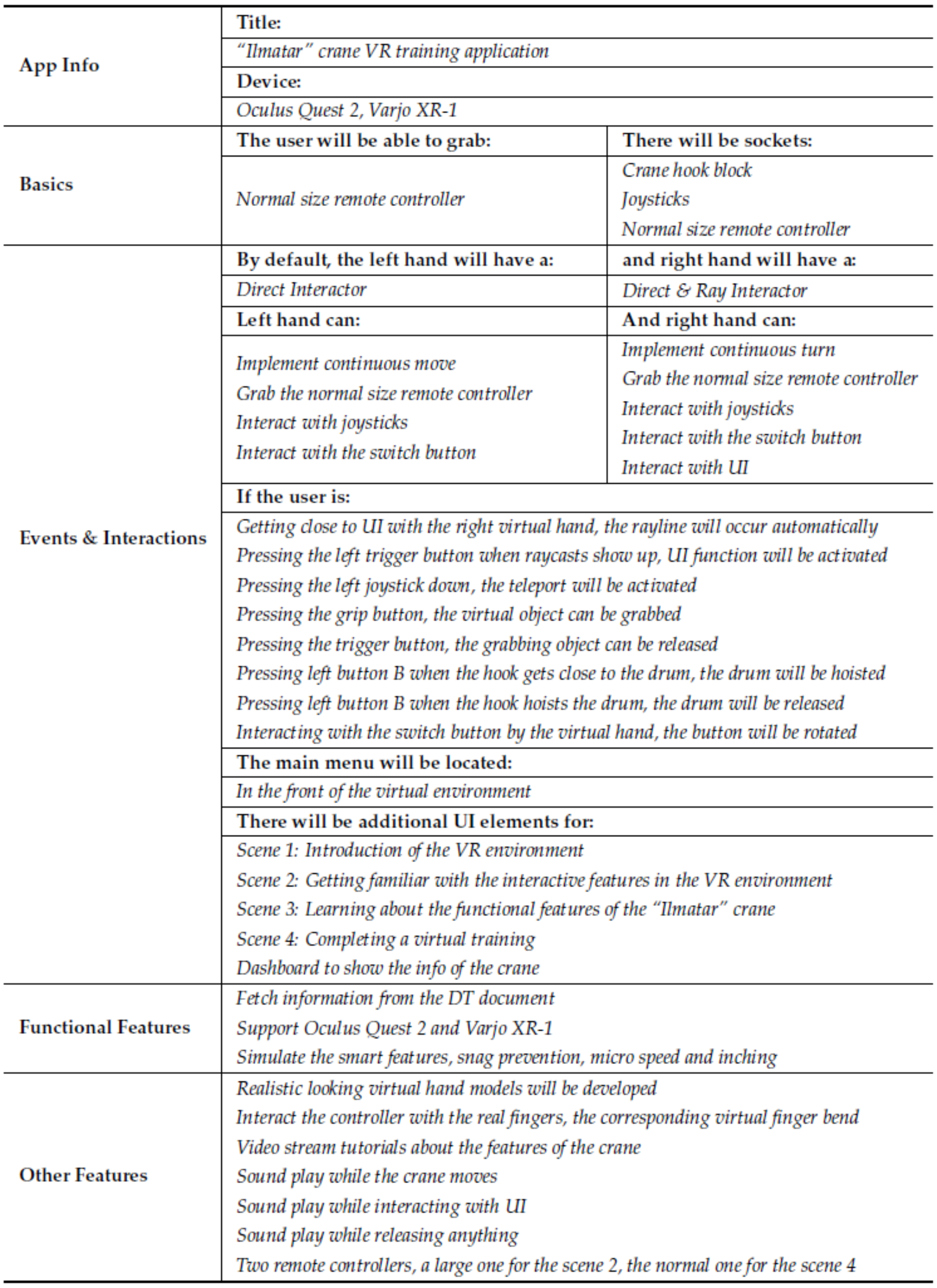
The use of VR technology as a method in industrial work training is a growing trend and active topic in research. VR enables the creation of a realistic look and feel world, providing an immersive and interactive yet safe experience for the user. To highlight the interactivity of the development framework, the first case is a VR training application of the ``Ilmatar'' crane, the development contents are based on the design document (Table 3). To enhance immersion and operability, the application was featured by:

- **Physical VR hand:** The two learning contents (the operation of the device controller, and the operation of the virtual remote controller of the crane) in the VR environment increased the operation complexity. The development of the physical VR hands eased the sense of unfamiliarity and improved the immersion.
- **Multi-Interactor:** Implemented the different Interactors to improve the user experience. Interactors are used for interacting with interactable. It contained three kinds of Interactors: Ray Interactor, used for interacting with interactable at a distance; Direct Interactor, used for directly interacting with interactables that are touching; and Socket Interactor, that would hold an interactable and raise an event when an interactable was placed into, or removed from, the Socket.
- **Multi-modal interaction capabilities:** video and sound streams were imported, video stream was used for the tutorial of the crane features instead of the text description, while audio functions were set up to give feedback on the operation.
- **Multi-scene:** Four scenes were created to give a progressive learning process, including the introduction of the VR environment, getting familiar with the interaction features in

the VR environment, learning about the functional features of the ``Ilmatar'' crane, and completing a virtual hands-on training (see Figure 17).

- **Design parameter synchronization:** The design parameters (Table 2) were stored in an independent script-based component in Unity, enabling the request of updated data from the external resource (DT document) through HTTP. In other words, the VR training application is a dynamic data-driven XR environment.

Table 3 VR training application} design document: The functional features and the interaction features were developed following the design document.



App Info	**Title:**	
	"Ilmatar" crane VR training application	
	Device:	
	Oculus Quest 2, Varjo XR-1	
Basics	**The user will be able to grab:**	**There will be sockets:**
	Normal size remote controller	*Crane hook block* *Joysticks* *Normal size remote controller*
Events & Interactions	**By default, the left hand will have a:**	**and right hand will have a:**
	Direct Interactor	*Direct & Ray Interactor*
	Left hand can:	**And right hand can:**
	Implement continuous move *Grab the normal size remote controller* *Interact with joysticks* *Interact with the switch button*	*Implement continuous turn* *Grab the normal size remote controller* *Interact with joysticks* *Interact with the switch button* *Interact with UI*
	If the user is:	
	Getting close to UI with the right virtual hand, the rayline will occur automatically *Pressing the left trigger button when raycasts show up, UI function will be activated* *Pressing the left joystick down, the teleport will be activated* *Pressing the grip button, the virtual object can be grabbed* *Pressing the trigger button, the grabbing object can be released* *Pressing left button B when the hook gets close to the drum, the drum will be hoisted* *Pressing left button B when the hook hoists the drum, the drum will be released* *Interacting with the switch button by the virtual hand, the button will be rotated*	
	The main menu will be located:	
	In the front of the virtual environment	
	There will be additional UI elements for:	
	Scene 1: Introduction of the VR environment *Scene 2: Getting familiar with the interactive features in the VR environment* *Scene 3: Learning about the functional features of the "Ilmatar" crane* *Scene 4: Completing a virtual training* *Dashboard to show the info of the crane*	
Functional Features	*Fetch information from the DT document* *Support Oculus Quest 2 and Varjo XR-1* *Simulate the smart features, snag prevention, micro speed and inching*	
Other Features	*Realistic looking virtual hand models will be developed* *Interact the controller with the real fingers, the corresponding virtual finger bend* *Video stream tutorials about the features of the crane* *Sound play while the crane moves* *Sound play while interacting with UI* *Sound play while releasing anything* *Two remote controllers, a large one for the scene 2, the normal one for the scene 4*	

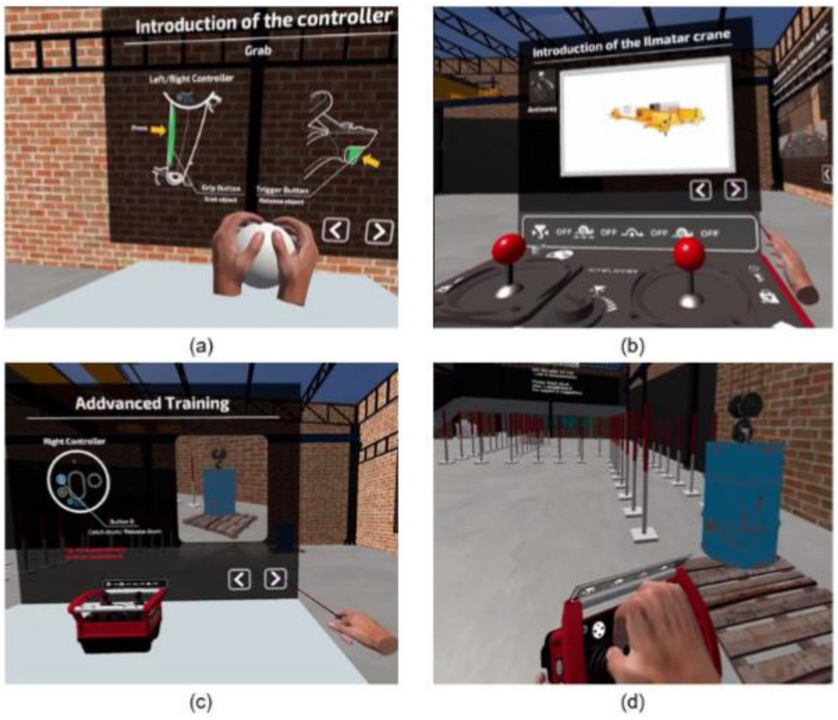

Introduction of the controller
Grab
Introduction of the Ilmatar crane
Addvanced Training
(a)
(b)
(c)
(d)

Figure 17 VR training app: (a) Be familiar with the fundamental features of the VR environment. (b) Interact with the large-scale remote controller to learn the smart features of the crane via the video stream. (c) Learn how to interact with the gradable same-scale remote controller. (d) Complete the realistic-looking training task.

The remote monitor prototyping application based on the developed XR environment is shown in Figure 18. The XR application development framework enables the minimization of repeated modeling development work across multiple platforms. We can easily develop a 3D environment by removing the relevant XR components from the XR environment in Unity. For the remote monitor app, through the OPC UA-MQTT wrapper, the predefined status and control data shown in Table 2 could be transmitted to the MQTT client in Unity via OPC UA-MQTT wrapper. In our research, the MQTT broker in the network layer was deployed in the public network, which allows the user to monitor the operation status of the crane on their device remotely. For the prototyping application, only the basic features were implemented. The Unity-based crane model synchronizes the movement trajectory from the OPC UA server. Furthermore, the operating data can be visualized on the dashboard. On the other hand, given the fact that PC, smartphones, or tablets would be commonly used as terminal devices, the scene roaming via screen touch and mouse control was developed.

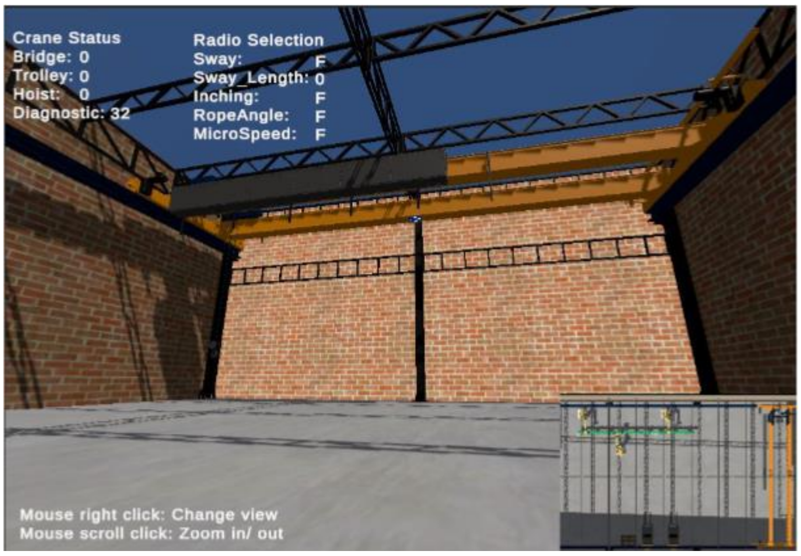


Figure 18 The screenshot of the remote monitor prototyping app. The dashboard is located in the upper left corner with operating data. Moreover, mouse control and screen touch can implement the view change.

Figure 19 displays the shortcut of the remote-control app. To guarantee the quality and safety of the data transmission, we applied the OPC UA-GraphQL wrapper and OPC UA-Unity client as the communication chains. The next section discussed the difference between two kinds of communication middleware. Given the data flow, the remote-control app contains bidirectional data transmissions, one is the same as above from the OPC UA server to the Unity client, while the other one is from Unity to the OPC UA server for the movement data of the crane's subsystems defined in the class control in Table 2. In addition, a 2D controller panel was developed for user interaction.

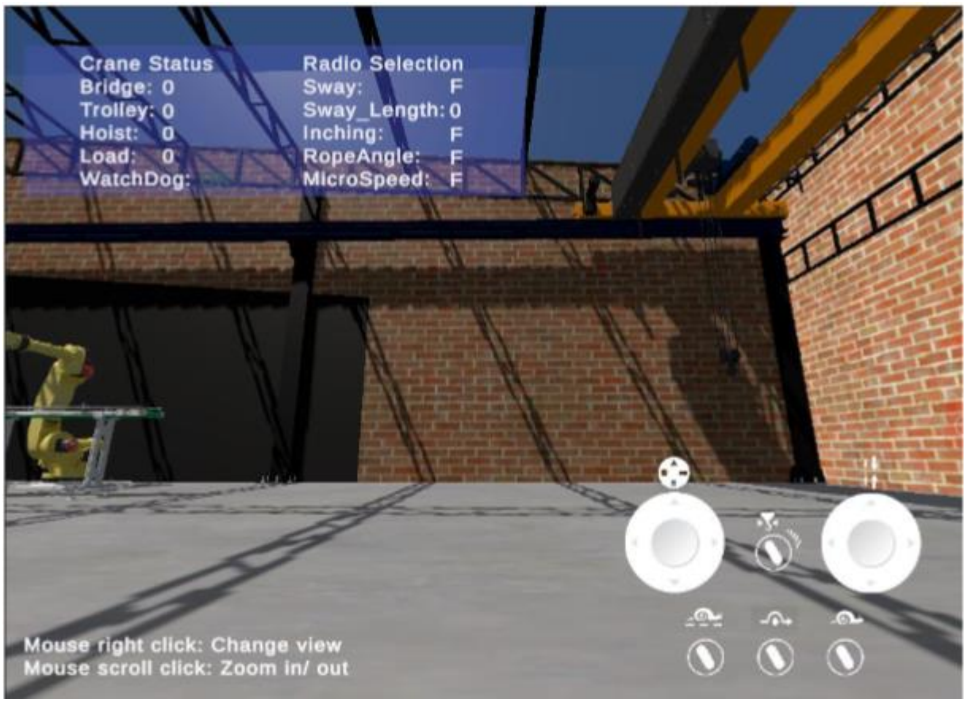
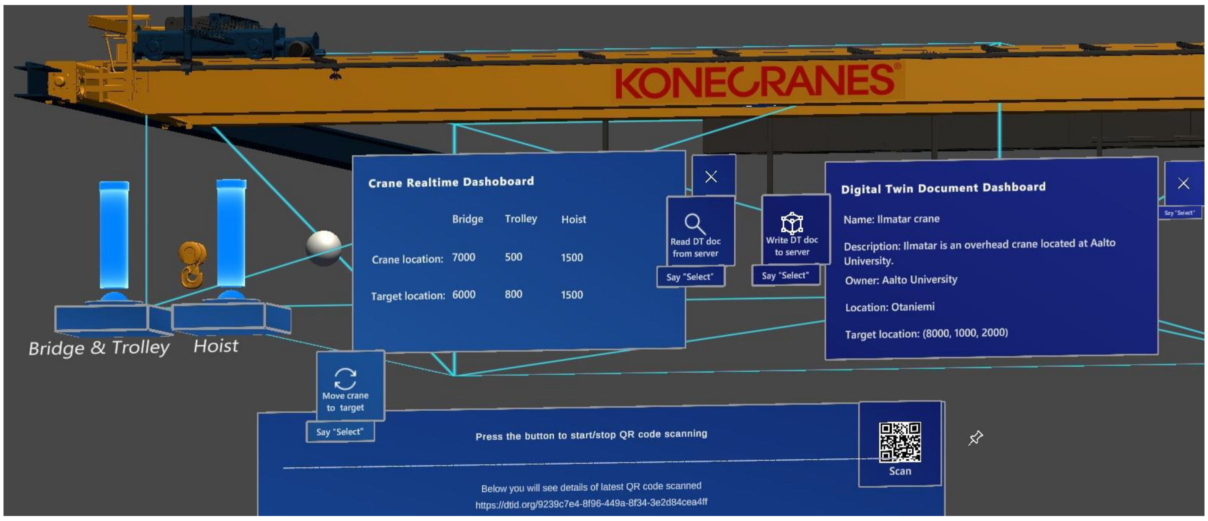


Figure 19 The screenshot of the remote-control prototyping app. The 2D remote controller is located in the lower and right corner.

2.3.2. XR interface for crane operation with DT documents

TwinXR is a method that uses descriptions of DTs of Smart Factory devices in creating and instantiating their XR applications. We develop a TwinXR-compatible MR application, “HoloCrane" to demonstrate the usage and validity of the method.

The “HoloCrane" application enables users to control a crane hologram with a virtual joystick or target positioning (Tu, Autiosalo, Jadid, Tammi, & Klinker, 2021), as well as view the crane real-time data and its linked DT document content via virtual dashboards. The safety zone indicator outlines the operationally safe range for the crane movement in three dimensions and therefore improves users' situational awareness. Instructions are available to guide users through the basics of using the application. Interface adjustment allows configuring the visibility of certain holograms. Through scanning the QR code of a Twin ID, the corresponding DT document will be linked to the application. Figure 20 shows the application in the Unity developer view. The application is deployed to and running on the Trimble XR10 with HoloLens 2 device in this implementation, but it can also be possibly deployed to other devices such as Meta Quest 2.



Crane Realtime Dashoboard

	Bridge	Trolley	Hoist
Crane location:	7000	500	1500
Target location:	6000	800	1500

Digital Twin Document Dashboard

Name: Ilmatar crane

Description: Ilmatar is an overhead crane located at Aalto University.

Owner: Aalto University

Location: Otaniemi

Target location: (8000, 1000, 2000)

Press the button to start/stop QR code scanning

Below you will see details of latest QR code scanned
https://dtid.org/9239c7e4-8f96-449a-8f34-3e2d84cea4ff

Figure 20 Screenshot of the implemented MR application “HoloCrane" (Unity developer view on desktop).

As shown in Figure 21, the application consists of a DT-doc module, two common XR modules, i.e. a control module and a visualization module, as well as the QR code scanning functionality:

- **DT-doc module** includes a DT-doc handler that enables the data flow and conversion of the DT documents on the server and its local copy in the MR application;
- **Control module** includes the features of virtual joystick control and target positioning. The value of the target position can flow in both directions between the control module and the local DT document;
- **Visualization module** includes the features of the crane real-time dashboard, the DT document dashboard, the safety zone indicator, the instruction, and the interface adjustment option. The DT document content for the dashboard, the safety zone

range for the indicator, the instruction text, and XR component visibility for interface adjustment are determined by the values stored in the local DT document.

- **QR code scanning** functionality enables Twin ID input through scanning the corresponding QR code. The inputted Twin ID is further forwarded to the DT-doc module and used to map the application to the corresponding DT document on the Twinbase server.

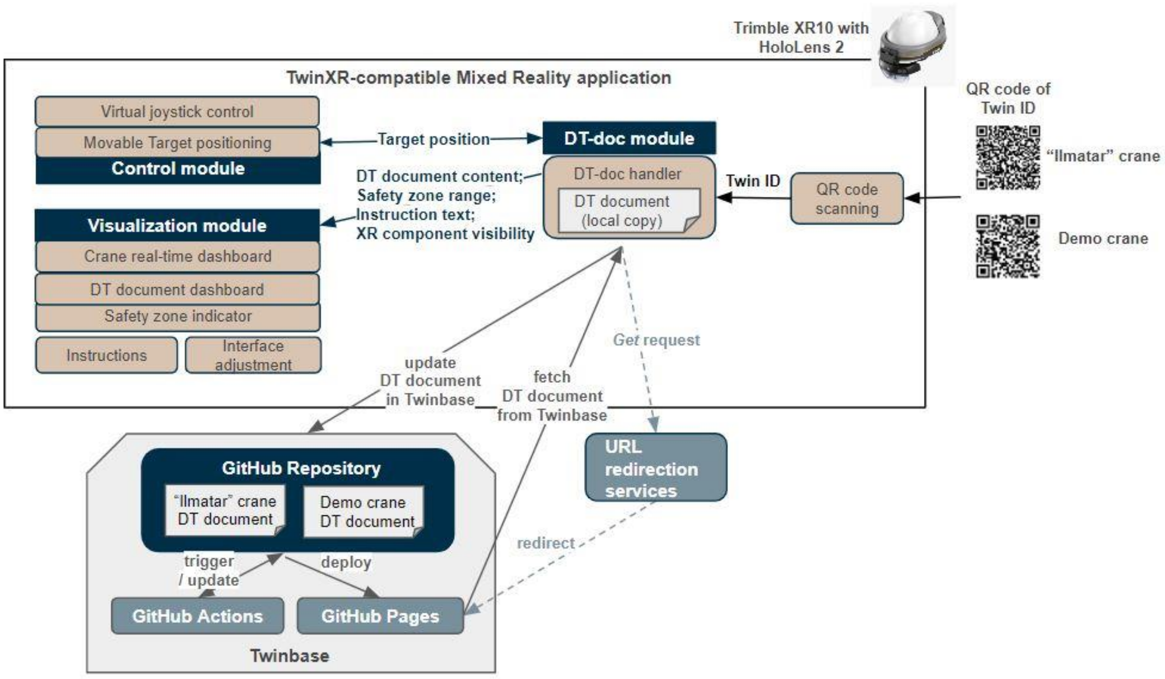
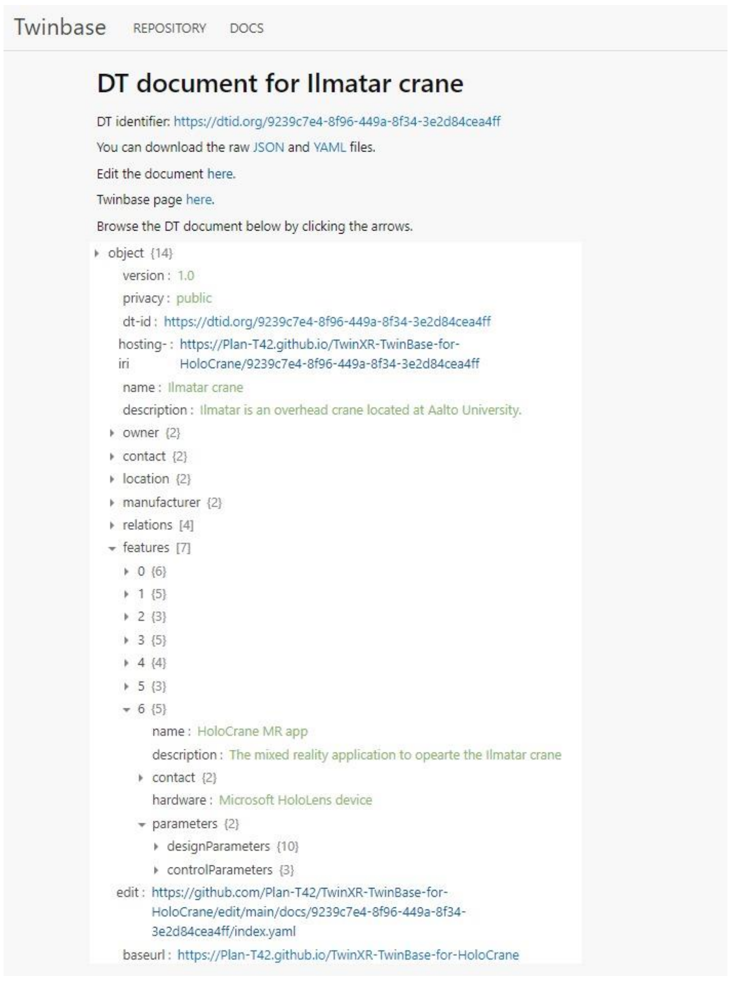


Figure 21 Architecture of the TwinXR method implementation for crane operation with a TwinXR-compatible MR application, and a Twinbase server with crane DT documents.

The process of using the application follows the workflow below: First, users would scan either the QR code of the Twin ID of the "Ilmatar" crane or the demo crane, and press *"Read DT doc from server"* button; This step triggers the DT-doc module to send a *get* request to the URL redirection services, which then redirect the Twin ID to the corresponding DT document on Twinbase; Consequently, the DT-module fetches the DT document and converts it to a local copy; Then, the visualization and control modules read parameters from the local DT document; Accordingly, the application is customized in terms of e.g. the content DT document dashboard, the location of the target hologram, the coverage of the safety zone indicator, .etc. Meanwhile, users can also move the target hologram, and press the *"Write to DT doc to server"* button to confirm; This step leads to an update on the local DT document with the current target location; The DT-doc module sends the modified local DT document to replace the one on Twinbase. The steps above are repeatable while using the application. In other terms, users can scan the QR code of another Twin ID, fetch or modify the DT document on the Twinbase server anytime according to need.

The implemented Twinbase server leverages the provided template and shares the same components: a GitHub repository, GitHub Actions, and GitHub Pages. The GitHub repository contains DT documents and other files, which are updated by the GitHub Actions and deployed to the GitHub Pages for distribution. For Twin IDs, this work uses the URL redirection service at dtid.org, hosted by Rebrandly. The Twin IDs are redirected to the corresponding twins on the GitHub Pages. As a special measure due to a bug in the *URL* property in *UnityWebRequest*, the Twin IDs used in this work also have an accompanying redirection towards the JSON version of the DT document.

Based on the defined XR ontology, we create two crane DT documents on Twinbase. One is for the “Ilmatar" crane. The other is for a demo crane of the same type as the “Ilmatar" crane but located in a different operating environment, thus with different safety zone, target location, etc. The demo crane DT document corresponds to an instance that is adapted from the “Ilmatar" DT document. Figure 22 shows the Twin page of the “Ilmatar" crane, with the XR ontology located under the level of *features.* The page for the demo crane shares the same layout, with only the values of certain terms differing. The Twin pages can be accessed via the corresponding Twin IDs. In our demonstration setup, the Twin IDs are also encoded into QR codes to enable easy access from the application.



DT document for Ilmatar crane

DT identifier: https://dtid.org/9239c7e4-8f96-449a-8f34-3e2d84cea4ff

You can download the raw JSON and YAML files.

Edit the document here.

Twinbase page here.

Browse the DT document below by clicking the arrows.

```
object {14}
  version : 1.0
  privacy : public
  dt-id : https://dtid.org/9239c7e4-8f96-449a-8f34-3e2d84cea4ff
  hosting- : https://Plan-T42.github.io/TwinXR-TwinBase-for-
  iri        HoloCrane/9239c7e4-8f96-449a-8f34-3e2d84cea4ff
  name : Ilmatar crane
  description : Ilmatar is an overhead crane located at Aalto University.
  owner {2}
  contact {2}
  location {2}
  manufacturer {2}
  relations [4]
  features [7]
    0 {6}
    1 {5}
    2 {3}
    3 {5}
    4 {4}
    5 {3}
    6 {5}
      name : HoloCrane MR app
      description : The mixed reality application to opearte the Ilmatar crane
      contact {2}
      hardware : Microsoft HoloLens device
      parameters {2}
        designParameters {10}
        controlParameters {3}
  edit : https://github.com/Plan-T42/TwinXR-TwinBase-for-
         HoloCrane/edit/main/docs/9239c7e4-8f96-449a-8f34-
         3e2d84cea4ff/index.yaml
  baseurl : https://Plan-T42.github.io/TwinXR-TwinBase-for-HoloCrane
```

Figure 22 Architecture of the TwinXR method implementation for crane operation with a TwinXR-compatible MR application, and a Twinbase server with crane DT documents.

2.3.3. Virtual Grinding machine

The RollResearch virtual grinding machine consist of a Siemens Sinumerik 840D sl numerical controller, and some additional devices, e.g., machine control panel, display, keyboard, etc. The Sinumerik 840D sl does not support virtualization as well as the new Sinumerik One where virtualization is already built in the system. The original plan was to use Sinumerik One, but the schedule by Siemens to bring out the Sinumerik One was delayed by over a year, and thus reduced its usefulness in this project. The current virtual grinder is therefore built around the Sinumerik 840D sl. Both controllers include a PLC CPU, which is used to monitor the machine and operate its supporting devices, e.g., lubrication pumps. The PLC CPU in the 840D is a S7-300 series CPU and in One S7-1500 series CPU.

The virtualization of the grinding machine is done mainly by setting the machining axes of the numerical controller to simulation mode. This allows the running of standard grinding machine NC programs. The modifications of the NC programs required by the virtualization are minimal. More changes were required in the PLC program, because no actual movements are done, which could be monitored, and no real supporting devices exist, which could be controlled. The devices of the virtual grinding machine were installed on a standard grinding machine operator’s panel, see Figure 23.

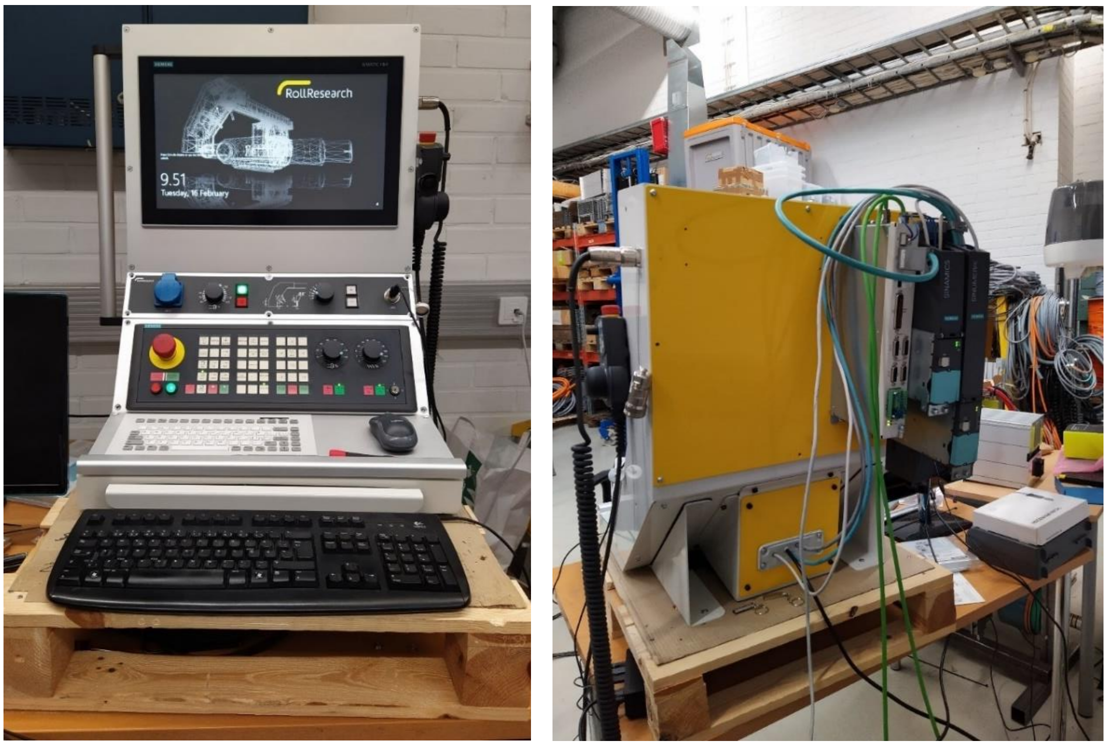


Figure 23 Hardware of Virtual Grinding Machine.

The project also included the programming of new interface software. One of the goals for the new software is to support virtualization better than the existing software. Other goals were to improve the user experience, to support the installation and commissioning phase of the machine, to get more information for service and there specially to collect long term data to plan the service in advance. The collection of production specific data will also be included in the new software, such as, grinding time, grinding efficiency, down times, etc.

In this project the interaction of the virtual grinding machine with other digital twins also played an important role. It is planned that the crane and the grinding machine can interact with each other. The grinding machine can signal to the crane its status (available for new work piece, busy, work piece ready to be picked up, ...), and the crane can signal its status back to the grinding machine (busy, lifting a new work piece, work piece ready in the machine, …). This communication will be done over OPC UA interface of the numerical controller. The OPC UA interface can be used for other data transfer. If, e.g., performance or status data of the grinding machine is required in the ERP of the (simulated) factory, this can be made available. The OPC UA link can be expanded much in the future.

2.4. Data processing and software updates

2.4.1. Scalable data processing

Modern IoT systems require an efficient and scalable IoT processing pipeline. Indeed, specific tooling for reading the data in from multiple data streams, transforming and manipulating it in several dimensions and finally outputting it in a useful format e.g. for representation purposes are needed. Moreover, as many IoT systems are built on top of cloud infrastructure, compatibility with different types of web services are important for creating impactful applications on top of the processed data.

Given these guidelines and the requirement to process data from multiple cranes simultaneously, we evaluated in this project a data processing framework created by Apache Foundation, namely Apache Flink. As Amazon Web Services (AWS) offers Apache Flink as a managed service with their own name Kinesis Data Analytics (KDA) which is easy to take into use, we used their service along with other AWS services (e.g. Kinesis Data Streams) that may be integrated to KDA easily. The goal of the evaluation was to find, if Apache Flink run on AWS, could be used as a scalable data processing platform for crane-collected time-series data. Figure 24 shows the high level test setup.

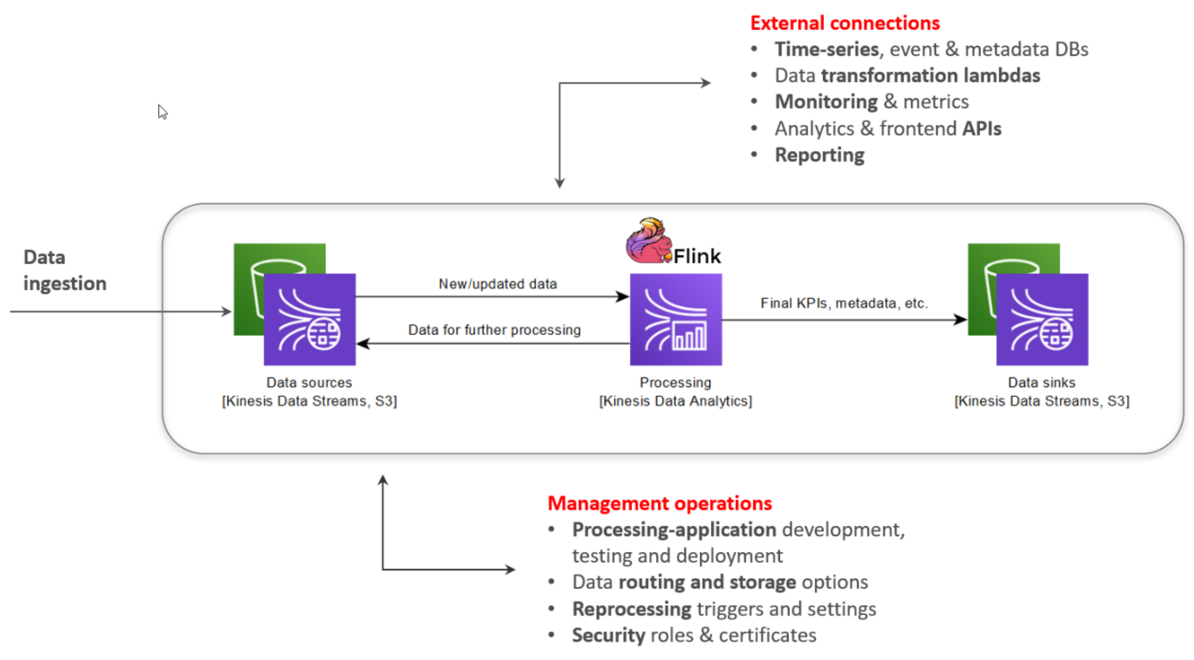

External connections
Time-series, event & metadata DBs
Data transformation lambdas
Monitoring & metrics
Analytics & frontend APIs
Reporting
Data ingestion
Flink
New/updated data
Data for further processing
Final KPIs, metadata, etc.
Data sources [Kinesis Data Streams, S3]
Processing [Kinesis Data Analytics]
Data sinks [Kinesis Data Streams, S3]
Management operations
Processing-application development, testing and deployment
Data routing and storage options
Reprocessing triggers and settings
Security roles & certificates

Figure 24 High-level data processing architecture used in the evaluation of Apache Flink. AWS offered managed service Kinesis Data Analytics was used as the practical alternative to ease the setup of the processing environment. In addition to the actual data path, several external data connections and management operations may be linked to the processing flow.

Test setup

For the test setup, we used AWS Cloud Development Kit (CDK) to deploy all the required components of a full IoT pipeline into AWS operated cloud. In addition to the above mentioned components, the pipeline was equipped with a long-term storage for the data, a data consumer Lambda that fed the processed data to Regatta Portal backend, a Remion offered IoT platform for operative IoT data, so that it could be explored and analyzed by humans using ready-made visualization components. Figure 25 shows the high-level test setup along with the details of the ingested data and system configuration.

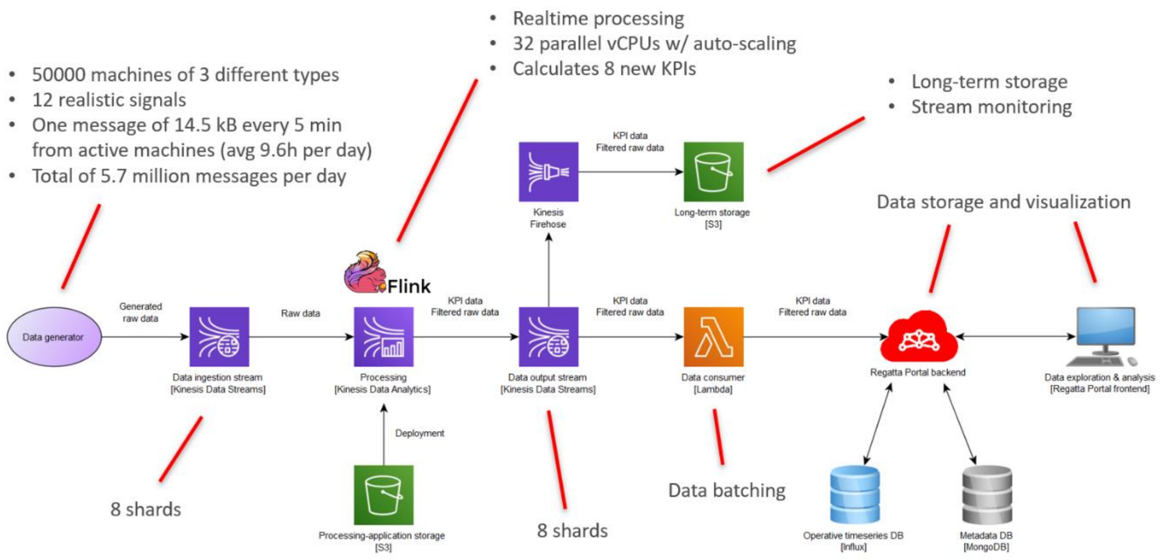
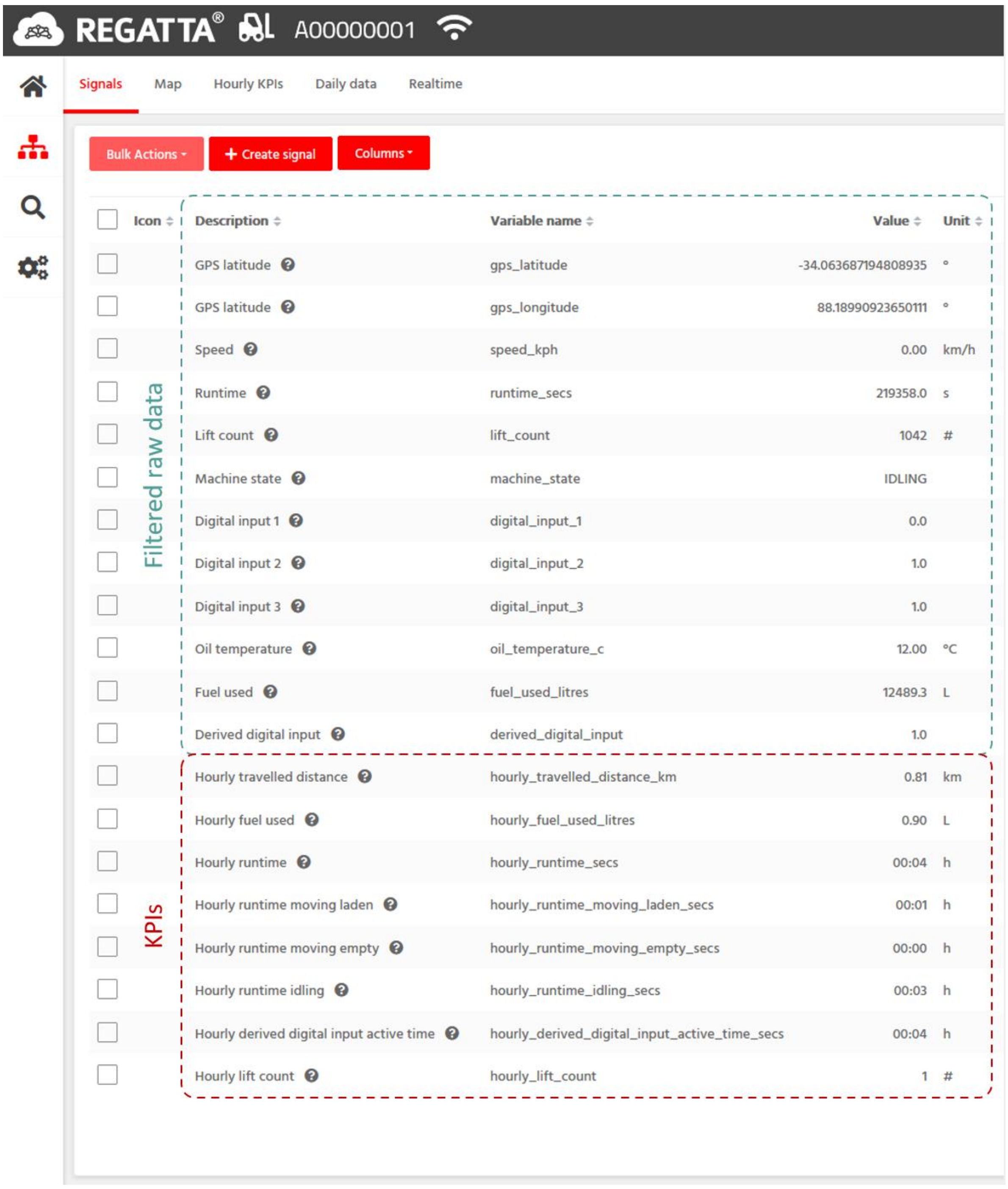

Realtime processing
32 parallel vCPUs w/ auto-scaling
Calculates 8 new KPIs
50000 machines of 3 different types
12 realistic signals
One message of 14.5 kB every 5 min from active machines (avg 9.6h per day)
Total of 5.7 million messages per day
Long-term storage
Stream monitoring
KPI data
Filtered raw data
Kinesis Firehose
Long-term storage [S3]
Data storage and visualization
Flink
Data generator
Generated raw data
Data ingestion stream [Kinesis Data Streams]
Raw data
Processing [Kinesis Data Analytics]
KPI data
Filtered raw data
Data output stream [Kinesis Data Streams]
KPI data
Filtered raw data
Data consumer [Lambda]
KPI data
Filtered raw data
Regatta Portal backend
Data exploration & analysis [Regatta Portal frontend]
Deployment
8 shards
Processing-application storage [S3]
8 shards
Data batching
Operative timeseries DB [Influx]
Metadata DB [MongoDB]

Figure 25 The actual deployed test setup included multiple supporting components and services. Altogether, they created a full-blown IoT pipeline that included features such as long-term data storage, operative data storage and hands-on tools for data exploration and analysis.

To account for realistic high-level loads, data from 50 000 virtual machines of 3 different types were used as an input for the processing system. Each machine was set to provide 12 signals and send data in every 5 minutes while they were in use. The list of the filtered raw signals and the calculated KPIs from them are shown in Regatta Portal in Figure 26 and examples of the processed KPIs in Figure 27.



Icon	Description	Variable name	Value	Unit
	GPS latitude	gps_latitude	-34.063687194808935	°
	GPS latitude	gps_longitude	88.18990923650111	°
	Speed	speed_kph	0.00	km/h
	Runtime	runtime_secs	219358.0	s
	Lift count	lift_count	1042	#
	Machine state	machine_state	IDLING	
	Digital input 1	digital_input_1	0.0	
	Digital input 2	digital_input_2	1.0	
	Digital input 3	digital_input_3	1.0	
	Oil temperature	oil_temperature_c	12.00	°C
	Fuel used	fuel_used_litres	12489.3	L
	Derived digital input	derived_digital_input	1.0	
	Hourly travelled distance	hourly_travelled_distance_km	0.81	km
	Hourly fuel used	hourly_fuel_used_litres	0.90	L
	Hourly runtime	hourly_runtime_secs	00:04	h
	Hourly runtime moving laden	hourly_runtime_moving_laden_secs	00:01	h
	Hourly runtime moving empty	hourly_runtime_moving_empty_secs	00:00	h
	Hourly runtime idling	hourly_runtime_idling_secs	00:03	h
	Hourly derived digital input active time	hourly_derived_digital_input_active_time_secs	00:04	h
	Hourly lift count	hourly_lift_count	1	#

Figure 26 Multiple realistic signals were used as an input to the data processing framework and they were brought to the Regatta Portal UI for exploration and analysis purposes along with the calculated KPIs from these raw data.

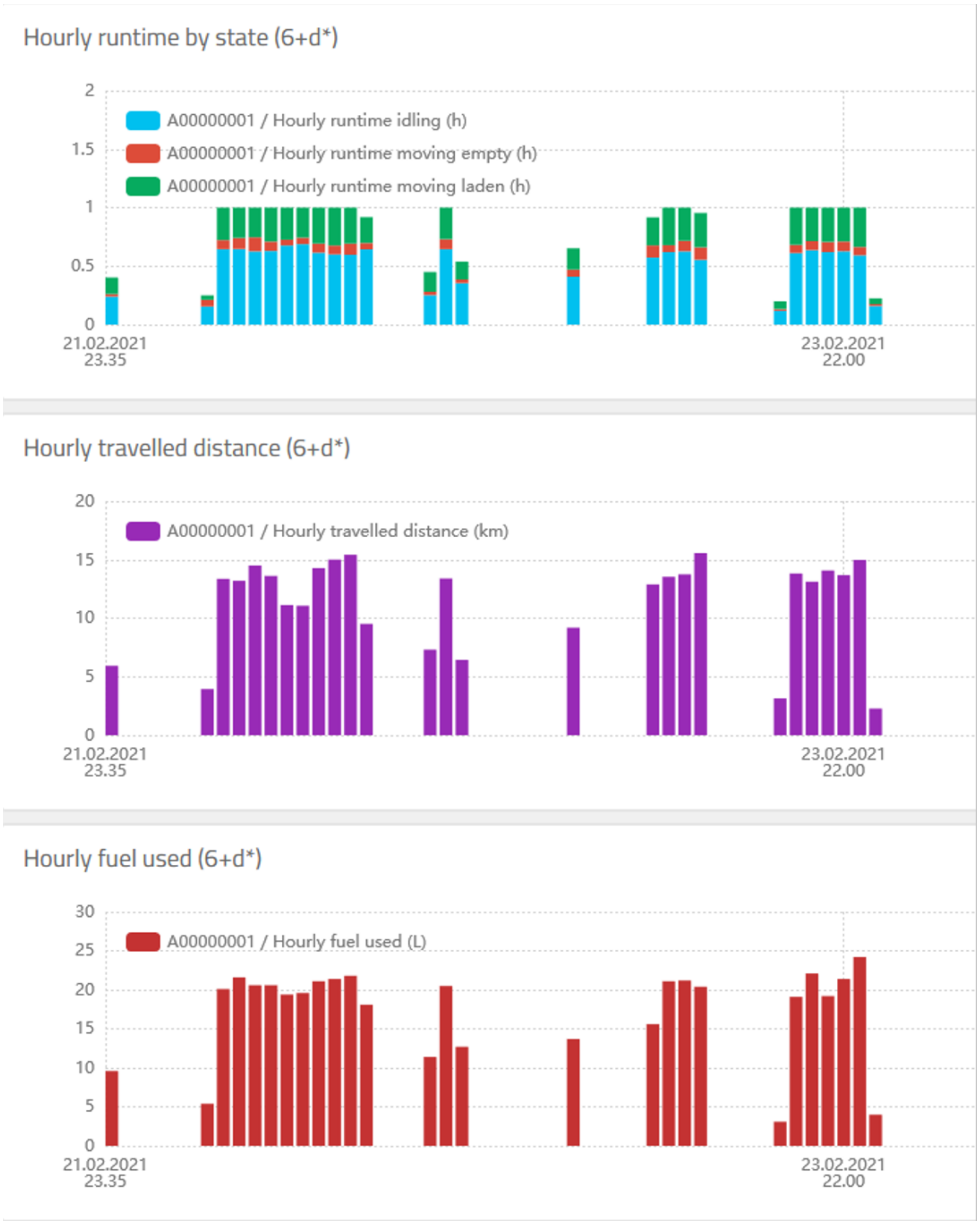


Figure 27 Examples of the calculated KPIs shown in Regatta Portal as histograms.

Results

We ran a 24-hour continuous stress test on the processing environment to see how the system would perform under extreme loads and what would be the limitations with the set configuration. As shown in Figure 28, we found out that the actual data processing was able to process the incoming data over 3 times faster than the realtime-rate of ingested data. Thus, data from about 160 machines could have been collected with the given rates and the system still would have not developed a queue for the data. The slowest performance was encountered in the data storage lambda, but its performance could have been scaled up to accommodate for the quicker processing.

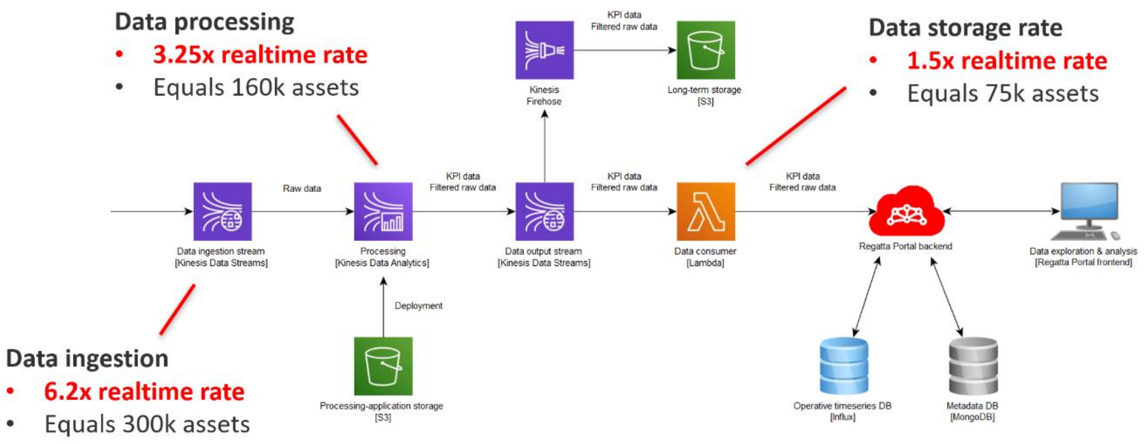


Figure 28 Achieved realtime processing rates with the selected configuration were all better than 1.

Our main finding was, that the demo architecture is suitable for processing realistic and realtime IIoT data at the intended scale. Moreover, it could be optimized and scaled up in AWS as needed, so the framework would suffice for computing at very large scale. Finally, the operating costs of the whole IoT processing pipeline of Figure 28 were found out to be about $0.05 / asset / month, which is very reasonable. Thus, we concluded the system to be suitable for processing crane data in realtime.

The main strengths of the evaluated system are:

- Flink provides a strong Java/Scala –based framework and platform for processing realtime data at scale
- Kinesis Data Analytics (KDA) eases Flink comissioning and deployment
- KDA operating-costs seem reasonable and many optimizations can be done

The main weaknesses of the developed IoT pipeline are:

- An analytical mindset with strong coding skills are required to master IIoT Flink applications
- Best practices for industrial IoT-data processing are not generally available, so they need to be sought on a case-by-case basis

2.4.2. Fuzzy modeling

In the project, a novel fuzzy-based approach for data modeling was developed. The work started as a master’s thesis by Keski-Heikkilä (2021) and later the method was further developed by translating the code from Java to Python. Based on Keski-Heikkilä’s thesis and further development, an article has been submitted to high quality journal: Elsevier Engineering Applications of Artificial Intelligence. The method development was motivated by the need of modeling and interpret the multivariate data that current machines equipped with multiple IoT sensor produce.

The fuzzy-based data modeling method allows examining the typical use of machine. For example, Figure 29 below shows the typical route of the crane. Because the data modeling method stores the typical use of a machine, it can be used for detecting changes in the machine operation. This can be used as a part of anomaly detection. Another use case for the developed method is optimization of the machine use. For example, suboptimal routes of the crane could be detected using method.

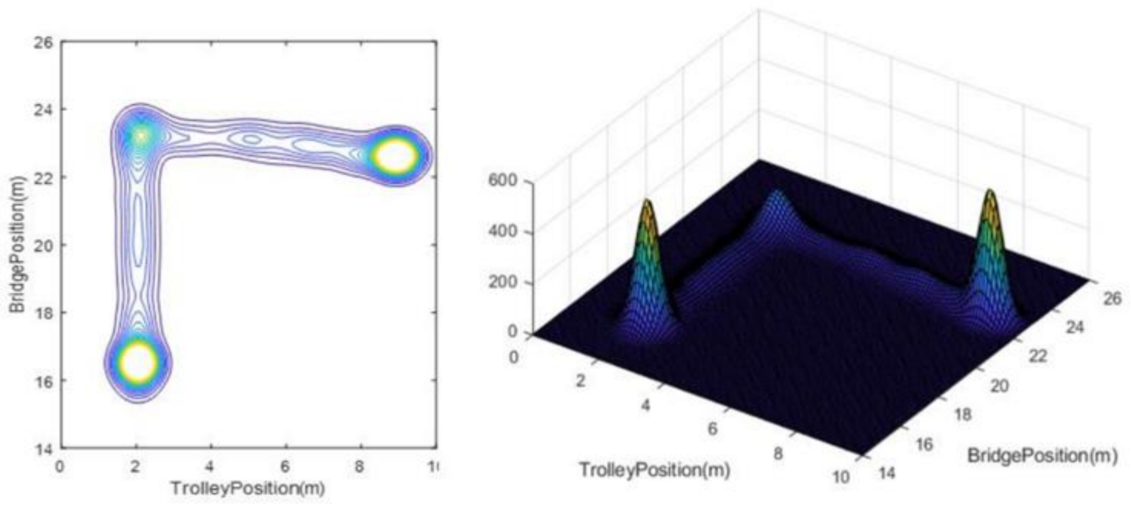


Figure 29 Fuzzy-based data modeling can be used to present the typical use of a machine. In this figure, the typical route of an overhead crane is visualized.

In a more technical level, the developed method first fuzzifies data, then finds the highest-grade fuzzy sets, finally combines these highest-grade fuzzy sets of different variables. After that data is time-bucketed which further saves storage space. The machine data is then stored into a time-series database, which allows efficient time-wise aggregation. Time-wise aggregation can be used examine machine operation using different time-ranges. The overall flow of fuzzy-based data modeling is illustrated in Figure 30. A comprehensive technical description of the modelling is available in (Keski-Heikkilä, 2021) and in the upcoming article.

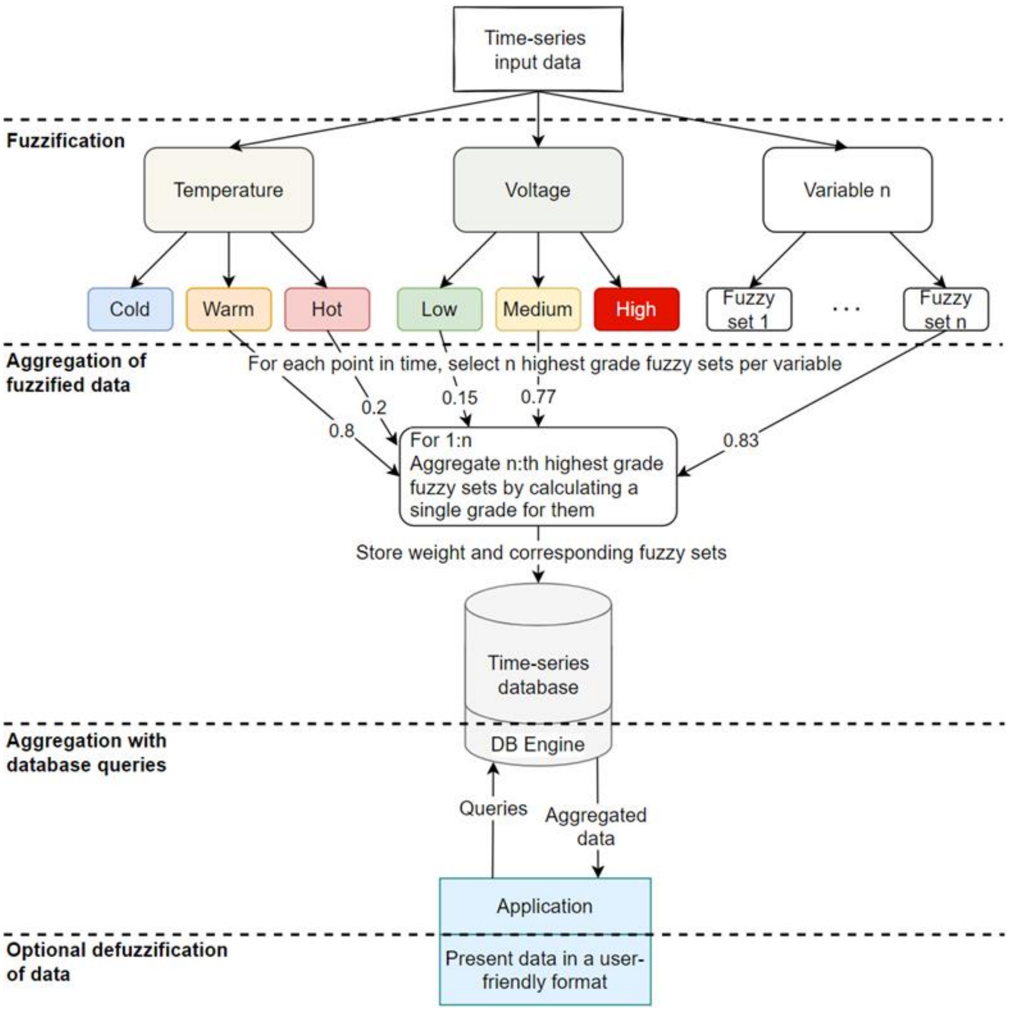

Time-series input data
Fuzzification
Temperature
Voltage
Variable n
Cold
Warm
Hot
Low
Medium
High
Fuzzy set 1
. . .
Fuzzy set n
Aggregation of fuzzified data
For each point in time, select n highest grade fuzzy sets per variable
0.2
0.15
0.77
0.8
0.83
For 1:n Aggregate n:th highest grade fuzzy sets by calculating a single grade for them
Store weight and corresponding fuzzy sets
Time-series database
Aggregation with database queries
DB Engine
Queries
Aggregated data
Application
Optional defuzzification of data
Present data in a user-friendly format

Figure 30 Overview of the fuzzy-based data modeling method.

2.4.3. Mass software updates

In industrial IoT systems gateway devices are used to collect, process, and transmit data to cloud from the attached machines or devices. As many gateway assets in large fleets are distributed around the world, remote mass management of their software and configuration in different lifecycle phases becomes important, for example, per business or security needs.

In this project, we developed a demonstrator user interface (UI) on top of Regatta IoT platform that can be used to remotely manage the software and configuration of the gateway devices in their different lifecycle phases. The demonstrator can be also extended to update the operational configuration of the machines behind the gateways.

Figure 31 shows a mass software update visualization from a prototype UI, that helps the user understand the current update status of the fleet of gateway devices. These update settings may be configured through a wizard type application, shown in Figure 32.

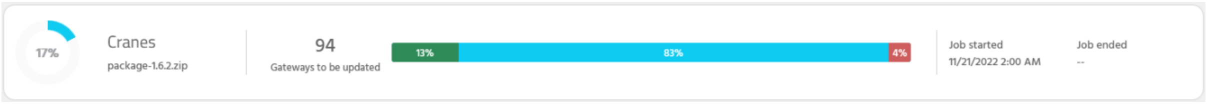
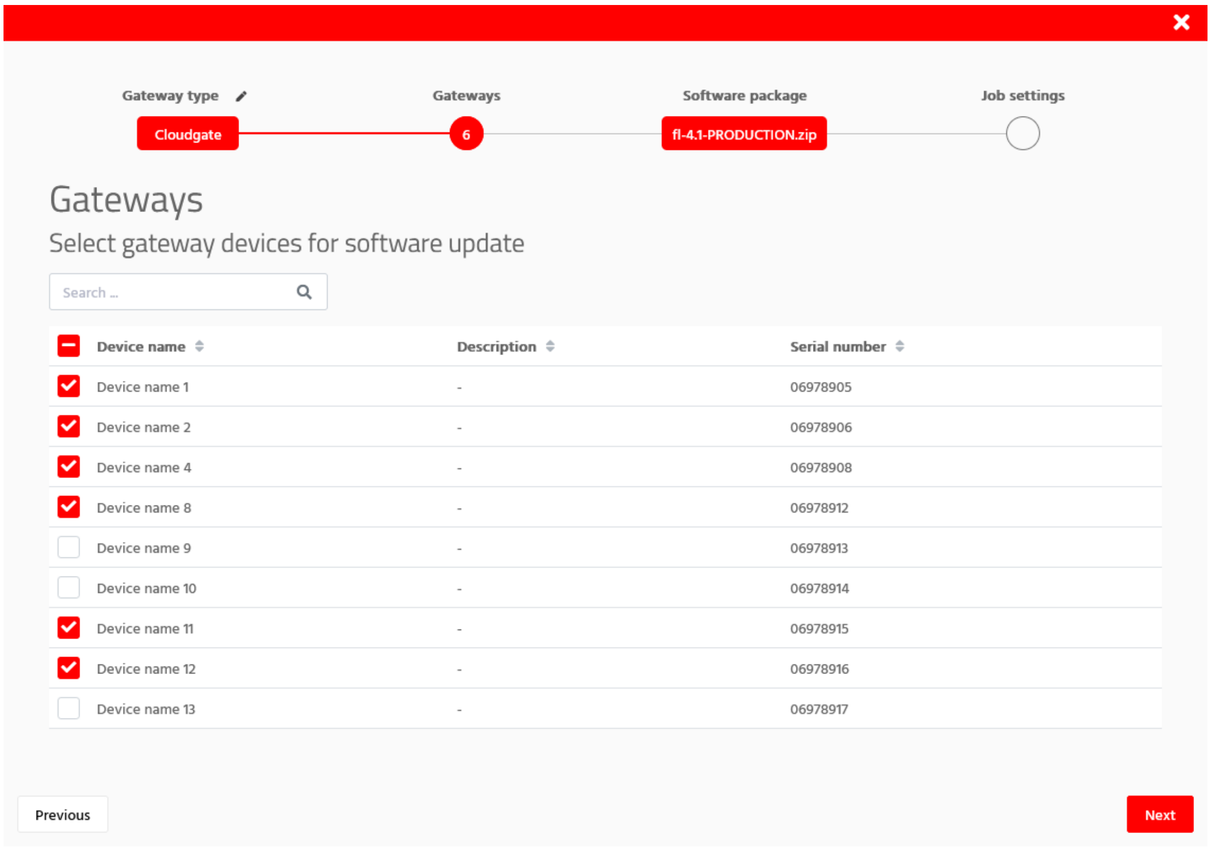

17%
Cranes
package-1.6.2.zip
94
Gateways to be updated
13% 83% 4%
Job started
11/21/2022 2:00 AM
Job ended
--

Figure 31 An example of software update visualization from which the user can monitor the status of the mass update job.



Gateways

Select gateway devices for software update

	Device name	Description	Serial number
✓	Device name 1	-	06978905
✓	Device name 2	-	06978906
✓	Device name 4	-	06978908
✓	Device name 8	-	06978912
	Device name 9	-	06978913
	Device name 10	-	06978914
✓	Device name 11	-	06978915
✓	Device name 12	-	06978916
	Device name 13	-	06978917

Figure 32 A software update configuration wizard with options to configure the gateway type, list of gateways and software package to be updated on the devices and the job settings for a mass update job.

2.5. Implementation schedule for demonstration

Main events have been collected on the Table 4.

Table 4 Implementation schedule for demonstration shown here with half year accuracy

Action	Starting Term	Ending Term	Responsible
Project duration	2019/2	2023/2	MACHINAIDE
Virtual Grinding Machine	2019/2	2023/1	RollResearch International
Hololens HMI for crane	2020/1	2020/2	AALTO
Scalable data processing	2020/1	2021/1	REMION
Smart factory with Tecnomatix	2020/2	2020/2	IDEAL
Fuzzy modelling	2020/2	2022/1	REMION
Communication between machines in smart factories	2021/1	2021/2	AALTO
Virtual Model with Varjo	2021/1	2021/2	AALTO
AIIC renovation	2021/1	2022/2	ACRE
OPC UA information model for crane	2021/1	2022/1	KONECRANES
Worklist app	2021/2	2023/1	IDEAL & KONECRANES
DT documents to control Smart Factory	2021/2	2022/1	AALTO
Hololens to DT document integration	2022/1	2022/2	AALTO
Mass software updates	2022/1	2023/2	REMION
Crane HW & SF update	2022/2	2023/1	KONECRANES & AALTO
Factory ontology system development	2022/2	2023/1	AALTO
Deploy XR application to physical platform	2022/2	2023/1	AALTO

3. Results

Results can be divided to concepts, software and demonstrations with physical machines including human-machine interfaces. In concepts the main focus is the development work done with Digital Twin Documents (DTD) and especially Digital Twin Web (DTW). Several scientific have been done around this topic, and especially doctoral dissertation by Juuso Autiosalo presents the results well (Autiosalo, Discovering the Digital Twin Web -from singular applications to a scalable network, 2021). DT document specifications, standardization and distribution is presented, as well as DTW. While this work explains methods to build standardized Digital Twins (DT), the challenge is that there are not yet widely used standards to build up DT´s and DTD´s. Thus, when different type of systems are connected, there could be need for brokers to transfer metadata between DT´s. Data Link could be one possible solution for this (Ala-Laurinaho, 2021). Now we have described how DT, DTD and DTW could be build. After that one common question is how can we use this concept and technology? In this project we have studied and demonstrated how we can use these documents for smart factory control (Mattila, Ala-Laurinaho, Autiosalo, Salminen, & Tammi, 2022) and how we can build Human Machine Interfaces (HMI) that use DTD´s as a source for metadata. There are publication submitted but not yet accepted that soon describes this work. Related to this, virtual model of AIIC research environment including Ilmatar Smart Crane and Virtual Grinding Machine has been made (Yang, et al., 2022). The virtual model has been used also to develop HMI´s for smart crane by using for example Hololens II, Varjo, Oculus or similar VR/XR headsets. At the later stage of Machinaide the virtual world will be connected to the physical machines. This interface work between physical and virtual worlds was delayed during the project as the Aalto Industrial Internet Campus laboratory went under renovation when pandemia closed down many activities at campus. During this time, the development of (Virtual) Grinding Machine was successful, and new software model supports virtualization and use of DT´s better than previous version. One of the findings was also that the new oncoming hardware/software version would enable even better interconnections to the DTW and plant simulation models that were developed during Machinaide project.

Machinaide has developed big concepts further and new (product) ideas have been emerged and these have been tested. These products will be developed further and multinational research work done in this project has build a stable platform for further development. We believe that after Machinaide the society and industry are well prepared to connect various systems together and to improve accessibility to the metadata. In addition, new HMI´s have been tested and we hope to demonstrate these during the last stages of project.

4. Abbreviations

ADT	Azure Digital Twins
AIIC	Aalto Industrial Internet Campus
API	Application Programming Interface
AR	Augmented Reality
AR/VR	Augmented Reality/ Virtual Reality
AWS	Amazon Web Services
CAD	Computer Aided Design
CPS	Cyber Physical System
DT	Digital Twin
DTD	Digital Twin Document
DTW	Digital Twin Web
HMI	Human Machine Interface
HTTP	Hypertext Transfer Protocol
IOT	Internet of Things
JSON	JavaScript Object Notation
MR	Mixed Reality
MQTT	Message Queuing Telemetry Transport
ODBC	Open Database Connectivity
O-DF	Open Data Format
OPC	Open Platform Communications
REST	Representational state transfer
PaaS	Platform as a Service
VR	Virtual Reality
WOL	Web Ontology Language
W3C	World Wide Web Consortium
XR	Extended Reality

5. References

Ala-Laurinaho, R. (2021). *API-based Digital Twins* (Doctoral dissertations 159 ed.). Espoo: Aalto University publication series. Retrieved from http://urn.fi/URN:ISBN:978-952-64-0594-0

Ala-laurinaho, R., Autiosalo, J., Nikander, A., Mattila, J., & Tammi, K. (2020, 12 18). Data Link for the Creation of Digital Twins. *IEEE Access, 8*, 228675-228684.

Autiosalo, J. (2018). Platform for industrial internet and digital twin focused education research, and innovation: Ilmatar the overhead crane. *IEEE 4th World Forum on Internet of Things (WF-IoT)*, (pp. 241-244). doi:10.1109/WF-IoT.2018.8355217

Autiosalo, J. (2021). *Discovering the Digital Twin Web -from singular applications to a scalable network* (Vol. DOCTORAL DISSERTATIONS 172/2021). Espoo: Aalto University publication series. Retrieved from http://urn.fi/URN:ISBN:978-952-64-0621-3

Autiosalo, J., Siegel, J., & Tammi, K. (2021, 10 13). Twinbase: Open-source server software for the Digital Twin Web. *IEEE Access, 9*, 140779 - 140798. Retrieved from https://ieeexplore.ieee.org/abstract/document/9568895

Keski-Heikkilä, T. (2021). *Multivariate fuzzy modelling of timeseries.* Espoo: Aalto Publication series.

Mattila, J. (2021). *Communication between machines.* Espoo: Aalto University publication series.

Mattila, J., Ala-Laurinaho, R., Autiosalo, J., Salminen, P., & Tammi, K. (2022, March 23). Using Digital Twin Documents to Control a Smart Factory: Simulation Approach with ROS, Gazebo, and Twinbase. *Machines*, 5. Retrieved from https://doi.org/10.3390/machines10040225

Tu, X., Autiosalo, J., Jadid, A., Tammi, K., & Klinker, G. (2021, 10 12). A Mixed Reality Interface for Digital Twin Based Crane. *Applied sciences, 11*(20). doi:https://doi.org/10.3390/app11209480

Yang, C. (2021). *Framework for Virtual Reality digital services leveraging digital twin-based crane.* Department of Mechanical Engineering. Espoo: Aalto University publication series.

Yang, C., Tu, X., Autiosalo, J., Ala-Laurinaho, R., Mattila, J., Salminen, P., & Tammi, K. (2022, June 14). Extended Reality Application Framework for a Digital-Twin-Based Smart Crane. *Applied Sciences*, 24. Retrieved from https://doi.org/10.3390/app12126030