




Cognitive plants through proactive self-learning hybrid digital twins

COGNITWIN

DT-SPIRE-06-2019 (870130)

Deliverable Report

Deliverable ID	D4.4	Version	V1.1
Deliverable name	Final Platform, Sensor and Data Interoperability Toolbox		
Lead beneficiary	NST		
Editor	Nenad Stojanovic (NST)		
Contributors	Ljiljana Stojanovic (Fraunhofer), Michael Jacoby (Fraunhofer), Stefan Marinkovic (NST), Arne Jørgen Berre (SINTEF), Dumitru Roman (SINTEF), An Lam (SINTEF), Anders Hansen (SINTEF), Enso Ikonen (UOULU), Jan Gunnar Dyrset (CYB), Nenad Stojanovic (NST), Perin Ünal (Tekno), Özlem Albayrak (Tekno), Adriaen Verheyleweghen (CYB)		
Due date	28.02.2023		
Date of final version	15.11.2023		
Dissemination level	Public		
Document approval	Frode Brakstad	15.11.2023	



The COGNITWIN project has received funding from the European Union's Horizon 2020 research and innovation programme under GA No. 870130

PROPRIETARY RIGHTS STATEMENT

This document contains information which is proprietary to the COGNITWIN consortium. The document or the content of it shall not be communicated by any means to any third party except with prior written approval of the COGNITWIN consortium.

Executive Summary

This D4.4 deliverable on COGNITIVE Final Platform, Sensor and Data Interoperability Toolbox describes a final specification and implementation of the COGNITWIN Toolbox in the areas of toolbox architecture, cyber security, data management and data spaces, further with sensors and real-time sensor/data processing. It refines and extends the D4.1, D4.2 and D4.3 deliverables by providing more technical details especially on new features. In particular it extends the description of already provided functionalities with new one developed during the last reporting period.

In this report we discuss the final state of the progress on technologies and methods that are being developed for COGNITIVE Platform, Sensor and Data Interoperability Toolbox, and which we have applied to the pilots. This has evolved from baseline through initial over updated to the final COGNITIVE Platform, Sensor and Data Interoperability Toolbox following period 3 in the project.

The D4.4 deliverable is accompanied with demonstrators that show how a subset of components play together. The demonstrators that show reusable patterns which can be replicated and adapted also in future industrial process industry settings.

The new contributions from the WP4 technology partners in the D4.4 update from D4.3 in particular includes the following;

- SINTEF – SINDIT (SINTEF Digital Twin) python framework is extended with Semantic Web Technologies to provide a formal standardized representation of heterogeneous data. The framework was applied to Sidenor pilot to enable the "cognition" aspect of the Cognitive Digital Twin. Accordingly, the data from processes and assets are transformed and integrated into the knowledge graph in order to support data-driven modelling. Domain expert knowledge was also added to SINDIT in the form of semantic rules to support automatic reasoning and decision-making. Details about the usage of SINDIT are reported in D2.4 and presented in the demonstration video of the Sidenor pilot.
- Cybernetica - Cybernetica OPC UA server has been extended with a module to exchange data with a PostgreSQL database.
- NST – Further development of the Streampipes framework, resulting in 1) development of CEP functionalities through Siddhi framework, 2) design and development of the element for unusuality detection, 3) design and development of the adapters for the numerical models (their integration in the Streampipes and 4) application on the Sidenor use case. Validation is done using the existing datasets
- Fraunhofer - The FA³ST (Fraunhofer Advanced Asset Administration Shell (AAS) Tools for Digital Twins) service has been enhanced with database persistence, HTTP asset connection, support for configuration, reconnection, and basic security, improved documentation, etc. The most important enhancement is related to real-time bidirectional connection with a physical asset, support for three types of interaction patterns, and default implementation for OPC UA, HTTP, and MQTT. A new video demonstrator of FA³ST has also been provided.
- UOULU - UOULU focused on the OD-tool, developed and published during the period. This tool is included in the FouMon component and provides an application/pilot independent tool for outlier detection and data quality improvement, to support data-driven modeling of system dynamics. The tool was experimented using data from the Sumitomo pilot. The OD-tool for Matlab is freely available (see COGNITWIN Toolbox portal). The OD-tool demonstration is

included in two videos, the other focusing on usage of developed tools and the other on application for solving the WP3 pilot problem.

- Tekno – A new tool was designed, developed, and validated in a real pilot environment. The developed tool, TIA CONTROL was also added to the Teknopar Industrial Automation (TIA) Platform, belong to the TIA IOT toolbox. TIA CONTROL tool enables the users to create for remote management and control of a machine or process. The tool uses On2RDB, a script to serialize ontology models on relational databases for the purposes of high query performance.

Table of Contents

- 1 Introduction..... 15
 - 1.1 Scope and purpose 15
 - 1.2 Relevance to other WPs and current and previous deliverables 15
 - 1.3 Structure of the deliverable 15
- 2 High level overview 17
 - 2.1 Integration and Interoperability considerations 17
 - 2.2 Design decisions 18
 - 2.2.1 StreamPipes for orchestration of the components..... 18
 - 2.2.2 Asset Administration Shells for Digital Twins 19
 - 2.2.3 Support of all different Big Data Types and different pipelines and OPC 19
 - 2.3 Conceptual architecture 20
 - 2.3.1 Current status..... 20
 - 2.3.2 Evolution of the architecture 21
- 3 COGNITWIN Interoperability Toolbox..... 23
 - 3.1 Introduction to the COGNITWIN Interoperability Toolbox 23
 - 3.2 End-to-End COGNITWIN Toolbox Pipeline Architecture 23
 - 3.2.1 Objectives, challenges and components 23
 - 3.2.1.1 COGNITWIN Toolbox end-to-end architecture - Digital Twin Pipeline 24
 - 3.2.1.2 COGNITWIN Pipeline architecture related to RAMI 4.0 26
 - 3.2.1.3 COGNITWIN Toolbox end-to-end architecture 29
 - 3.2.1.4 Digital Twin - Data Acquisition/Collection..... 30
 - 3.2.1.5 Digital Twin Data Representation..... 31
 - 3.2.1.6 Digital Twin Hybrid and Cognitive Analytics Models..... 32
 - 3.2.1.7 Digital Twin - Action/Interaction, Visualisation and Access 33
 - 3.2.1.8 COGNITWIN Toolbox – Method layer 34
 - 3.2.1.9 COGNITWIN Toolbox Web Portal 34
 - 3.2.2 Detailed description of the activities performed 37
 - 3.2.3 Progress beyond State of the Art or State of the Practice 37
 - 3.2.4 Summary of the key achievements 37
 - 3.2.5 Conclusion 38
 - 3.3 Interoperability Orchestration with Streampipes 38
 - 3.3.1 Objectives, challenges and components 38

3.3.2	Detailed description of the activities performed	41
3.3.3	Progress beyond State of the Art or State of the Practice	42
3.3.4	Summary of the key achievements	42
3.3.5	Conclusion	42
3.4	Communication and Semantic Interoperability with OPC UA, FMI etc	43
3.4.1	Objectives, challenges and components	43
3.4.2	Detailed description of the activities performed	46
3.4.3	Progress beyond State of the Art or State of the Practice	47
3.4.4	Summary of the key achievements	47
3.4.5	Conclusion	48
3.5	Big Data Pipeline Deployment Framework	48
3.5.1	Objectives, challenges and components	48
3.5.2	Detailed description of the activities performed	50
3.5.3	Progress beyond State of the Art or State of the Practice	50
3.5.4	Summary of the key achievements	51
3.5.5	Conclusion	51
4	Digital Twin Cloud Platform, Data Space and Cyber Security	52
4.1	Overview of Digital Twin Cloud Platform, Data Space and Cyber Security	52
4.2	Cloud Platforms	52
4.2.1	Objectives, challenges and components	52
4.2.2	Detailed description of the activities performed	59
4.2.3	Progress beyond State of the Art or State of the Practice	60
4.2.4	Summary of the key achievements	60
4.2.5	Conclusion	60
4.3	Security and IDS – International Data Spaces	60
4.3.1	Objectives, challenges and components	60
4.3.2	Detailed description of the activities performed	67
4.3.3	Summary of the key achievements	67
4.3.4	Conclusion	69
4.4	Digital Twin API – AAS	70
4.4.1	Objectives, challenges and components	70
4.4.2	Detailed description of the activities performed	71
4.4.3	Progress beyond State of the Art or State of the Practice	72
4.4.4	Summary of the key achievements	73

4.4.5	Conclusion - Next steps after completion of the project	73
4.5	Digital Twin Graph support for Simulation and Cognition	73
4.5.1	Objectives, challenges and components	73
4.5.2	Detailed description of the activities performed	76
4.5.3	Progress beyond State of the Art or State of the Practice	76
4.5.4	Summary of the key achievements	77
5	Sensors, Understanding Sensor Data & Quality Assurance	78
5.1	Objectives, challenges and components	78
5.1.1	Hydro	79
5.1.2	Elkem	81
5.1.3	Saarstahl	84
5.1.4	Sidenor	84
5.1.5	Noksel	85
5.1.6	Sumitomo SHI FW (SFW)	86
5.1.7	Data QA Tools Analysis	88
5.1.8	Data connectors	89
5.2	Summary of the key achievements	90
5.3	Conclusion	90
6	Realtime sensor/data processing - Connecting and Synchronizing Assets and Digital Twins.....	92
6.1	Objectives, challenges and components	92
6.2	Detailed description of the activities performed	92
6.3	Progress beyond State of the Art or State of the Practice	94
6.4	Summary of the key achievements	95
6.5	Conclusion	95
7	Reflection on the use cases – from the WP4 perspective.....	96
7.1	Analysis of requirements from the pilots.....	96
7.2	HYDRO Pilot	96
7.3	SIDENOR Pilot	98
7.4	ELKEM Pilot.....	100
7.5	SUMITOMO SHI FW Pilot.....	102
7.6	SAARSTAHL Pilot.....	103
7.7	NOKSEL Pilot.....	105
8	Demonstrators	111
8.1	Demonstrator of FA ³ ST (Fraunhofer Advanced AAS Tools for Digital Twins) at M42	111

8.2	Demonstrator of FA ³ ST (Fraunhofer Advanced AAS Tools for Digital Twins) at M30	111
8.3	Demonstrator of Building a Digital Twin	111
8.4	Demonstrator of TIA IoT and TIA DATA.....	118
8.5	Demonstrator of Streampipes (integration of numerical model)	118
8.6	Demonstrator of the COGNITWIN Toolbox.....	118
9	References.....	123
10	Annex – COGNITWIN Toolbox Components.....	126
10.1	Toolbox Components - COGNITWIN Interoperability Toolbox	126
10.1.1	End-to-End COGNITWIN Toolbox Pipeline Architecture	126
10.1.1.1	COGNITWIN Toolbox Portal.....	126
10.1.2	Interoperability Orchestration (StreamPipes).....	127
10.1.2.1	Adapt_ST – Nissatech	127
10.1.2.2	TIA STREAM	130
10.1.3	Adapters and Semantic Interoperability (OPC UA ++)	132
10.1.3.1	FUSE OPC-UA.....	132
10.1.3.2	Cybernetica OPC UA DA Server	134
10.1.3.3	TIA DATA Industrial Big Data Analytics.....	135
10.1.3.4	TIA STORAGE	136
10.1.3.5	Grafterizer-SDQ – for Sensor Data Curation	137
10.1.4	Big Data Pipelines Deployment	140
10.1.4.1	Big Data Pipelines Deployment Framework.....	140
10.2	Toolbox - Cloud Platform, Data Space, Security, Digital Twin.....	142
10.2.1	Cloud Platform.....	142
10.2.1.1	Bedrock Platform / Toolbox – Pipeline and Components	142
10.2.1.2	TIA IOT: Industrial Internet of Things Platform	145
10.2.1.3	TIA SENSOR.....	148
10.2.1.4	TIA PLC.....	149
10.2.1.5	TIA CONTROL.....	149
10.2.2	Security and IDS – International Data Spaces	150
10.2.2.1	Trusted Factory Connector (IDS)	150
10.2.2.2	IDS Connectors - SINTEF	152
10.2.3	Digital Twin API – AAS	155
10.2.3.1	FA ³ ST – Fraunhofer Advanced AAS Tools for Digital Twins	155
10.2.4	Digital Twin Graph support for Simulation and Cognition	159

- 10.2.5 Outlier detection OD-tool..... 161
- 10.3 Toolbox Components- Realtime sensor/data processing 163
 - 10.3.1 Real-time data preprocessing with complex event processing:..... 163
 - 10.3.1.1 StreamPipes Siddhi-Processor..... 163
 - 10.3.1.2 Editor for CEP patterns..... 165
 - 10.3.2 Video and image sensor data & processing 167
 - 10.3.2.1 Honir 167
 - 10.3.2.2 Lodur..... 170
 - 10.3.3 Data preprocessing..... 171

List of Figures

Figure 1: Cognitwin approach for digital twins	18
Figure 2: High-level architecture to integrate DTs with sensors connected to assets.....	20
Figure 3: DT integration with IDS	21
Figure 4: Initial COGNITWIN conceptual architecture	22
Figure 5: COGNITWIN approach based on the integration of DTs and StreamPipes.....	22
Figure 6: Big Data and AI Pipeline and the European AI and Robotics Framework	24
Figure 7: Big Data and AI Pipeline Architecture – Applied for Digital Twins	25
Figure 8: COGNITWIN Digital Twin pipeline related to RAMI 4.0.....	26
Figure 9: COGNITWIN Digital Twin Pipeline related to the BDVA Reference Model	27
Figure 10: COGNITWIN Digital Twin Pipeline related to the DAIRO/Adra Framework.....	28
Figure 11: COGNITWIN Toolbox with components for the pipeline steps and D4.4 focus in red box..	30
Figure 12: Digital Twin - Data Acquisition/Collection.....	31
Figure 13: Digital Twin Data Representation	32
Figure 14: Digital Twin Hybrid and Cognitive Analytics Models.....	33
Figure 15: Digital Twin - Action/Interaction, Visualisation and Access	34
Figure 16: COGNITWIN Toolbox Web Portal – with Digital Twin Pipelines and Components	35
Figure 17: Snapshot from Teknopar COGNITWIN Toolbox	36
Figure 18: Architecture of StreamPipes – showing adapter libraries for OPC-UA, MQTT and REST ...	39
Figure 19: Connecting Microsoft Azure IoT Edge to Cloud support Azure Data Lake and IoT Data Hub	54
Figure 20: Digital platform at the Edge – Microsoft Azure	54
Figure 21: Digital platform in the Cloud – Microsoft Azure	55
Figure 22: Existing Digital Platform architecture – including Microsoft Azure architectural components	55
Figure 23: Existing Digital Platform architecture – including SAP architectural components	56
Figure 24: Pipeline architecture of the TIA IOT from Teknopar	57
Figure 25: Architecture of the Bedrock open source based pipeline from SINTEF.....	59
Figure 26: International Data Spaces connecting different platforms.....	63
Figure 27: High-level overview of the GAIA-X architecture.....	65
Figure 28: Combination of IDS technologies and GAIA-X architecture – with COGNITWIN annotations.	66

Figure 29: Internal Connector architecture pattern – from IDS Connector component (SINTEF)	67
Figure 30: The architecture using Admin Shell IO	68
Figure 31: The architecture using Eclipse BaSyx	68
Figure 32: Setup of example AAS	69
Figure 33: The generic four-layer software architecture of SINDIT along with various technologies deployed in the stack [WAS+22]	75
Figure 34: Overview of the SINDIT components instantiated for manufacturing environment [WAS+22]	75
Figure 36: Left: Subset of time-series showing thermodynamic model predictions (red) and acoustic predictions (blue). Right: Scatterplot of acoustic vs thermodynamic predictions.....	87
Figure 37: <i>UML class diagram of FA³ST Service Data Model and AAS Metamodel explaining linking of asset connections with submodel elements</i>	93
Figure 38: <i>FA³ST Service architecture w.r.t. asset connections</i>	93
Figure 39: <i>Protocols delivered with FA³ST Service and the supported interaction patterns</i>	94
Figure 40: Hydro existing Digital Platform with Trusted Data Layer	97
Figure 41: IoT architecture for the Hydro pilot showing interconnectivity of data and toolbox components.....	98
Figure 42: Cognitwin Pipeline in Sidenor use case	99
Figure 43: Pipeline for acyclic data.....	100
Figure 44: Pipeline for cyclic data.....	100
Figure 45: Planned Digital Twin pilot for the Elkem pilot.....	101
Figure 46: Image analytics pipeline for the Elkem pilot	102
Figure 47: Digital Twin pipeline architecture for the SFW pilot.....	103
Figure 48: The Digital Twin – Machine Learning pipeline for the Saarstahl pilot case	104
Figure 49: The Digital Twin – Operational pipeline for the Saarstahl pilot case	105
Figure 50: Digital Twin pipeline for the NOKSEL pilot case	106
Figure 51: Noksel Digital Twin pipeline	108
Figure 52: Kafka – Spark – PostgreSQL connections	108
Figure 53: MQTT to AAS to NodeRed	109
Figure 54: Use of AAS data in pipeline	109
Figure 55: Data from StreamPipes to dashboard	110
Figure 56: AAS model for DT in JSON format (shortened version)	113
Figure 57: DT service with a standardized model and standardized interfaces	114

Figure 58: Connected DT 114

Figure 59: Extension of StreamPipes with DT-specific components 115

Figure 60: A pipeline example 115

Figure 61: DT integrated with the StreamPipes pipeline 115

Figure 62: Data-souverain DT 116

Figure 63: UI in IDS-Apps in the customer IDS connector 116

Figure 64: Overview and software architecture of the demonstrator 117

Figure 65: COGNITWIN Toolbox Portal access 118

Figure 66: Digital Twin Data Acquisition Components 119

Figure 67: Digital Twin and Data Space Components 120

Figure 68: Digital Twin Analytics – for Machine Learning and First Principles Models 121

Figure 69: Digital Twin Visualisation and Control Components 122

Figure 70: Developed pipeline (arrows represent data flow) 128

Figure 71: Displayed notification 129

Figure 72: Visualization of outputs of Keras Neural Network (Brick Degradation Class), Task Duration (Time Between Heats) and Sidenor Measurements Simulation (Ladle Information, Kwh_rr) 129

Figure 73: *A Sample Pipeline for Cassandra* 131

Figure 74: A sample pipeline for Fiware 132

Figure 75: FUSE fuel characterization during one week CFB operation 133

Figure 76: Coupling of the PROFINET subnets with the PN/PN Coupler 146

Figure 77: PLC definition on OPC, and Sample Tag List Defined 146

Figure 78: Pipeline for OPC -> MQTT -> Kafka in JSON 147

Figure 79: Pipeline to demonstrate Kafka data saved on to Cassandra database 147

Figure 80: Interaction between technical components of IDS Reference Architecture Model 152

Figure 81: The basic principle of ellipsoidal peeling. Removal of few outliers can improve the performance of identification significantly 162

Figure 82: Siddhi wrapper that enables users to utilize CEP inside StreamPipes element 164

Figure 81: The basic principle of ellipsoidal peeling. Removal of few outliers can improve the performance of identification significantly 172

List of Tables

Table 1: Communication – challenges, requirements and solutions	43
Table 2: Cloud platform – challenges, requirements and solutions	52
Table 3: CyberSecurity – challenges, requirements and solutions	61
Table 4: Data lake and storage – challenges, requirements and solutions.....	61
Table 5: Digital Twin Models – challenges, requirements and solutions.....	69
Table 6: Sensors – challenges, requirements and solutions	78
Table 7: Sensors and data sources used in the Hydro pilot model	80
Table 8: Sensors and data sources used in the Elkem pilot model	82
Table 9: Sensors and data sources used in the Sairstahl pilot model	84
Table 10: Sensors and data sources used in the Sidenor pilot model.....	84
Table 11: Sensors and data sources used in the Noksel pilot model	86
Table 12: Sensors and data sources used in the SFW pilot model.....	87
Table 13: Novel components for COGNITWIN pilots	90
Table 14: Real time sensor data processing – challenges, requirements and solutions.....	90
Table 15: Overview of WP4 components in use and (possible use) by pilots.....	96
Table 16: Bedrock Toolbox components per 1 Jan 2021	143

Acronyms

AAS	Asset Administration Shell
AI	Artificial Intelligence
APICS	Aluminium Production Control System
CAD	Computer-Assisted Design
CEP	Complex Event Processing
CFB	Circulating Fluidized Bed
CFD	Computational Fluid Dynamics
CMOS	CMOS - Complimentary Metal-Oxide Semiconductor
CO	Carbon Monoxide
CSTR	Continuous Stirred-Tank Reactor
DT	Digital Twin
DAQ	Data Acquisition
DCS	Distributed Control System
EAF	Electric Arc Furnace
FPGA	Field Programmable Gate Array
FTIR	Fourier-Transform Infrared
GTC	Gas Treatment Center
HF	Hydrofluoric Acid
IDS	International Data Spaces
IoT	Internet of Things
IR	Infrared
ICPV	Industrial Control Panel and Visualization
JSON	JavaScript Object Notation
LSTM	Long Short-Term Memory
MEWMA	Multivariate Exponentially Weighted Moving Average
ML	Machine Learning
MPC	Model-Predictive Control
MQTT	Message Queuing Telemetry Transport
OPC-UA	OPC Unified Architecture

PPBM	Pragmatism in Physics-Based Modelling
SDO	Service Data Objects
SWP	Spiral Weld Pipes
TMLL	Teknopar Machine Learning Library
UKF	Unscented Kalman filter
WER	Word Error Rate

1 Introduction

1.1 Scope and purpose

The deliverable D4.4 is a result of all WP4 tasks. It defines the final specification of the COGNITWIN platform, sensor and data interoperability toolbox. It is accompanied with demonstrators that show how a subset of components play together. The demonstrators can be seen in the COGNITWIN YouTube video channel <https://www.youtube.com/@cognitwin9786/videos> and are also linked to from the digital twin pipelines and the various toolbox components in the COGNITWIN Toolbox portal, <https://cognitwin.github.io/toolbox/>.

The deliverable serves as a common project reference across all the technical work packages (WP1-WP5).

1.2 Relevance to other WPs and current and previous deliverables

The deliverable describes the main concepts and functionalities to be offered by the COGNITWIN toolbox. As such it is relevant not only to the technical deliverables of the project, but also to the use-case deliverables where the requirements and the usage of the components is described. There are two deliverables, which are the most closely related to the present deliverable. Specifically:

- Deliverable D4.3 “Updated Platform, Sensor and Data Interoperability Toolbox” contains the description of COGNITWIN components as of month 30. This deliverable D4.4 refines and extends D4.3 by providing further technical details and explaining new implemented functionalities. It can be considered as the final step in the specification and implementation of the COGNITWIN components.
- Deliverable D5.4 “Final Hybrid AI and Cognitive Twin Toolbox“, which has the same development timeline as D4.4, provides specification of the ML, hybrid and cognitive services. By providing the services for platform, sensor and data interoperability toolbox, D4.4 enables the integration of WP5 intelligent services into cognitive digital twins.

Deliverables D4.4 and D5.4 have been written in an aggregated way to be complete without a need for the reader to fully read the previous initial toolbox descriptions in D4.3, D5.3, D4.2 and D5.2. The goal is to provide a sound basis for implementing and integrating the COGNITWIN toolbox and its components. For some of the elements that have not been worked on in the last period (due to issues with Scortex), more details can be found in the previous deliverable D4.3 and D4.2.

1.3 Structure of the deliverable

The rest of the deliverable is structured as follows:

- Section 2 following this introductory paragraph, provides a high-level overview of the COGNITWIN toolbox. It discusses the integration considerations and justifies the decisions made. It presents the current version of the COGNITWIN architecture and explains its evolution.
- Section 3 illustrates the concepts and the components of the COGNITWIN toolbox needed to achieve interoperability among the components as well as with the external systems.

- Sections 4 discusses the needs and requirements for the cloud platforms, data spaces and cyber security and clarifies how it will be realized. Additionally, it provides an update to the specification and implementation of the AAS-complaint digital twin API.
- Section 5 focuses on the selection and installation of sensor hardware in all pilots that have requested new components.
- Section 6 is devoted to description of the COGNITWIN real-time processing services with image analytics and FPGA support.
- Section 7 reports on the application of the technical results in the COGNITWIN use cases.
- Section 8 lists the COGNITWIN demonstrators and provides links to them.

Finally, the deliverable includes a list of the various specifications of the COGNITWIN components in Annex – COGNITWIN Toolbox Components.

2 High level overview

2.1 Integration and Interoperability considerations

The goal of WP4 is to provide methods and tools for digital twins. A digital twin is a digital replica of a physical asset (e.g. sensor, machine, system, etc.) that captures attributes and behaviors of that asset. It is typically materialized as a set of multiple isolated models that are either empirical or first-principles based.

In the context of the COGNITWIN project, the digital twins should not be developed from scratch, but rather the already existing components/systems should be considered. Additionally, the new services to model the behavior of a digital twin have already been developed by different partners. However, the partners use different technologies, develop components in several programming languages, use different protocols, etc. In previous WP4 deliverable (D4.1) we have already identified a list of the components to reuse or extend. The components are of a different granularity level (e.g. from storing a data in a database over covering one phase in big data value chains, until realizing a whole (I)IoT system). These components form the COGNITWIN toolbox, which on the other hand should be open for any change (e.g. adding a new component or updating an existing one).

Just having a list of components is not enough to realize use cases. Both the already existing components and the components that were developed in COGNITWIN should be made interoperable, wherever it is appropriate. However, there are no predefined pipelines, i.e. the way the components are combined is application specific.

To support interoperability between the heterogeneous components, two options are possible:

- Integrate each component with all other components – this means that $n*(n-1)$ interfaces must be implemented. In addition, each new component added would require the implementation of interfaces to all components. Any change to the interfaces of a component would require the change of all interfaces already implemented for that component. Consequently, not only integration between components would be a time-consuming and labor-intensive activity, but maintenance would also be very difficult.
- Define standardized interfaces for each component. This would mean that n interfaces should be developed. The disadvantage is that a wrapper should be implemented for each component. However, no change is required when a component is changed.

We have chosen the 2nd option. As a framework for integration and orchestration of the COGNITWIN components we decided to use the StreamPipes¹ framework. The rationale for this choice is explained in Section 2.2.1.

Combining toolbox components in pipelines is not enough to realize the COGNITWIN vision. There is a need to develop digital twins. The digital twins should not be proprietary solutions but rather should be developed based on standards. Therefore, we decided to build digital twins based on the Asset Administration Shell, mainly as this standard is focused on the industrial domain. The detailed justification is included in 2.2.2.

¹ <https://streampipes.apache.org/>

The consequence of this decision is that the digital twin API and the StreamPipes pipelines which model the behavior of a physical asset have to be integrated. The proposed solution is explained in section 2.3.

The proposed approach is shown in Figure 1.

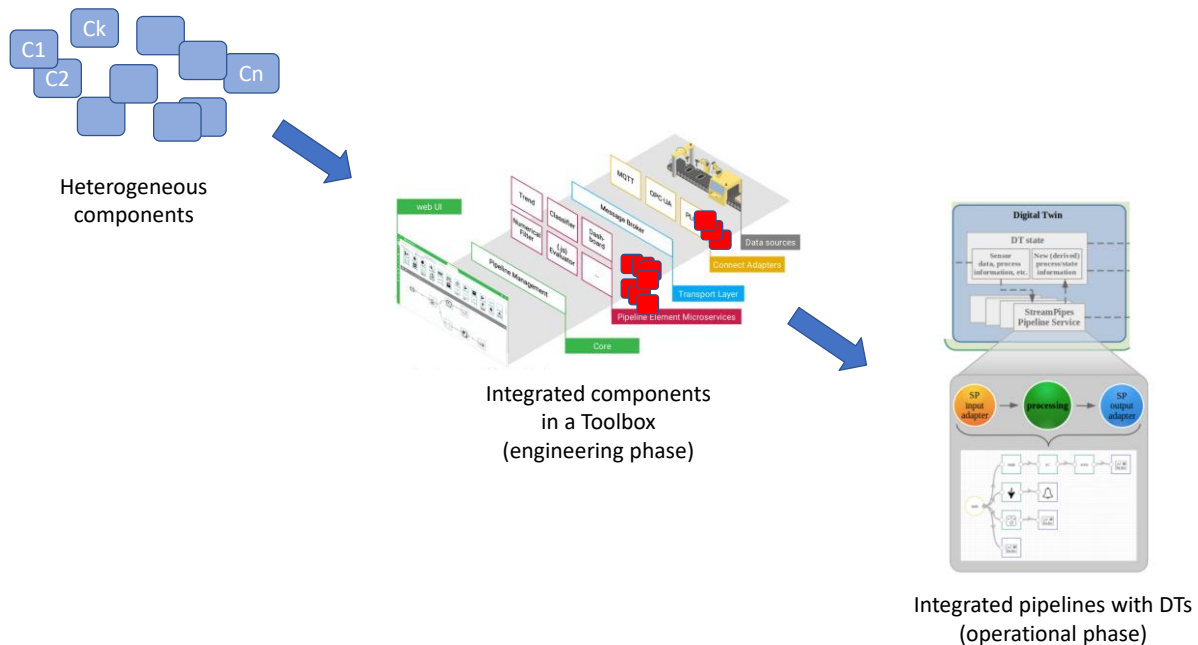


Figure 1: Cognitwin approach for digital twins

2.2 Design decisions

2.2.1 StreamPipes for orchestration of the components

Although both Node-RED and Apache StreamPipes enable the graphical creation of processing pipelines in the sense of a "pipes and filters" approach, a closer look reveals various differences between them, both in terms of the target user group and technical level.

On a technical level, Node-RED is described as a solution for "low-code programming". The focus is on the development of data-driven applications by means of a graphical user interface for users with low programming skills or little programming effort. This means that during creation there must be a basic understanding of underlying data structures, e.g. the structure of the JSON objects used for transmission. In contrast, Apache StreamPipes is aimed at business users without programming skills. The interface for creating pipelines abstracts from the underlying data structure and thus requires less technical understanding. While Apache StreamPipes uses a semantic description level, for example, to automatically hide non-compatible data fields of an input data stream in a data processor and thus reduce user errors, Node-RED uses a purely syntactic manipulation of the parameters at the level of JSON documents.

The differences at the technical level are most apparent in the implementation of the runtime environment. Node-RED relies here on a Node.js runtime environment, i.e. a so-called single host runtime processes the incoming data of a processor. Apache StreamPipes uses a wrapper architecture, which allows the development of a pipeline element in different execution environments. Currently, in addition to a lightweight wrapper for edge nodes with low processing power, wrappers for scalable

Big Data systems such as Apache Flink and Apache Kafka also exist. One advantage of this architecture is that the execution layer can also be realized in different programming languages. A Python wrapper for Apache StreamPipes is currently also available as a prototype, which e.g. simplifies the use of ML-based processors.

Besides that, StreamPipes can be more seen of an end-to-end toolbox with the pipeline editor as just one module out of several modules that aim to support non-technical users in analyzing IIoT data. Therefore, other modules are available to easily connect data, to visualize data and to explore data.

2.2.2 Asset Administration Shells for Digital Twins

The success and usefulness of DTs depends heavily on standardized interfaces to interact with the DTs. We investigated and compared the following state-of-the-art digital twin and IoT standards: Asset Administration Shell, W3C Web of Things and Digital Twin Definition Language, NGS-LD, OData and OGC SensorThings API. The detailed analysis is published as open access [Jau+20]. Based on this work we decided to use the Asset Administration Shell (AAS) standard to implement DTs in this project. Although the AAS standard is (partially) still work in progress, we deem it best suited for this project because of its strong focus on and background in industry. To our knowledge, it is the only available DT standard that has this strong connection to industry and we are convinced that this is essential to be recognized and become well-accepted and adapted in real-world industrial plants in the future. The AAS standard is also the only DT standard that explicitly supports industry-typical protocols and data formats like OPC UA and AutomationML. The fact that it is still work in progress seems acceptable considering that this applies to all DT standards, i.e. there is no complete and "ready-to-use" standard available to this time as the domain of DT (in industry) is still rather new.

Within this project we are implementing relevant parts of the AAS standard alongside its development. We currently focus on the implementation of executable DTs which we refer to as AAS Services as well as a repository to register, manage and discover running AAS Service called AAS Registry. Our vision is that this set of DT tools could be extended into an open source software ecosystem for easy creation, import/export, deployment, and management of DTs (according to the AAS standards). Therefore, we are designing the software with future extensibility in mind. This is done by introducing technology-agnostic interfaces wherever possible, e.g. for de-/serialization (to support de-/serialization using different data formats), data storage and access (to support different kind of databases), and network protocols (to support integration with any kind of existing or proprietary physical devices).

2.2.3 Support of all different Big Data Types and different pipelines and OPC

The strategy for the use of orchestration and connections through the use of StreamPipes and Digital Twin representations through AAS might not be optimal in all situations. StreamPipes and AAS is initially aimed at supporting IOT and Telemetry data, and structured data, including APIs for services and events. It is initially not aimed at supporting media data like video and camera (RGB, Infrared, HF laser etc.). It is further also not aimed at supporting Natural Language Processing with Speech/Audio and/or complex graph structures. Further extensions, or alternative solutions, might be needed to support such data types, which exist in some of the COGNITWIN pilot cases.

Further, a complex system such as a processing plant or factory will essentially be a "System of systems" with "Pipelines of pipelines" and "Twin of Twins". This means that not all pipelines should be best realized as StreamPipes pipelines. In the current plants and factories there is a lot of pre-

existing pipelines that already is working. With well-defined interfaces of components on the pipeline they can be accessed and reused also outside of initial pipelines.

2.3 Conceptual architecture

2.3.1 Current status

One of the main objectives of WP4 is to enable creation of DTs for assets. Figure 2 shows a high-level architecture view how this was realized. The state of an asset is monitored via (internal) sensors. The sensor data is forwarded to the DT and optionally combined with additional external sensors such as ambient sensors for temperature or humidity. A DT exposes a well-defined DT API offering access to properties, services and events. The integration of models, e.g. ML-based or physic-based, is realized by expressing the models as StreamPipes pipelines. Each pipeline can collect relevant data about the state of the DT over time and continuously output new virtual/calculated properties that are made available via the DT API to the outside world. An example for such a virtual property could be the estimated number of remaining heating cycle that a ladle can safely endure.

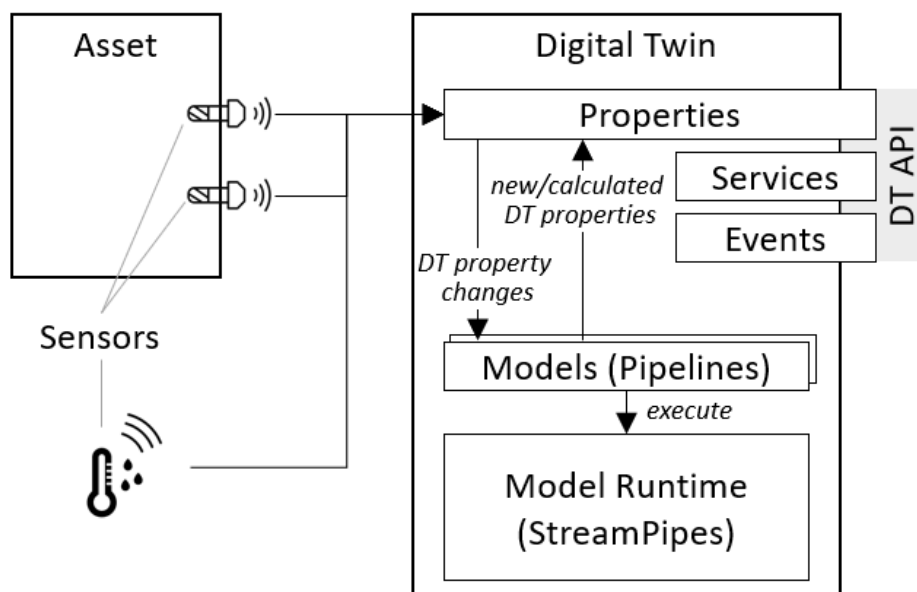


Figure 2: High-level architecture to integrate DTs with sensors connected to assets

A special issue that needs further discussion is how to deal with storage and access of historical data. This is a general issue with the current perception of the DT concept as all DT standards and almost all implementations seem to agree that DTs only reflect the current state of an asset and do not provide any means to storage or access historical data. However, in many use cases this is required, often for legal reasons. Most models typically also require some knowledge about the previous state(s) of the asset to make predictions. For models this is solved by using a stream processing approach (StreamPipes) which allows each model to keep track of historical data as needed. Unfortunately, this does not solve the problem of accessing historical values via the DT API, e.g. historical values of a virtual/calculated property. There are basically two possible approaches; either delegate this to some external system or introduce some kind of data store for historical data to the DT and define new API

operations to access this data. We are currently discussion which approach is best suited for our pilot and also are reaching out to DT SDOs to investigate if such functionality is likely to be added to available standards in the future.

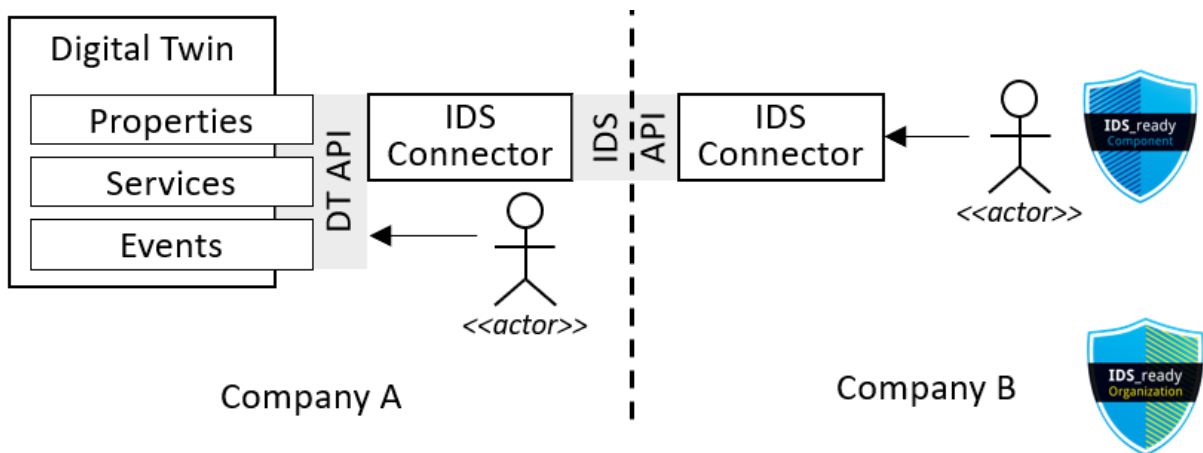


Figure 3: DT integration with IDS

For use and integration of DTs across company borders, we propose to integrate DTs with the IDS to ensure security and confidentiality of exchanged information. Figure 3 shows a conceptual integration of DT into IDS. From inside the company that owns/hosts the DT (i.e. *Company A*), an actor, which can be either human but typically an application, can interact directly with a DT via the DT API. Actors from outside the company, i.e. from *Company B*, will have to use the IDS API to interact with the DT. This requires both *Company A* and *B* to each provide a (properly configured) IDS connector component. Additionally, if the actor wanting to interact with the DT across company boundaries is an application, it needs to be certified to be «IDS ready».

2.3.2 Evolution of the architecture

The COGNITWIN conceptual architecture has been evolved during project execution. We started from the architecture included in the DoW and made it more concrete based on the analysis of the COGNITWIN use cases and the project vision. This architecture was published in [AB2+20] and is shown in Figure 4. The deliverable focuses only on the part in the grey box in Figure 4. The components related to hybrid twins and cognitive twins are described in WP5 deliverables (e.g. D5.4).

We use the DT API to model data, models and services (e.g. Metadata Repository in Figure 4) and to provide standardised interfaces to all these digital twin entities (cf. Access Services and Discovery Services).

Our solution has to be opened for new models and services (of any type). To provide support for both model execution and service registration we decide to use StreamPipes. This framework among many other things enables (i) to register new data-processors, i.e. COGNITWIN services; (ii) to deploy a service and (iii) to run a deployable service that receives data requests over the standardized DT API and connects to the asset. This means that many building blocks shown in Figure 4 (e.g. Service Registry, Pre-processing service, Model Executor) are supported by StreamPipes as well as our extension of it. The data itself or the models can be stored either internally in a digital twin or can be references from it.

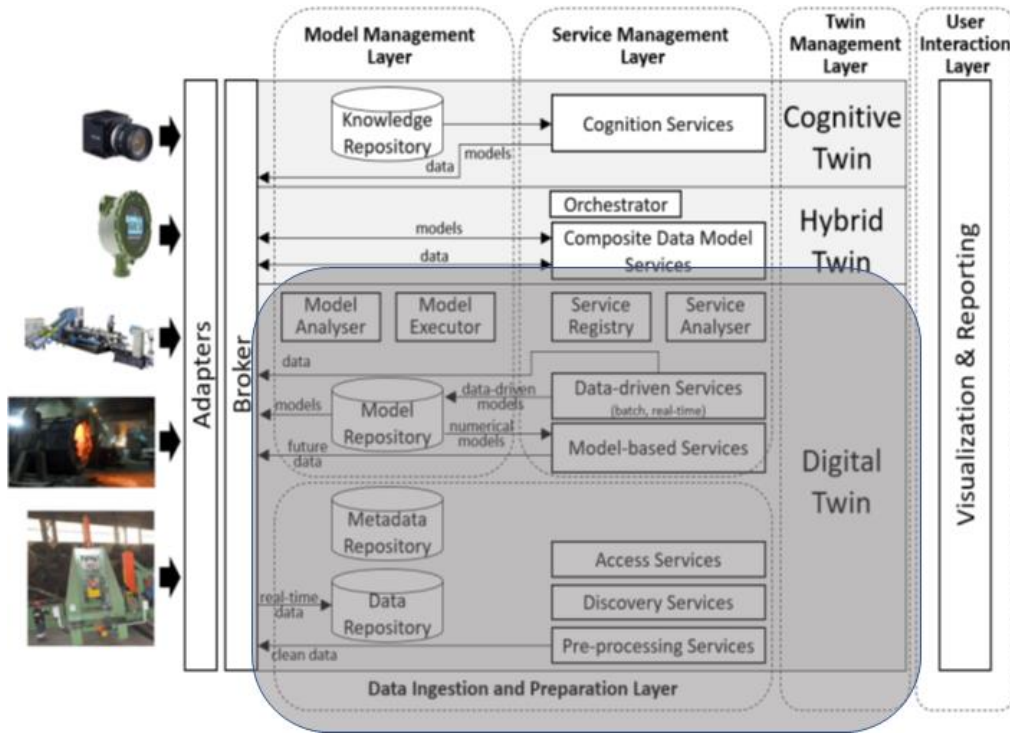


Figure 4: Initial COGNITWIN conceptual architecture

The dependencies between all above mentioned entities are shown in Figure 5. COGNITWIN covers all building blocks for digital twins: whereas AAS-based digital twins are used for the asset management and the standardized interfaces, the StreamPipes provides an environment to host and orchestrate different models, components, services.

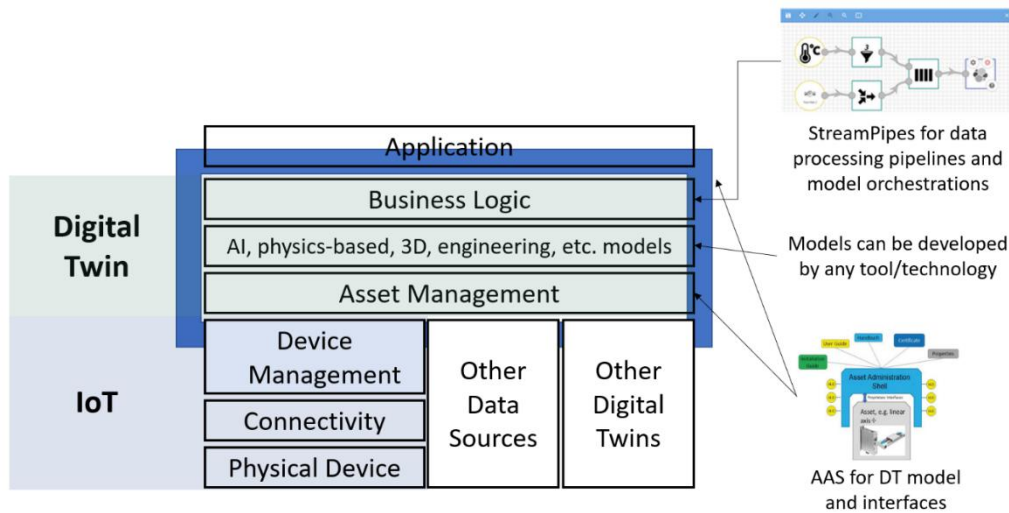


Figure 5: COGNITWIN approach based on the integration of DTs and StreamPipes

3 COGNITWIN Interoperability Toolbox

3.1 Introduction to the COGNITWIN Interoperability Toolbox

The COGNITWIN project objective 4 - COGNITWIN Interoperability Toolbox as a Service – states this to be *A reference architecture* for the cognitive elements including of Big Data, Databases, IoT, Smart Sensors, Machine Learning, and AI technologies that realizes hybrid modelling, self-adaptivity and cognitive recognition, leveraging/extending the existing work into relevant communities. It will be measured through the percent of reuse of the elements of the architecture in the implementation of the use cases.

IEEE has defined interoperability as “the ability of two or more systems or components to exchange information and to use the information that has been exchanged” [IEE90].

A newer definition of Interoperability states: Interoperability is a characteristic of a product or system, whose interfaces are completely understood, to work with other products or systems, at present or in the future, in either implementation or access, without any restrictions [INT21].

The overall task for the COGNITWIN Interoperability Toolbox has been divided into four subtasks that are presented in the following sections.

1. End-to-End COGNITWIN Toolbox Pipeline Architecture
2. Interoperability Orchestration with StreamPipes
3. Adapters and Semantic Interoperability with OPC UA and others
4. Big Data Pipeline Deployment Framework

3.2 End-to-End COGNITWIN Toolbox Pipeline Architecture

3.2.1 Objectives, challenges and components

This subtask includes the specification of an end-to-end architecture of the COGNITWIN Toolbox. We have not aimed at delivering one technically integrated platform for Big Data, IoT & AI/ML technologies, but instead a toolbox. In order to address not only greenfield developments, but also to properly address brownfield integration and interoperability with proprietary solutions and standards, we have taken into account different existing platforms (open source as well as commercial). The different pilot partners already have existing IoT platforms in operation. Such as Microsoft IoT Edge and IoT Hub and/or SAP systems with Enterprise Service Bus/ESB technologies and Data Lakes. Existing Big Data, IoT and AI platforms will thus be considered to be platforms to be interoperable with.

The work is also related to the outcomes of the Industrial Internet Consortium (IIC) Task Groups (“Distributed Data Interoperability and Management (DDIM)” co-chaired by Fraunhofer IOSB and “Digital Twin Interoperability”) as well as the IoT-EPI projects (Task Force “Platform Interoperability”) which some partners are involved in. To realize complex industrial scenarios, Digital Twins (DT) should be capable of capturing characteristics of an asset as specified by the vendor/manufacturer, its state during run time, as well as how an asset interacts with other assets in a complex system. This is usually related to different interoperability problems, which should be resolved using semantic technologies, e.g., semantic models or ontologies.

The next section describes the COGNITWIN project approach with a toolbox end-to-end architecture based on a Digital Twin Pipeline.

3.2.1.1 COGNITWIN Toolbox end-to-end architecture - Digital Twin Pipeline

We have adopted the Big Data and AI Pipeline description that has emerged from work in the Big Data Value Association (BDVA) and the AI Framework for the DAIRO Association, and which have been described by the DataBench project [BER+21], together with a link to the BDVA Reference Model and DAIRO AI framework reference model.

In COGNITWIN we have extended this perspective for the focus of Digital Twins into a new Digital Twin pipeline.

The four step Digital Twin pipeline also has a mapping to the technical areas in the SRIDA for AI, Data and Robotics Partnership [ZIL+20]. In the COGNITWIN project we have further specialized this for the context of Digital Twins as shown in Figure 7.

The COGNITWIN Pipeline approach has been related to the Industrie 4.0 RAMI 4.0 Reference architecture, as shown further below.

The following Framework for Big Data and AI pipelines is based on Big Data Value Association (BDVA) reference architecture. In order to have an overall perspective on Big Data and AI systems.

The Big Data and AI Pipeline Framework is based on the elements of the BDV (Big Data Value Association) Reference Model [ZIL+17]. In order to have an overall usage perspective on Big Data and AI systems a top level generic pipeline has been introduced in order to understand the connections between the different parts of a Big Data and AI system in the context of an application flow. The following figure depicts this pipeline, following the Big Data and AI Value chain.

The BDV Reference Model shown to the right in Figure 6 has been developed by the BDVA, taking into account input from technical experts and stakeholders along the whole Big Data Value chain as well as interactions with other related Public-Private Partnerships (PPPs). An explicit aim of the BDV Reference Model in the SRIA 4.0 document is to also include logical relationships to other areas of a digital platform such as Cloud, High Performance Computing (HPC), IoT, Networks/5G, CyberSecurity etc. The model for the European AI, Data and Robotics framework [ZIL+20] is shown to the left, with the Big Data and AI pipeline from [BER+21] shown in the middle and also in more detail further below.

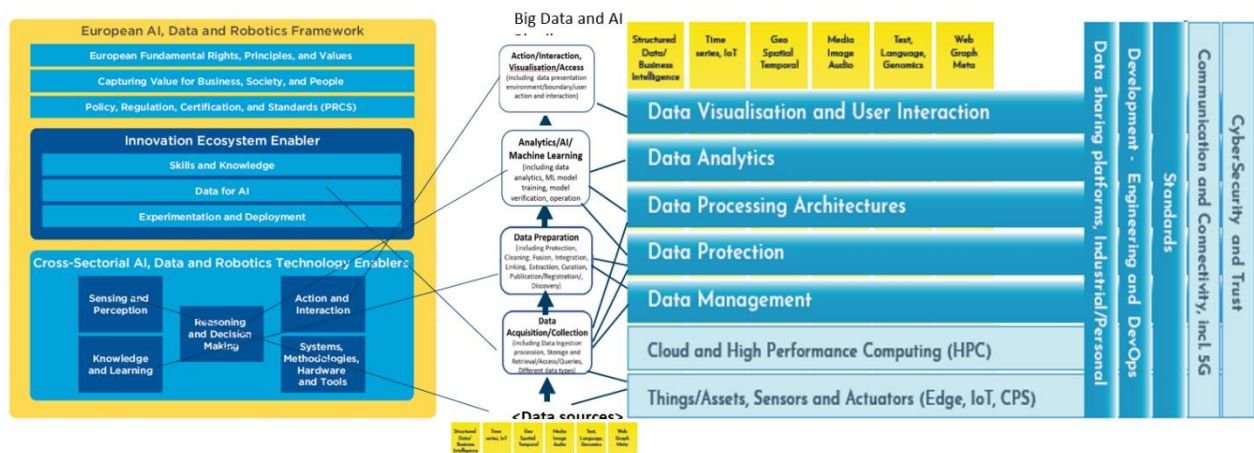


Figure 6: Big Data and AI Pipeline and the European AI and Robotics Framework

The steps of the Big Data and AI Pipeline Framework are also harmonized with the ISO SC42 AI Committee standards. It is in particular harmonized with the steps of Collection, Preparation, Analytics

and Visualization/Access steps within the Big Data Application Layer of the recent international standard ISO 20547-3 Big data reference architecture within the functional components of the Big Data Reference Architecture [ISO20]. It is further also harmonised with the emerging pipeline steps in the ISO SC42 AI standard ISO/IEC 23053 “Framework for Artificial Intelligence (AI) Systems Using Machine Learning (ML)”. This describes a Machine learning pipeline with the related steps of Data Acquisition, Data Pre-processing, Model Deployment and Operation.

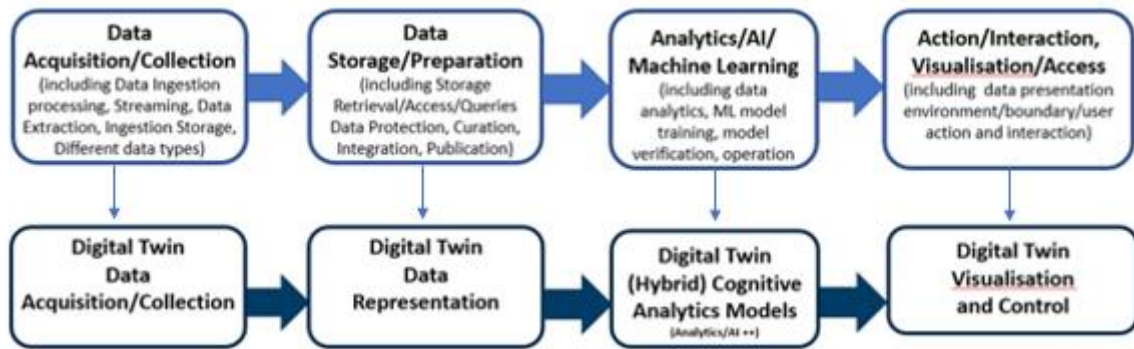


Figure 7: Big Data and AI Pipeline Architecture – Applied for Digital Twins

The proposed pipeline architecture shown at lower part of Figure 7 starts with data acquisition and collection to be used by the DT. This step includes acquiring and collecting data from various sources, including streaming data from the sensors and data at rest.

Following the data acquisition and collection, the next step is the DT data representation in which the acquired data is stored and pre-processed. DT (Hybrid) Cognitive Analytics Models step of the pipeline enables integration of multiple models and the addition of cognitive elements to the DT through data-analytics. Finally, the DT Visualisation and Control step of the pipeline provides a visual interface for the DT, and it provides interaction between the twin and the physical system.

3.2.1.2 COGNITWIN Pipeline architecture related to RAMI 4.0

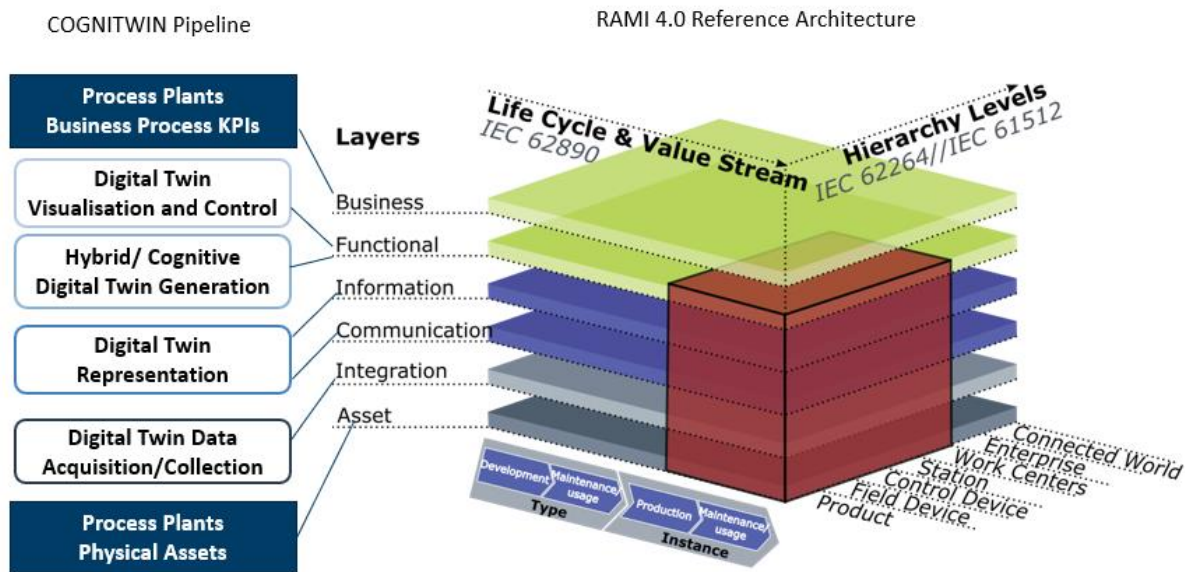


Figure 8: COGNITWIN Digital Twin pipeline related to RAMI 4.0

The most well-known reference architecture in the Manufacturing domain is RAMI 4.0 from the Industrie 4.0 initiative. This is thus a relevant common framework for presenting and discussing different architectural aspects. The COGNITWIN Digital Twin pipeline maps into the Layers of RAMI 4.0 as follows:

- Process plants business processes maps into the Business layer
- Digital Twin Visualisation and Control maps into the Functional layer – interactions
- Hybrid/Cognitive Digital Twins maps into the Functional layer - services
- Digital Twin Representation maps into the Information and Communication layer
- Digital Twin Data Acquisition maps into the Integration layer
- Process plants physical assets maps into the Asset layer

The Hierarchy level dimension of RAMI can be used to analyse the various focus areas between the pilots– with Digital Twin focus ranging from the product through the levels from field and control to the whole factory and supply chains.

In order to achieve the interconnection of the DT from the **Asset layer** through the **Integration layer** with the **Communication layer** with the overall physical system(s) follow non-intrusive and high-performance edge computing approaches and utilizing standard communication approaches (e.g. REST, OPC UA, MQTT) in order to transfer their data in the virtual counterparts within their respective systems.

Furthermore, and following RAMI 4.0, the DTs in terms of the **Information layer** the projects utilize the DTs to serve their own cases, however all aim at predicting or optimizing different parameter / values / variations etc. i.e. they aim to identify and produce valuable information toward enabling a

situational awareness and even more so derive to cognition as to what is important and happening at real- (or near real-) time from the twinned assets.

In the **Functional layer** we observe that the goal is to create (utilizing the DTs) self-aware machines or processes with embedded knowledge towards creating the actual plant decision support system (with cognitive capabilities) for the involved processes and assets of each plant.

Lastly and at the highest layer of the RAMI 4.0, that of the **Business layer**. The business goals that are envisaged to be facilitated by the applications of DTs towards a cognitive plant.

The Life cycle & Value Stream dimension of RAMI can be used to analyse the different focus on the aspect of (Development/Usage – Type/ Instance). A digital twin can further be reflected on various levels in the hierarchy dimension. Typical in COGNITWIN the focus is on the station/work center level but the pipeline model can be adapted to all levels.

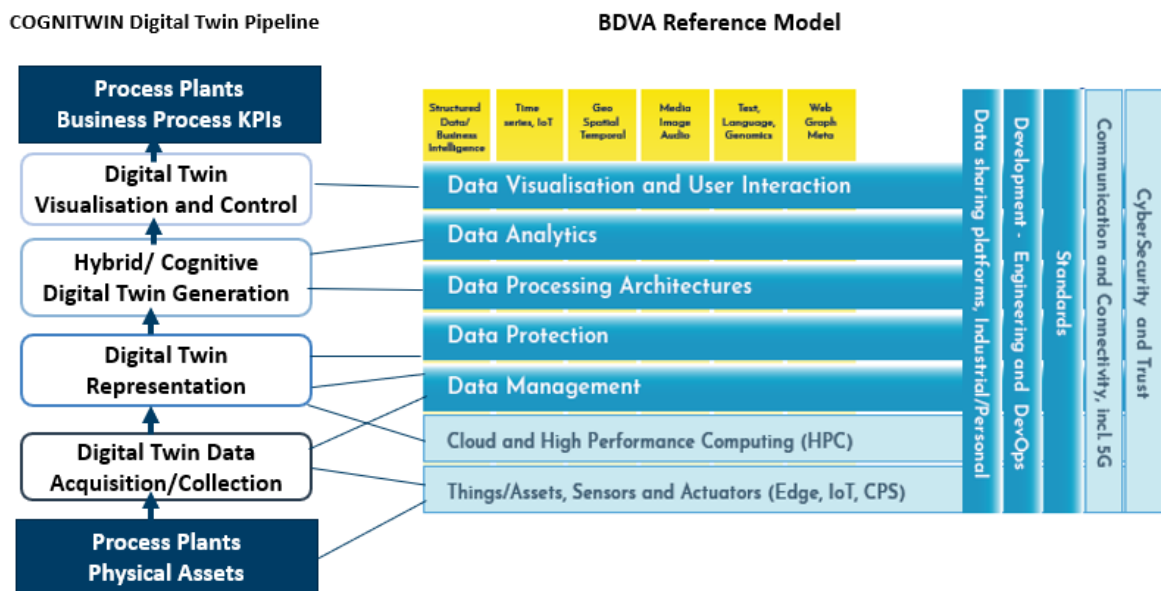


Figure 9: COGNITWIN Digital Twin Pipeline related to the BDVA Reference Model

The COGNITWIN Pipelines reflects the functionalities of the various levels in the BDVA Reference model but combines these into a logical data flow instead of describing these as independent areas – as there is now explicit flow associated with the layers in the BDVA model.

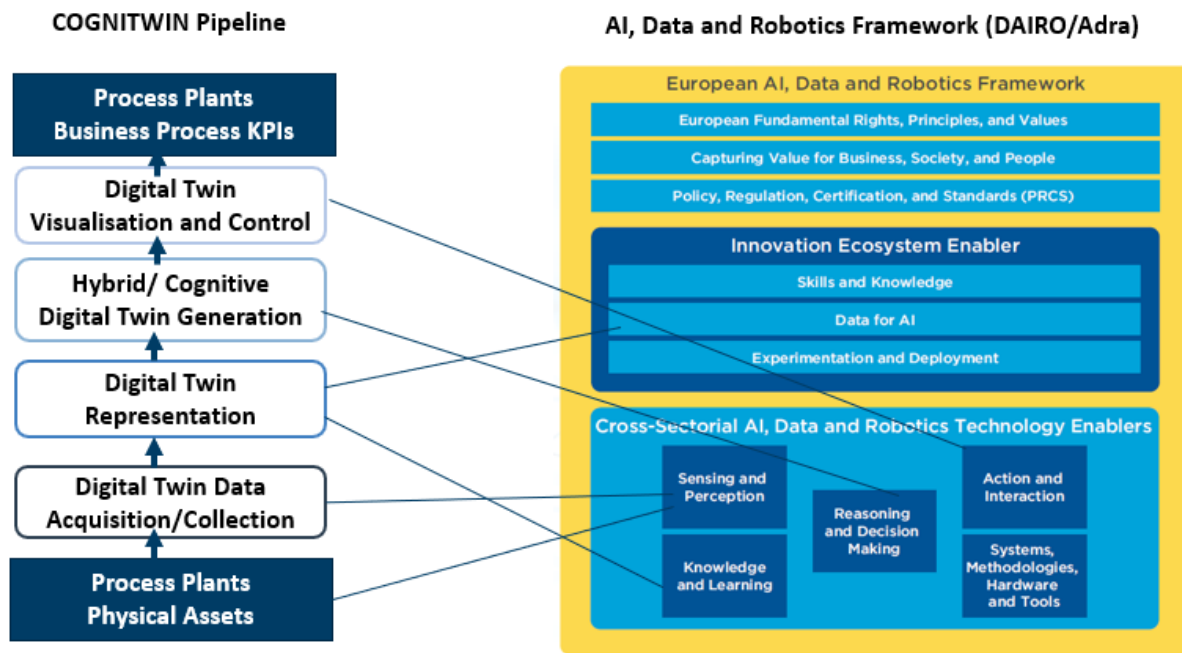


Figure 10: COGNITWIN Digital Twin Pipeline related to the DAIRO/Adra Framework

Properties of a cognitive computing system include support for contextual, adaptive, iterative and stateful and interactive aspects. This can be aligned with the four areas of the European AI SRIDA model and correspondingly to the COGNITWIN Digital Twin pipeline as follows:

- **Sensing and perception** - for Contextual aspects – Understands and extracts contextual elements such as meaning, time, location, process and others based on multiple sources of information.
- **Knowledge and learning/ data (real time /historic)** – for Adaptation: Cognitive computer systems must learn as information/data changes, and as goals and requirements evolve. They must resolve ambiguity and tolerate unpredictability. They must be engineered to feed on dynamic data in real time, or near real time. In relation to context, this characteristic takes care of feeding on dynamic data and then processing it in order to form the eventual context and come up with solutions or conclusions.
- **Reasoning and decision making** – for Iteration/state: They must aid in defining a problem by asking questions or finding additional source input if a problem statement is ambiguous or incomplete. They must “remember” previous interactions in a process and return information that is suitable for the specific application at that point in time.
- **Action and Interaction** – being Interactive. They must interact easily with users so that those users can define their needs comfortably. They may also interact with other processors, devices, and Cloud services, as well as with people.

The following gives examples related to the various cognition areas.

Sensing and Perception/Context: A number of different sensors are being introduced in the different pilots, typically by use of existing sensors or by retrofitting new sensors into existing/brown field environments. This is including sensors for the various human senses and perception areas such as - touch (with pressure, temperature and vibration), sight (with various kinds of image spectra), hearing (with various kinds of audio/sound/speech), smelling (with various kinds of gas and liquid detectors).

All of the pilots support various kinds of measurement sensors. The concept of soft/virtual sensors is possible to introduce in order to create higher level sensor abstractions.

The COGNITWIN project provides support for RGB cameras with image analytics with associated deep learning for the understanding of the movement of steel bars, and is also using deep learning in the context of infrared cameras for images for high temperature analysis, including possible use of high-performance FPGA hardware for image deep learning-based analysis. Further support is also provided for audio/speech to text analysis based on natural language processing.

Knowledge and learning/ data/Adaptation: Many different data representations are being used to capture and evolve/learn knowledge. With a basis in heterogenous data often collected into data lakes, the use of semantics/ontologies in combination with knowledge graphs is a strategy by many projects, and, like COGNITWIN is using knowledge representation through the graph structures of the asset hierarchy of the Asset Administration Shell/AAS.

Reasoning and decision making: All of the pilots use hybrid models that combine mathematical/physical models with historic data and purely data-driven model approaches. Reasoning and decision making are typically based on analysing and understanding variations, including their root causes, as well as their impacts. The COGNITWIN project has introduced the concept of hybrid digital twins in order to highlight the need to combine multiple models (both physical and data driven) for a digital twin. A hybrid DT extends a "classic" DT by intertwining different models to achieve higher predictive capabilities and thus offer greater potential for industrial applications. Indeed, by supporting symbiosis between models, hybrid DTs will allow for integrative prediction of unusual behaviour rather than 'simple' anomaly prediction. The further evolution to Cognitive Digital Twins extends the hybrid DT by marrying expert knowledge with the power of hybrid analytics. The synergy with expert knowledge makes it possible to find solutions to previously unforeseen situations.

Action and Interaction: The environment context for action and interaction contains both machines and equipment in a control loop context and human operators. Cognitive control implies autonomous interaction with the environments in order to impact the situation towards achieving the goals of the system. The COGNITWIN project also opens for the use of speech to text/knowledge processing in the interactions between human operators and the system.

3.2.1.3 COGNITWIN Toolbox end-to-end architecture

The COGNITWIN project is built around six pilots: **Non-Ferrous pilots** (Aluminium, Silicon) (WP1), **Steel pilots** (WP2) and **Engineering pilot** (Boiler, Furnace) (WP3). These pilots that will be supported by the COGNITWIN Toolbox:

COGNITWIN Platform, Data and Sensor Interoperability Toolbox (WP4). This supports the overall coordination of the COGNITWIN toolbox evolution with a focus on the toolbox support for interoperability and portability across platforms, including cloud support, data sharing and potential exchange through the Industrial Data Space, and further support for sensors and real time sensor data processing.

COGNITWIN Hybrid AI and Cognitive Twin Toolbox (WP5) This part of the COGNITWIN Toolbox establishes the AI/Machine Learning support for Digital Twins, and enhances this with sensor analysis

with deep learning and deep learning performance support. It focuses on establishing the Hybrid Digital Twin and the Cognitive Digital Twin toolbox support.

Figure 11 shows various components in the COGNITWIN Toolbox that can be selected in order to create Digital Twin pipelines in different application settings.

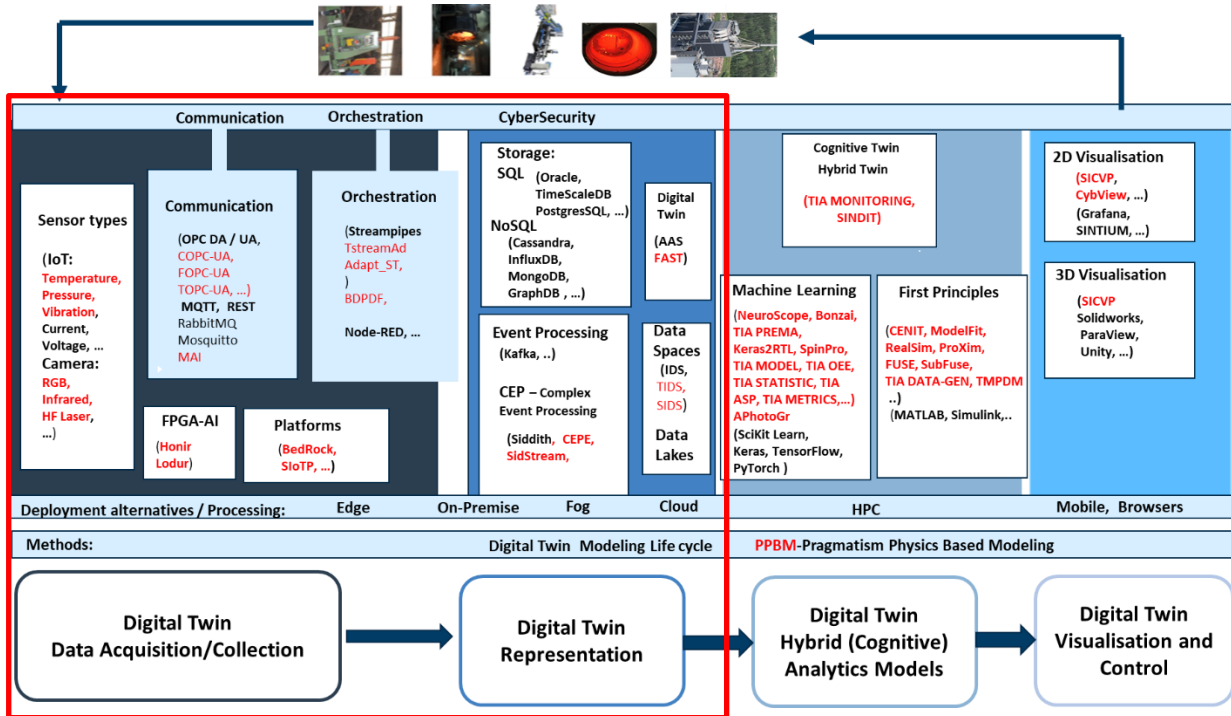


Figure 11: COGNITWIN Toolbox with components for the pipeline steps and D4.4 focus in red box

Figure 11 shows further, in the red box, the areas of the Digital Twin pipeline on Digital Twin Data Acquisition and Representation which are focused on in this D4.4 deliverable document. The areas of Digital Twin Analytics Models and Visualization are further described in the companion D5.4 deliverable document.

Below descriptions of each of the four pipeline steps are provided.

3.2.1.4 Digital Twin - Data Acquisition/Collection

This Digital Twin step maps to *Data acquisition and collection* from various sources, for input to the Digital Twin. This includes both streaming data and data extraction from relevant external data sources and sensors. It includes support for handling all relevant data types and also relevant data protection handling for this step. In the Digital Twin sensor context this includes various data types such as numerical values from sensors, but also images from RGB and Infrared camera sensors, as well as HF laser and others. This step is often associated with the use of both real-time and batch data collection, and associated streaming and messaging systems. It uses enabling technologies in the area using data from things/assets, sensors and actuators to collect streaming data-in-motion as well as connecting to existing data sources with data-at-rest. Often, this step also includes the use of relevant communication and messaging technologies. This step maps to *AI Data Acquisition/Collection* by using enablers from Sensing and Perception technologies, which includes methods to access, assess, convert and aggregate signals that represent real-world parameters into processable and communicable data assets that embody perception. Experiences from the initial pilots of the COGNITWIN project has

shown that the usage of OPC – both DA and UA is much in use for the access to sensor and machinery data. *Digital Twin Data Acquisition and Collection* is taking place from factory machinery and assets with connected devices, and controllers through protocols and interfaces like OPC UA and MQTT.

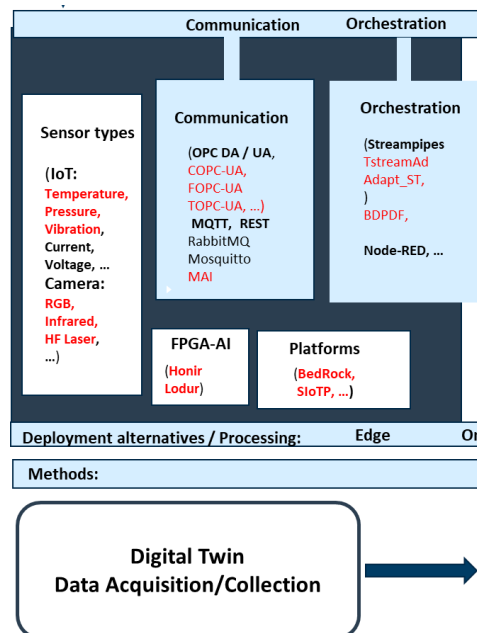


Figure 12: Digital Twin - Data Acquisition/Collection

Figure 12 shows the area of Digital Twin - Data Acquisition/Collection in the COGNITWIN context with relevant technologies for this area.

3.2.1.5 Digital Twin Data Representation

This Digital Twin step maps to *data storage/preparation* by use of appropriate storage systems and data preparation and curation for further data use and processing. Data storage includes the use of data storage and retrieval in different databases systems – both SQL and NoSQL, like key-value, column-based storage, document storage and graph storage and also storage structures such as file systems. Tasks performed in this step also include further data preparation and curation as well as data annotation, publication and presentation of the data in order to be available for discovery, reuse and preservation. Further in this step, there is also interaction with various data platforms and data spaces for broader data management and governance. This step is also linked to handling associated aspects of data protection. This steps map to *AI Data Storage/Preparation* by using enablers from Knowledge and learning technologies, including data processing technologies, which cover the transformation, cleaning, storage, sharing, modelling, simulation, synthesizing and extracting of insights of all types of data both that gathered through sensing and perception as well as data acquired by other means. This will handle both training data and operational data. It will further use enablers for Data for AI which handles the availability of the data through data storage through data spaces, platforms and data marketplaces in order to support data driven AI. With a focus on Digital Twin representation there is a question of which representation approaches to take. In the early survey of Digital Twin API standards and approaches there has been an analysis of various Digital Twin API approaches. *Digital Twin Data Representation* in various forms, based on the sensor and data sources

connections involving event processing with Kafka and storage in relevant SQL and NoSQL databases combined with Digital Twin API access opportunities being experimented with, such as the Asset Administration Shell (AAS).

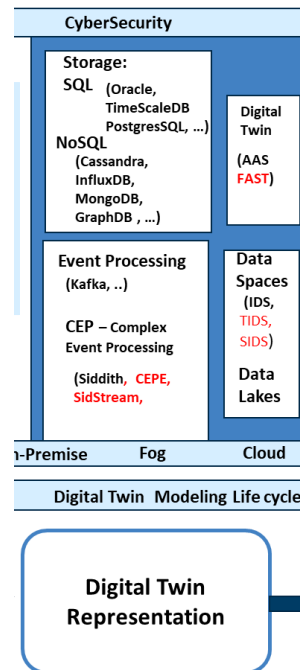


Figure 13: Digital Twin Data Representation

Figure 13 shows the area of Digital Twin Data Representation in the COGNITWIN context with relevant technologies for this area. The COGNITWIN provided technologies (in red) are described later in this deliverable D4.3.

3.2.1.6 Digital Twin Hybrid and Cognitive Analytics Models

This Digital Twin step maps to *Analytics/AI/Machine Learning* that handles data analytics with relevant methods, including descriptive, predictive, and prescriptive analytics and use of AI/Machine Learning methods and algorithms to support decision making and transfer of knowledge. For Machine learning, this step also includes the subtasks for necessary model training and model verification/validation and testing, before actual operation with input data. In this context, the previous step of data storage and preparation will provide data input both for training and validation and test data, as well as operational input data. This step maps to *AI Analytics/AI/Machine Learning*: using enablers from Reasoning and Decision making which is at the heart of Artificial Intelligence. This technology area also provides enablers to address optimisation, search, planning, diagnosis and relies on methods to ensure robustness and trustworthiness. *Digital Twin Hybrid (Cognitive) Analytics* with AI/Machine learning models based on applying and evaluating different AI/machine learning algorithms. This is further extended with first-principles physical models – to form a Hybrid Digital Twin.

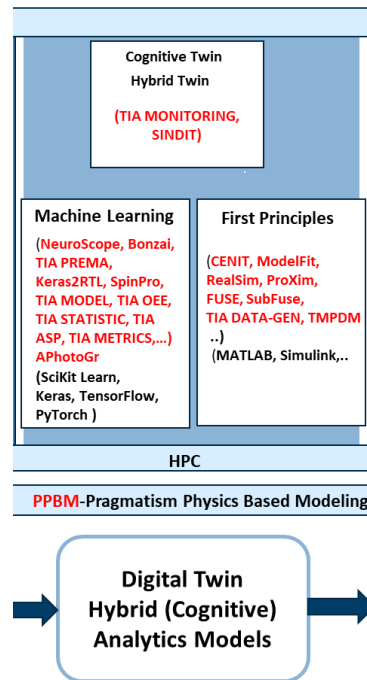


Figure 14: Digital Twin Hybrid and Cognitive Analytics Models

Figure 14 shows the area of Digital Twin Hybrid and Cognitive Analytics Models in the COGNITWIN context with relevant technologies for this area. The COGNITWIN provided technologies (in red) are described in deliverable D5.2.

3.2.1.7 Digital Twin - Action/Interaction, Visualisation and Access

This Digital Twin step maps to *Action/Interaction, Visualisation and Access* (including data presentation environment/boundary/user action and interaction) identifies the boundary towards the environment for action/interaction, typically through a visual interface with various data visualisation techniques for human users and through an API or an interaction interface for system boundaries. This is a boundary where interactions occur between machines and objects, between machines, between people and machines and between environments and machines. The action/interaction with the system boundaries can typically also impact the environment to be connected back to the data acquisition/collection step, collecting input from the system boundaries. This step maps to *AI Action/Interaction, Visualisation and Access*: using enablers from Action and Interaction – where Interactions occur between machines and objects, between machines, between people and machines and between environments and machines. This interaction can take place both through human user interfaces as well as through various APIs and system access and interaction mechanisms. The action/interaction with the system boundaries can typically also be connected back to the data acquisition/collection step, collecting input from the system boundaries.

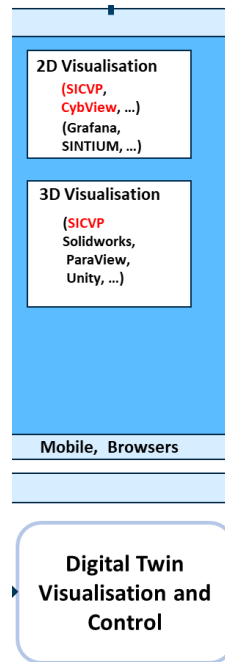


Figure 15: Digital Twin - Action/Interaction, Visualisation and Access

Figure 15 shows the area of *Digital Twin Visualisation and Control*, including the use of 3D models and dashboards suitable for interacting with Digital Twin data and further data access and system control through control feedback to the plant/factory.

3.2.1.8 COGNITWIN Toolbox – Method layer

Besides components, tools and pipelines, the COGNITWIN Toolbox also contains accompanying methods, techniques and methodologies – which includes procedures and guidelines.

In this context there is also preparations for a method for Digital Twin Modeling – in order to provide support for the whole Digital Twin life cycle. This will, among else, be harmonized with Digital Twin life cycle steps as defined in ISO 23247 Digital Twin framework for manufacturing and further emerging standards from ISO SC41 IoT and Digital Twin.

There are also method components associated with other perspectives of the pipeline, explicitly the approach and techniques for first order modelling. In particular there is currently a component PPBM for Pragmatism Physics Based Modelling, described further in D5.2.

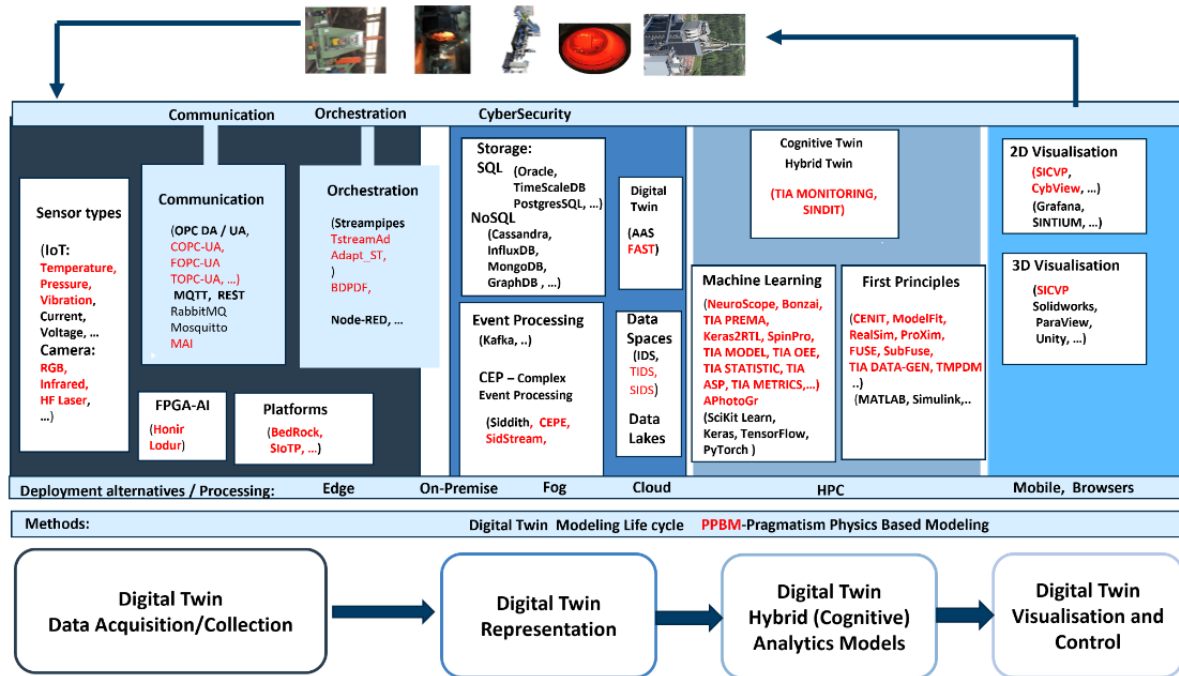
3.2.1.9 COGNITWIN Toolbox Web Portal

The COGNITWIN Toolbox web portal has been created to manage all of the partner components/tools/pipeline that will be included in the COGNITWIN Toolbox. In the final version at the end of the project the description of the potential reusable Digital Twin pipelines for the six different pilots have been focused on.

COGNITWIN Toolbox Portal

The COGNITWIN Toolbox is structured according to a defined Digital Twin pipeline, comprising a set of different Digital Twin (DT) supporting components. These component will typically be connected and configured together in different ways for different pipeline instances in various application contexts.

As shown in the figure, four main steps for the pipeline have been defined, corresponding to 1) DT data acquisition tools & services, 2) DT representation tools & services, 3) DT analytics tools & services, and 4) DT visualization and control tools & services that will be explained below.



Digital Twin Pipelines

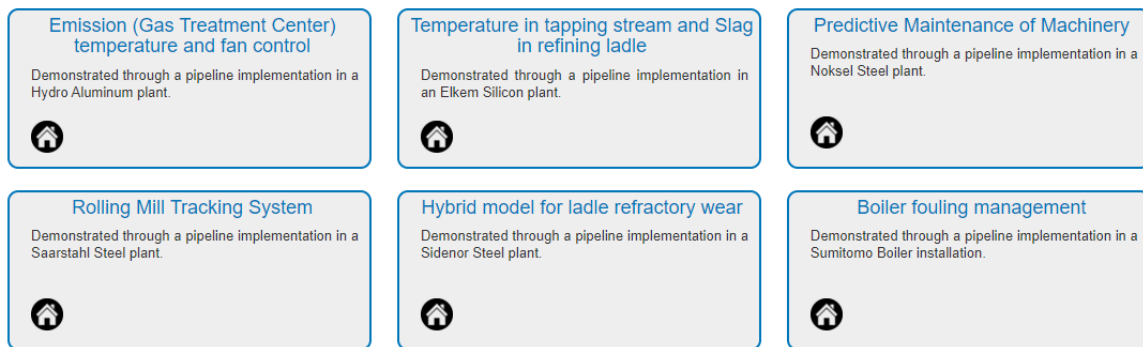


Figure 16: COGNITWIN Toolbox Web Portal – with Digital Twin Pipelines and Components

Figure 16 shows the COGNITWIN Toolbox Web Portal – including components from all areas of the COGNITWIN Toolbox.

The COGNITWIN project is now promoting the following 6 Digital Twin pipelines:

- DT Emission (Gas Treatment Center) temperature and fan control
- DT Refining ladle Tapping Temperature and Slag
- DT Predictive Maintenance of Machinery
- DT Rolling Mill Tracking System
- DT Ladle refractory wear

- DT Boiler fouling management

The COGNITWIN Toolbox contains all the components and related example pipeline configurations both for the COGNITWIN Platform, Sensor and Data Interoperability Toolbox (from WP4) and the COGNITWIN Hybrid AI and Cognitive Twin Toolbox (WP5). The COGNITWIN Toolbox Portal can be found online at <https://cognitwin.github.io/toolbox/>. The Toolbox Portal contains links to the various COGNITWIN Toolbox components and eventually also links to demonstrators and pilot demonstrators, and is being continually updated during the project.

Project partners have also looked into how the Toolbox approach can be used for different market presentations – i.e. Teknopar has created a toolbox portal.

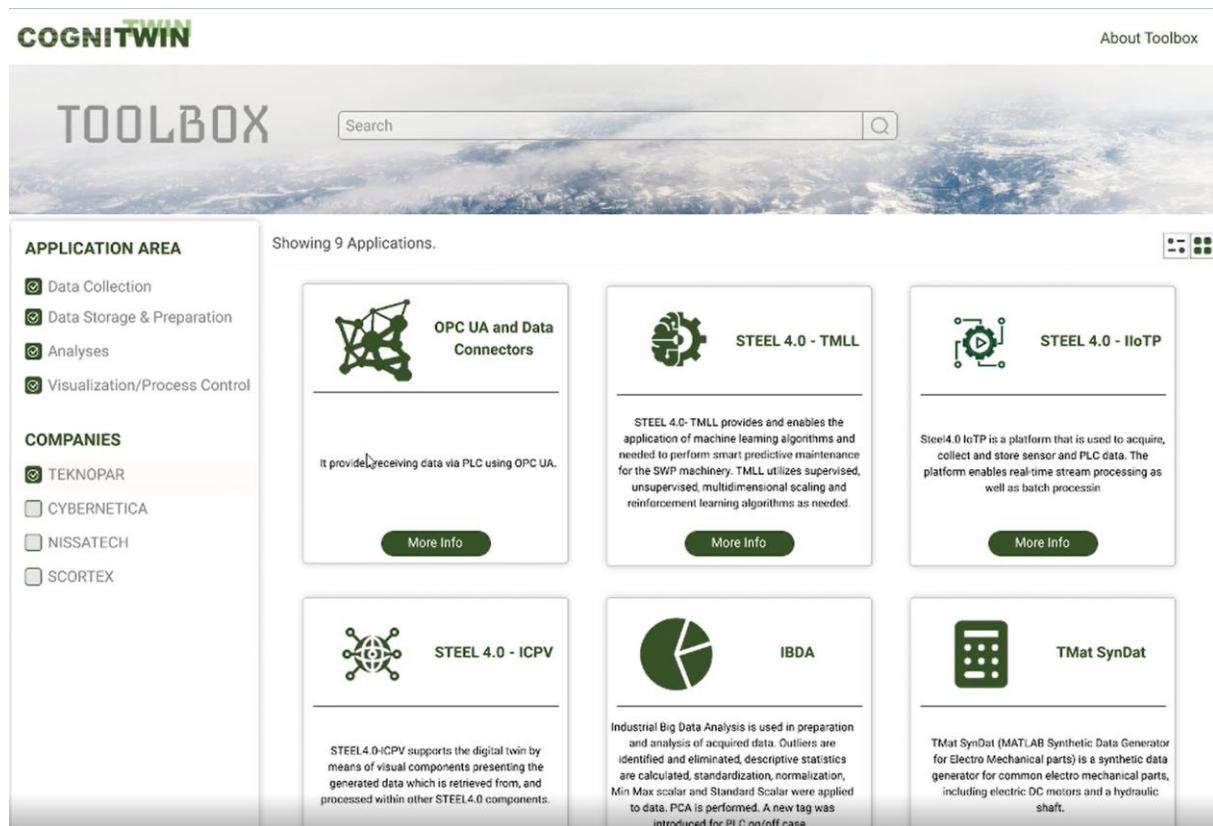


Figure 17: Snapshot from Teknopar COGNITWIN Toolbox

Teknopar has further created a dedicated COGNITWIN Toolbox as part of their marketing portal and prepared this for further promotion of COGNITWIN components as shown in Figure 17. Tool names in the Toolbox can be configured. The names of some of the tools have been changed: STEEL 4.0 - TMML has become TIA MODEL, STEEL 4.0- IIoTP has become TIA IoT, TMat SynDat has been replaced with TIA DATA-GEN, BDVA has been changed with TIA – DATA, and STEEL 4.0-ICPV has become TIA UX.

In the context of collaboration with the 6 SPIRE-06 projects we have also been working on how to share pipeline and components descriptions together with use cases through the IoT-Catalogue <https://www.iiot-catalogue.com/>.

3.2.2 Detailed description of the activities performed

Activities performed include:

1. Analysis of the use of Reference Models from D4.1 document
2. Selection of BDVA and the Big Data and AI Pipeline as a reference model basis
3. Creation of the Digital Twin pipeline framework
4. Establishment of the COGNITWIN overall Toolbox architecture
5. Mappings of the COGNITWIN components into the Toolbox
6. Initial version of the COGNITWIN Toolbox Portal
7. Pipelines introduced into the Toolbox
8. A toolbox portal for TEKNOPAR components have been developed
9. MES integration to the digital twin platform has been performed.
10. Data migration scripts were written and used to migrate data from Cassandra to PostgreSQL.
11. Data preprocessing was performed to fill in and clean data.

3.2.3 Progress beyond State of the Art or State of the Practice

The end-to-end COGNITWIN Toolbox pipeline architecture is based on the technical areas of the BDV Reference Model and the DAIRO / ADRA AI Framework with the associated Big Data and AI Pipeline described by the DataBench project [BER21].

The extension and adaptations of this for Digital, Hybrid and Cognitive Twins is a progress beyond state of the art and practice provided by the COGNITWIN project.

3.2.4 Summary of the key achievements

Key achievements include:

1. Establishment and description of the Digital Twin pipeline steps
2. Mapping of the various COGNITWIN Toolbox components into the pipeline steps
3. Description of the various Toolbox components with a common template
4. Creation of a web-based portal to present the pipeline steps with associated components
5. Population of the COGNITWIN Toolbox with the initial description of COGNITWIN components
6. Pipelines introduced into the toolbox
7. Toolbox portal developed by TEKNOPAR has been populated and the portal is ready for demonstration
8. Evolution the COGNITWIN Toolbox portal
9. Further evolution of the pipeline architecture between relevant components for the use in the various pilots - consolidate and combine the pipeline architectures
10. Further inclusion possibilities of additional components for pipelines – including more references to relevant open source and third-party software that has been used

3.2.5 Conclusion

The COGNITWIN Toolbox portal will continue to be maintained as a collection of the COGNITWIN pipeline implementations for the various pilots, and will have the links to the components and services involved and the partners responsible for providing these.

3.3 Interoperability Orchestration with Streampipes

3.3.1 Objectives, challenges and components

In the context of the COGNITWIN project, the digital twins should not be developed from scratch, but rather the already existing components/systems should be considered.

Based on the components in the COGNITWIN Toolbox the aim is further to make it as easy as possible to support interoperability among components which might be suitable to connect to each other in a pipeline. This will support the ease of reusability and synergy of different components potentially provided by different organisations. The objective of the Interoperability orchestration with StreamPipes is to facilitate more easily the configuration and connections among components in a pipeline ensuring both syntactic and semantic interoperability.

Problem definition

In order to provide a solution for a big data challenge, it is required to combine several components in the form of a pipeline and minimize the trade-off between the efficiency and flexibility. The efficiency refers to the low effort in building a pipeline and flexibility refers to the low effort in extending and rebuilding a pipeline, which are common activities in maintaining a data-driven industry solution.

Indeed, although many organizations recognize Big Data analysis's significance, they still face critical challenges when implementing data collection, data processing and data analytics into their process [Bar+19]. One reason for this is that multiple experts, ranging from technical to domain experts, need to be involved in specifying such complex workflows, and workflow steps need to be mapped dynamically to heterogeneous computing and storage resources to ensure scalability [Ran+17].

Possible Solutions

In order to support interoperability and integration among different interacting components there are many approaches that can be taken. One approach is to ensure that components have well defined interfaces, and also that the interfaces, when possible, is based on established standards.

To support the connections between the output from one component with the input from another components a number of orchestration/workflow support services has evolved.

There is a large variety of Big Data workflow solutions that share similar design principles while fulfilling the needs of various groups of users and use cases. We carried out an analysis of the most promising workflow tools (chosen based on their mass user base and relevance), including Pachyderm², Apache

² <https://www.pachyderm.com>

Airflow³, Snakemake⁴, Apache NiFi⁵, Node-RED⁶, StreamPipes⁷, Argo Workflows⁸, NextFlow⁹, and Conductor¹⁰.

The various frameworks area aimed at different user groups – from analytics/domain experts to system developers. We identified Node-RED and StreamPipes as two orchestration solutions that had good user-oriented graphical pipeline tools.

Although both Node-RED and Apache StreamPipes enable the graphical creation of processing pipelines in the sense of a "pipes and filters" approach, a closer look reveals various differences between them, both in terms of the target user group and technical level.

We have further introduced the design decisions related to StreamPipes for orchestration of components in section 2.2.1 StreamPipes for orchestration of the components. We discussed the different aspects of Node-RED and Apache StreamPipes in the introductory section D2.2. on Design decisions – where we concluded to go forward with a priority for StreamPipes.

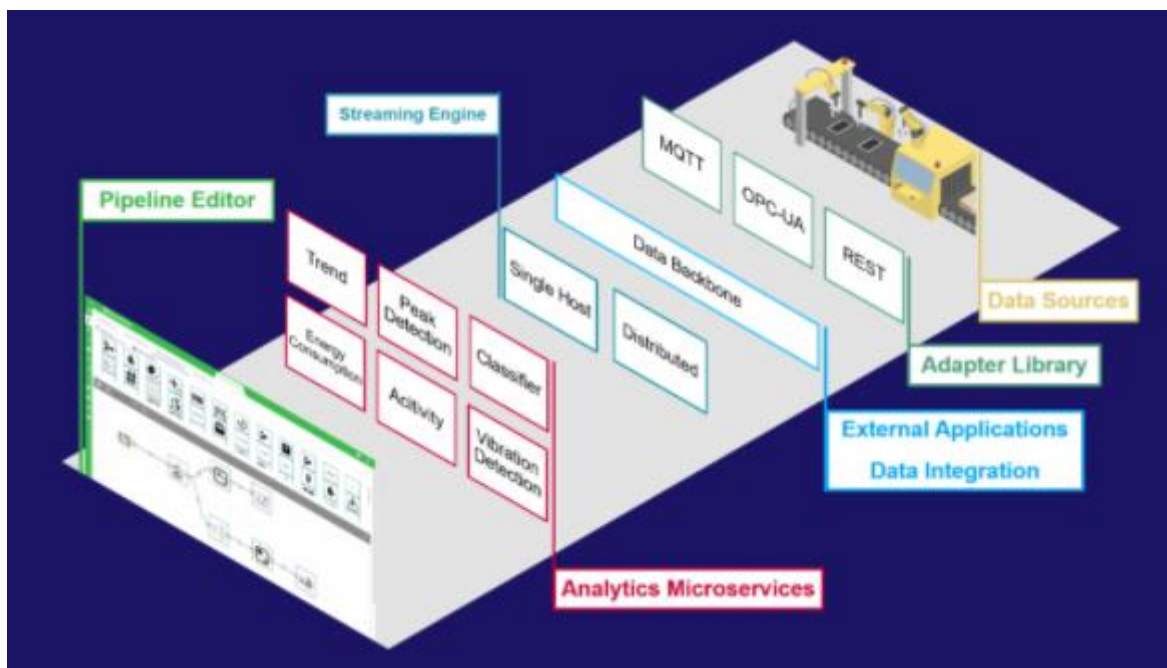


Figure 18: Architecture of StreamPipes – showing adapter libraries for OPC-UA, MQTT and REST

³ <https://airflow.apache.org>

⁴ <https://snakemake.readthedocs.io>

⁵ <https://nifi.apache.org>

⁶ <https://nodered.org>

⁷ <https://streampipes.apache.org/>

⁸ <https://argoproj.github.io/argo>

⁹ <https://www.nextflow.io>

¹⁰ <https://netflix.github.io/conductor>

Figure 18 shows the architecture of StreamPipes, including already existing adapters for OPC-UA, MQTT and REST.

The COGNITWIN project has implemented StreamPipes as the orchestration tool in the Sidenor pilot and the results are very encouraging. There are also experimentations with StreamPipes adapters for the Noksel pilot and the Sumitomo SHI FW pilot.

Independently from a particular pilot, there are two main benefits of using StreamPipes for the orchestration of components:

1. It enables an efficient creation of the end-to-end pipelines.
2. It supports an easy extension of the pipes, or the introduction of new pipelines.

StreamPipes can be categorized as an orchestration engine, which is oriented toward data-intensive applications. It means that it has a very strong support for all phases in data-driven processing, starting from the adapters for various data sources, wrappers for different processing components and solid support for the visualization and reporting.

StreamPipes offers user-friendly interfaces for adding new elements in existing pipelines and incremental debugging of the pipelines. It means that it is possible to combine existing elements (e.g. data-driven models) with the new elements (e.g. numerical models) in various ways, based on how the corresponding pipes are connected. By knowing that many pipelines can exist in an environment, it is easy to develop validation strategies for a combined approach. This is very important for the creation of Hybrid twins, where the way of combining different models is not predefined and a high level of flexibility is required by an orchestration engine.

Pipelines can easily be modified and extended with other elements, both built-in and custom-built, using built-in Editor or combined into one pipeline. Additionally, pipeline elements are encapsulated as standalone microservices, which enables joining elements created by different partners and run on different machines into one pipeline.

We successfully used it to utilize CEP advantages from within StreamPipes. Just an example: we have implemented query which tells us how many heats from the last N (e.g., 5) had more than M (e.g., 100) anomalies:

```
define stream MEWMAOutput(anomalies object);
from MEWMAOutput#window.length(5)[list:size(anomalies) >= 100]
select count() as count
insert into OutputStream;
```

This query uses output from MEWMA element and calculates the number of heats that have more than M anomalies.

Regarding outputs from MEWMA and KNN elements, we singled out following queries that can be applied:

- Test whether there were more than M anomalies in a single heat during time window of length N . For example, if there were more than 10 anomalies in a time window of 5 minutes, raise a warning. (uses output from MEWMA Streampipes element)
- Test whether there were too many anomalies in single heat with same root-cause. For example, if there were more than 100 anomalies with the same root-cause parameter, this

can indicate that some sensor or part of machine is faulty and should be checked. (*uses output from MEWMA element*)

- During last N heats, cycle is classified as **1** (*indicating that the degradation is greater than given threshold*) with an increasing probability (certainty). This can indicate that the ladle will soon be unusable and should be monitored closely or even repaired/replaced. (*uses output from KNN element*)
- Test whether during the last N heats there were more than M anomalies and that cycle was classified as **1** (*indicating that the degradation is greater than given threshold*) - extension to the implemented query in pipeline #2. This situation may lead to even greater process instability. (*uses outputs from MEWMA and KNN elements*)
- Test whether during the last N heats there were more than M anomalies and that cycle was classified as **0** (*indicating that the degradation is lesser than given threshold*) - extension to the implemented query in pipeline #2. In this case it could be better to disregard output of KNN element, as it may be faulty. (*uses outputs from MEWMA and KNN elements*)

These advantages are realized in the component Adapt_ST by Nissatech- Set of adapters for StreamPipes-based Toolkits. Teknopar has created TStreamPipes-Adapters (TIA STREAM) that enable non-experts to create data stream pipes that links to data sinks Cassandra and Fiware.

3.3.2 Detailed description of the activities performed

Activities performed include:

1. Analysis of different orchestration technologies
2. Selection of Streampipes as the most relevant option
3. Proof of Concept: First experiments with Streampipes in pilot pipelines
4. MVP: Implementation of an end-to-end solutions for Sidenor pilot
5. Knowledge reuse: Creation of Adapt_ST - Set of adapters for StreamPipes-based Toolkits by Nissatech
6. Creation of **TIA DATA** (by Teknopar) — Adapters to gets data from data sources (such as Kafka or MQTT) as input. The data is consumed by Cassandra and Fiware and processed via TIA STREAM. The addition of the support for this provides new opportunities for the interoperability with different plant data sources and storage services.
7. Creation of TIA ASP, component enables ML nonexperts to select one or more previously trained ML/DL models of TIA MODEL and execute the selected one on stream data and visualize them to the user.
8. Adding new pretrained models to the TIA MODEL (by TEKNOPAR) - The component enables users of the TIA MODEL to add new pretrained models.
9. In ApacheStreamPipes to be integrated with TIA ASP ,-a new functionality is added to inform user that the new model selected is not a valid model, before the selected model is added to the list of pretrained ML/DL models.

3.3.3 Progress beyond State of the Art or State of the Practice

The Interoperability Orchestration with StreamPipes is supported through the implementation and use of StreamPipes adapters for various technologies. The extension and adaptations of this for integration with Digital Twin components is progress beyond state of the art provided by the COGNITWIN project.

Another progress beyond state of the practice is the creation of the generic pipelines for the tool wear challenges, which are combining processing of product, process and tool/equipment data using StreamPipes orchestration. Main problem in the tool wear challenges is that the (training) data for extreme usage of the tools is missing. The reason is that the usage of a tool stops before it will be broken, so that the data related to breakage is missing. The models which can simulate that destruction process are also missing. Therefore, there is a need for an extensive and intelligent usage of existing data and our approach provides a processing pipeline for an incremental improvement of the understanding how intensive the wearing process is, based on the knowledge learned from past data. Validation is done through the realization of an end-to-end solution for selected industry problems, which has demonstrated the benefits for the creation of complex pipes for tool wear.

Adapt_ST (by Nissatech) - COGNITWIN Toolbox component - Set of adapters for StreamPipes-based Toolkits. Demonstrated through examples from Sidenor – but is also representative examples/templates for StreamPipes adapters that can be used by other StreamPipes adapters in COGNITWIN pipelines.

TIA STREAM, TIA DATA and TIA ASP (by Teknopar) – COGNITWIN Toolbox components – Adapters to gets data from data sources with Kafka as input. The data is consumed by Cassandra and Fiware. The addition of the support for this provides new opportunities for the interoperability with additional plant data sources and storage services.

3.3.4 Summary of the key achievements

Key achievements include:

1. Analysis of different technologies for component orchestration
2. Solution design and identification of relevant technologies
3. Prototype implementation and experimental evaluation – in particular for the use of StreamPipes
4. NissatTech - Creation of various StreamPipe adapters – Adapt_ST
5. Teknopar - Creation of various StreamPipe adapters –TIA ASP
6. Creation of TIA ASP and TIA MODEL, Adding new pretrained models to the StreamPipes Generic ML Component
7. Support development of StreamPipes adapters for relevant components
8. Further integration of StreamPipes in pipeline architectures

3.3.5 Conclusion

Orchestration with Streampipes has been demonstrated to be a good system for orchestration among components in a Digital Twin context. It is a large library of available adapters and new adapters for new environments can easily be made.

3.4 Communication and Semantic Interoperability with OPC UA, FMI etc

3.4.1 Objectives, challenges and components

This subtask addresses the communication and interoperability interfaces for IoT and also links to the IIoT platforms (including use of OPC UA and others) used in pilots in order to ensure efficient deployment in pilots. We assume that IIoT platforms have implemented some of the above-mentioned services and provide open interfaces for 3rd party applications.

Table 1: Communication – challenges, requirements and solutions

Pilots	Communication (by task 4.2) – Challenges, Requirements and Solutions
Hydro	<ul style="list-style-type: none"> • Challenge: Weather data needs to be made available behind firewall on closed process network • Requirement: Allow application running on process network to access internet data from without unacceptable security risk • Solution: Cybernetica OPC UA server, SINTEF Weather API in combination with agreed upon protocol for industry partner (weather service runs on local machine)
Elkem	<ul style="list-style-type: none"> • Challenge: Integration with existing process control system • Requirement: Operators must be able to perform their normal work routine without conflict with the new IR camera equipment. • Solution: Engage external consultant for programming job • Challenge: To avoid to have to many different communication protocols in use, OPC DA is still present. • Requirement: Use OPC UA access whenever possible. • Solution: Use Elkem and Cybernetica OPC UA servers
Saarstahl	<ul style="list-style-type: none"> • Challenge: To handle the video stream data in a suitable way • Requirement: RTSP stream for live video data, JSON via messaging queue/rest service for communication of tabular data. • Solution: For training purposes, video data exchange with technical partner via hard drive.
Noksel	<ul style="list-style-type: none"> • Challenge: Lags in Apache Kafka topics causing delays in data visualisation • Requirement: Real-time • Solution: Apache Kafka topics were grouped based on machine components. Each topic is composed of two partitions. Data coming from each topic is listened by two separate bridges. • Challenge: Some data transferred contained NULL values when the machine is restarted • Requirement: No NULL values transferred • Solution: Transfer of NULL values are prevented via Bridges • Challenge: Missing data coming from MES • Requirements: Data consistency

	<ul style="list-style-type: none"> • Solutions: Interviews with domain experts and thorough analysis of data inconsistencies • Challenge: Interoperability of data • Requirement: Control of air conditioner in the welding cell • Solution: Knowledge map creation and storing ontology in relational data base (ON2RDB)
Sidenor	<ul style="list-style-type: none"> • Challenge: Avoid having to many different communication protocols in use • Requirement: Use OPC UA access when suitable • Solution: Initially to provide data from the Sidenor systems with a possibility to set up further dedicated connections
Sumitomo SHI FW	<ul style="list-style-type: none"> • Challenge: Reliable and timely communication • Requirement: Process operation data needs to be communicated in real time to the analysis/modelling tool environment. Analysis outcomes must be communicated to user/decision maker (eg automation system screen) or directly back to plant DCS (automatic control). • Solution: A basic solution via OPC-UA network connections, alternative solutions in cloud-based systems required. Toolbox component: FUSE-OPCUA

Pilot type	Pilot	Task	WPS5 Toolbox COMPONENTS																																				
			DFKI: Generation of photorealistic data	DFKI: Neuroscope	Nissatech: D2 Lab-C	Nissatech: StreamPipes Sildhfi-Processor	SINTEF: BEDROCK	TEKNOPAR: TIA DATA (TIA STREAM, TIA STORAGE)	CONTROL	TEKNOPAR: TIA APPS (TIA MODEL, TIA PREMA)	TEKNOPAR: TIA UX (TIA DASHBOARD, TIA UX)	TEKNOPAR: TIA ASP	TEKNOPAR: TIA APPS (TIA DATA-GEN)	SINTEF: Open Framework and Tools (SOFT)	Machine learning for hybrid models (SINTEF)	Pragmatic framework for development of hybrid models (SINTEF)	Cybernetica CENT	Cybernetica Cognitive CENT	Cybernetica ModelFit	Cybernetica RealSim	Cybernetica Viewer	Cybernetica Proxim	Cybernetica OPC UA Server	Benzai (Scortex)	Como (Scortex)	Keras / tensorflow (Scortex)	Finite Markov Chains Matlab toolbox (MCPC) (UOULU)	SpinPro (Fraunhofer)	Weather API Interface MAPI (SINTEF)	Correlation Analysis Toolbox (SINTEF)	FUSE Fuel state estimation (UOULU)	SubFUSE Subspace Identification and state estimation (UOULU)	FouMon Fouling monitoring (UOULU)	FouCon Fouling management and control (UOULU)	SINDIT (SINTEF)				
Non-ferrous	Hydro	Plant Digital Twins with ML/AI					(x)										x	(x)	(x)	(x)	(x)																		
	Elkem	Plant Digital Twins with ML/AI					(x)											x	(x)	(x)	(x)																		
Steel	Saarstahl	Plant Digital Twins with ML/AI	x	x																																			
	Sidenor	Plant Digital Twins with ML/AI					(x)			(x)																													
	Noksel	Plant Digital Twins with ML/AI					(x)	x	x	x	x	x	x																										
Engineering	Sumitomo	Plant Digital Twins with ML/AI																(x)	(x)	(x)	(x)																		

In the context of first order models for Digital Twins – the Twin representation is often based on a combined set of equations – often PDE (Partial Differential Equations). For the model representations, exchange and interoperability of these first order models there is a need for other interoperability mechanisms than OPC UA. One of the most popular approaches for this is FMI – the Functional Mock-up Interface.

The Functional Mock-up Interface (FMI) is a free standard that defines a container and an interface to exchange dynamic models using a combination of XML files, binaries and C code zipped into a single file. It is supported by 150+ tools and maintained as a Modelica Association Project on GitHub. The code is released under the 2-Clause BSD, the docs under the CC-BY-SA License.

The latest version of the Functional Mock-up Interface (FMI) standard per February 2023 is version 3.0¹¹ - with the purpose to (a) exchange dynamic models between tools and (b) define tool coupling for dynamic system simulation environments.

*FMI for Model Exchange*¹² - The intention is that a modeling environment can generate a C code representation of a dynamic system model that can be utilized by other modeling and simulation environments. Models are described by differential, algebraic and discrete equations with time-, state- and step-events. If the C code describes a continuous system, this system is solved with the integrators of the environment where it is used. The models treated by this interface can be large for usage in offline or online simulation, or they can be used in embedded control systems on microprocessors.

FMI for Co-Simulation - The intention is to provide an interface standard for coupling of simulation tools in a co-simulation environment. The data exchange between subsystems is restricted to discrete communication points. In the time between two communication points, the subsystems are solved independently from each other by their individual solver. Master algorithms control the data exchange between subsystems and the synchronization of all simulation solvers (slaves).

The two interface standards have many parts in common. In particular, it is possible to utilize several instances of a model and/or a co-simulation tool and to connect them together. The FMI standard is relevant for the interoperability of first order Digital Twin models developed in particular for Hybrid Digital Twins in WP5.

One of the goals of this task will be to provide such an abstraction layer with interoperability interfaces. Based on the experiences among the pilot organisations, and also the general process industry agreement as expressed by the German Industrie 4.0 initiative, OPC-UA is the preferred mechanism for provisioning and access to data related to sensors and actuators.

The implementations differ somewhat, with methods and extensions tailored to the needs of each pilot and data platform. As a result, there are three unique tools developed for this task:

- A server application with a database query extension developed by Cybernetica.
- An OPC-to-MATLAB communication tool developed by University of Oulu.
- A data connector environment for PLC communication developed by TEKNOPAR.

Problem definition

This subtask addresses the interoperability interfaces for IoT and also links to the IIoT platforms (including use of OPC UA and others) used in pilots in order to ensure efficient deployment in pilots. We assume that IIoT platforms have implemented some of the above-mentioned services and provide open interfaces for 3rd party applications.

In the context of first order models for Digital Twins – the Twin representation is often based on a combined set of equations – often PD (Partial Differential Equations). For the model representations, exchange and interoperability of these first order models there is a need for other interoperability mechanisms than OPC UA. One of the most popular approaches for this is FMI – the Functional Mock-up Interface.

¹¹ <https://fmi-standard.org/>

¹² <https://github.com/modelica/fmi-standard/releases/download/v2.0.2/FMI-Specification-2.0.2.pdf>

Possible Solutions

OPC is the preferred interoperability standard for the secure and reliable exchange of data in the industrial automation space and in other industries. It is platform independent and ensures the seamless flow of information among devices from multiple vendors. The OPC Foundation¹³ is responsible for the development and maintenance of this standard.

The OPC standard is a series of specifications developed by industry vendors, end-users and software developers. These specifications define the interface between Clients and Servers, as well as Servers and Servers, including access to real-time data, monitoring of alarms and events, access to historical data and other applications.

The initial aim of OPC was to abstract PLC specific protocols (such as Modbus, Profibus, etc.) into a standardized interface allowing HMI/SCADA systems to interface with a “middle-man” who would convert generic-OPC read/write requests into device-specific requests and vice-versa.

As part of the COGNITWIN Toolbox in particular the partners University of Oulu, Cybernetica and Teknopar have used and extended OPC UA adapters based on the needs in the pilots.

FUSE OPC-UA is a tool enabling the necessary data transfer required for on-line operation of FUSE state estimation tool. This includes transfer of measurement data from plant to the computations in FUSE, FouMon and FouCon components (Matlab) and propagation of outcomes for further use (e.g. to StreamPipes or company asset management services). The FUSE OPC-UA communication tool is based on setting up an OPC-UA server (Prosys Simulation Server) and related OPC-UA clients for communication of data.

Cybernetica OPC UA Server is a general purpose OPC UA server supporting the Data Access (DA) interface. It can be used as a hub for exchanging real-time data from processes with other clients that support OPC UA. The OPC UA server has a plugin API that allows specialized plugins to be developed. These can be used to collect and distribute data from other data sources (like databases, process control systems or simulators).

TIA DATA from Teknopar is used in preparation and analysis of sensor data retrieved from PLC. The purpose of TIA DATA is to generate meaningful information to support digital twin for state estimation and process control. The sensor data, such as temperature, pressure and vibration, voltage, and current are transmitted to MQTT over OPC, and then to Kafka in JSON format. Being a data streaming platform, Apache Kafka transmits real-time data with a low error margin and short latency. Real time data received by Kafka is then transmitted to the Python-based server, where the attribute extraction process is performed.

DataGraft Grafterizer SDQ – for Sensor Data Curation and Quality check (SDQ) – is a possible technology to use for curation and transformation of sensor data. It has not been worked on in the last phase of the project, but is being considered for potential use in the context of sensor data quality analysis.

3.4.2 Detailed description of the activities performed

Activities performed include the following:

¹³ <https://opcfoundation.org/>

FUSE OPC-UA has been developed (designed, implemented, and verified) in 2/2020. The tool originates from solving the WP3 pilot problem on fuel characterization, as a part of the heat exchanger fouling monitoring problem. The communication of data between Matlab – OPC-UA – StreamPipes has been implemented and tested with the FUSE state estimation tool.

Cybernetica OPC UA Server - Some measurements at the Elkem pilot are not available on OPC and are only stored in Elkem's Oracle database. To have these measurements available on OPC, Cybernetica has created a bespoke extension to the Cybernetica OPC UA Server for the Elkem pilot. This extension queries a set of tables in the Oracle database to extract the relevant measurements and publishes them to OPC. This simplifies the employment of the digital twin, as it allows for using OPC as the single data source.

Teknopar TIA DATA has evolved through the following activities: Data has been preprocessed. Data variations and LCL-UCL calculations have been performed on data. Energy consumption data has been monitored and analyzed. For the NOKSEL pilot, the dockerization of the developed components have been performed. Systems resource utilization is measure by Prometheus, Docker Composer file was updated. TEKNOPAR has modified MQTT-Kafka bridge for additional controls, such as transfer of data having electrical current value as zero. OPC UA server used (KEPserverX) has been deployed to the virtual server. MES integration to the Digital Twin has been established. PG-CHAMELEON was used to connect to the MES. The system is dockerized. AAS and IDS related studies have been performed. A data source was developed in Java in order to use the AAS data in the AASX Server in Apache StreamPipes. In order to prove that AAS data can be received on StreamPipes, a pipeline was created that displays the received data on the dashboard screen. AAS Data Source application is dockerized. Within the scope of AAS application, applications were made with Eclipse Basyx and Admin Shell IO components (AASX Package Manager and AASX Server) for hosting and presenting AAS data. A script was written to transfer data from Cassandra database to PostgreSQL database. Existing bridge written in Java is rewritten in Python. Two new bridges were added: Kafka to PostgreSQL and Kafka to Cassandra. A generic, rule-based data transfer in between MQTT to Kafka was implemented.

Migration from timeseries database Cassandra to PostgreSQL was completed. MES integration has been completed. Energy consumption data is visualized. User and role authentication and authorization were determined and developed. Search menu for energy analyzer was developed. Toolbox design and development were started by customizing Apache StreamPipes. As time series database, PostgreSQL was used with TimeScaleDB.

3.4.3 Progress beyond State of the Art or State of the Practice

The Adapters that have been provided are based on the existing OPC UA technologies. The progress in the project has been to adapt the partner OPC UA components of **FUSE OPC-UA**, **Cybernetica OPC UA** and **Teknopar TIA DATA** for the additional technology needs experienced in the pilots, with the needs for additional links for OPC UA connections, such as Matlab, Oracle and also initial link to StreamPipes.

3.4.4 Summary of the key achievements

Key achievements include the following enhancements to partners adapters and OPC UA technologies, related to identification of OPC UA adapter needs emerging from pilots and associated solution design, implementation and experimental evaluation.

- **FUSE OPC-UA** from UOULU has been extended to support Matlab – OPC-UA – StreamPipes links.
- **Cybernetica OPC UA** has been extended for integration with Oracle through queries to extract data for publication to OPC.
- **TIA DATA** from Teknopar has been extended for the links to, and curation of connected data from various sources.
- TEKNOPAR has migrated data in timeseries database to PostgreSQL with TimeScaleDB.
- Further evolution of the OPC UA and adapters based on any further needs of the pilots

3.4.5 Conclusion

OPC UA is the most used communication technology for the data acquisition for digital twins within COGNITWIN, and it is an important part of the COGNITWIN Toolbox. The partners COGNITWIN University of Oulu, Cybernetica and Teknopar have used and extended OPC UA adapters based on the needs in the pilots.

3.5 Big Data Pipeline Deployment Framework

3.5.1 Objectives, challenges and components

The subtask objective is to come up with an approach for effectively and efficiently to deploy and execute Big Data pipelines/workflows on heterogeneous infrastructures (cloud/fog/edge), while at the same time making it possible for various stakeholders to be involved in the design, deployment, and execution of Big Data pipelines.

The challenges are related to modeling Big Data pipelines and optimally deployment and execution of pipelines on available resources, while providing high level of usability and generality.

The components are realized as a software framework addressing various aspects of Big Data pipelines modelling, deployment, and execution.

The Big Data Pipelines Framework is complementary to the StreamPipes approach in the sense that it is focused on scalability aspects related to deployment and execution of pipelines, enabling dynamic scaling of execution of individual steps in the pipelines. While approach like StreamPipes are focused on pipeline modelling/UI and library of operations, the proposed framework is focused on deployment and scaling executions of pipelines (including individual steps granularity) on heterogeneous resources ranging from Edge to Fog to Cloud to HPC resources; the synergy can be explored at the pipelines modelling level, where deployment/scaling aspects (with granularity at individual steps) can be embedded in pipelines specifications.

Problem definition

Big Data pipelines/workflows are composed of multiple orchestrated steps, such as workflow activities that perform various data analytical tasks. They are different from business and scientific workflows since they are dynamic, process heterogeneous data, and are executed in parallel instead of a sequential set of operators. Although many organizations recognize Big Data analysis's significance, they still face critical challenges when implementing data analytics into their process [Bar+19]. One reason for this is that multiple experts, ranging from technical to domain experts, need to be involved

in specifying such complex workflows, and workflow steps need to be mapped dynamically to heterogeneous computing and storage resources to ensure scalability [Ran+17]. Providing a scalable, general-purpose solution for scalable Big Data workflows deployment and execution that a broad audience can use is an open research issue.

The challenges in devising an applicable generalized solution come from the fact that bottlenecks can occur on an individual workflow step level – for example, when the throughput of one step is lower than the others. Thus, scaling up the entire workflow does not address the scalability issues and needs to be done on the individual step level. This issue becomes worse by the fact that scalability needs to be organized and orchestrated over heterogeneous computing resources. Furthermore, scaling up individual steps introduces race conditions between step instances that attempt to process the same piece of data simultaneously. Another major challenge is achieving usability by multiple stakeholders as most Big Data processing solutions are focused on ad-hoc processing models that only trained professionals can use. However, organizations typically operate on specific software stacks, and getting experts in Big Data technology can introduce costs that are not affordable or practical. Even if an organization has the necessary technical personnel, data workflow steps pertain to specific domain-dependent knowledge, which is possessed by the domain experts rather than the data scientists who set up the data workflows. In this respect, this framework proposed here aims to provide an approach that allows conceptualization of Big Data workflows to support the high-level definition of complex data processing across multiple types of parameters, inputs, and outputs, and scalable execution of Big Data workflows. Accordingly, we extracted the following requirements for our framework for supporting Big Data pipelines on heterogeneous resources:

- A workflow definition mechanism with a clear separation between design- and run-time aspects and not limited to a specific technology stack and/or application domain and ad-hoc processing models;
- A workflow run-time support that considers workflows separate units, rather than as a single unit, for individual workflow steps; and
- An execution method with event driven execution method and support for race condition free parallel execution.

Possible Solutions

A review of existing technologies in light of the above-mentioned requirements indicate the following technology areas as relevant element of a possible solution:

- Message-oriented Middleware (MOM) – for achieving race-condition-free consistency and concurrency for scaling the homogeneous Big Data workflow steps by using a synchronization mechanism across the different step instances. MOM provides an infrastructure for loosely-coupled and asynchronous inter-process communication using messaging capabilities. In the context of Big Data workflows, the middleware can act as a medium for communication, whereby step instances coordinate passing intermediate results through sending/receiving messages in MOM queues.
- Software Containers Technology – for deploying distributed scalable applications (such as Big Data workflows), providing transparent means for infrastructure management and easy

scalability of individual application sub-components. Orchestration tools use a configuration file to define container images, network, and related deployment schemes of an application.

- Domain-Specific Languages (DSLs) – for addressing the modeling aspects and usability of Big Data workflow specifications, and offering better domain-specificity, while at the same time improving collaboration between domain experts and developers.

The abovementioned technologies are used as underlying technologies for the proposed framework. The DSL is used for defining workflow steps at a high-level (including dependencies) and composing them into workflows. Storage configurations, data preparation, and step-level data processing and transformation operations are also specified. Individual workflow steps are wrapped as containers, and container orchestration tools are used in managing the heterogeneous resources, allowing workflows step containers to be deployed. MOM is used for storing intermediate and output data and the data exchange mechanism during workflow execution. As workflow steps are deployed across heterogeneous distributed environments, the data exchange mechanism is an essential element that binds the workflow steps together by allowing them to pass data. In this way, where various stakeholders can be involved in the creation, deployment, and execution of Big Data workflows. Domain-experts can be responsible for extracting data processing requirements and structuring the high-level design of workflow steps. Technical experts provide the concrete programmatical implementations of the steps – for example, data scientists may provide workflow step-specific analytical models and data preparation code. Finally, DataOps experts are engaged in deploying and maintaining data workflows in production settings and monitoring data quality and related infrastructure status.

3.5.2 Detailed description of the activities performed

Activities performed include:

- Problem definition related to Big Data pipelines on heterogenous resources
- Conceptual design of the framework
- Identification of relevant technologies, including message-oriented middleware, software containers, and domain-specific languages
- Review of relevant state of the art in Big Data workflows deployment and execution
- Solution design including: framework architecture, first version of the DSL, step design, inter-step communication
- Prototype implementation
- Experiments to evaluate the scalability of the solution
- StreamPipes GUI has been modified to include components developed in the project, Prototype is implemented and used in the NOKSEL pilot., and it can be applied to the other pilots as well.

3.5.3 Progress beyond State of the Art or State of the Practice

There is a large variety of Big Data workflow deployment and execution solutions that share similar design principles while fulfilling the needs of various groups of users and use cases. In comparison with such solutions, our framework aims to be highly usable for the non-technical experts for pipelines deployment, have out of the box run time support for software containers, and be generic.

With respect to the focus on use of StreamPipes for Pipeline connections in the COGNITWIN project this approach is complementary, and we will seek to find a possible synergy between these.

3.5.4 Summary of the key achievements

Key achievements include:

- Conceptualization of Big Data pipelines and their scalable deployment/execution on heterogenous infrastructures
- Solution design and identification of relevant technologies
- Prototype implementation and experimental evaluation
- Establishment of a big data pipeline

3.5.5 Conclusion

The COGNITWIN pilot pipelines have not had an extensive need for specialised big data workflows but some possibilities for future research has been identified and further development of big data pipelines has been continued in other projects like the EU DataFlow project, <https://datacloudproject.eu/>, - enabling the big data pipeline lifecycle on the computing continuum.

4 Digital Twin Cloud Platform, Data Space and Cyber Security

4.1 Overview of Digital Twin Cloud Platform, Data Space and Cyber Security

The overall task for the Digital Twin Cloud Platform, Data Space and Cyber Security has been divided into four subtasks that are presented in the following sections.

- Cloud Platform
- Security and IDSs – International Data Spaces
- Digital Twin API - AAS
- Digital Twin Graph support for Simulation and Cognition

4.2 Cloud Platforms

4.2.1 Objectives, challenges and components

A cloud platform will be used to gather data from plant assets and will be utilized to arrive at real-time operational insights. In addition to infrastructure services that manage the connectivity, authentication, and the edge devices, the cloud level will provide basic services and business services. The basic services are used to ingest data in the cloud, and to clean, integrate and store data that is received. The business services are intelligent, data-driven services, such as data analytics or asset performance management services (to be developed in WP5) to detect trends and/or create insights about assets and to find the root causes of failures.

Table 2: Cloud platform – challenges, requirements and solutions

Pilots	Cloud platform (by task 4.2) – Challenges, Requirements and Solutions
Hydro	<ul style="list-style-type: none"> • Challenge: An Industrial IoT platform is needed • Requirement: For Hydro it is necessary be able to access and use data from the existing Industrial IoT platform. • Solution: Hydro has already established a comprehensive platform for Industrial IoT that is continuously collecting data from thousands of signals coming from machines, sensors, PLCs, and other plant control systems. The raw data is uploaded to a data lake system where it is stored in its original form. Cleansed or aggregated data can also be stored in the data lake whenever this is beneficial. Selected, use case specific data can be prepared and distributed to users through a Trusted Data Layer (TDL).
Elkem	<ul style="list-style-type: none"> • Challenge: An Industrial IoT platform is needed • Requirement: Utilise the existing sensor data collection platform. • Solution: Use the existing databases and associated platform
Saarstahl	<ul style="list-style-type: none"> • Challenge: Use the existing platforms of Saarstahl as the basis • Requirement: Utilise the existing Saarstahl infrastructure • Solution: Embed the solution into the Saarstahl infrastructure
Noksel	<ul style="list-style-type: none"> • Challenge: Data losses due to OPC Server's (KepServerEX) being turned off by the operators • Requirement: No data loss

	<ul style="list-style-type: none"> • Solution: OPC server is moved to virtual servers that are on the same networks the PLC
Sidenor	<ul style="list-style-type: none"> • Challenge: Need to use the existing infrastructure • Requirement: Ensure solutions that do not open up the current protected infrastructure • Solution: Provide necessary data so experimental setups can be realised outside of the existing infrastructure
Sumitomo SHI FW	<ul style="list-style-type: none"> • Challenge: Challenges are in bringing all necessary computational tools to the cloud environment, to communicate online data in to the cloud environment, and securely feed back automatic control information to local DCS. • Requirement: Requirements include robustness and data safety/security. • Solution: Cloud platform solution may be cloud dependent. Basic solution includes OPC-UA

Pilot typ	Pilot	Task	WP4 Toolbox COMPONENTS																			
			IMA1 (SINTEF)	Cybernetica OPC UA (Cybernetica)	FUSE OPC UA (UOLU)	TEKNOPAR: TIA DATA (TIA STREAM, TIA STORAGE)	BD Pipelines DF (SINTEF)	Honir (Scortex)	Lodur (Scortex)	Bedrock (SINTEF)	TEKNOPAR: TIA IOT (TIA SENSOR, TIA PLC, TIA FAST (Fraunhofer)	TEKNOPAR: TIA DATA-GEN	SINDIT (SINTEF)	StreamPipes + Siddhi (Fraunhofer)	CEP Editor (Fraunhofer)	IDS Connectors (SINTEF)	Trusted Factory Connector (Fraunhofer)	COGNITWIN Toolbox Portal (SINTEF)	TEKNOPAR: TIA UX	Cybernetica Viewer (Cybernetica)	Sensor library (SINTEF)	Sensor data quality framework (SINTEF)
	Hydro	Digital Twin Cloud Platform, Data Space and Cyber Security	x	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)
	Elkem	Digital Twin Cloud Platform, Data Space and Cyber Security	x	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)
	Saarstahl	Digital Twin Cloud Platform, Data Space and Cyber Security	(x)	(x)	(x)	(x)	x	x	x	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)
	Sidenor	Digital Twin Cloud Platform, Data Space and Cyber Security	(x)	(x)	(x)	(x)	(x)	(x)	(x)	x	x	(x)	x	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)
	Noksel	Digital Twin Cloud Platform, Data Space and Cyber Security	(x)	(x)	x	(x)	(x)	(x)	(x)	(x)	(x)	x	x	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)
	Sumitomo	Digital Twin Cloud Platform, Data Space and Cyber Security	(x)	x	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)

Problem definition

Many combined edge/cloud digital platforms are already deployed among the COGNITWIN partners, including digital platforms such as Microsoft Azure Edge/Cloud and SAP solutions. It is not relevant to replace these with alternative solutions, but rather to ensure that the contributions through the COGNITWIN Toolbox can enhance and build further on these. The following describes some of the deployed edge/cloud platforms.

The cloud connection is realized by the Azure IoT Edge runtime, which is a collection of programs that turn a device into an IoT Edge device by enabling it to receive code to run at the edge and communicate the results.

The IoT Edge runtime installed on the IoT Edge device is responsible for the following functions:

- Install and update workloads on the device
- Maintain Azure IoT Edge security standards on the device
- Ensure that IoT Edge modules are always running

- Report module health to the cloud for remote monitoring
- Manage communication between downstream devices and IoT Edge devices
- Manage communication between modules on an IoT Edge device
- Manage communication between an IoT Edge device and the cloud

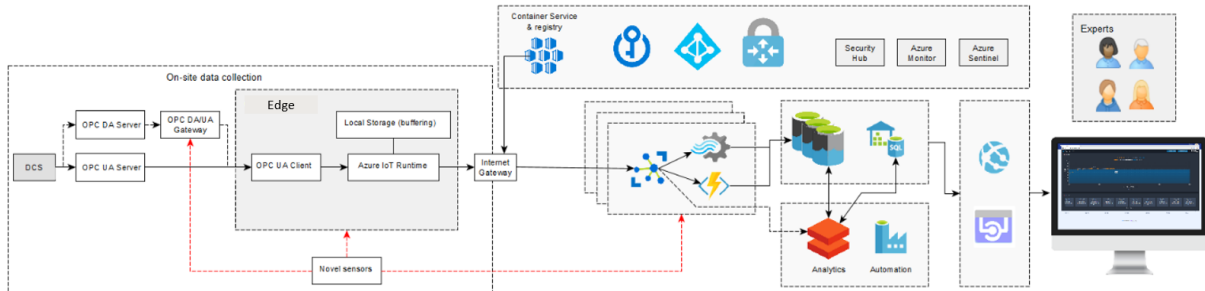


Figure 19: Connecting Microsoft Azure IoT Edge to Cloud support Azure Data Lake and IoT Data Hub

After the edge device has provided the online data to the cloud, there are plenty of possibilities to develop online analytics, as all the tools and components of Microsoft Azure environment are then available (see Figure 19). For example, Azure Data Lake enables the developers, data scientists, and analysts to store data and do processing and analytics across platforms and languages, by means of batch, streaming, and interactive analytics. Another useful Azure compatible component is the Databricks, which can be used to process large workloads of data by fully managed Spark clusters, and can also help in data engineering and data exploring by Machine learning. The visualization tools in the Azure cloud, such as the Time Series Insights, may help in formulating of new kinds of analytics-based flexible process condition monitoring services.

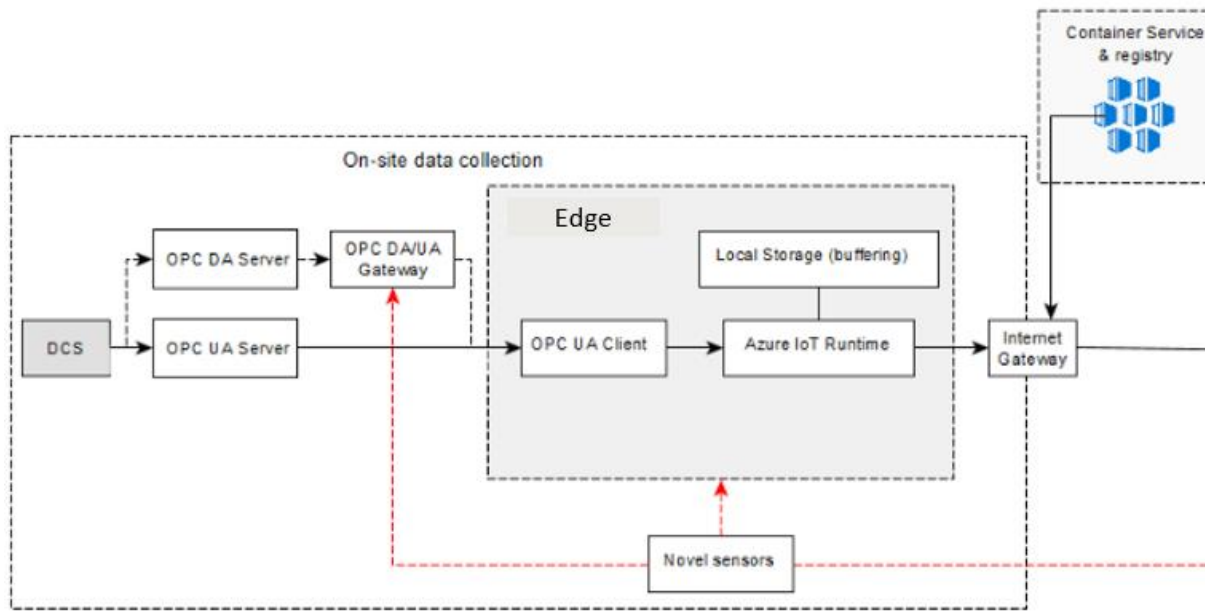


Figure 20: Digital platform at the Edge – Microsoft Azure

Figure 20 shows a more detailed picture of the usage of OPC DA and UA servers connected to the Azure IoT Edge Runtime.

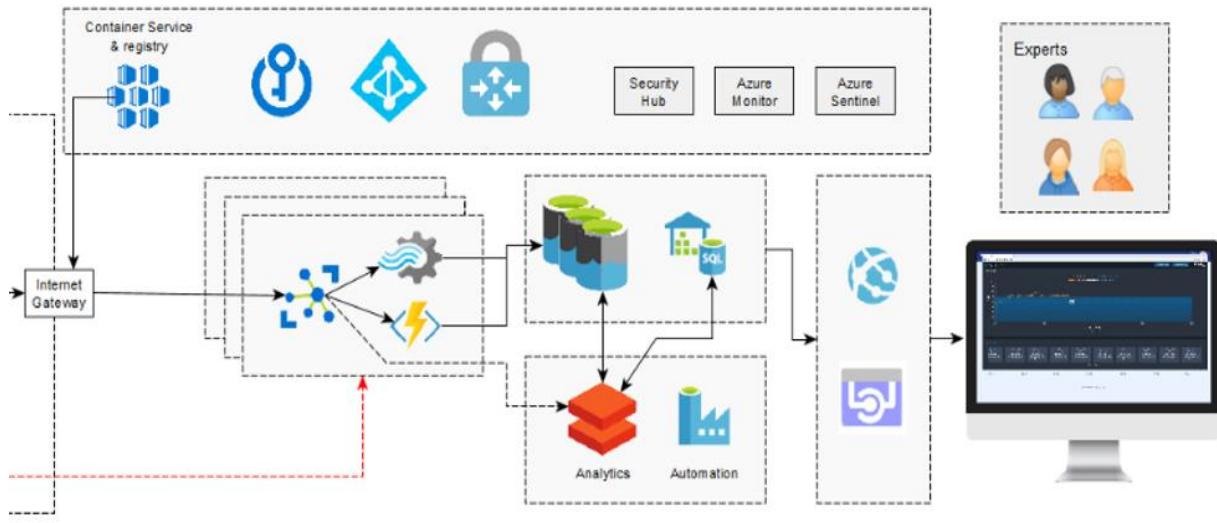


Figure 21: Digital platform in the Cloud – Microsoft Azure

Figure 21 shows a more detailed picture of the various Azure IoT Cloud components with related services and support for analytics processing and automation.

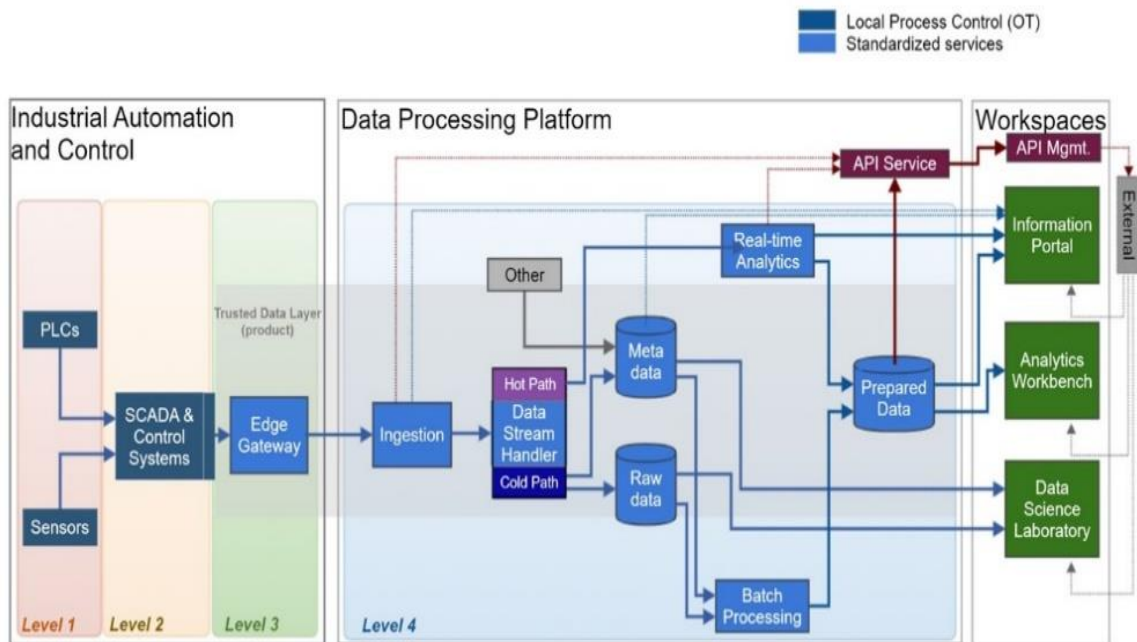


Figure 22: Existing Digital Platform architecture – including Microsoft Azure architectural components

Figure 18 illustrates the architectural setup of a digital platform including Microsoft architectural components – with connections that supports both realtime and batch data access as input to workspaces for data access and analytics.

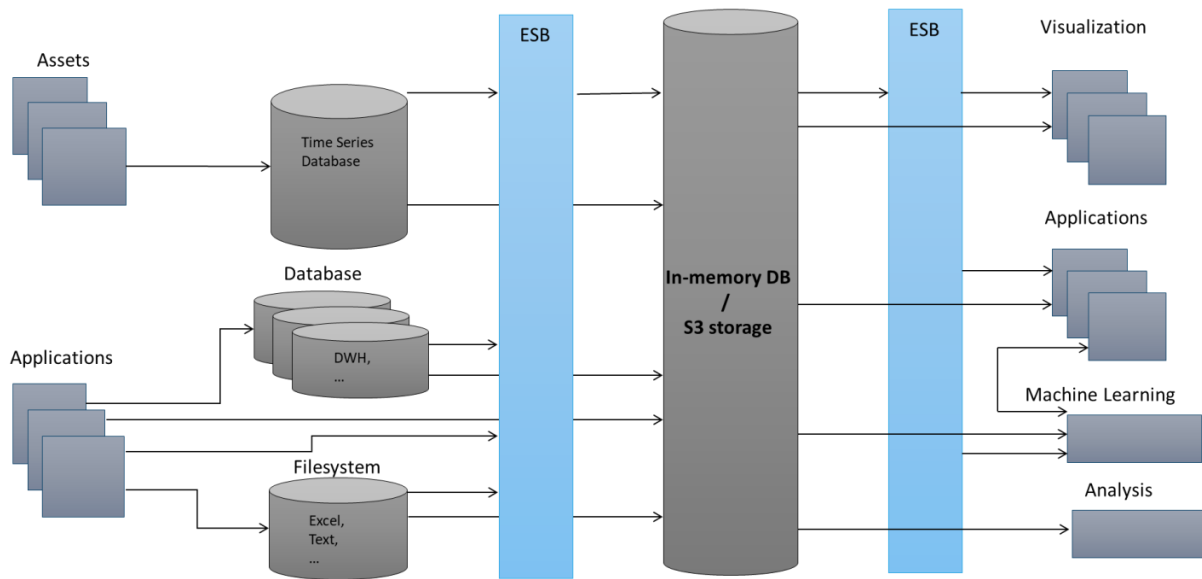


Figure 23: Existing Digital Platform architecture – including SAP architectural components

The figure above illustrates the architectural setup of a digital platform including SAP architectural components – with connections through an Enterprise Service Bus (ESB) – typically with a message communication with the MQTT protocol, and implementations of this with technologies like RabbitMQ, Mosquitto or others.

Possible Solutions

A review of existing technologies in light of the above-mentioned requirements indicate the following technology areas as relevant element of a possible solution:

In this context some of the pilots already have large cloud platforms investments, including in particular the use of the Microsoft Azure Cloud platform, which are being used as part of the infrastructure for sensor data collection – and also SAP.

In addition, we have partners that are developing and providing platforms with sequenced pipelines of components, in particular Teknopar with their IoT Apache based platform (already deployed for the Noksel pilot) and SINTEF with the Bedrock platform (being considered in conjunction with existing platforms for the Sidenor and Sumitomo SHI FW pilots).

Proposed approach: Use the existing data lake platforms of the partners (Microsoft Azure IoT Hub, SAP and others) in conjunction with the components of the COGNITWIN Toolbox – offer additional cloud service/storage when needed – i.e. through the BedRock or IoT open-source components/platforms.

TIA IOT (from Teknopar) is a platform that is used to acquire, collect and store sensor and PLC data. The platform enables real-time stream processing as well as batch processing. TIA AIOT platform provides a drag-and-drop easy-to-use interface and enables non-technical users to easily create stream pipeline elements and pipelines.

PLC data is collected by OPC and later via MQTT is passed to Kafka. Data in Kafka is consumed by Cassandra and PostgreSQL. The data are stored in the Cassandra database with three columns: id, time, and value. The id column shows the component to which the data belong. The JSON format streams of the data transferred to the Cassandra database are presented to users as a log file.

In the context of COGNITWIN, the IoTP output will be useful both in real-time condition monitoring and conducting the predictive maintenance. The Teknopar Industrial Data Security technology has been integrated in the TIA IOT platform.

A fully asynchronous communication structure with the event-bus method is used for the transmission of data collected from the source with OPC. Data transmission is provided in the JSON format. In the architecture managed on the basis of Microservice, Cassandra is used as the NoSQL.

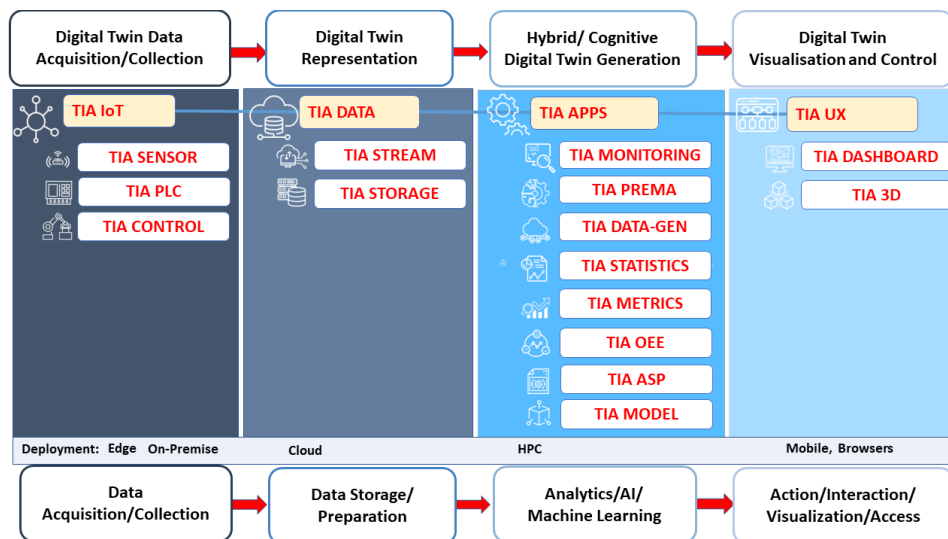


Figure 24: Pipeline architecture of the TIA IOT from Teknopar

In the 2nd period components developed by TEKNOPAR were dockerized. Prometheus software was added to the system in order to measure the resource usage in the system, and therefore the Docker-Compose file was updated. Prometheus shows the system resource consumed by Docker Containers and the total resource consumed in the system instantly via Grafana. PGAdmin and the Portainer technologies that allow us to manage containers have been updated, and hence the Docker-Compose file has been updated. KEPServerEX software was moved to a virtual server. Data filtering was applied, and data was exported in .csv format using PGAdmin interfaces. Data transferred to the servers in the pilot site, was zipped as tar.gz and transferred to the analysis servers.

TIA PLC (from Teknopar) - The purpose of the TIA PLC is to gather data and process data on the PLC. Different PLC's can be coupled together upon requests. TIA PLC is a module that includes the service where PLC selection, programming and installation is made and the software that can read data from the PLC. Within the 26 years of experience we have gained in the industry, the programming of many different brands/models of PLCs, mainly SIEMENS PLCs, and manufacturer-independent, has been carried out. With the module, the PLC software actively used in the field is provided with the optimum update service. If there is no PLC used in the field, it is programmed by selecting and installing PLC in line with the needs. With the TIA PLC software module, it can be directly connected to the PLC installed in the enterprise without any intermediate software and start collecting data. The received data is transmitted to different modules with many communication protocols such as MQTT, TCP/IP. TIA PLC software is easy to install, scalable, monitorable and flexible.

TIA SENSOR (from Teknopar) - It is the service module in which the most suitable sensors for the machines are selected and installed with TEKNOPAR's 26 years of experience in the industry. With the experience in sensor selection, meetings with the machine manufacturer and operators working in the field are held and a high degree of success is achieved with the optimum number of sensors. The connections of the installed sensors to the existing PLC or another device are carried out by using standards and protocols suitable for the relevant system needs.

TIA STORAGE (from Teknopar) - TIA STORAGE is the module that enables the storage of data received from field components such as sensors, PLCs, actuators, devices, MES or machines. Compatible with automation technologies, it can be easily integrated with all kinds of equipment and data sources thanks to its modular structure. TIA STORAGE can use many databases with its direct integration with relational databases, NoSQL databases and time series databases. TIA STORAGE can work distributed. Thanks to this feature, there is no performance loss in the increasing amount of data. The module is easy to install and use. Some databases that TIA STORAGE can use: PostgreSQL

- MySQL
- MS SQL
- InfluxDB
- TimescaleDB
- CrateDB
- Apache Cassandra

TIA STREAM (from Teknopar) - TIA STREAM is the module used to transfer data from storage to related TIA PLATFORM modules or different sources. TIA STREAM is compatible with automation technologies and can be easily integrated with all kinds of equipment and data sources thanks to its modular structure. With the store and forward feature of the module, there is no data loss in connection interruptions that may occur due to hardware or software reasons. TIA STREAM The data received from the source can be transferred to the relevant module/modules in real time. With its optimized structure, there is no delay in data transmission. TIA STREAM supports many communication protocols, is scalable, can be run distributed, easy to install and easy to use.

TIA CONTROL (from Teknopar) - TIA CONTROL is a software module for remote management and control of a machine or process. With the module, it allows instant interventions by sending remote commands to the machines working in the field in case of malfunction or emergency. TIA CONTROL works on a web-based basis. Independent of the environment, it can keep the process under control only with an internet connection. TIA CONTROL can work in integration with other TIA PLATFORM modules. The module is flexible and easy to install and use.

BedRock (from SINTEF) - The BEDROCK Toolbox is a flexible and easily deployable software bundle of modules developed as a platform to be deployed on various servers – with one instance running on local machines and servers at SINTEF Industry. The list of available toolbox modules consists of open-source components and SINTEF in-house developed code. The framework is applied as the foundation for digital twinning R&D activities, process control and data analytics.

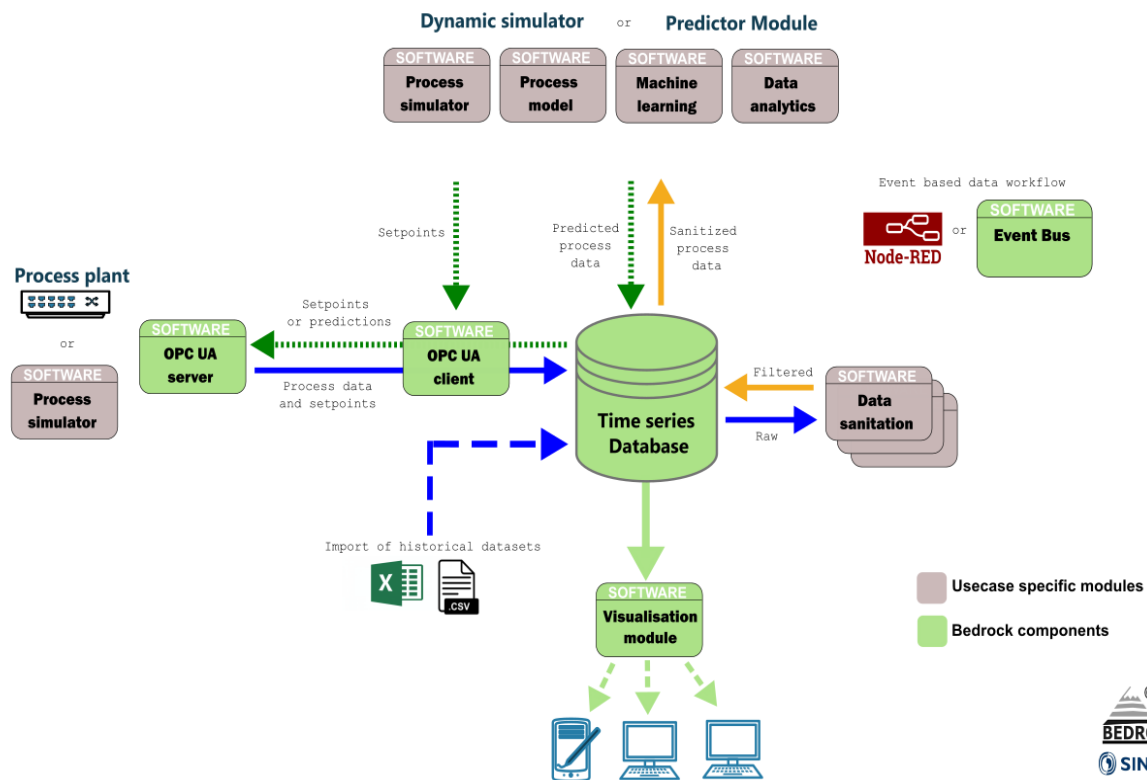


Figure 25: Architecture of the Bedrock open source based pipeline from SINTEF

4.2.2 Detailed description of the activities performed

The **Bedrock** platform toolbox with code repository has since the last milestone undergone restructuring to allowing for improved configuration and remote deployment.

The underlying components in the bundle of the updated repository are still containerized, but the deployments are now centrally configured from a hierarchy of Ansible playbooks. This enables deployment and administration of multiple remotely installed instances on different servers and projects and allows for flexible selection of modules in customized deployments based on the needs. It also enables options for cloud deployment.

The updated code structure allows for easy addition of new components to the toolbox.

TIA IOT is a platform that is used to acquire, collect and store sensor and PLC data. The platform enables real-time stream processing as well as batch processing.

- Two PLCs were connected by PN/PN Coupling.
- Sensor data was reorganized by means of a new coding system.
- OPC gateways and tags were updated and finalized for new coding system.
- Kafka topics have been redefined for optimized performance.
- Cassandra database tables were updated to store component-based data per table.
- Scripts to create OPC IoT gateways, database tables, Kafka topics, etc. were updated
- MQTT-Kafka bridge, previously developed in Java, was developed in Python.
- MQTT-Kafka bridge has been developed to transfer data based on rules that can be defined in config.yaml file.

- The bridge listening to Kafka and writing to PostgreSQL was developed in Python.
- The bridge listening to Kafka and writing to Cassandra was developed in Python.
- Data transfer script transferring data from Cassandra to PostgreSQL database has been developed in Python.

4.2.3 Progress beyond State of the Art or State of the Practice

Many of the pilot partners already have one or more comprehensive baseline platforms to support their operations (i.e. SAP and/or Microsoft). These platforms will be connected in various way to the offerings from the COGNITWIN Toolbox, most typically through OPC UA based interfaces or by use case adapted data connections.

4.2.4 Summary of the key achievements

The **TIA IOT** platform is in operative use in the Noksel pilot – and the platform has been updated for additional services and functionality needed by the COGNITWIN pilot.

In the second period the TIA IOT platform's quality attributes especially in performance and scalability was improved.

The **Bedrock** platform toolbox has been updated for more flexible cloud deployment

The pilot partners have described their existing platforms, where there might be some needs for extensions/connections with these for further trusted data sharing with external organisations – like researchers or other collaborating organisations.

4.2.5 Conclusion

The **TIA IOT** platform is at the end of COGNITWIN in operative use in the Noksel pilot – and the platform has been updated for additional services and functionality needed by the COGNITWIN pilot.

The **Bedrock** platform toolbox has been used for experimental deployments, but is also now ready for further use in larger deployments.

4.3 Security and IDS – International Data Spaces

4.3.1 Objectives, challenges and components

Security is vital at all levels and ensures that data/information/services are always managed in a secure way. The goal of this task is not to prescribe one approach for the whole project, but to help selected parties to work with each other, without being hindered by data and model access issues. Based on the results of the *International Data Spaces initiative (IDS)*, we have the mechanisms to support *IDS connectors* and the methods and tools for usage control by specifying the way in which data and models should be handled after access has been granted. The actual choice of what to provide here will depend on the strategies and needs for data sharing in the context of the different pilot partners. In the first phase of the project we see that it has been sufficient to share the data by either providing access to the data within a trusted subset of the pilot partners running digital platform or by providing selected export of data at various time intervals. For the second phase of the project there will be an evaluation of the further needs and requirements, to evaluate and decide if an IDS implementation will be suitable.

Table 3: CyberSecurity – challenges, requirements and solutions

Pilots	CyberSecurity (by task 4.2) – Challenges, Requirements and Solutions
Hydro	<ul style="list-style-type: none"> • Challenge: Sharing of online industrial data from Hydro is very challenging due to experiences in the past where the data systems have been attacked from the outside with the intent of bringing down the production. • Requirement: Strict security rules implies that most of the plant process system will be offline with respect to external access. • Solution: Strict security access will be applied – and dedicated access will be done through the Trusted Data Layer.
Elkem	<ul style="list-style-type: none"> • Challenge: Due to the risk of cyber attack the process system is isolated from external access. • Requirement: Avoid external intruders. • Solution: Ensure that the systems in the plant are not externally connected.
Saarstahl	<ul style="list-style-type: none"> • Challenge: Due to the risk of cyber attack the process system is isolated from external access. • Requirement: Ensure closed loop system without connection to the exterior • Solution: Ensure that the systems in the plant are not externally connected.
Noksel	<ul style="list-style-type: none"> • Challenge: Security • Requirement: Data should be accessed by authorized and authenticated users, data should be coded • Solution: Authentication and authorization was performed. Data encryption via coding was done
Sidenor	<ul style="list-style-type: none"> • Challenge: Strict process plan security need to be enforced • Requirement: Ensure no intruder possibilities • Solution: Ensure that the systems in the plant are not externally connected.
Sumitomo SHI FW	<ul style="list-style-type: none"> • Challenge: Strict process plan security • Requirement: Use the existing secure management system • Solution: Ensure that the existing infrastructure is being used

Table 4: Data lake and storage – challenges, requirements and solutions

Pilots	Data lake, storage, Data Space (by task 4.2) – Challenges, Requirements and Solutions
Hydro	<ul style="list-style-type: none"> • Challenge: Ensure controlled and trusted data access. • Requirement: Use the existing Trusted Data Layer. • Solution: The method used in this pilot is that some research partners are supplied with off-line data for analyses purposes, while a few persons from the partners get access to online data. The partner working with the control application must be able to access the online data.

Elkem	<ul style="list-style-type: none"> • Challenge: IR cameras generate large amounts of data if each captured frame is stored. • Requirement: Store images with lower framerate, and only selectively when tapping is in progress for instance. • Solution: Cybernetica InSight with algorithms for detecting active tapping/refining/casting. • Challenge: Use the existing infrastructure whenever possible, i.e. Elkem databases and data lake platform. • Requirement: It is not a process need/requirement to exchange information with external organisations as part of the process lifecycle within the plant control. • Solution: Provide database access on demand through secured access.
Saarstahl	<ul style="list-style-type: none"> • Challenge: Resolve the management of large volumes of video data • Requirement: Ensure management of both training and production data • Solution: For productive setting, video data will not have to be persisted. Training data is saved in storage (x TB, type of storage...). Tracked IDs saved in DWH.
Noksel	<ul style="list-style-type: none"> • Challenge: When data gets bigger, time to query database is increased • Requirement: Data should be queried in real time • Solution: Data was split into tables that are based on machine components • Challenge: The increased in the size of the data resulted in data analysis difficulties • Requirement: Efficient and fast data analysis • Solution: Data analysis is performed on Apache Spark that works distributed
Sidenor	<ul style="list-style-type: none"> • Challenge: Ensure data access taking into account the needs for secure access protection • Requirement: Data exchange • Solution: Provide explicit data access for the needed experimental data, consider separate system access.
Sumitomo SHI FW	<ul style="list-style-type: none"> • Challenge: Large amounts of on-line require design of edge computing • Requirement: Speed and price must remain feasible. • Solution: To be completed by the pilot owner, service provider

Problem definition

Trusted and secure data exchange and access is fundamental in the context of digital platforms and digital twins for process plants. Related to this the project pilot partners already have a number of data and access control mechanisms in place, such as a Trusted Data Layer, and strong restrictions on remote connections.

Possible Solutions

The project objectives and scope has explicitly stated the goal to relate to the *International Data Spaces initiative (IDS)* and the emerging technologies from the International Data Spaces Association.

The International Data Spaces Association (IDSA)¹⁴ is the evolution of IDS (Industrial Data Space) which itself was an initiative lead by Fraunhofer ISST, in cooperation with ATOS, T-Systems, and the idea is promoted by the German Federal Ministry of Education and Research. IDSA is characterized by the focus on information ownership, with the aim of enabling clear and fair exchanges between data providers and consumers. To this end it suggests a reference distributed architecture that accomplishes this goal is illustrated in Figure 26.

Broadening the perspective from an individual use case scenario to **interoperability** and a platform landscape view, the IDS Reference Architecture Model positions itself as an **architecture that links different cloud platforms through policies and mechanisms for secure data exchange and trusted data sharing** (through the principle of data sovereignty). Over the IDS Connector, industrial data clouds, individual enterprise clouds, on-premise applications and individual, connected devices can be connected to the International Data Space ecosystem.

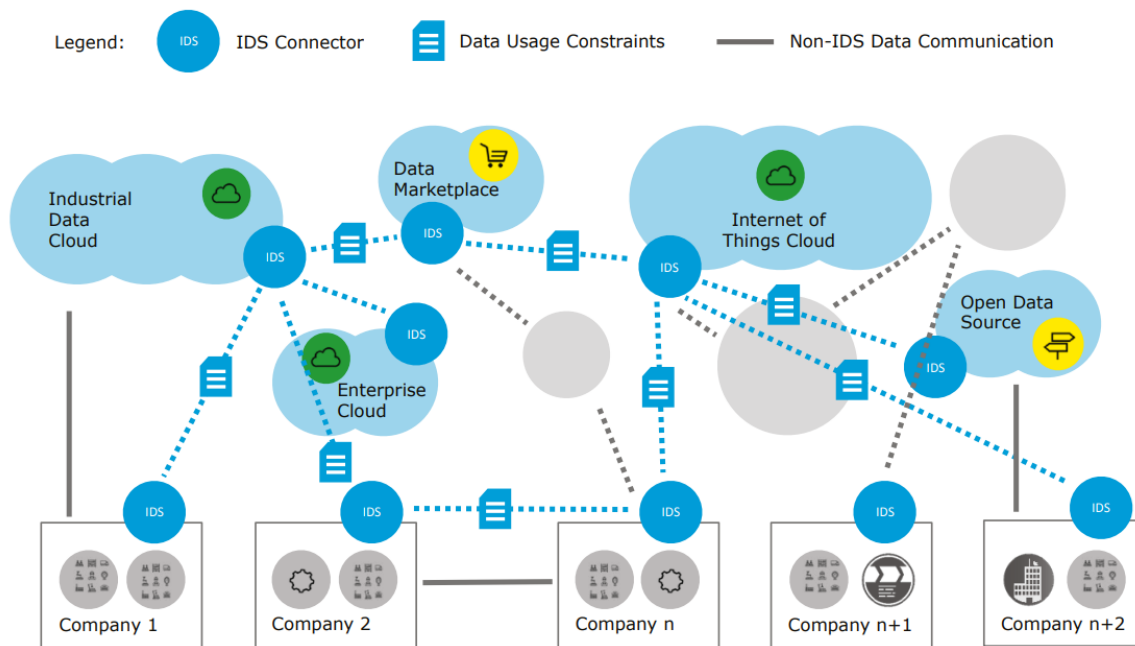


Figure 26: International Data Spaces connecting different platforms

This IDS Reference Architecture¹⁵ as cited from here, support various mechanisms for secure and trusted data transfer including in particular the following:

- **Secure communication.** The concept of Trusted Connector is introduced
- **Identity Management** for identification/authentication/authorization enhancing. There is use of certificates issued by a Certificate Authority (CA).

¹⁴ <https://www.internationaldataspaces.org/>

¹⁵ <https://internationaldataspaces.org/download/19016/>

- **Trust Management** that uses Cryptographic methods such as PKI (Public Key Infrastructures).
- **Trusted Platform** for trustworthy data exchange, which defines the minimal requirement for Security Profiles that should be verified by IDS connectors. It also defines the capacity to perform integrity verification of the rest of the involved connectors.
- **Data Access control.** IDS defines authorization criteria based on the previously defined Security Profiles.
- **Data Usage Control.** IDS checks and regulates that data processing is according to the intended purposes defined by the original data owner.

In addition to the current development in IDS it is relevant to follow the evolution of the GAIA-X initiative that now is incorporating IDS.

The **GAIA-X** [GAIAX20] initiative as cited from here, aims for the creation of a federated, open European data infrastructure, enabling the interconnection of centralised and decentralised data infrastructures in order to turn them into a homogeneous, user-friendly system. The following description is created from the GAIA-X technical architecture description. GAIA-X will define the technical principles which foster the implementation of the European Data Strategy. Data Sovereignty, i.e. the execution of full control and governance by a data owner over data location and usage, is one of the core principles of GAIA-X. The requirement of data sovereignty has led to the following high-level requirements for a GAIA-X implementation:

- **Openness and transparency:** specifications will be accessible to all GAIA-X participants, technical steering and roadmap definitions are conducted in a public process.
- **Interoperability:** Participants are able to interact with each other in a defined way. Self-description and policies are used to manage interactions between data providers and data consumers.
- **Federation:** standardized access and multiple decentralized implementations operated by autonomous providers.
- **Identity and trust systems** to manage the interaction between GAIA-X participants, without building upon the authority of a single corporation or government.

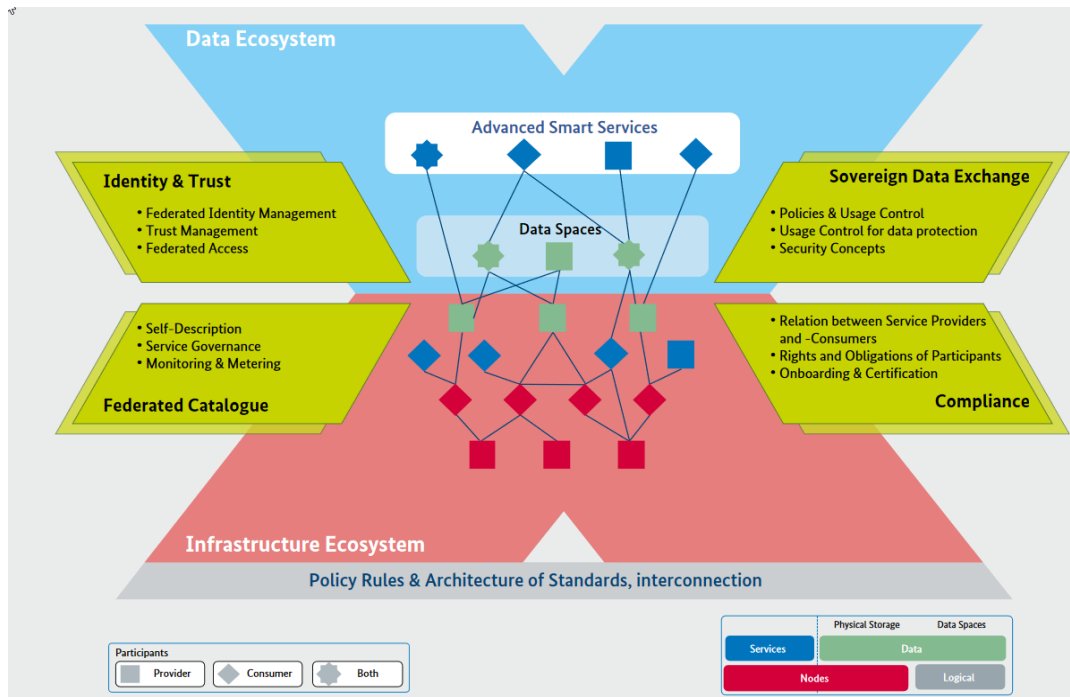


Figure 27: High-level overview of the GAIA-X architecture

The core architectural elements in GAIA-X are *assets*, *participants* and *catalogues*. Participants are natural or legal persons that can act as provider, consumer, data owner and visitor. Providers can host multiple user accounts. Assets can be either a *Node*, a *Service*, a *Service Instance* or a *Data Asset*. Hereby, a node is in general a computational resource like a data centre or an edge computing device, and nodes can be organized in hierarchies. Services can be deployed on nodes and describe a cloud offering. A service instance is the concrete realization of a service running on a node. All nodes, services and service instances are associated with a provider. Data assets are data sets which can be either searched, provided or consumed by either another service or a participant, are hosted on a node, and are owned by a participant. GAIA-X data assets are content- and structure agnostic and provide metadata and a self-description. Self-descriptions which describe the characteristics of assets and participants and catalogues are the elements which implement the publication and discovery assets and participants.

The architecture of GAIA-X fosters the development of digital ecosystems and structures them into *Infrastructure Ecosystems* and the *Data Ecosystems*. The infrastructure ecosystem comprises hereby services to transfer, process and store data. Stakeholders of the infrastructure ecosystem can be cloud service providers, edge clouds, HPC providers etc. Under the data ecosystem, actors along the data value chain are summarized. This could be for example data providers, data owners, data consumers, or smart service providers.

The following figure illustrates how the GAIA-X architecture incorporates IDS elements¹⁶, and how also COGNITWIN technologies could be positioned related to this.

¹⁶ <https://internationaldataspaces.org/download/19016/>

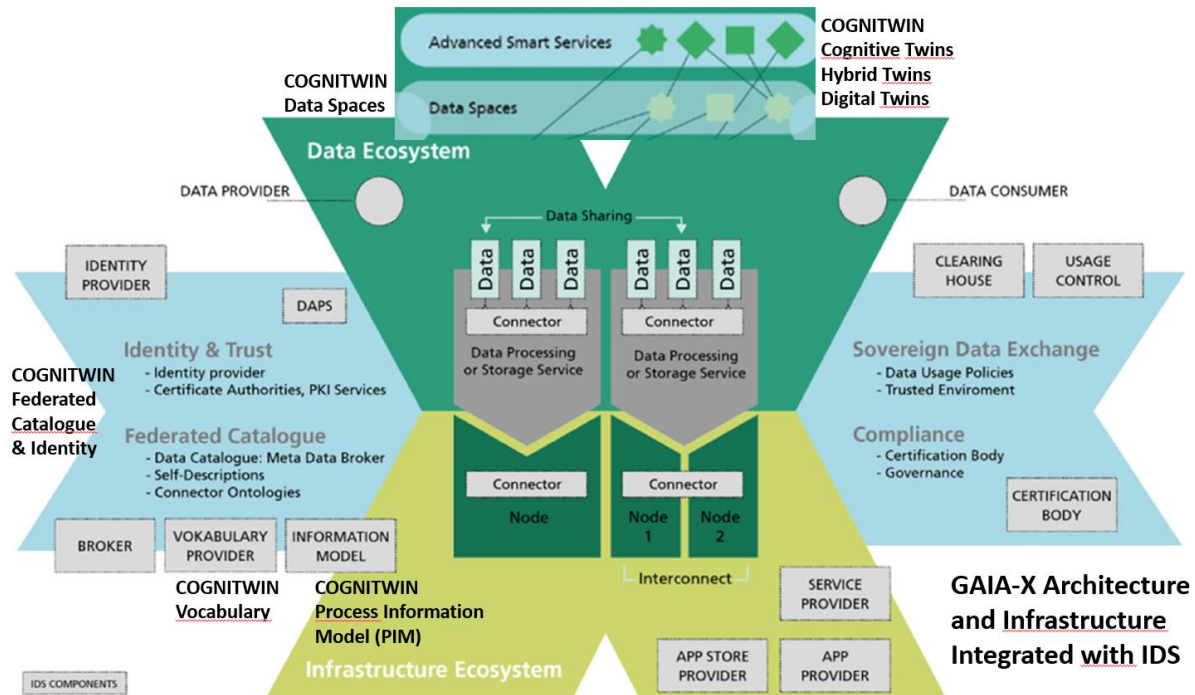


Figure 28: Combination of IDS technologies and GAIA-X architecture – with COGNITWIN annotations.

Both IDS and GAIA-X are currently executing use case/pilot initiatives related to Industry 4.0 use cases that is worth to follow in order to ensure possible future synergies with these.

Proposed approach: Establish necessary IDS connectors when this is deemed suitable, compliant also with the emerging GAIA-X/IDS architecture as a federated data infrastructure for Europe – supporting data sovereignty and control. It is not any intention to implement a fully GAIA-X compliant infrastructure in the COGNITWIN project, but COGNITWIN partners Fraunhofer and SINTEF are involved in IDS and GAIA-X and will ensure that the architectures and approaches in COGNITWIN will be possible to take advantage of IDS and GAIA-X technologies in the future.

The focus in the first period was on the potential use of IDS technologies in COGNITWIN on the **Trusted Factory Connector (IDS)** provided by Fraunhofer. This could be further enhanced in the final parts of the project, pending requirements from the pilots.

IDS Connectors– SINTEF - The Figure 29 shows the architecture of a connector with the common execution core and a custom application part, as described in the SINTEF IDS Connector component.

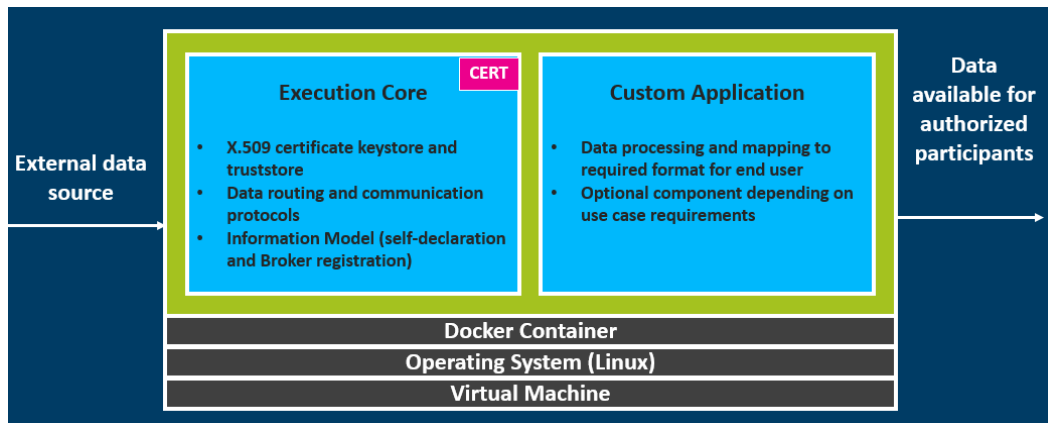


Figure 29: Internal Connector architecture pattern –from IDS Connector component (SINTEF)

This component is available as a framework for the creation of new connectors for data providers or data consumers in an IDS conformant Data Space, if the further pilot requirements would imply the need for this.

4.3.2 Detailed description of the activities performed

IDS Connectors– SINTEF

The presentation of possibilities to create context specific IDS connectors presented in deliverable D4.1 is still relevant for future project phases, although no activities has been performed on this during the last or current period.

4.3.3 Summary of the key achievements

TEKNOPAR integrated data security technology into the TIA IOT platform. User authentication and authorization were realized both in the application and the system level.

The TIA IOT platform's quality attributes especially in performance and scalability was improved. TEKNOPAR established data transfer between two hosts using IDS Connectors. The connector providing data read AAS data from AASxServer providing AAS data and sent data to IDS connector configured as Consumer Connector.

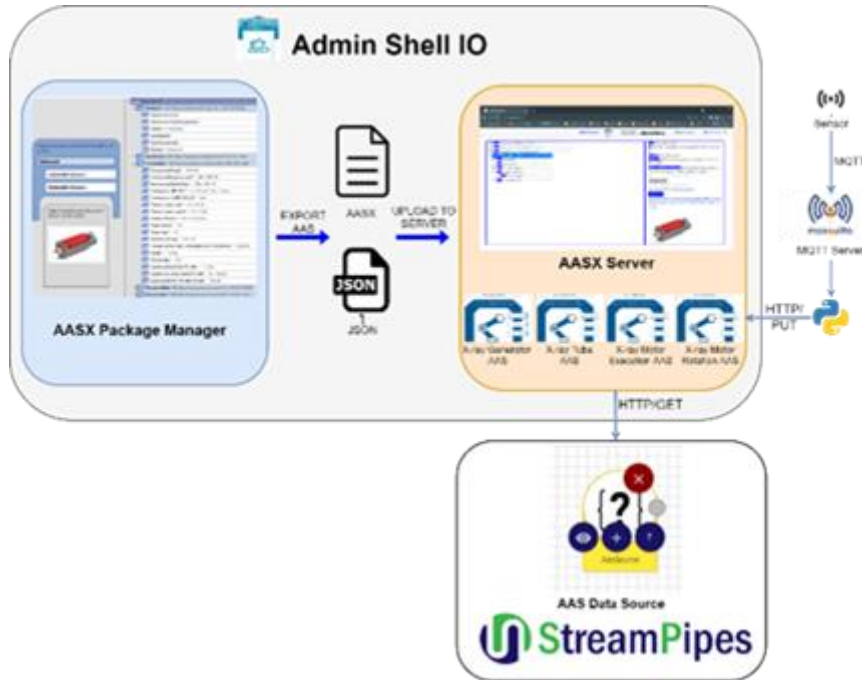


Figure 30: The architecture using Admin Shell IO

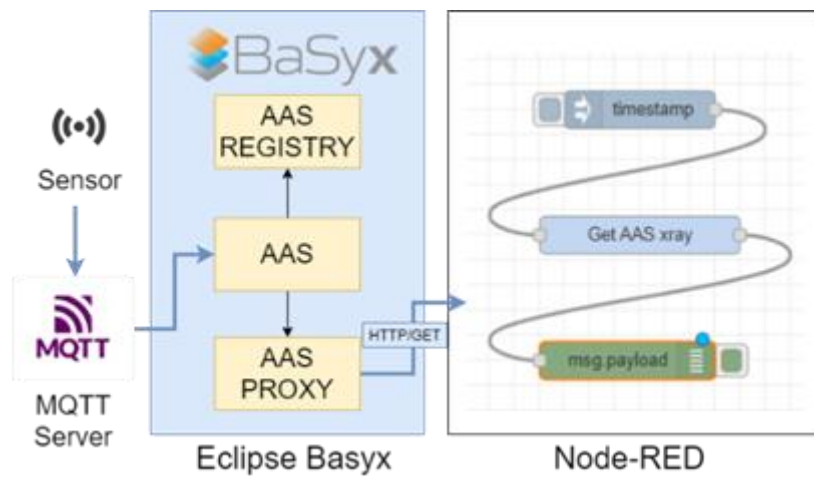


Figure 31: The architecture using Eclipse BaSyx

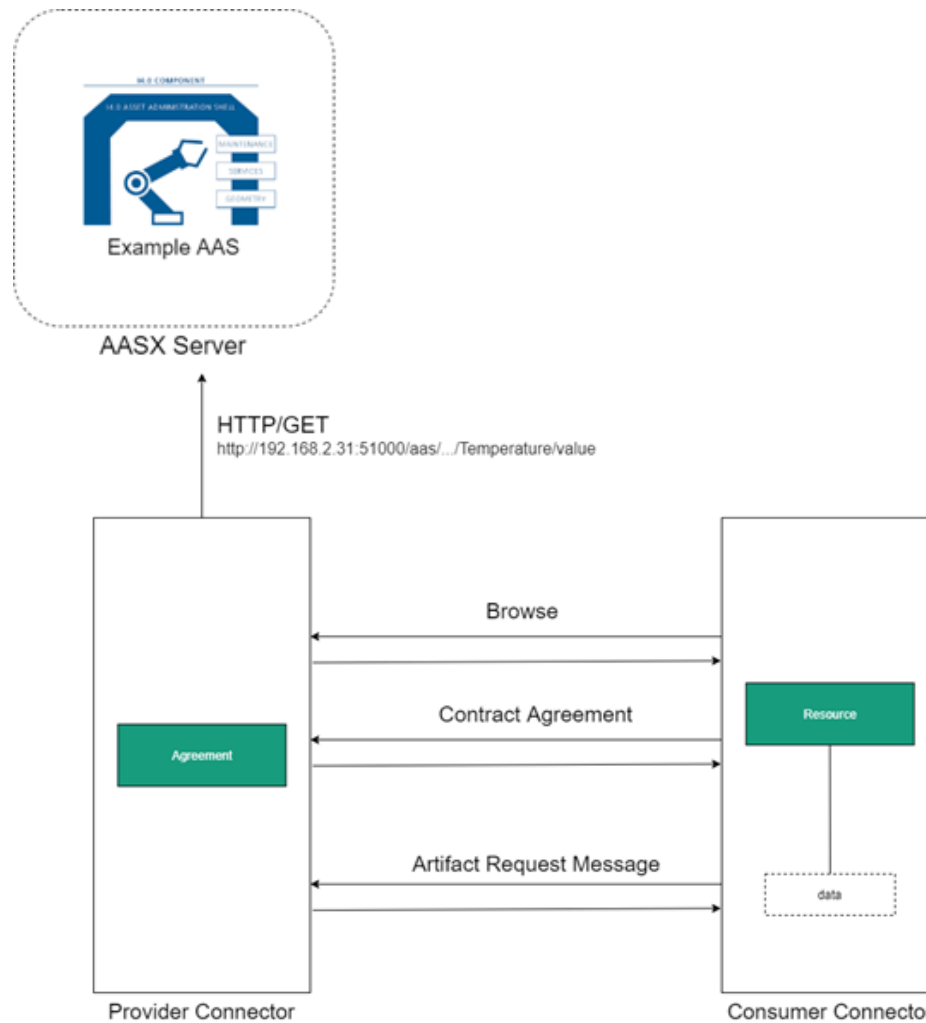


Figure 32: Setup of example AAS

4.3.4 Conclusion

The concept and architectures of Data Spaces is in active development in Europe, in particular under the umbrella of European Common Data Spaces, and in particular the European Manufacturing Data Space. It is, however, a prerequisite for successful use of data spaces that there is a situation where data (and/or digital twins) need to be exchanged and managed across a number of autonomous organisations typically in a supply chain or in a supply network. In COGNITWIN each of the focused pilots are only within one organisation and not across organisations and the benefits of a data space architecture has thus been decided not to be necessary.

Table 5: Digital Twin Models – challenges, requirements and solutions

Pilots	Digital Twin Models, data representation (by task 4.3) – Challenges, Requirements and Solutions
Hydro	<ul style="list-style-type: none"> • Challenge: Ensure the best possible representations for data driven and physical models • Requirement: Use the best possible Digital Twin representations.

	<ul style="list-style-type: none"> • Solution: Use the sensor data models in combination with physical models – consider experimentation with standards like AAS in order to facilitate for future digital twin interoperability.
Elkem	<ul style="list-style-type: none"> • Challenge: Need for temperature sensor data models as well as physical models • Requirement: Use the best possible Digital Twin representations. • Solution: Use the sensor data models in combination with physical models – consider experimentation with standards like AAS in order to facilitate for future digital twin interoperability.
Saarstahl	<ul style="list-style-type: none"> • Challenge: Neural network detecting billets for digital representation of status quo in blooming train. • Requirement: Vast amounts of training data required • Solution: Set up routine for mass generation of synthetic training data for DL network.
Noksel	<ul style="list-style-type: none"> • Challenge: Not sufficient machine breakdown data • Requirement: Data driven model for predictive maintenance • Solution: Synthetic data generator and balanced data over sampling • Challenge: Data has empty columns • Requirement: Data driven model will be developed used the data • Solution: empty data is filled by “back fill” and “forward fill”
Sidenor	<ul style="list-style-type: none"> • Challenge: Ensure a suitable data representation • Requirement: Manage the available data in a Digital Twin representation • Solution: Provide AAS Digital Twin (<i>DT</i>) representation of Sidenor assets based on Asset Administration Shell (<i>AAS</i>) standard Use Sensors and Data Acquisition App (<i>DAA</i>) for reading sensor values and providing them to <i>DT</i> StreamPipes (<i>SP</i>) pipeline services for data processing
Sumitomo SHI FW	<ul style="list-style-type: none"> • Challenge: Ensure data representation suitable for the required models • Requirement: Provide data in a suitable form for the models • Solution: Provide data in the form needed by the mathematical models and also create StreamPipe interfaces for thsi

4.4 Digital Twin API – AAS

4.4.1 Objectives, challenges and components

In the previous deliverables we have introduced the FA³ST ecosystem. FA³ST stands for Fraunhofer Advanced Asset Administration Shell (AAS) Tools for Digital Twins. The key component is the FA³ST Service, which is a Java-based open-source implementation of the reactive AAS and cornerstone of the FA³ST ecosystem for Digital Twins. It is released under Apache 2.0 License.

During the final 3rd period, the focus was on:

- Further development and improvement
- Release of multiple versions
- Documenting the code

4.4.2 Detailed description of the activities performed

In our paper “FA³ST Service – An Open Source Implementation of the Reactive Asset Administration Shell” [Jac+22], we provide details about the FA³ST Service that implements the concept of a reactive/Type 2 AAS and is one of the key components of the FA³ST ecosystem. The FA³ST Service was developed to be easy to use for end users. At the same time, it supports an open architecture and thus allows for extensibility. The additional key selling point of FA³ST is that a digital twin is in control of all asset connections.

During the reporting period, there were several releases of the FA³ST Service. Here we report only on new features. More information on them as well as on internal changes & bugfixes can be found at: <https://faaast-service.readthedocs.io/en/latest/changelog/changelog>.

Release version 0.2.0 - New Features

- Persistence
 - File-based persistence added
 - Each persistence implementation can now be configured to use a given AAS model as initial value
- Asset Connection
 - HTTP asset connection added
 - Basic authentication (username & password) added for OPC UA, MQTT and HTTP
 - Introducing protocol-agnostic library for handling different payload formats including extracting relevant information from received messages as well as template-based formatting of outgoing messages (currently only implemented for JSON)
- HTTP Endpoint
 - API extensions for Submodel Interface and Asset Administration Shell Serialisation Interface
 - Support for output modifier
 - CORS support
 - now returns status code 405 Method Not Allowed if URL is correct but requested method is not supported
- Support for valueType=DateTime
- Support for Java 16
- Improved robustness (e.g. against common invalid user input or network issues)
- Improved console output (less verbose, always displays version info)
- Improved documentation

Release version 0.3.0 - New Features

- Asset Connection

- OPC UA
 - Automatic reconnect upon connection loss
 - Add ParentNodeId to OpcUaOperationProviderConfig
 - Introduce mapping between IdShort and Argument Name in OpcUaOperationProviderConfig
- MQTT
 - Automatic reconnect upon connection loss
- HTTP
 - Now supports adding custom HTTP headers
- Improved JavaDoc documentation
- Improved security through automatic vulnerabilities check before release
- Added example how to implement custom asset connection

Release version 0.4.0 - New Features

- Improved logging for new CLI arguments

Latest development version (0.5.0-SNAPSHOT) - New Features & Major Changes

- Improved exception handling in CLI - upon error starter application should now correctly terminate with error code 1
- OPC UA Endpoint
 - Additional parameters available in configuration
- Docker container now runs using a non-root user
- Base persistence configuration updated

4.4.3 Progress beyond State of the Art or State of the Practice

The FA³ST Service implements the model and API defined by the AAS specification. Additionally, it embraces an open architecture that allows for easy extension and customization. FA³ST Service allows closing the gap between digital twins and the (physical) assets by providing the means to connect and synchronize them using arbitrary communication protocols.

Key FA³ST features are:

- supports several data-formats for the Asset Administration Shell Environment: json, json-ld, xml, aml, rdf, opcua nodeset
- easy configuration via JSON file
- easily expandable with 3rd party implementations for endpoint, message bus, persistence, asset connection
- uses existing open-source implementation of AAS data model and de-/serializers admin-shell-io java serializer and admin-shell-io java model
- synchronization between multiple endpoints
- connecting to assets using arbitrary communication protocols
- can be used via command-line interface (CLI), as docker container or embedded library

4.4.4 Summary of the key achievements

Key achievements include:

- FA³ST release: [GitHub - FraunhoferIOSB/FAAAS-Service: FA³ST - Fraunhofer Advanced Asset Administration Shell Tools \(for Digital Twins\)](#)
- FA³ST documentation: [FA³ST Service — FA³ST Service documentation \(faaast-service.readthedocs.io\)](#)
- FA³ST paper: M. Jacoby, F. Volz, C. Weißenbacher and J. Müller, "FA³ST Service – An Open-Source Implementation of the Reactive Asset Administration Shell," *2022 IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA)*, Stuttgart, Germany, 2022, pp. 1-8, doi: 10.1109/ETFA52439.2022.9921584.
- Three demonstrators based on FA³ST:
 - Demonstrator for the research factory in Karlsruhe - AAS modelling for a physical device, data acquisition, use of the AAS API for data access and visualization.
 - Demonstrator for HMI 2022 - Interaction between digital twins in real time and on demand and their use in data spaces.
 - Demonstrator for SPS 2022 – Multiple digital twins and modelling, collection and sharing of carbon footprint data.

4.4.5 Conclusion - Next steps after completion of the project

Next milestone is to release version 1.0.0 to Maven Central and DockerHub. Additionally, the FA³ST service will be further developed to support business logic integration with or into digital twins.

4.5 Digital Twin Graph support for Simulation and Cognition

4.5.1 Objectives, challenges and components

The subtask objective is to come up with flexible mechanisms to represent Digital Twin (DT) data, covering assets, processes, and sensor data, effective and efficient storage of the data, and support for analytics tasks on top of the data with a focus on simulation of processes.

The challenges are related to flexible data models for DT data, scalable storage and access to data, and integration of various aspects in a unifying framework.

The components are realized as a software framework addressing data representation, storage and access, and example of analytics (focus on process simulation).

The DT data representation aspect of the framework is orthogonal to standards such as the Asset Administration Shell (AAS) in the sense that it is focused on basic data structures (graph) for representing assets and processes without going into the domain-specific semantics of such elements, while at the same time being directly usable in a programming/development environment by typical developers. Standards such as AAS can be used to add semantics/annotations to elements stored in the graph data structures. Furthermore, the graph structures are persistently stored and used for simulation purposes.

Problem definition

The concept of a Digital Twin can encompass a large variety of aspects. Representing and managing various data related to DTs is a key enabler for tasks such as DT analytics/simulation. Finding the right data representation mechanisms and identifying relevant technologies for representing, storing, accessing and using DT data for the purpose of simulations is a challenging task for many organizations that lack data management competence. In this context, we want to support such organizations with tools that facilitate the management of DT data, and established the following requirements for our framework for supporting DTs related data:

- Cover generic aspects related to assets, processes, and related sensors, without going into domain-specific semantics.
- Support for flexible data structures to manipulate data related to the above aspects.
- Support for data storage and access.
- Support for analytics, with focus on discrete event and flow simulation for the processing and manufacturing industries.
- Python-friendly software stack based on open-source components (libraries and databases).

Proposed Solutions

In the paper [WAS+22], we introduced SINDIT¹⁷ - SINTEF Digital Twin Architecture – a generic software architecture that provides interface to connect various tools for the purpose of building knowledge graph based Digital Twins. Figure below illustrates the four-layer architecture of SINDIT. The Physical Layer contains the physical assets such as machines or sensors. The Data Layer contains all persistent storage back-ends. The Business Layer contains (i) descriptions of machines and processes, (ii) services for event detection, capacity estimation, and simulation, and (iii) federated access to relevant data sources. The Application Layer exposes the relevant outputs of the Business Layer. Depending on the application domains, different libraries and databases can be exploited to realize SINDIT architecture Figure 34.

¹⁷ <https://github.com/SINTEF-9012/SINDIT>

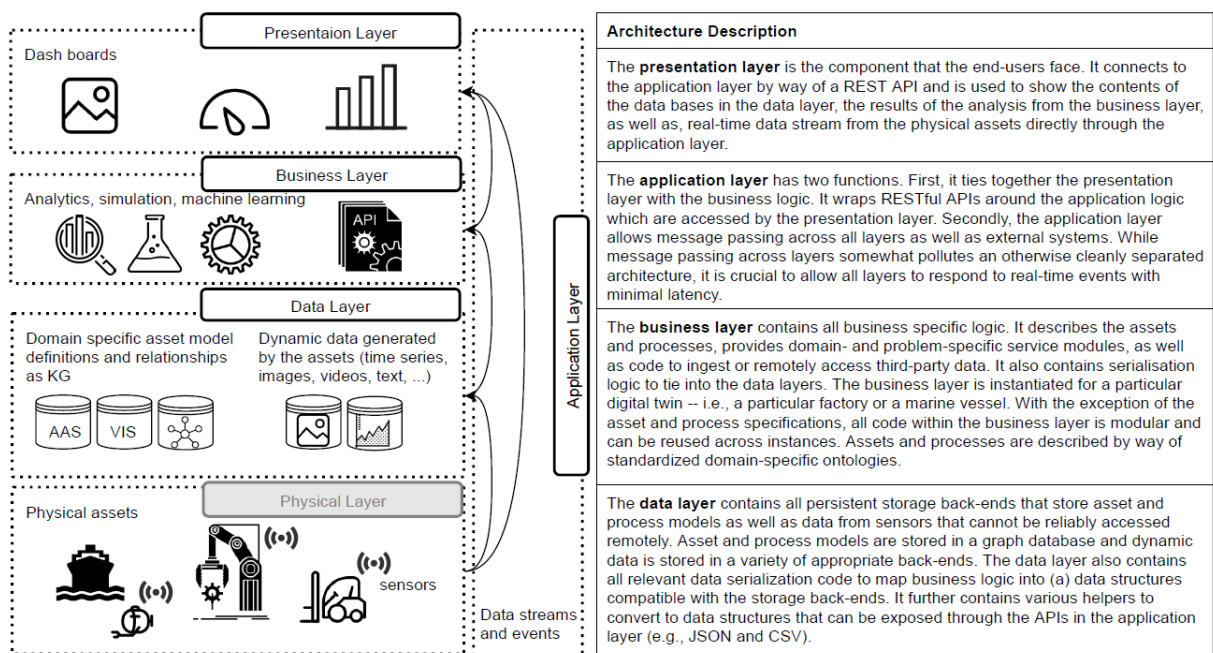


Figure 33: The generic four-layer software architecture of SINDIT along with various technologies deployed in the stack [WAS+22]

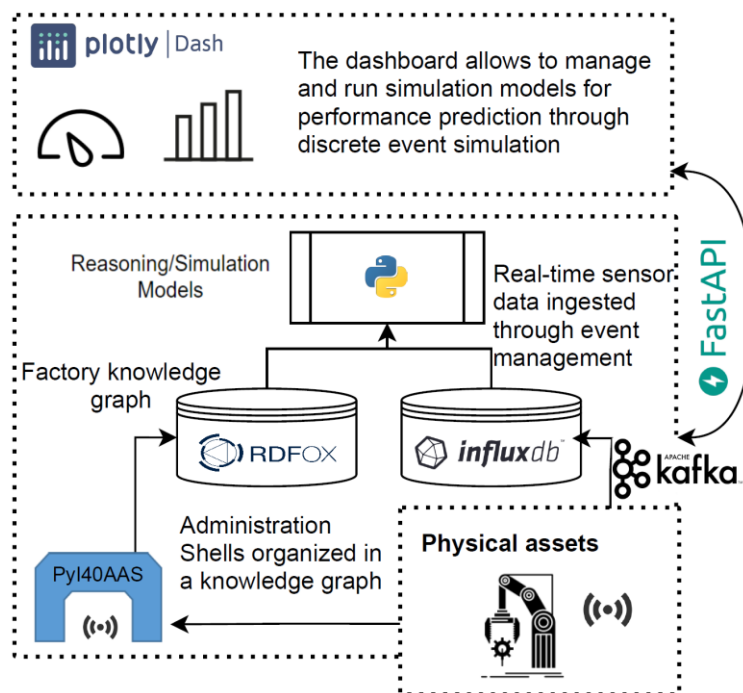


Figure 34: Overview of the SINDIT components instantiated for manufacturing environment [WAS+22]

To develop a generic infrastructure for Knowledge Graph Based Digital Twin in the manufacturing domain, different graph-based technologies, databases and library are investigated and integrated into SINDIT. As shown in the figure above, we employed the AAS standard as a generic modelling method for assets such as machines, sensors, storage, products as well as their relationships or manufacturing

processes. We exploited the Python library PyI40AAS¹⁸ to build such AASs. There are two types of databases employed to build the Digital Twin: (i) InfluxDB¹⁹ for real-time data generated by the sensors and machines, and (ii) RDFox²⁰ is the triplestore for the semantic knowledge graph representing the process parameters, assets information and the factory graph. This semantic knowledge graph provides a standardized knowledge representation for the (Hybrid) Digital Twins as well as support domain knowledge integration. On top of this knowledge graph, semantic reasoning rules and simulation models are introduced in order to enable the "cognition" aspect of the Cognitive Digital Twins (CDT). Furthermore, SINDIT also provides an API implemented with FastAPI²¹ to support exchanging or modifying the AASs models and the knowledge graphs via REST services. These REST services are also used to build a simple dashboard with Plotly Dash²² for the visualization of the Digital Twins and reasoning/simulation results.

SINDIT is provided as open-source framework under MIT license and is available in our GitHub repository²³. SINDIT is also added to the COGNITWIN toolbox under the "Digital Twin Representation Tools and Services" category.

SINDIT was used in Sidenor pilot to integrate both acyclic data (e.g., process parameters) and cyclic data (e.g., timeseries sensor data) from the production into the knowledge graph in order to enable a formal data representation for the CDT. The output of the hybrid models (ML or physical models) is also integrated into the graph in order to support automatic reasoning and decision-making. Finally, domain expert knowledge for making assessment (e.g., whether the ladle should be demolished or repaired based on current real-time sensor measurements and prediction outputs) is also integrated into SINDIT under the form of reasoning rules (IF-THEN rules) which will be executed over the knowledge graph in order to assist the human operator in making maintenance assessment. Detailed about this application was reported in D2.4.

4.5.2 Detailed description of the activities performed

Activities performed include:

- Conceptual design of the framework
- Identification and evaluation of relevant technology stack including technologies, flexible data models, libraries, and databases
- Solution design including: framework architecture, selection and deployment of components, libraries and databases
- Python-based prototype implementation

4.5.3 Progress beyond State of the Art or State of the Practice

¹⁸ <https://pypi.org/project/pyi40aas/>

¹⁹ <https://www.influxdata.com/>

²⁰ <https://www.oxfordsemantic.tech/>

²¹ <https://fastapi.tiangolo.com/>

²² <https://plotly.com/dash/>

²³ <https://github.com/SINTEF-9012/SINDIT>

Representing Digital Twins using Knowledge Graphs (KG) paradigm has recently been proposed in the literature [Ban+17], [Gar+18]. By use of ontologies and various data mapping, KGs can offer a mechanism for harmonizing data in a DT context, on top of which various analytics tasks such as simulations, predictive maintenance, etc., could be performed. The use of heavy-weight semantics is however challenging in practice, especially for developers. Our approach is focused on lightweight graph data structures that would allow developers an easy entry into flexible modelling of assets and processes, in addition to smooth integration with timeseries data.

4.5.4 Summary of the key achievements

Key achievements include:

- Design of framework for data management in the context of DT integrating flexible graph-based data representation (for generic aspects of assets, processes, and sensor data), data storage and simulation support
- Solution design and identification of relevant technologies
- Python-based prototype implementation covering data representation, storage, and simulation (including visualization)

5 Sensors, Understanding Sensor Data & Quality Assurance

The objectives of COGNITWIN Task 4.3 may be briefly summarized:

- 1) To assist the pilots with selection of suitable sensors, or to aid with development of new sensors or sensor concepts where commercial solutions cannot be found.
- 2) To develop digital tools for local (edge) processing of sensor data. This is especially relevant in situations where sensors generate large amounts of data (images, video, AC waveforms) that must be reduced to essential components before transfer to permanent storage.
- 3) To ensure data quality at the sensor level, using knowledge of the physical sensor and relevant processes to make decisions about data quality that could not be made at the analytics level.
- 4) To arrange robust methods of extracting and transferring data from the physical sensors or local clients to the data storage layer.

At its conception, the COGNITWIN project sought to include a broad range of industries with very different challenges to be solved. Although this approach promises a rich toolbox, it also poses challenges with finding general solutions within the project – i.e. tools that are relevant for multiple pilot cases. This is especially true on the "bottom" sensor layer, where digital tools necessarily must be tailored to the sensor type. E.g., machine vision cannot be applied to a vibration waveform.

We therefore structure most of this section by pilot – instead of by component – to provide necessary background and context for the choice and implementation of each component. Subsequently, we will discuss some of the generic digital tools that have been developed or made available for sensor data management at the edge.

5.1 Objectives, challenges and components

Table 6: Sensors – challenges, requirements and solutions

Pilots	Sensors (inc. data quality) (by task 4.3) – Challenges, Requirements and Solutions
Hydro	<ul style="list-style-type: none"> • Challenge: Improved but still somewhat unstable HF sensor • Requirement: Introduce new TDL gas monitor for HF in pot stack - Quality information for HF sensor such that state and parameter estimation techniques integrated in the hybrid Digital Twin can be applied effectively • Solution: Laser status signal from OPC UA and anomaly detection method
Elkem	<ul style="list-style-type: none"> • Challenge: High quality data from infrared cameras in a harsh environment with moving equipment. • Requirement: Accurate calibration of infrared cameras • Solution: Proper protection and cooling of IR camera • Challenge: Identify and use suitable thermal sensors. • Requirement: Introduce 1μm Si-CMOS thermal imager for monitoring of refining process, and 3.9μm MBM thermal imager for monitoring of tapping process. • Solution: Sensors have been evaluated and now installed.

Saarstahl	<ul style="list-style-type: none"> • Challenge: Need for HD video camera for Blooming and Mill Train • Requirement: HD camera for industrial environment • Solution: Introduction of Machine vision cameras - FullHD Cameras
Noksel	<ul style="list-style-type: none"> • Challenge: Provide necessary sensor data to monitor machinery • Requirement: Make use of suitable sensors, including vibration sensor and energy sensors for motors, in addition to pressure and temperature sensors. • Solution: Four new sensors have recently been installed, including Analog vibration sensor on DC motor and Energy analyzers.
Sidenor	<ul style="list-style-type: none"> • Challenge: Collect necessary process data related to ladle heat • Requirement: Initially collect data from the process system • Solution: Use collected process data, consider solutions for use of infrared temperature camera but challenging due to closed ladle.
Sumitomo SHI FW	<ul style="list-style-type: none"> • Challenge: Necessary sensors include on-line process operation measurement data, especially heat exchanger temperatures and flows, both at flue gas and steam side. Also other process/combustion measurement indicators, fuel feed characterization sensors/soft sensors may be beneficial. • Requirement: DVR (data validation & reconciliation) of sensor data required. • Solution: Basic DVR is done both at DCS (by DCS system diagnostics) and monitoring tool ends (in Matlab, incl. OD-tool).

The sensor analysis and current usage for each pilot is further described below.

5.1.1 Hydro

Four sources of data are currently expected for the Hydro GTC pilot: alumina certificates for alumina quality information, GTC process data, Electrolysis cell data and Weather data including weather forecasts for improved predictive capabilities.

In addition to sensors that predate the COGNITWIN project, the Hydro pilot will make use of one new component: a newly installed tunable-diode-laser HF gas monitor, installed in the GTC off-gas channel. The instrument data are routed via PLC to OPC-UA and into the Aluminium Production Control System (APICS), from APICS the data is relayed to various visualisation systems and are stored in the Trusted Data Layer (TDL). The TDL is the used for off-line modelling and data driven adjustments.

Today all vital data is logged once every minute and where the logged values are time averages. Time averaged data is built from instruments measuring “continuously”, such as temperature T100, pressure transducers and HF sensor with sampling rate down to 4 Hz. Additional lab data, either on shift basis (1/8h) or day basis (1/24h) is available.

The HF laser was chosen primarily based on positive previous experience with this sensor type in similar applications at Hydro. The manufacturer has previously worked with Hydro to tailor their sensors to HF applications, and they are familiar with Hydro's processes and challenges. Furthermore, the sensor is delivered with integrated air purging to protect optical viewports from scaling or corrosion – an absolute requirement in the dusty and toxic environment of the off-gas channel.

Other possible sensor technologies have been considered but were found to be either too complex for permanent installation (open-path FTIR, gas chromatography) or unable to withstand the aggressive chemical environment in the off-gas channel (electrochemical, photoionization). Many technologies for gas sensing (optochemistry, organic) give only qualitative detection or have long transient fall times. Finally, there are emerging technologies (metal-organic frameworks) that were considered too immature for the pilot objectives.

The new hardware components were installed in the early phases of the project and have been connected to Hydro's internal data lake and logging data since August 2020. The main upcoming challenges involve quality assurance of the data. A key issue in this regard is that the source data in the TDL does not indicate if a sensor is offline – it simply continues to report the last known value. Logic must therefore be implemented to determine if a sensor is "flat-lining," and whether this is expected or unexpected.

A list of sensors and data sources is provided in Table 7.

Table 7: Sensors and data sources used in the Hydro pilot model

COMPONENT TYPE	PARAMETER	LOCATION
Data source	Air temperature	Weather data provided by the meteorological institute
Data source	Relative humidity	Weather data provided by the meteorological institute
Data source	Air pressure	Weather data provided by the meteorological institute
Data source	Wind speed	Weather data provided by the meteorological institute
Process input	Absolute humidity	Calculated from air temperature and relative humidity
Sensor	Temperature of gas into each filter compartment	Temperature sensors in filter inlet ducts
Sensor	Temperature of outlet water from integrated heat exchangers	Temperature sensors in heat exchanger pipes
Sensor	Temperature of gas from potroom	Temperature sensor in potroom duct
Combined sensor data	Differential plant pressure	Pressure sensors in potroom doct and outlet ducts from filter
Sensor	Air pressure in pulse air tanks	Pressure sensors in air tanks above each filter
Normalized sensor data	Secondary alumina airlift current	
Normalized sensor data	Flow of secondary alumina leaving secondary silo (to potroom)	
Normalized sensor data	Flow of primary alumina leaving primary silo (to GTC)	

Sensor	Alumina level in secondary silo	Light sensor in silo
Sensor	Alumina level in primary silo	Light sensor in silo
Sensor	HF-concentration in gas from potroom	HF-laser in potroom duct
Process input	Secondary alumina feed to electrolysis cells	Average of sensor measurement above each electrolysis cell

5.1.2 Elkem

In addition to sensors that predate the COGNITWIN project, the Elkem pilot uses a total of three new components: a thermal camera (1) above the ladle during refining, a second thermal camera (2) installed near the tap hole of the furnace and a third thermal camera (3) installed at the casting station. The thermal cameras are delivered by Optris²⁴ (1 and 3) and DIAS²⁵ (2) respectively.

Selection of correct thermal imagers is of paramount importance. Different thermal cameras are optimized for different temperature ranges and tend to have best resolution and accuracy in their preferred range. At temperatures up to 1200°C specialized infrared sensors are required, whereas at higher temperatures it is possible to use cheaper Si-CMOS sensors. Most cameras have optical filters that transmit only a narrow range of wavelengths; an industrial IR-camera is therefore typically characterized and marketed according to its operating wavelength(s).

Of equal importance is to choose a camera that is a good match for the physical characteristics of the target system and environment. A key concept is *spectral emissivity*, i.e., the amount of thermal radiation emitted by a surface at each wavelength. Different materials have different emissivity spectra, and spectra tend to change with temperature and thermodynamic phase. When imaging subjects comprising different material surfaces or mixed fluids, it is worth considering whether to choose an operating wavelength where the subject elements have matching emissivity (if such a wavelength exists) or highly mismatched emissivity. In the former case, one can obtain accurate temperature measurements without knowledge of the subject, e.g., knowing slag-metal ratio in metal processing, or knowing the position of various objects in an image. In the latter case, the thermal images may be utilized, for instance, to measure mixture ratios or perform segmentation tasks – given that the subject temperature is *á priori* known with sufficient accuracy.

Gases also have emissivity spectra that may be harnessed to measure gas temperature; however, the same emissions may interfere with the subject. Understanding the composition of the process atmosphere and the spectral signatures of the component gases is important to avoid costly mistakes.

For the Elkem case, one component camera was chosen primarily to achieve good emissivity matching and accurately measure temperature of unknown fluid mixtures. The other camera was chosen for its ability to penetrate hot process gases with high content of CO and other fumes.

Thermal cameras will be connected to the pilot data lake in two phases. In the initial phase, data will be streamed over ethernet to a virtual machine (VM) on the local plant server. The VM runs a

²⁴ <https://www.optris.global/thermal-imager-optris-pi-1m>

²⁵ http://www.dias-infrared.com/pdf/pyroview640f_eng_mail.pdf

proprietary application provided by the component manufacturer, which processes each image in the video stream and delivers one or more estimated temperatures to a data log. This data log serves as a "virtual sensor" that provides input to the process batch – one batch per ladle.

In the second step, the ambition is to deliver an adaptive machine vision toolbox that can perform both image- and video analysis and adapt highly customized measurement schemes: flow measurements, segmentation and morphology, texture analysis, etc.

A list of sensors and data sources is provided in Table 8. Sensor and PLC data streams are uploaded via OPC-UA to Cybernetica's CENIT server. However, some manual measurements at the Elkem pilot are not available on OPC and are only stored in Elkem’s Oracle database. To have these measurements available on OPC, Cybernetica has created a bespoke extension to the Cybernetica OPC UA Server for the Elkem pilot. This extension queries a set of tables in the Oracle database to extract the relevant measurements and publishes them to OPC. This simplifies the employment of the digital twin, as it allows for using OPC as the single data source.

Table 8: Sensors and data sources used in the Elkem pilot model

COMPONENT TYPE	PARAMETER	SENSOR AND LOCATION
Sensor	Temperature of tapped material	Thermal camera in front of tap hole
Sensor	Ladle weight before and after tapping	Multiple load cells: tapping wagon, scales on floor
Sensor	Ladle weight before and after refining	Load cell on ladle truck
Process input	Furnace and tap hole used	Manual registration
Process input	Ladle used	Manual registration
Sensor	Bath temperature before and after refining	Thermocouple (dip sampling) at refining station
Sensor	Bath temperature during refining	Thermal camera
Sensor	Bath chemistry before refining	Lab analysis of dip sample
Sensor (NEW)	Slag content in product. Temperature of metal during casting.	IR camera at near casting belt
Sensor	Metal chemistry after refining	Lab analysis of end sample (after casting)
Sensor	Level of metal bath	Radar rangefinder above bath
Sensor	Mass of additives	Multiple load cells under silos
Process input	Manual addition (plunging)	Manual registration
Sensor	Lance position	Position sensor
Method	Yield of alloying materials	Calculated from mass of additives, chemical analysis

Method	Refining effect	Calculated from mass of additives, chemical analysis
Process input	Material grade	Manual registration
Process input	Recipe	Manual registration
Sensor	Mass of cast metal	Load cell under dumper
Sensor	Cast metal temperature before crushing	Pyrometer above dumper

Since D4.3, Elkem has successfully installed and demonstrated a thermal camera at the casting station. The camera views the runner where liquid metal is entering from the ladle, as well as a portion of the casting belt as depicted in Figure 36. Image analysis can be used to quantify the amount of slag that may follow the liquid metal out from the ladle. Most customers have a very limited tolerance for slag in their processes, so it is important to be able to detect possible slag contamination before the product is shipped to the customer.

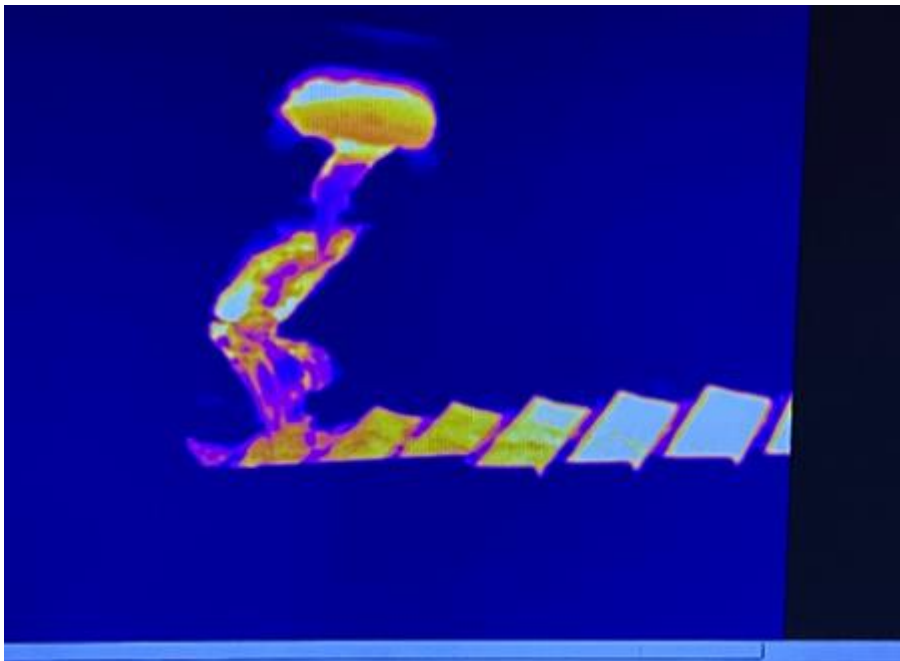


Figure 36 – Screenshot from IR camera monitoring casting.

The cameras are operated through *Cybernetica InSight*, an OpenCV-based tool for data acquisition and edge analysis of camera data. The tool has been augmented for the Elkem pilot to allow use with thermal imagers, and new APIs have been developed for communication with the camera models used by the Elkem pilot. At the time of writing, the refining camera is fully online, whereas the tapping camera has been operated manually and data analyzed offline.

There is an ongoing effort to develop computer vision algorithms for feature extraction. Edge analytics is an absolute requirement, due to the large data volumes (approx. 1GB/s) generated by each camera. Algorithms are being developed offline by researchers at Elkem and SINTEF and will eventually be implemented in *Cybernetica InSight*.

Another ongoing effort seeks to mitigate the effects of dust and vapor deposits on camera lenses. Dirty lenses cause significant sensor drift. The refining camera is particularly affected, due to its top-down installation. Elkem will lead the effort to reduce the effects of lens degradation and develop routines for timely cleaning.

5.1.3 Saarstahl

The data provided for the tracking system is a video stream stemming from 3 high-resolution surveillance cameras and possibly some additional video or image data. Sample video data will be enhanced with synthetic artefacts to provide a large training set for the deep neural network ML system.

To achieve the necessary image processing speed, much of the edge analysis will be performed on local FPGA units. These, along with associated adapters and interfaces, are described in detail in Task 4.4. Other sensor data in the Use Case, e.g. signals used as trigger to start billet identification, are accessed via OPCUA.

A list of sensors and data sources is provided in Table 9.

Table 9: Sensors and data sources used in the Saarstahl pilot model

COMPONENT TYPE	PARAMETER	LOCATION
Sensor	HD video camera	Blooming train
Sensor	HD video camera	Mill train
Process input	Input triggers for billet ID system	Mill train
Process input	Billet ID	Blooming train

5.1.4 Sidenor

Process data in the Sidenor pilot is itemized by ladle heat, i.e., a database entry is generated for each ladle/batch of steel produced. In addition to these, Sidenor is collecting data on refractory wear each time a ladle is repaired or decommissioned. This data comprises a reference/training set for the process data.

The MES data is stored in an Informix database with 1-second granularity, whereas the heat process data is stored in a separate MySQL database. Due to security the other partners could not be given direct access to Sidenor data at this stage. The data exchange is therefore currently done manually by Sidenor, exporting the relevant data from the different data sources internally, and copied as files to the other partners. Later in the project Sidenor plans to provide a mirror of the data system, allowing model developers to read online and historic data. Writing data will not be allowed.

A list of sensors and data sources is provided in Table 10.

Table 10: Sensors and data sources used in the Sidenor pilot model

COMPONENT TYPE	PARAMETER
----------------	-----------

Process input	MES data from refining process
Process input	Date of production
Process input	ID number of the ladle used in the heat (numbers from 1 to 16)
Process input	Heat ID Number
Sensor	% Sulphur composition in the steel after vacuum
Sensor	Electrical consumption
Sensor	Kg of lime added during Secondary Metallurgy
Process input	Time with steel
Sensor	Kg of lime added during Secondary Metallurgy
Sensor	% Sulphur composition in the steel at tapping
Sensor	Burner Preheating
Process input	Vacuum time (<1 Torr)
Sensor	Stirring gases(Ar/N2)
Sensor	Kg of FluroSpar added during Secondary Metallurgy
Sensor	Tons of Liquid Steel
Process input	Billet format
Process input	Desulphuration rate
Process input	Time heating during Secondary Metallurgy
Sensor	% of EAF Slag measured at tapping
Sensor	% Mn composition at tapping

5.1.5 Noksel

The Noksel pilot collects data from a set of PLCs on the factory floor. At the PLCs, distributed sensor data is bundled, and uploaded over Profinet to an OPC gateway and MQTT broker.

In addition to sensors that predate the COGNITWIN project, the Noksel pilot will make use of one new component: a newly installed vibration sensor for condition monitoring of motors. The sensor is manufactured by IFM. Their website may be consulted for additional technical detail and specifications.²⁶

The component has been in operation since autumn 2020. The component is a two-terminal electromechanical (piezo-MEMS) device that provides a single analogue output. The signal is digitized and logged by a local PLC.

The component is not a pure accelerometer: it contains on-board rectification and prefiltering and reports RMS vibration level with a 10Hz low-pass topology.

The drawback of such a sensor, as opposed to an unfiltered accelerometer, is that it erases information about the underlying waveform. Thus, it is not appropriate for, e.g., vibration spectrum analysis. The

²⁶ <https://www.ifm.com/de/en/product/VTV122>

component has, however, two major advantages: it supports low-rate or intermittent sampling (accelerometers must be sampled at a rate twice the desired Nyquist frequency, typically thousands of times per second), and it does not require any edge analysis or calibration: the readout is an *ad-hoc* vibration strength parameter that can be digested at face value by top-level analytics.

Alternative vibration sensors with digital and/or WiFi-output and on-board spectrum analysis were considered; however, the simpler analog sensor was selected on the recommendation of the motor manufacturer, who has also assisted with mechanical installation of the component. The power and flexibility of Noksel's PLCs allows for virtually any sensor interface; it is therefore not a priority to find wireless – or even digital – solutions.

A subset of sensors and data sources for the Noksel pilot is provided in Table 11. For confidentiality reasons, NOKSEL wishes not to release a comprehensive list of sources.

Table 11: Sensors and data sources used in the Noksel pilot model

COMPONENT TYPE	PARAMETER	LOCATION
Sensor	Current	Motors
Sensor	Temperature	Various machine components and environment
Sensor	Vibration	Motors
Sensor	Pressure	Various machine components
Sensor	Oil temperature	Various machine components
Sensor	Oil pressure	Hydraulic Power Unit
Sensor	Oil contamination	Hydraulic Power Unit
Process input	Edge milling rpm	SWP
Process input	Alarm data	All machine components
Process input	Status data	All machine components
Process input	Production parameters	SWP

A cognitive digital twin has been developed and in operation at NOKSEL Iskenderun facility where environmental temperature data has been used.

5.1.6 Sumitomo SHI FW (SFW)

In the first iteration, the SFW pilot will contain no new components. However, the project has delivered an investigation of acoustic condition monitoring with the aim of providing a new component to the raw data layer.

The objective of the study was to determine whether active and/or passive acoustic sensing methods could detect the buildup of fouling on the primary heat exchangers in SFW's boilers. Initial investigations were performed in SINTEF's lab facilities, using sample piping from SFW. This was followed by a 6-month test deployment at the pilot plant, with regular off-line data transfer and analysis.

Table 12: Sensors and data sources used in the SFW pilot model

COMPONENT TYPE	PARAMETER	LOCATION
Sensor	Temperature of air supply	Secondary air duct
Sensor	Temperature of furnace bed	Furnace bed
Sensor	Temperature of fluidization	Various locations in furnace chamber
Sensor	Flue gas temperature	Various locations in flue
Sensor	Water temperature	Heat exchanger supply lines
Sensor	Steam temperature	Heat exchanger output lines
Sensor	Air supply flow	Air supply duct, flue
Sensor	Coolant flow	Water and steam pipes
Sensor	Fuel supply rate	Infeed silos
Process parameter	Fuel mixing ratios	
Sensor	Flue gas composition	

The SFW pilot case contained an exploratory activity to develop a sensor system for detecting, measuring, and monitoring fouling on heat exchanger surfaces.

Since D4.2, a preliminary acoustic concept has been refined using the science of modal analysis. Sumitomo SHI FW has developed numerical simulations to estimate the sensitivity of the method. SINTEF has tested and characterized candidate vibration sensors and designed an active-sensing solution. Finally, the system has been deployed at the SFW Pilot, and various forms of data analysis have been explored, with particularly emphasis on the highest-quality data from the final 60 days.

In parallel, analytical tools have been developed by U. Oulu that estimate fouling through a sophisticated energy balance calculation. This calculation results in a quantitative estimate of the heat exchange coefficient of the steam pipes. With appropriate normalization, this quantity is used as a proxy for fouling level.

Since physical ground-truth data cannot be obtained, the two methods have been compared in a "blind-leading-the-blind" approach. This comparative analysis shows good agreement between these completely separate methods.

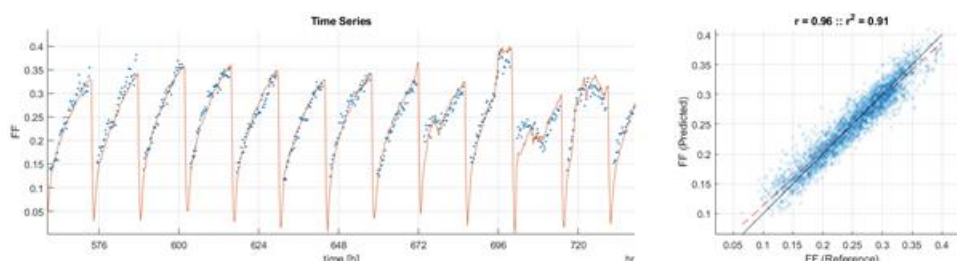


Figure 35: Left: Subset of time-series showing thermodynamic model predictions (red) and acoustic predictions (blue). Right: Scatterplot of acoustic vs thermodynamic predictions.

5.1.7 Data QA Tools Analysis

Data QA at the sensor level may be performed on single variables or groups of connected sensors. The advantages of local quality assurance are twofold. Firstly, bad, missing, or corrupt data is detected early and prevented from comingling with or "infecting" good data. Secondly, on-site computing resources are the ideal place for operator-data interaction, where data boundaries may be set based on the experience, intuition, and domain expertise of plant personnel.

Sensor-level QA should require minimal input from external sources such as process data or sensors with differently structured data; high-level analytics involving the full data landscape are more appropriately implemented in the AI/ML toolbox.

Some common data QA tasks are listed below. Additional tools may be developed according to the needs of the pilots.

- Recognition of error codes in sensor metadata (where available).
- Detection of physically improbable data: out of range, discontinuous, noisy, or otherwise violating process physics.
- Detection of missing data. Some sensors will return a default value, e.g. zero or -1, when no value is received. Others will return the last-known value (see Hydro description above), leading to a "flat-line." In the latter case, contextual logic is required to determine if the constant value is expected for this data source.
- Zero-point drift. This is most easily detected as a monotonic trend in periodic statistics (daily or weekly min/max/mean values)
- Comparative analysis, e.g. monitoring the strength of correlation between two data sets.

Ultimately, at the sensor edge it should be a focus of data QA to interpret the sensor and detect un-physical behavior. If a sensor is reporting improbable values or exhibiting behavior that is uncharacteristic of the process in question, this is much easier to detect and repair locally than after the data has been mixed, merged, and dressed in the pipelines.

We expect, therefore, that the QA system will be a point-of-entry for operator expertise, by allowing the user to input, e.g., typical process values, saturation and zero-point sensor values, and adjust tolerance of outliers and spikes in the time series.

Data Quality issues could be divided into detectable and undetectable errors. The detectable errors can be automatically identified and cleaned while undetectable errors must be evaluated by sampling and can be model by data quality matrices that can be analysed by machine learning algorithms [BLA+08]. There are various tools for both type of errors that were investigated and analysed during the last period of the project.

For detectable errors, the following tools are evaluated to support data validation and cleaning:

- Panda Profiling²⁷ provides Python API to quickly identify data quality issues such as missing values, data type mismatch as well as provide descriptive statistics about the data.

²⁷ <https://github.com/pandas-profiling/pandas-profiling>

- Great Expectations²⁸ also provides API for automatic data testing, documentation and profiling. Great Expectations supports testing data quality with a library of expectations that are declarative statements that can be evaluated by computer and semantically meaningful to humans. Expectations can be defined by users and covers different kinds of common data issues such as "*expect column values to not be null*" or "*expect column median to be between*".
- DataGraft²⁹ is a cloud-based service support data transformation (interactively building, modifying, and sharing of repeatable and reusable data transformations), data cleaning and reconciliation. This service can be used to repair common data quality problems such as missing, incorrect or badly formatted values, and duplicated records.
- OD-tool is a Matlab implementation tool that supports data driven modeling of system dynamics, a general-purpose data quality tool focusing on systematic detection and removal of outliers in a batch data. Most importantly, the tool implements the ellipsoidal peeling method, based on detection of outliers using statistical analysis, and exploiting the potential that a removal of outliers can significantly improve the performance of process identification. OD-tool is included in the set of tools in the FouMon component and is openly available for download. The tool has been applied in the context of the Sumitomo SHI FW pilot.

To repair erroneous data with undetectable errors, existing machine learning methods for data quality will be evaluated. Particularly, ERDRE framework³⁰ - a machine learning pipeline for erroneous data repair – can be adapted to enable automatic detection of anomalies in data for both short-term streaming data and long-term historical data as well as repair and clean the data.

5.1.8 Data connectors

It is an unspoken, and possibly coincidental consensus in the project to use OPC-UA for sensor-to-storage communication. Of the six pilots, four are using some form of OPC-UA: Hydro, Elkem, Noksel and Sumitomo SHI FW.

The implementations differ somewhat, with methods and extensions tailored to the needs of each pilot and data platform. As a result, there are three unique tools developed for this task:

- A server application with a database query extension, as described in Section 5.1.2, developed by Cybernetica.
- An OPC-to-MATLAB communication tool developed by U. Oulu.
- A data connector environment for PLC communication developed by TEKNOPAR.

Since D4.2, the Elkem pilot has come partially online, with one camera connected to Cybernetica's OPCUA server.

The Sumitomo SHI FW pilot has its new sensor installed and logging, but the remote data extraction is performed manually. Sumitomo SHI FW, SINTEF, and the pilot plant are in discussion about whether

²⁸ <https://greatexpectations.io/>

²⁹ <https://datagraft.io/>

³⁰ <https://github.com/SINTEF-9012/Erdre>

to upgrade to an online sensor. Since these are experimental devices that will be removed at end of project, streamlined data uplink is low priority.

5.2 Summary of the key achievements

To date, the key achievements of Task 4.3 are the selection and installation of sensor hardware in all pilots that have requested new components as well as analyzing the requirement for sensor data QA and identify relevant tools as describe above. The activities are summarized in the table below.

Table 13: Novel components for COGNITWIN pilots

PILOT	COMPONENT	STATUS
Hydro	TDL gas monitor for HF in pot stack	Installed and online
Elkem	1µm Si-CMOS thermal imager for monitoring of refining process	Temporarily installed, logging video data to local client.
Elkem	3.9µm MBM thermal imager for monitoring of tapping process	Installed. Manual data collection.
Noksel	Analog vibration sensor on DC motor	Installed and online
Noksel	Control panel to manipulate environmental temperature in the welding generators room	Designed, implemented and installed
Saarstahl	Machine vision cameras	Installed and online
Sumitomo SHI FW	Accelerometers	Installed for limited test campaign. Now decommissioned.
	Fouling monitoring data processing	Outlier tool implemented.

5.3 Conclusion

At the top of this section, we outlined the four goals of Task 4.3:

- 1) Selection of sensors and development of novel sensors
- 2) Digital tools for edge data processing
- 3) Data quality assurance
- 4) Data connectors

During the project all four of these have been addressed for each pilot. New sensors have been selected, tested, and brought online through appropriate connectors - generally based on OPC-UA. Edge analytics, typically through a local PC client, have been developed to filter raw data into robust measurements, and QA systems have been implemented as necessary, with additional QA tools made available through the COGNITWIN toolbox for possible future implementation.

Table 14: Real time sensor data processing – challenges, requirements and solutions

Pilots	Real time sensor data processing (by task 4.4) – Challenges, Requirements and Solutions
Hydro	<ul style="list-style-type: none"> • Challenge: Local soft real time sensor data will be needed • Requirement: The required timing should allow for OPC UA data access • Solution: Use OPC UA data access
Elkem	<ul style="list-style-type: none"> • Challenge: Local real time sensor data is needed for temperature analysis. • Requirement: Ensure local/edge processing • Solution: Support imagery/thermal sensor processing at the edge with the sensor associated software.
Saarstahl	<ul style="list-style-type: none"> • Challenge: Video images need to be analysed in real time to understand the billets movement • Requirement: Neural Network inference speed needs to be optimized for real time event handling. • Solution:
Sidenor	<ul style="list-style-type: none"> • Challenge: The refractory analysis is not strict realtime • Requirement: Provide data as needed for the Digital Twin and model analysis • Solution: Uses collected data without strict realtime needs
Noksel	<ul style="list-style-type: none"> • Challenge: Latency for machine data • Requirement: Real time processing • Solution: Application of pretrained models on stream data using SPARK
Sumitomo SHI FW	<ul style="list-style-type: none"> • Challenge: The fouling analysis is done based on soft realtime data • Requirement: Provide the necessary processing support for the analysis • Solution: Sensor data access through OPC UA provides the necessary support, tool for data preprocessing.

Pilot typ	Pilot	Task	WP4 Toolbox COMPONENTS																				
			MAI (SINTEF)	Cybernetica OPC UA (Cybernetica)	FUSE OPC UA (UOLU)	TEKNOPAR: TIA DATA (TIA STREAM, TIA STORAGE)	BD Pipelines DF (SINTEF)	Honir (Scortex)	Lodur (Scortex)	Bedrock (SINTEF)	TEKNOPAR: TIA IOT (TIA SENSOR, TIA PLC, TIA FAST (Fraunhofer)	TEKNOPAR: TIA DATA-GEN	SINDIT (SINTEF)	StreamPipes + Siddhi (Fraunhofer)	CEP Editor (Fraunhofer)	IDS Connectors (SINTEF)	Trusted Factory Connector (Fraunhofer)	COGNITWIN Toolbox Portal (SINTEF)	TEKNOPAR: TIA UX	Cybernetica Viewer (Cybernetica)	Sensor library (SINTEF)	Sensor data quality framework (SINTEF)	
	Hydro	Realtime sensor/data processing												(x)	(x)								
	Elkem	Realtime sensor/data processing												(x)	(x)								
	Saarstahl	Realtime sensor/data processing												(x)	(x)								
	Sidenor	Realtime sensor/data processing								x				x	(x)								
	Noksel	Realtime sensor/data processing			x									x	(x)								
	Sumitomo	Realtime sensor/data processing												(x)	(x)								

6 Realtime sensor/data processing - Connecting and Synchronizing Assets and Digital Twins

6.1 Objectives, challenges and components

As Digital Twins (DTs) are a digital representation of a physical (or virtual) asset, synchronization between the asset(s) and the DT is crucial. The main objective therefore was to provide the necessary means to connect and synchronize assets to the FA³ST Service³¹ representing the DT. Assets come in all shapes and forms, meaning there is no standardized API or even communication protocol to talk to them which is the biggest challenge to overcome when developing a generic approach to asset connectivity.

6.2 Detailed description of the activities performed

To address the challenge of asset heterogeneity, we introduced the concept of *asset connections* in FA³ST Service by adding the software interface *AssetConnection*. This interface defines the three types of generic interactions that communication protocols can potentially support: read/write a value, subscribe to a value, and execute an operation. Defining such an interface allows to decouple the DT (here represented by FA³ST Service) from all potential concrete implementations of this interface for different communication protocols.

Figure below shows how asset synchronization can be configured by linking submodel elements of an AAS to different providers of an asset connection. An asset connection holds a set of key-value pairs for each of the three provider types where the key is an AAS reference representing the submodel element the provider should be applied to and the value is the provider configuration. This is represented in figure below by qualified associations between the asset connection and the different provider types. The design allows for flexible configuration of asset connections and supports complex use cases, e.g., reading/writing the value of an element via HTTP while keeping its values up-to date via an MQTT or OPC UA subscription. More information can be found in [Jac+22].

³¹ <https://github.com/FraunhoferIOSB/FAAST-Service>

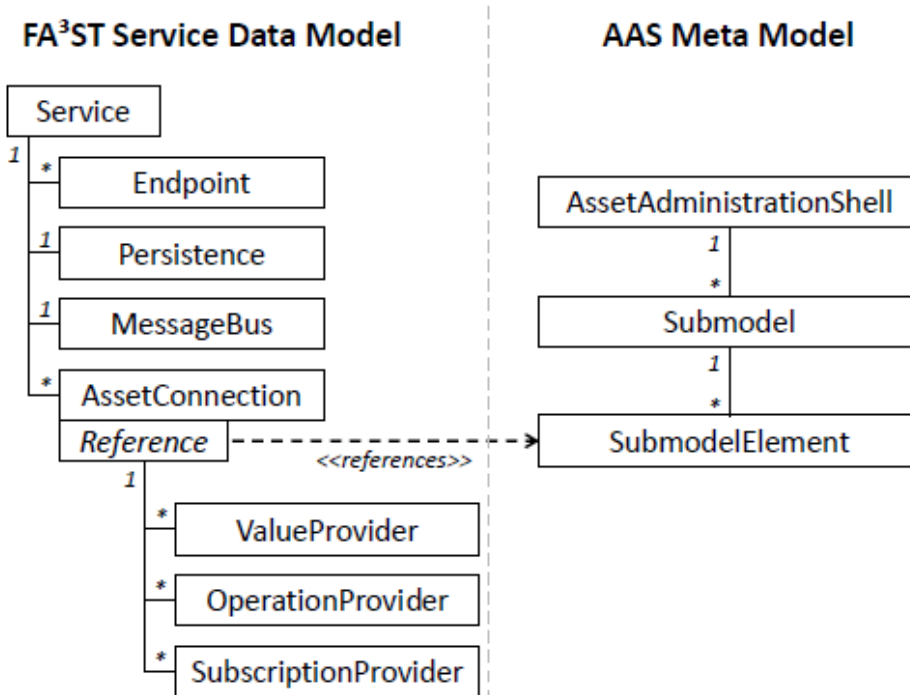


Figure 36: UML class diagram of FA³ST Service Data Model and AAS Metamodel explaining linking of asset connections with submodel elements

Besides the *AssetConnection* interface, FA³ST Service also provides three implementations for the most common communication protocols used in industrial plants: OPC UA, HTTP and MQTT.

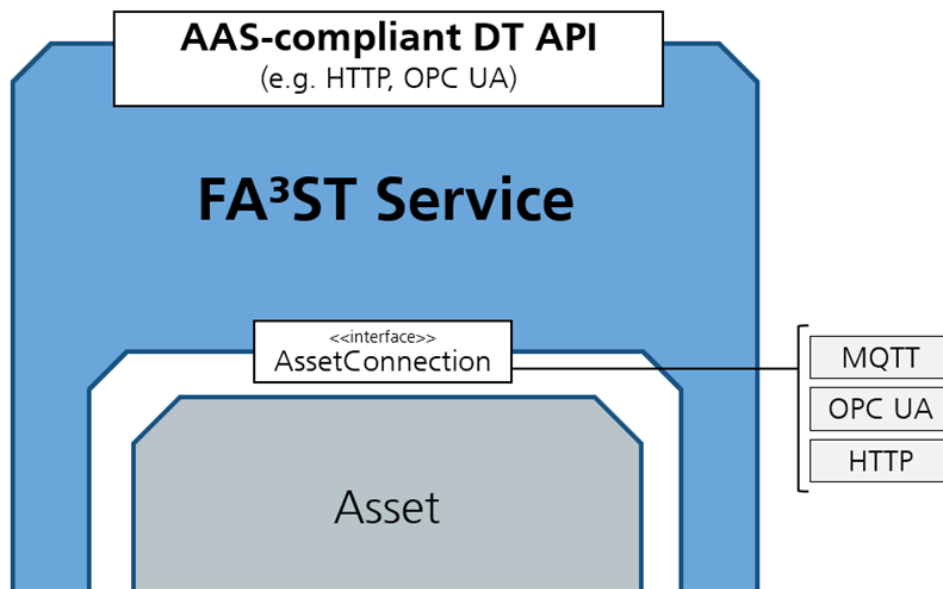


Figure 37: FA³ST Service architecture w.r.t. asset connections

Protocol	Read	Write	Execute	Subscribe
HTTP	✓	✓	✓	✓
MQTT	✗	✓	✗	✓
OPC UA	✓	✓	✓	✓

Figure 38: Protocols delivered with FA³ST Service and the supported interaction patterns

FA³ST Service also offers a user-friendly way to setup and use any implementation of the *AssetConnection* interface via configuration file, i.e. without requiring the user to write any code. This not only works for the already developed implementations (OPC UA, HTTP, and MQTT) but also for any custom or third-party implementation available in the future.

A concrete example for OPC UA asset connection could look like this:

```
{
  "@class": "de.fraunhofer.iosb.ilt.faaast.service.assetconnection.opcua.OpcUaAssetConnection",
  "host": "opc.tcp://localhost:4840",
  "valueProviders":
  {
    "(Submodel){IRI}urn:aas:id:example:submodel:1,(Property){ID_SHORT}Property1":
    {
      "nodeId": "some.node.id.property.1"
    },
    "(Submodel){IRI}urn:aas:id:example:submodel:1,(Property){ID_SHORT}Property2":
    {
      "nodeId": "some.node.id.property.2"
    }
  },
  "operationProviders":
  {
    "(Submodel){IRI}urn:aas:id:example:submodel:1,(Operation){ID_SHORT}Operation1":
    {
      "nodeId": "some.node.id.operation.1"
    }
  },
  "subscriptionProviders":
  {
    "(Submodel){IRI}urn:aas:id:example:submodel:1,(Property){ID_SHORT}Property3":
    {
      "nodeId": "some.node.id.property.3",
      "interval": 1000
    }
  }
}
```

6.3 Progress beyond State of the Art or State of the Practice

The concept of asset connections is not yet mentioned or even defined in DT standards although it is an essential key functionality of a DT. Although other DT implementations also identified this issue, they typically do not provide easy-to-use solutions to this problem and often leave it up to a developer to manually implement this connectivity in code. To our knowledge, our approach is the only one so far to understand asset connectivity as a crucial functionality which should reside within the DT itself and therefore should be performed by the DT itself rather than some external tool or code.

Providing a generic interface for easy integration of arbitrary communication protocols as well as our user-friendly approach for establishing asset connections without writing code both represent further progress beyond the state of the practice and improves usability of DTs in real-world scenarios.

FA³ST Service can be configured via a JSON-based configuration file or code. Another approach to configuring and mapping asset connections would be to store this information in the AAS model itself (e.g., as specialized type of submodel) [Jac+22]. For FA³ST Service we decided to keep this information separated from the actual AAS model, as it might contain sensitive data such as credentials or details about internal network structure.

6.4 Summary of the key achievements

We have developed an open-source implementation of the DT that can easily be connected to and synchronized with assets using arbitrary communication protocols (OPC UA, HTTP, and MQTT are provided, others require implementing the *AssetConnection* interface) which is a crucial functionality of a DT but rather neglected by existing DT implementations. We also provide a simple and user-friendly way to configure and establish connectivity between assets and DTs.

6.5 Conclusion

Synchronizing DTs with their underlying asset(s) is an essential task for DTs but unfortunately not part of any specification and often ignored by implementations. With the concept of asset connections in FA³ST Service we deliver a communication protocol-agnostic approach to solve this problem.

7 Reflection on the use cases – from the WP4 perspective

7.1 Analysis of requirements from the pilots

Table 15: Overview of WP4 components in use and (possible use) by pilots

Pilot type	Pilot	Task	WP4 Toolbox COMPONENTS																				
			MAI (SINTEF)	Cybernetica OPC UA (Cybernetica)	FUSE OPC UA (UOULU)	TEKNOPAR: TIA DATA (TIA STREAM, TIA STORAGE)	BD Pipelines DF (SINTEF)	Honir (Scortex)	Lodur (Scortex)	Bedrock (SINTEF)	TEKNOPAR: TIA IOT (TIA SENSOR, TIA PLC, TIA)	FAST (Fraunhofer)	TEKNOPAR: TIA DATA-GEN	SINDIT (SINTEF)	StreamPipes + Siddhi (Fraunhofer)	CEP Editor (Fraunhofer)	IDS Connectors (SINTEF)	Trusted Factory Connector (Fraunhofer)	COGNITWIN Toolbox Portal (SINTEF)	TEKNOPAR: TIA UX	Cybernetica Viewer (Cybernetica)	Sensor library (SINTEF)	Sensor data quality framework (SINTEF)
Non-ferrous	Hydro	WP4																					
	Hydro	COGNITWIN Interoperability Toolbox													(x)				x				
	Hydro	Digital Twin Cloud Platform, Data Space and Cyber Security	x	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)			(x)	(x)			(x)	(x)		
	Hydro	Sensors, Understanding Sensor Data & Quality Assurance	x																			(x)	(x)
	Hydro	Realtime sensor/data processing													(x)	(x)							
	Elkem	WP4																					
Elkem	Elkem	COGNITWIN Interoperability Toolbox													(x)				x				
	Elkem	Digital Twin Cloud Platform, Data Space and Cyber Security	x	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)			(x)	(x)			(x)	(x)		
	Elkem	Sensors, Understanding Sensor Data & Quality Assurance	(x)																			(x)	(x)
	Elkem	Realtime sensor/data processing													(x)	(x)							
	Steel	Saarstahl	WP4																				
	Saarstahl	COGNITWIN Interoperability Toolbox													(x)				x				
Saarstahl	Digital Twin Cloud Platform, Data Space and Cyber Security		(x)	(x)	(x)	(x)	x	x	(x)	(x)	(x)	(x)	(x)		(x)	(x)			(x)	(x)			
Saarstahl	Sensors, Understanding Sensor Data & Quality Assurance	(x)																			(x)	(x)	
Saarstahl	Realtime sensor/data processing													(x)	(x)								
Sidenor	Sidenor	WP4																					
	Sidenor	COGNITWIN Interoperability Toolbox													(x)				x				
	Sidenor	Digital Twin Cloud Platform, Data Space and Cyber Security		(x)	(x)	(x)	(x)	(x)	(x)	x	x	(x)	x			(x)	(x)			(x)	(x)		
	Sidenor	Sensors, Understanding Sensor Data & Quality Assurance	(x)								x											(x)	(x)
	Sidenor	Realtime sensor/data processing									x				x	(x)							
	Noksel	WP4																					
Noksel	Noksel	COGNITWIN Interoperability Toolbox													(x)				x				
	Noksel	Digital Twin Cloud Platform, Data Space and Cyber Security		(x)	(x)	x	(x)	(x)	(x)	(x)	(x)	x	x	(x)		(x)	(x)			(x)	(x)		
	Noksel	Sensors, Understanding Sensor Data & Quality Assurance	(x)			x																(x)	(x)
	Noksel	Realtime sensor/data processing				x									x	(x)							
	Engineering	Sumitomo	WP4																				
	Sumitomo	COGNITWIN Interoperability Toolbox													(x)				x				
Sumitomo	Digital Twin Cloud Platform, Data Space and Cyber Security		(x)	x	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)		(x)	(x)			(x)	(x)			
Sumitomo	Sensors, Understanding Sensor Data & Quality Assurance	(x)																			(x)	(x)	
Sumitomo	Realtime sensor/data processing													(x)	(x)								

Updated pipelines for the pilots – as reflected in the updated COGNITWIN Toolbox portal includes the components identified to support the various pilots within the various technology areas.

7.2 HYDRO Pilot

Hydro has already established a comprehensive platform for Industrial IoT that is continuously collecting data from thousands of signals coming from machines, sensors, PLCs, and other plant control systems. The raw data is uploaded to a data lake system where it is stored in its original form. Cleansed or aggregated data can also be stored in the data lake whenever this is beneficial. Selected, use case

specific data can be prepared and distributed to users through a Trusted Data Layer (TDL), as shown in Figure 39.

Data from multiple sources are handled and processed for use in the planned pilot pipeline at Karmøy. Process data are collected on Open Platform Communications (OPC UA) tags and then stored in the data lake and distributed via the Trusted Data Layer.

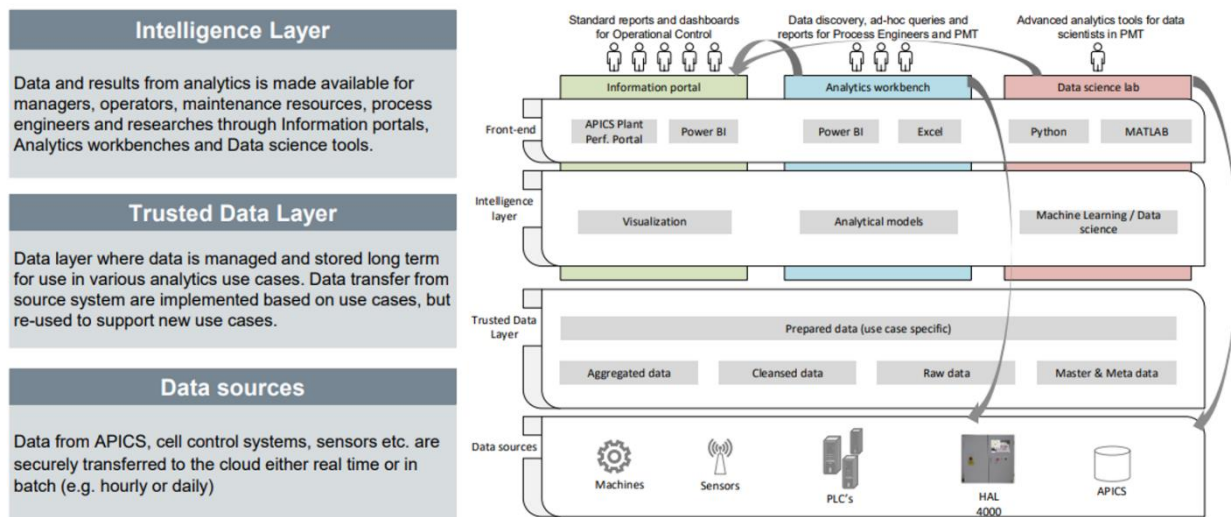


Figure 39: Hydro existing Digital Platform with Trusted Data Layer

The implemented pilot pipeline for online use is illustrated in Figure 40. Process data from the Gas Treatment Centre and the electrolysis cells are collected on Hydro’s own OPC UA server. At the same time, a pilot specific Cybernetica OPC UA server collects additional relevant data such as local weather measurements and primary alumina certificates for incoming shipments. OPC tags from both servers are made available to the digital twin applications. The weather data is collected using the SINTEF MET API Interface (MAPI) that has been created in the COGNITWIN project.

Outputs from the digital twin are stored in a PostgreSQL database on a local machine and the results can be visualised using a Cybernetica Web Viewer application. Data visualisation and investigation of the results by operators (OT users) allows for model-based adjustments and an advisory control loop.

The current approach is that all data communication between the different modules in the pipeline can be done via OPC UA.

Process measurements established during the duration of the COGNITWIN project (HF laser data) were originally only available for offline modelling adjustments. As of the end of the project, HF laser data is available online via Hydro’s own OPC UA server and is used by the digital twin for online model adaptation. The Trusted Data Layer (TDL) is then only needed for offline reference data.

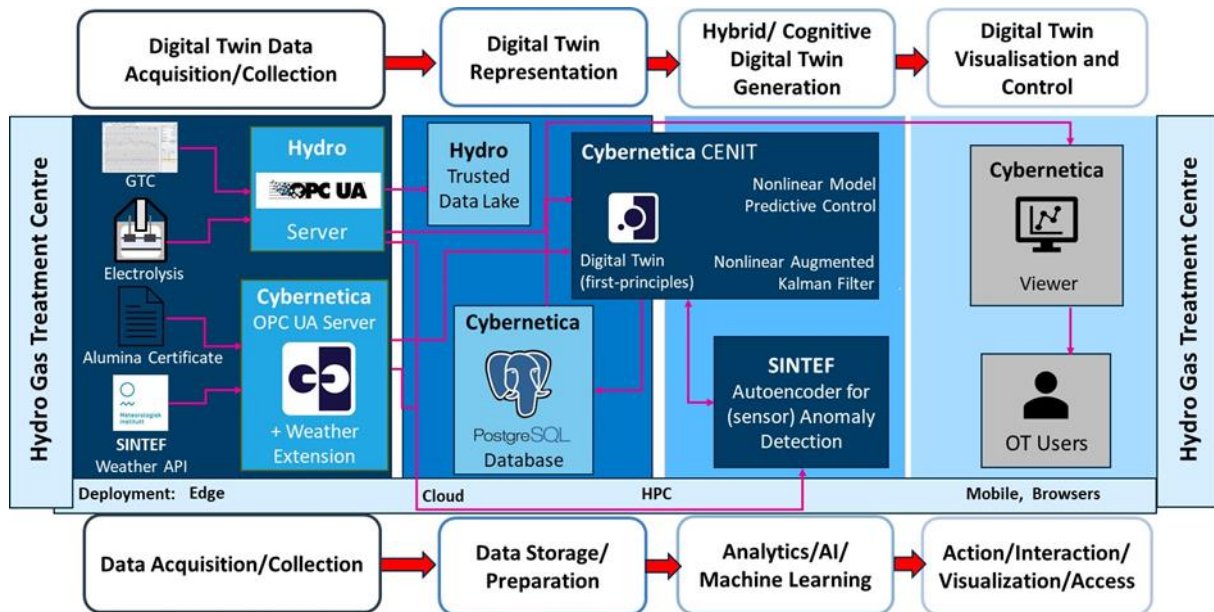


Figure 40: IoT architecture for the Hydro pilot showing interconnectivity of data and toolbox components.

Data sharing and direct access

Sharing of online industrial data from Hydro is very challenging due to experiences in the past where the data systems have been attacked from the outside with the intent of bringing down the production. In addition, there is strong scepticism against letting outsiders (research partners) have general access to data.

The method used in this pilot is that some research partners are supplied with off-line data for analyses purposes, while a few persons from the partners get access to online data. The partner working with the control application must be able to access the online data.

In addition, the data collection infrastructure has, as for any large company, grown dynamically over years, and it is not simple to get access to the needed data, even seen from the inside. Improving on this is outside the scope for the current project. The chosen strategy is to provide the access mediated through the Trusted Data Layer.

These challenges limit the possibilities for externals (research partners, academia) to investigate the impact of "nearby data". "Nearby data" is here data that is thought not to be of importance, but which still could have some relevance.

7.3 SIDENOR Pilot

In this section we explain the main achievements from WP4 perspective in the context of Sidenor pilot. The figure below shows high-level overview of required components and their interaction.

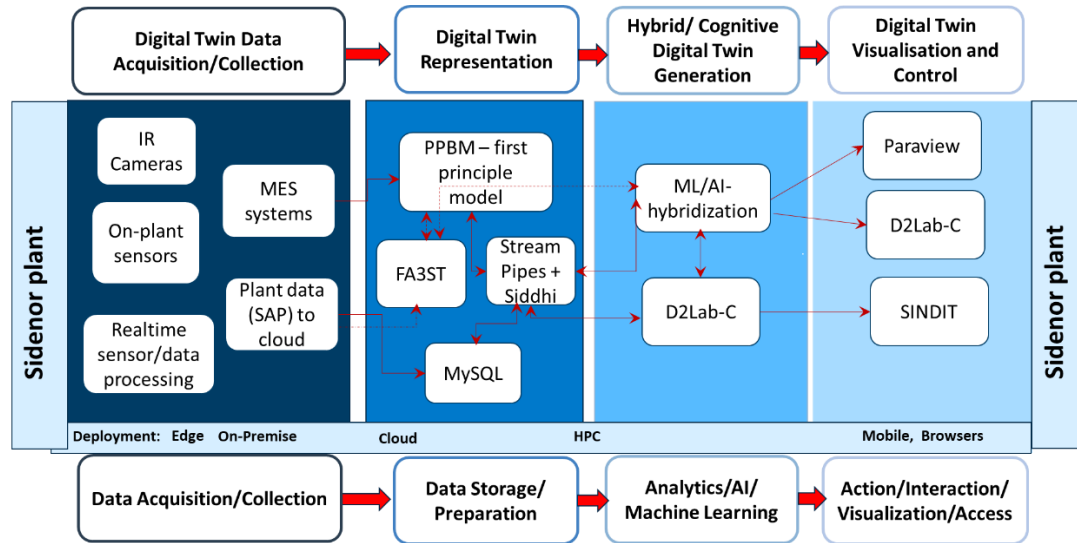


Figure 41: Cognitwin Pipeline in Sidenor use case

The main components are:

- Digital Twin (DT) representation of Sidenor assets based on Asset Administration Shell (AAS) standard (FA3ST).
- Sensors.
- Data Acquisition App (DAA) for reading sensor values and providing them to DT.
- StreamPipes (SP) pipeline services for data processing.
- SINDIT for heterogenous data integration and cognition support.

The following components have been developed and integrated into StreamPipes:

- Sidenor Measurements Buffer (Pipeline #1) – It buffers property values for current cycle of heats that are used by other elements
- Factored Contributions (Pipeline #1) – It calculates contributions of each parameter to the wear of ladle walls – preprocessing step
- Keras Neural Network (Pipeline #1) – It uses Neural Network model to infer state of the ladle
- MEWMA (Pipeline #2) – It performs MEWMA on input data and outputs detected anomalies along with the information which parameter caused it.
- CEP (Pipeline #2) – It uses Siddhi engine to perform Complex Event Processing on its input, thus providing application of complex logic in pipeline

The following pipelines have been implemented:

- Pipeline for acyclic data (Pipeline #1, figure below) - Uses acyclic data to calculate state of the ladle using Neural Network. In addition, it triggers notification when value of particular property goes above certain threshold, displays time between consecutive heats and information about each heat.

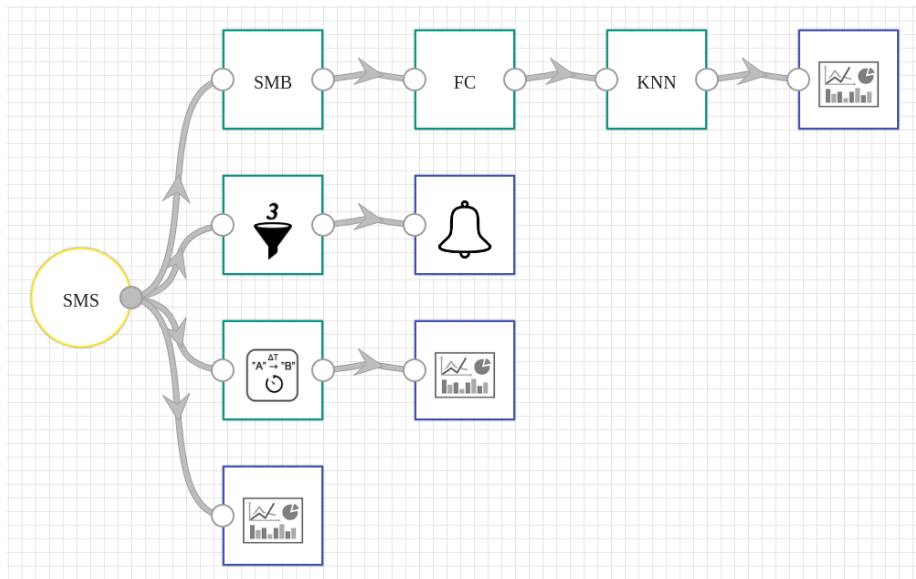


Figure 42: Pipeline for acyclic data

- Pipeline for cyclic data (Pipeline #2, figure below) - Performs MEWMA (Multivariate Exponentially Weighted Moving) analysis on cyclic data and employs CEP (Complex Event Processing) with Siddhi to detect trends in data.

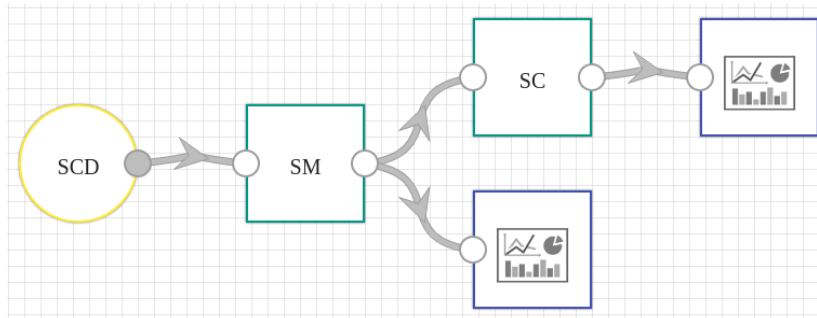


Figure 43: Pipeline for cyclic data

The state of the asset/DT is monitored with sensors located at Sidenor. This state is represented by cyclic data (*time-series data*) and acyclic data (*single values*) for the ongoing ladle usage (*heat process*). Sensor readings are collected with Data Acquisition App which then updates the state of the DT using exposed DT API (*DT property update*).

This data can be polled by an operator that wants to see process information or used by SP pipeline services (*DT services*). These services represent processing components that use available data/information to calculate/infer new information regarding ladle usage. Operator can access this new information, as well – basic process information (*sensor data*) along with the new/inferred one represent overall state of the DT. These DT properties answer operator’s questions regarding the process, such as “What is the ladle/brick state?”, “Should the ladle be repaired?”, “How many heats can the ladle last?”, etc., and provide general information about ongoing heat process – all through DT API.

7.4 ELKEM Pilot

The pilot pipeline for online use in the Elkem pilot case is shown in Figure 44. Data from the process (PLC/ sensors) are made available as tags on Elkem’s OPC server. Additional data from manual measurements are stored in an Oracle database also maintained by Elkem. A tailor-made Cybernetica OPC UA server has been developed to distribute these measurements via OPC UA.

Thus, all data from the process is available for the digital twins on OPC UA. Two digital twin models have been developed: one physics-based twin of the ladle implemented using Cybernetica CENIT and a data driven (AI) slag model developed by SINTEF. Data from the twins are stored on a local PostgreSQL database that can hold both historical and current values. This data can be visualized for the operators using a Cybernetica Viewer application.

The components of the digital twin communicate via the Cybernetica OPC UA Server. Integration with StreamPipes was considered, but was deemed unnecessary for this pilot as the Cybernetica OPC UA Server already facilitated all the necessary signals.

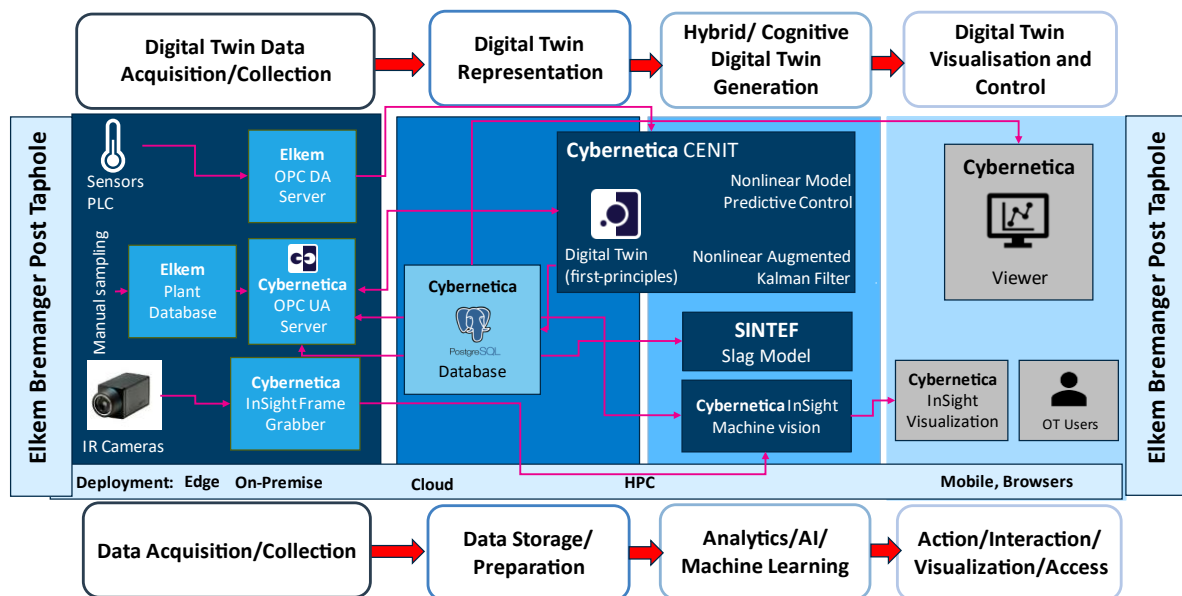


Figure 44: Planned Digital Twin pilot for the Elkem pilot

The evaluation of new sensors for the ELKEM pilot has been described in more detail in chapter 4 – related to WP4 Task 4.2 on sensors and sensor data.

The three thermal imaging cameras planned for the project, one at the refining step, one at the tap hole, and one for the casting step have all been installed at the plant.

To enable the real-time potential of either model, reliable temperature measurements are needed. Due to the challenging operational environment (dust, smoke, flames, high temperature) in addition to the network requirement of the high data transfer, the reliability of the images from the cameras have not been good enough for online use by the models. These challenges have also hindered the development of the machine vision algorithms as the image quality had to be scaled down due to network capacity limits.

The molten subject in the thermal images comprises a mixture of metal, slag, and various bits of solid. Different components have different thermal emissivity, leading to uncertain calibration factors, providing a challenge from a methodological standpoint.

For the camera at the refining station, a set of machine vision algorithms have been developed to measure the metal temperature, slag coverage and slag amount on the ladle surface. The algorithms have been realized online in Cybernetica InSight, but has not been tested over longer periods of time due limitations in the plant network capacity.

For the camera at the tapping station, the algorithms have been developed offline to detect positions of moving equipment, tapped slag and temperature of the tapped metal.

The structure of the online machine vision system is shown in Figure 45.

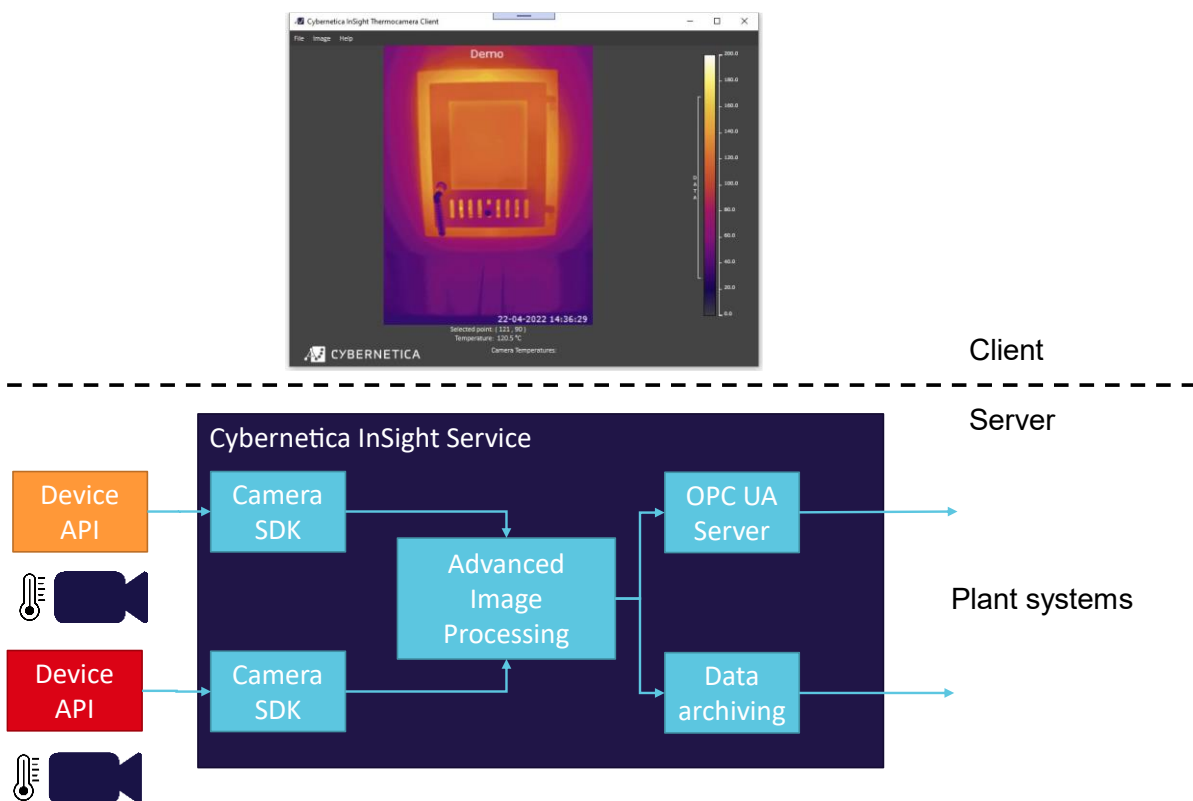


Figure 45: Image analytics pipeline for the Elkem pilot

7.5 SUMITOMO SHI FW Pilot

The Engineering pilot (Sumitomo SHI FW, SFW) considers monitoring and control of heat exchange surfaces in biomass combustion. The evolution of fouling on heat exchange surfaces cannot be directly measured, but information can be obtained via acoustic sensing (see Section 5 on Sensors, Sumitomo plant). An alternative is to use standard process measurements on flows, temperatures and pressures over the heat exchangers, and estimate the heat transfer from flue gases to the water-steam side (see D5.3). In the cases where plant combusts multiple fuel fractions, on-line characterization of the incoming fuel feed characteristics can provide important information for fouling monitoring. For this

purpose, a state estimation tool has been constructed to estimate the uncertain input fuel fragments in the fuel (see D5.2).

Figure 46 shows the SFW pipeline architecture with the Digital Service Platform, connected to the DCS via an OPC/DA connection, and an edge device that is connected to Digital Service Platform as an OPC client.

For the Sumitomo SHI FW pilot, an example setup of data communication was demonstrated via an OPC-UA tool, consisting of free server software (Prosys Simulation Server) and existing properties of Matlab (Mathworks) and StreamPipes (Apache). The OPC-UA setup is described in the Appendix Component Description. The links between Matlab - OPC-UA - StreamPipes have been implemented, tested and demonstrated with the FUSE state estimation tool.

Model-based monitoring and control requires up-to-date models. In order to ensure good data quality, an outlier detection tool was implemented on Matlab, to support the data-driven part of process modelling.

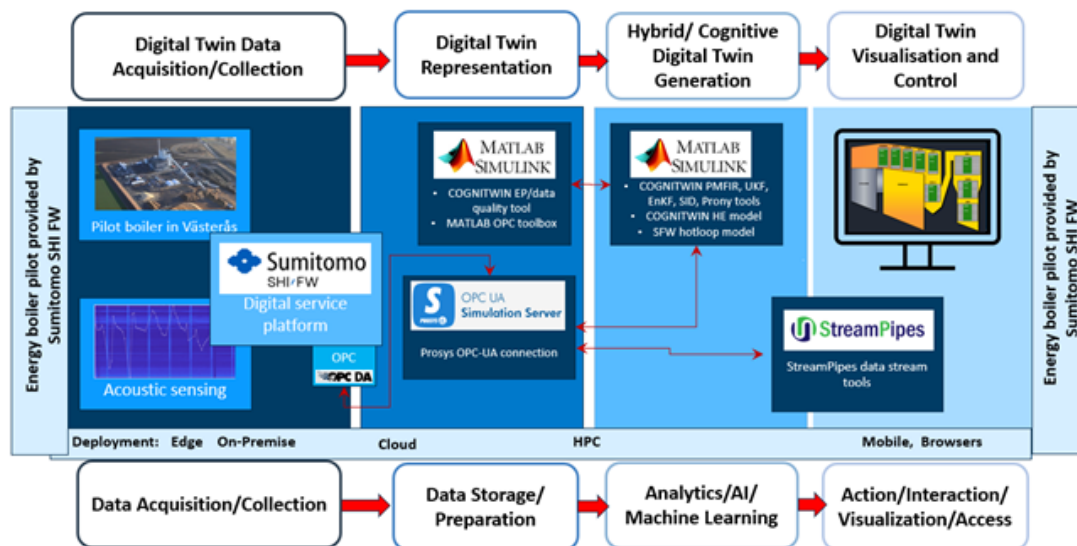


Figure 46: Digital Twin pipeline architecture for the SFW pilot

7.6 SAARSTAHL Pilot

This progress is related to the installation of optical tracking sensory hardware onsite at the Saarstahl production facility, the implementation of integrational components that allow the interoperability of the optical tracking system described in the Use-Case with Saarstahl production planning systems, and the photogrammetric capturing of the Saarstahl production plant to form a generative 3D model that allows the creation of training data for a tracking system.

A technical issue for the Use-Case is that from the realizable camera angles, the rolled bars cannot be separated optically. This means, that rolled bards overlap in the image. A consequence is that the envisioned software architecture consisting of a neural network for the semantic segmentation (i.e. pixel-wise labelling) of rolled bards, followed by a manually programmed component for the linking of bars to sequences, will not work. The reason is that for overlapping bars, a semantic segmentation will

lose the information that the two objects are separate bars, an information that cannot be retrieved later. Rather than using this two-component approach, we will need to shift more responsibility to the machine learning system. We finally decided to deploy a custom network architecture, that exploits temporal coherence in the problem. The network received as input the camera image from the current frame, plus the correct instance segmentation from the previous frame and generates a solution (instance segmentation) for the current frame. This problem is dramatically easier to solve compared to frame-by-frame approaches, as consecutive frames exhibit a large degree of coherence. Obviously, the approach faces problems when a new billet enters the field of view for the first time, but this case can easily be handled (algorithmically) as special case.

Unfortunately, the solution is very specific to the Use-Case, so little insight for the further development of Neuroscope was gained from this. On the positive sight, the exploitation of temporal coherence in the presented form (providing a network with a solution from previous frames) is a relatively fresh research idea and might trigger research in different areas of application.

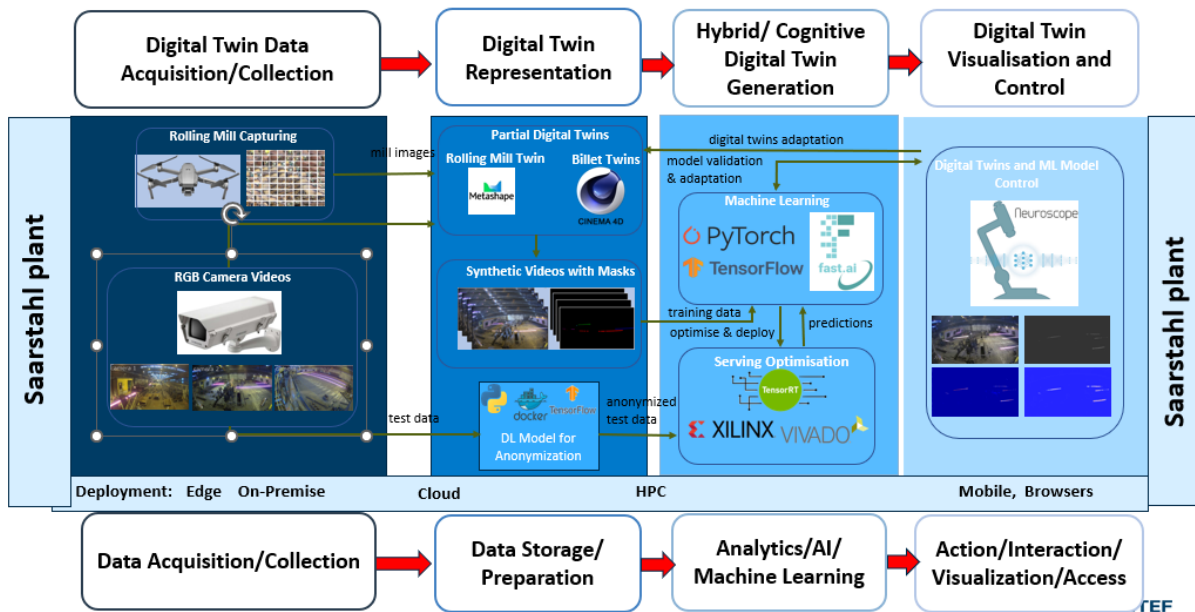


Figure 47: The Digital Twin – Machine Learning pipeline for the Saarstahl pilot case

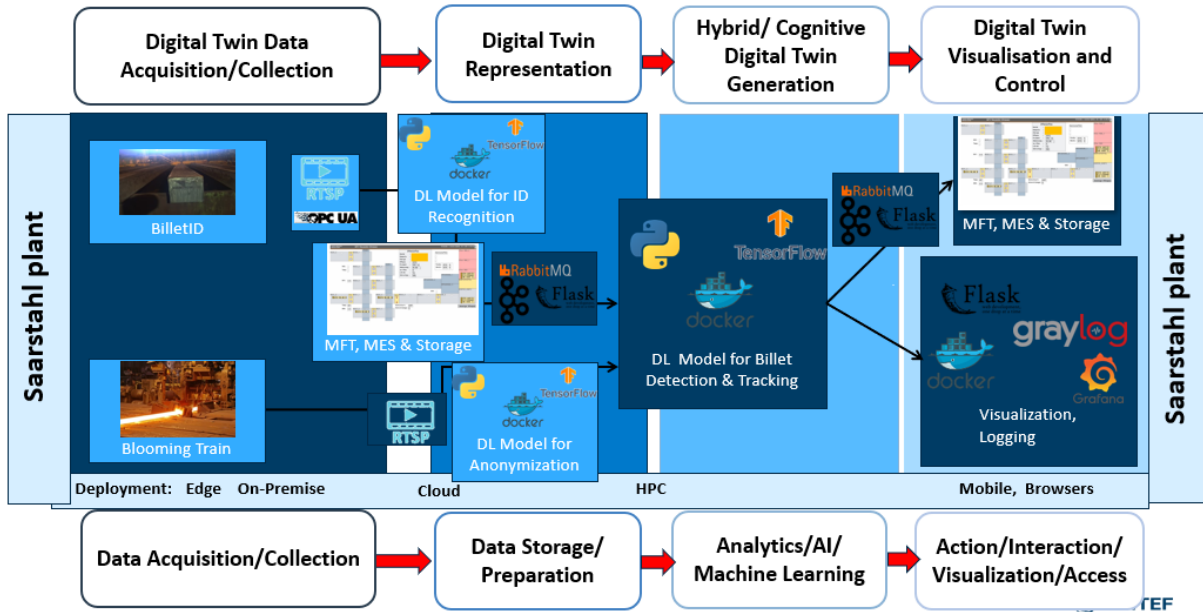
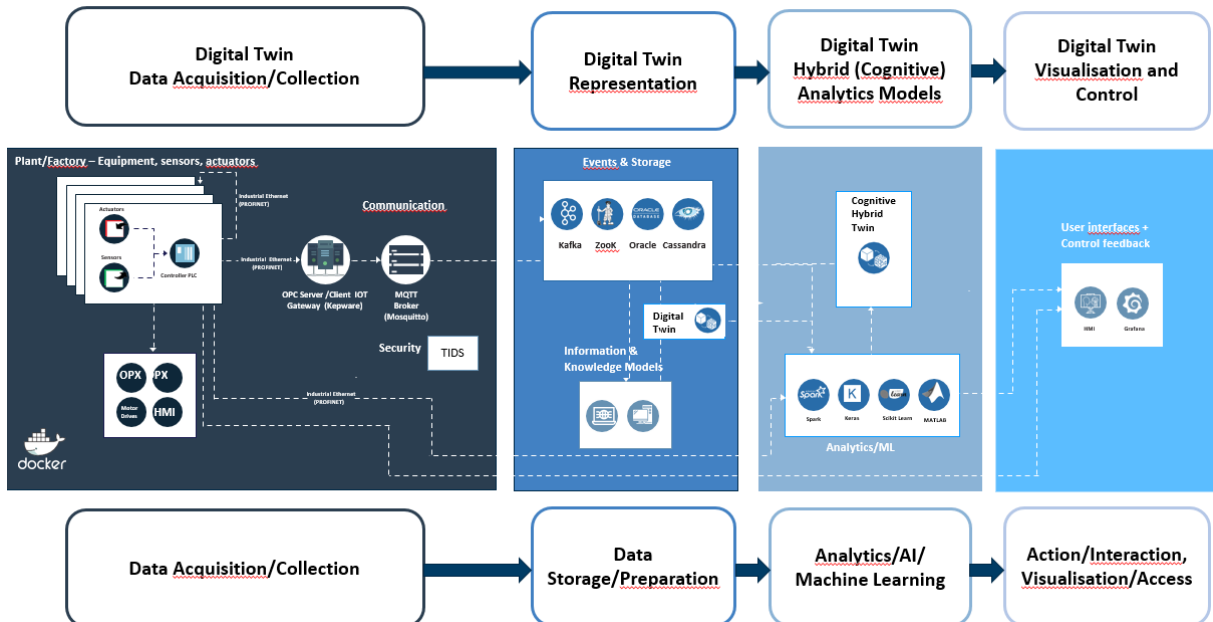


Figure 48: The Digital Twin – Operational pipeline for the Saarstahl pilot case

7.7 NOKSEL Pilot

The NOKSEL pilot has progressed according to the plan. All of the sensor installations have been completed. Sensor data was coded by means of a new coding system. For each sensor installed, the frequency of data collection and a corresponding tag is created in OPC. IIoTP enables PLC data to be collected by OPC and later via MQTT is passed to Kafka.



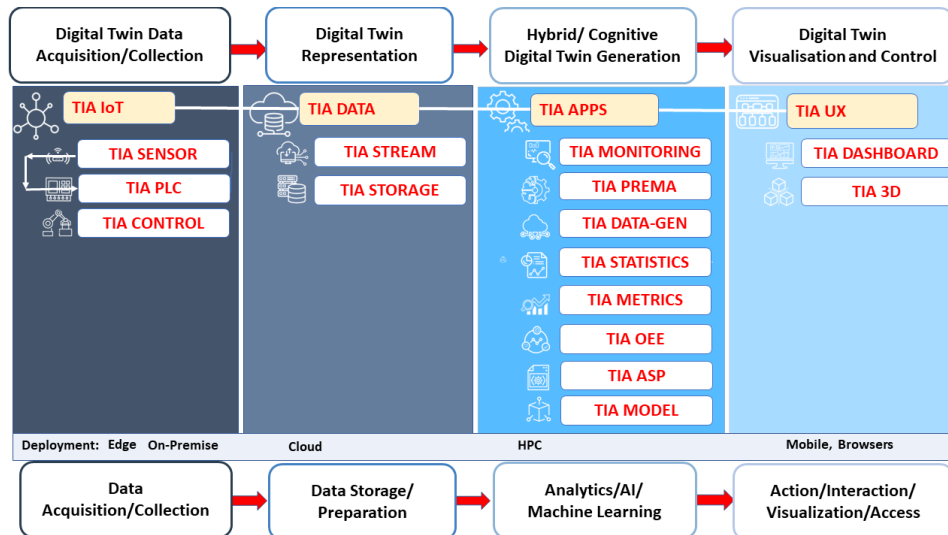


Figure 49: Digital Twin pipeline for the NOKSEL pilot case

TEKNOPAR’s TIA IOT Platform has been developed and data acquired and collected from the installed sensors have been stored in database. One of the challenges regarding the architecture of IIoTP was related to determination of types and number of topics consumed by Kafka. Configurations were updated to avoid lag time that occurred during database writes.

TIA DATA – Industrial Big Data Analysis is used in preparation and analysis of acquired data. Outliers are identified and eliminated, descriptive statistics are calculated, standardization, normalization, Min Max scalar and Standard Scalar were applied to data. PCA is performed. A new tag was introduced for PLC on/off case.

For non-experts to be able to create pipelines where data is retrieved by Cassandra and Fiware Orion Context Broker, Cassandra and Fiware Orion Context Broker were added to the end of the pipelines in Apache StreamPipes.

TIA CONTROL system that is composed of both hardware and software element has been designed, developed and installed at NOKSEL. The purpose of TIA CONTROL is to manipulate the temperature level in the closed room where air conditioners are controlled to set environmental temperature.

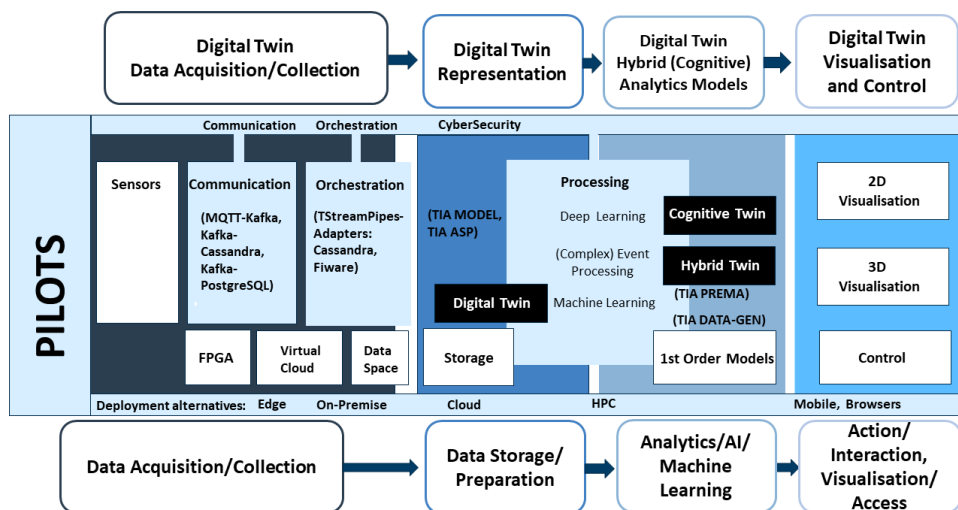
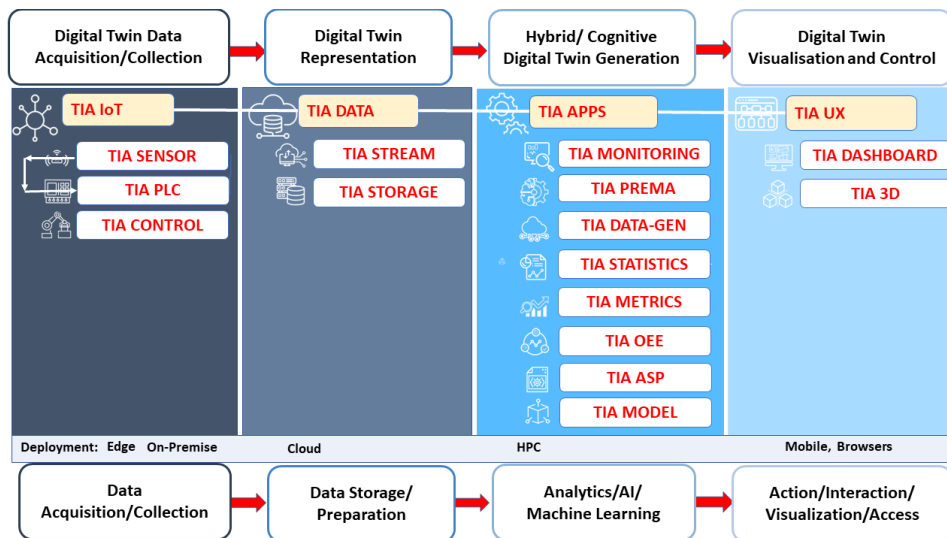
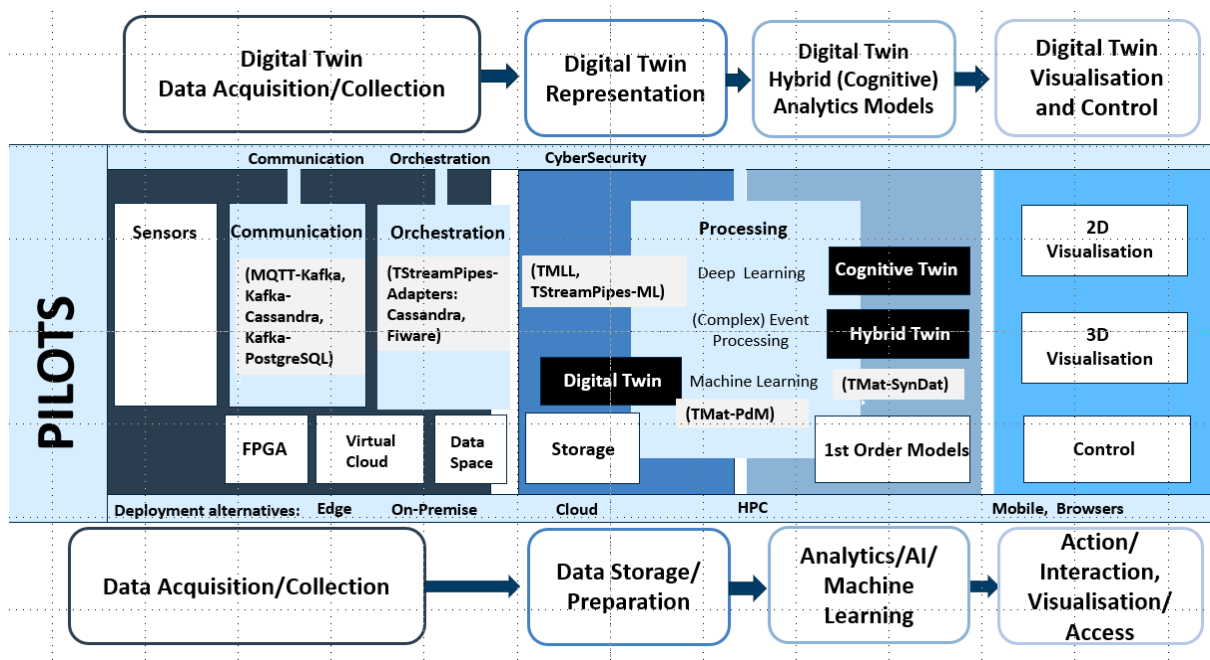


Figure 50: Noksel Digital Twin pipeline

Tools that are in the toolbox within the scope of D4.4 in the above Figure 51 are: TIA CONTROL, TIA PLC, TIA SENSOR, TIA STORAGE and TIA STREAM.

The technologies used in the tools are detailed as follows: The bridge that transfers data from MQTT to Kafka was rewritten in Java in order to improve the performance. As part of the communication in the data acquisition/collection of the digital twin pipeline, two new bridges, Kafka to Cassandra and Kafka to PostgreSQL were written in Python. Two new adapters were added for the TStreampipes-Adapters components: Cassandra and Fiware.

MQTT-Kafka bridge was developed to pass data based on rules defined in the config.yaml file. For example, if a rule in the configuration file states that for a defined sensor only certain values will be transferred, the bridge allows only these values to be transferred for the data coming from the identified sensors. Tests of the rule based data transfer bridge are in progress.

Eleven topics of the Apache Kafka broker have been listened and merged in real-time. The merged topic data are processed and turned into pivot. Based on the type of data (sensor, alarm or state) means, max calculations are processed. Backward and forward data filling were applied. Based on user selected scaling preference, either standard scaling or min-max scaling were applied.

Dockerization of the components, including all of the developed bridges, was conducted. KepserverEX was moved to the virtual server.

By means of a script written in Python, data from Cassandra was transferred to PostgreSQL. NOKSEL MES system is connected to the twin data space, and data retrieved from NOBIS (MES of NOKSEL) is written to PostgreSQL

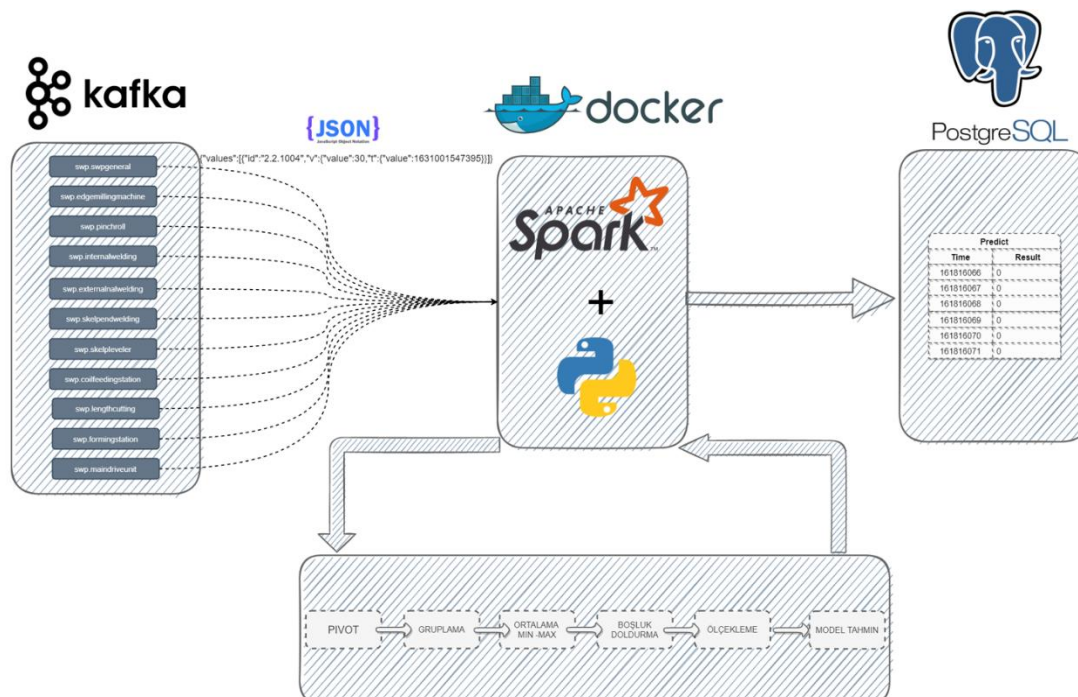


Figure 51: Kafka – Spark – PostgreSQL connections

Two proof of concept studies regarding AAS were performed:

POC1- Eclipse Basyx Application: AAS for some assets were generated. Sensor data coming via MQTT was read and transferred to Eclipse Basyx application. To demonstrate that AAS data in Eclipse Basyx is accessible, a flow in Node-RED was developed and, the AAS data was read in NodeRED. Below diagram presents the flow of related studies:

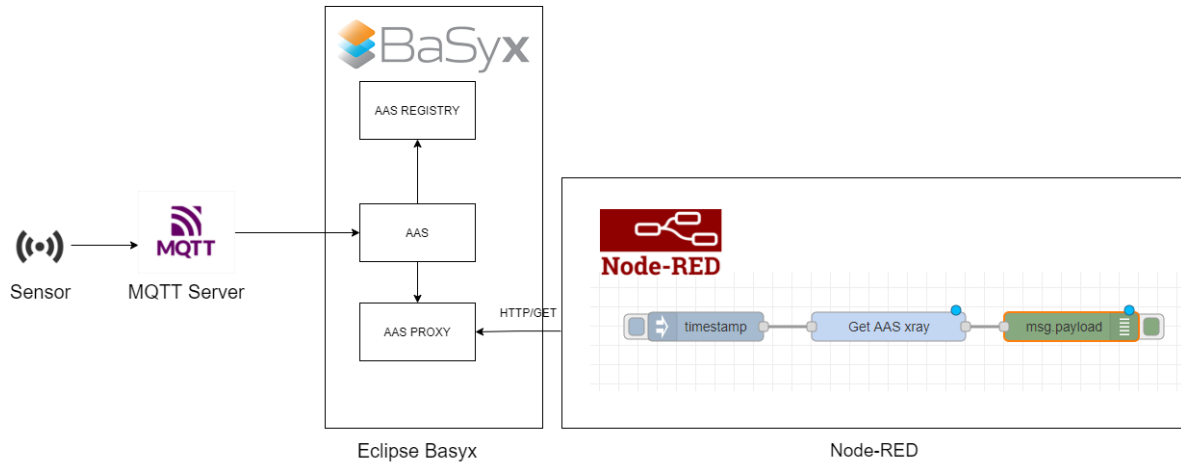


Figure 52: MQTT to AAS to NodeRed

POC2- AAS Package Manager Application: AAS in the AASX Package Manager were exported and then imported to the AASX Server. It was demonstrated that by means of AASX Server, REST/API utilization to the data was possible. Below diagram presents the architecture used in the POC:

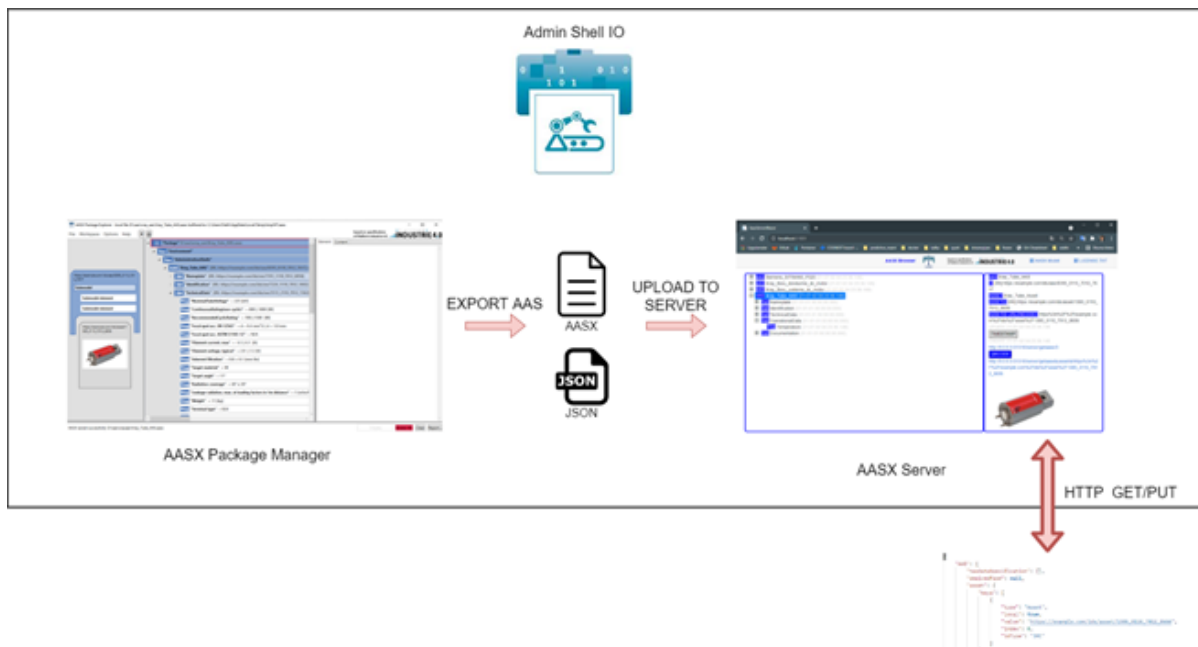


Figure 53: Use of AAS data in pipeline

In order to utilize AAS data of AASX Server in Apache StreamPipes, a data source was developed in Java. To demonstrate that Apache StreamPipes can get AAS data, a pipeline was created in Apache StreamPipes and data coming from AASX Server to the generated datasource was displayed on a

dashboard. AAS data source is dockerized. Below diagram present the pipeline in Apache StreamPipes (on the left) and the data presented in the dashboard (on the right).

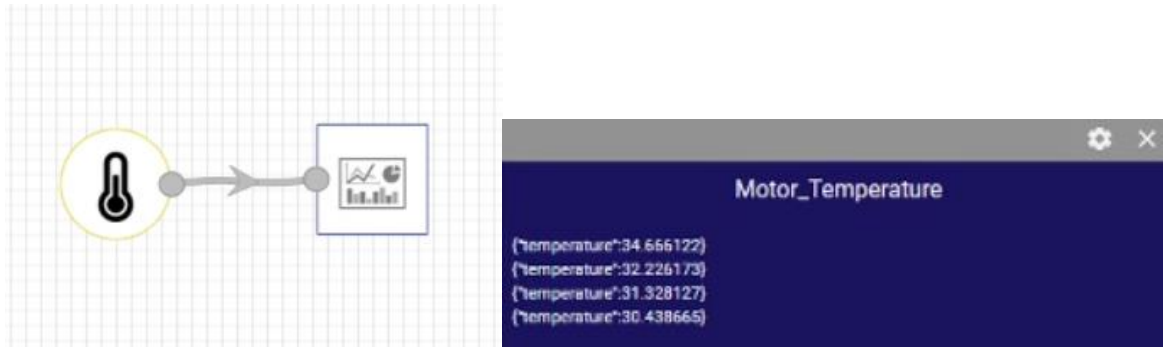


Figure 54: Data from StreamPipes to dashboard

8 Demonstrators

8.1 Demonstrator of FA³ST (Fraunhofer Advanced AAS Tools for Digital Twins) at M42

This demonstrator shows how to connect and synchronize assets and digital twins using FA³ST (Fraunhofer Advanced AAS Tools for Digital Twins) Service. The focus is on three different types of interaction patterns: (i) reading and writing property values, (ii) subscribing to changes in the asset, and (iii) executing operations and returning results. The implementation was done for three different protocols, namely HTTP, OPC UA and MQTT.

The video of this demonstrator is available via the COGNITWIN YouTube channel.

[COGNITWIN FA³ST Asset Connection Fraunhofer \(Final COGNITWIN Demonstrator D4.4\) - YouTube](#)

8.2 Demonstrator of FA³ST (Fraunhofer Advanced AAS Tools for Digital Twins) at M30

This demonstrator presents FA³ST (Fraunhofer Advanced AAS Tools for Digital Twins) Service as a Java-based implementation of the Asset Administration Shell (AAS) specification which is a Digital Twin specification with focus on industrial production. FA³ST Service is designed to be easily extendable and is published as open source (<https://github.com/FraunhoferIOSB/FAAST-Service>). The video shows how FA³ST Service aligns with FA³ST in general which aims to provide a set of tools to support users and developers of Digital Twins throughout the Digital Twin lifecycle. Basic functionality of the FA³ST Service such as reading and writing values via HTTP and OPC UA as well as synchronizing an asset via MQTT is in a demonstration.

The video of this demonstrator is available via the COGNITWIN YouTube channel.

[Industrie 4.0-compliant and Data-Sovereign Digital Twins - YouTube](#)

8.3 Demonstrator of Building a Digital Twin

In this section we describe how to build a digital twin based on the technologies developed in the COGNITWIN project. The video of this demonstrator is available via the COGNITWIN YouTube channel:

<https://www.youtube.com/@cognitwin9786/videos>

Step 1: DT Modelling

A first step is to define a digital twin model for a given asset. According to the ISO 23247 standard³², an asset can be a sensor, equipment, machine, production line, software, product, process, etc. Since we have developed an AAS-compliant digital twin API, the digital twin model has to be based on the

³² ISO 23247 standard - Digital Twin framework for manufacturing

AAS meta model. It can be created either manually or by using the AASExplorer³³. The model can be created in different formats (such as JSON, OPC UA, etc.). Figure 55 shows an asset administration shell in JSON format for the asset selected for the demonstrator.

³³ <https://www.plattform-i40.de/PI40/Redaktion/DE/Newsletter/2019/Ausgabe21/2019-21-Praxisbeispiel2.html>


```
{
  "assetAdministrationShells": [
    {
      "idShort": "MachineAAS",
      "identification":
        {
          "idType": "IRI",
          "id": "urn:aas:id:cognitwin:demo:aas:1"
        },
      ...
    }
  ],
  "assets": [
    {
      "idShort": "Machine",
      "identification":
        {
          "idType": "IRI",
          "id": "urn:aas:id:cognitwin:demo:asset:1"
        },
      "kind": "Type"
    }
  ],
  "submodels": [
    {
      "idShort": "Status",
      "identification":
        {
          "idType": "IRI",
          "id": "urn:aas:id:cognitwin:demo:submodel:1"
        },
      "kind": "Instance",
      "submodelElements": [
        {
          "idShort": "Temperature",
          "category": "PARAMETER",
          "kind": "Instance",
```

Figure 55: AAS model for DT in JSON format (shortened version)

The demonstrator will explain the entities of the DT model.

Step 2: DT Creation

Based on the DT model shown in Figure 55 and by using the FAST software (see section 10.2.3.1) we created a DT service. It is a software component that provides the standardized DT services to external systems/users. Currently, we provide support for HTTP/REST and OPC UA interfaces. For this demonstrator, we decided to use only HTTP/REST interfaces.

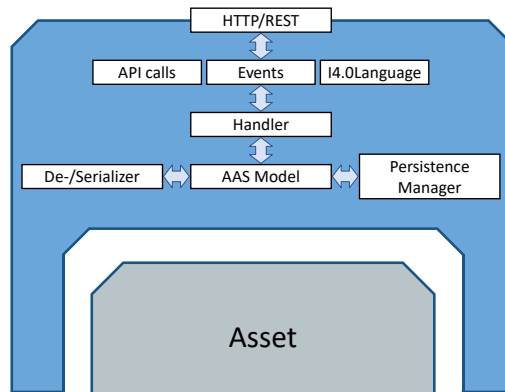


Figure 56: DT service with a standardized model and standardized interfaces

The demonstrator will show how to use FAST to create a digital twin, how to register it with a registry and how to access its entities.

Step 3: DT Connection

The next step is to connect to the physical asset. While the PI4.0 specifies interfaces to external systems, the asset connections could be based on the proprietary protocols. Therefore, we implemented a specialized module to connect to the selected asset. The result of this extension is shown in Figure 57. It is a deployable AAS service that connects to the asset and receives data requests and sends responses back over the standardized DT API.

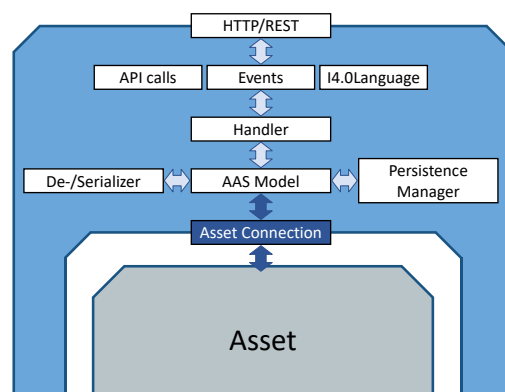


Figure 57: Connected DT

The demonstrator will show how to use FAST to connect to the asset and how to get or set the values of one of its parameters.

Step 4: DT Services

In order to model behavior of an asset and to provide support for real-time processing we decided to use StreamPipes. This requires integration between FAST and StreamPipes. Two DT-specific processors have been developed for StreamPipes: the DT adaptor and the DT data sink. This is shown in Figure 58.

They should be used for all digital twins to be developed by using FAST DT API in order to provide support for the model execution and orchestration of services.

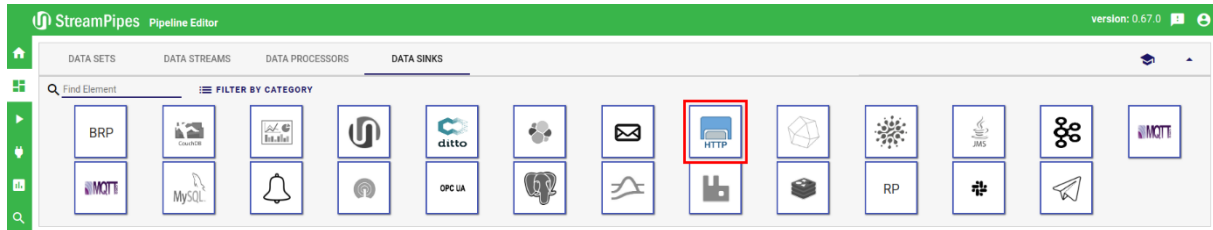


Figure 58: Extension of StreamPipes with DT-specific components

To model behavior of the digital twin, we created a pipeline in StreamPipe. In addition to two DT-specific extension of StreamPipes, the pipeline contains also the Siddhi processor (see section 10.3) that was extended in COGNITWIN. The pipeline is shown in Figure 59.

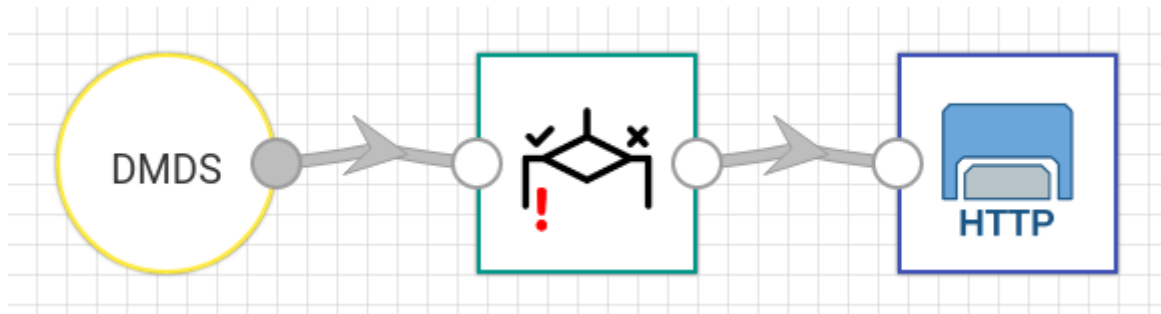


Figure 59: A pipeline example

The pipeline is integrated with the digital twin based on the approach explained in section 2. The result is shown in Figure 60.

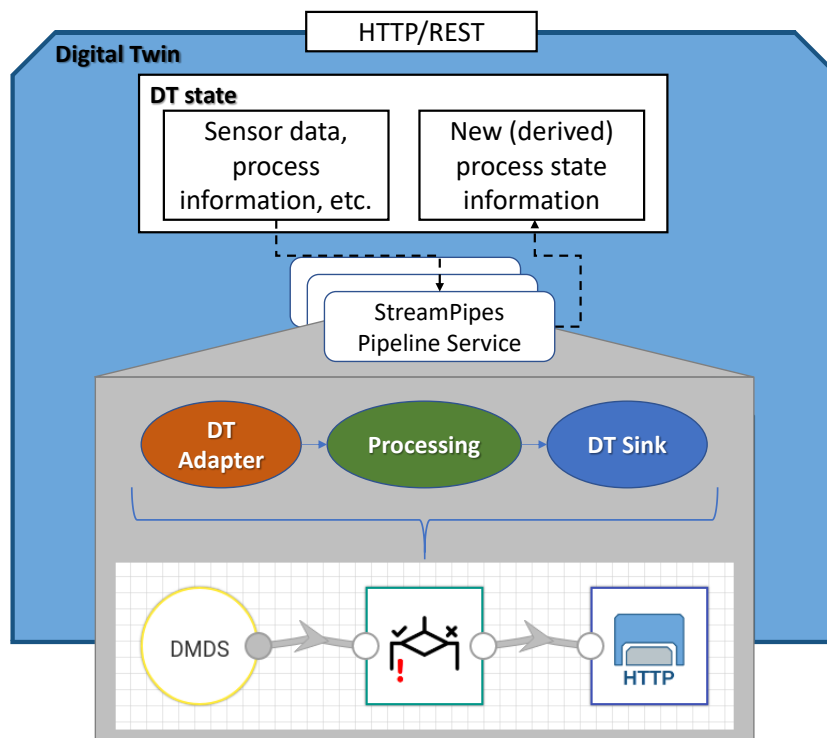


Figure 60: DT integrated with the StreamPipes pipeline

The demonstrator will show how the DT connects to the pipeline in StreamPipes and how the real-time data (both raw sensor data and the calculated data) flows between the DT and the pipeline.

Step 5: DT Usage

At the application level there are several applications that use services provided by the digital twin. To allow the owner of the digital twin to have full control over the digital twin data, models and services, we customized the Fraunhofer Factory Trusted Connector³⁴ and integrated the digital twin into it. In addition, we developed a customer IDS connector with two operator applications and defined a usage policy so that the digital twin data can only be used by one of those applications. The results of the IDS-extensions of the digital twin are shown in Figure 61. Figure 62 shows the user interface of those two IDS operator apps, illustrating the impact of usage policies.

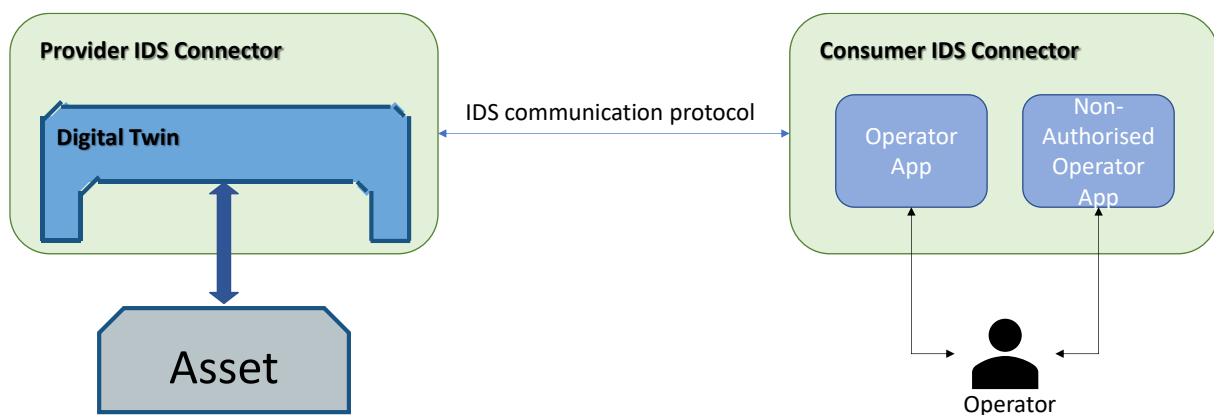


Figure 61: Data-soverain DT

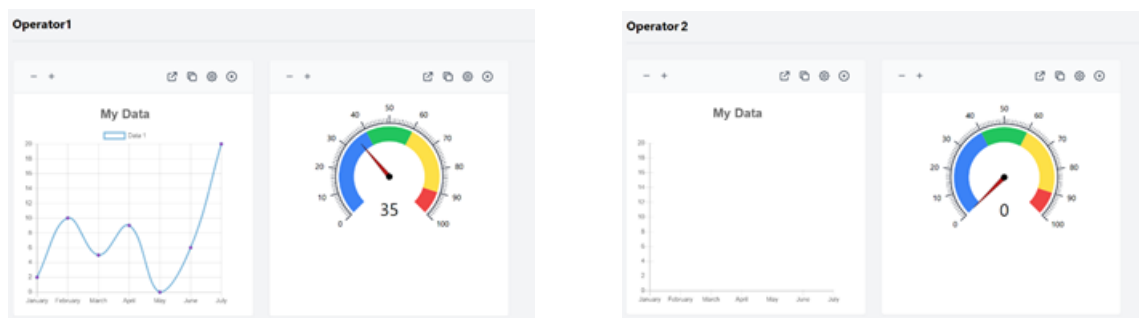


Figure 62: UI in IDS-Apps in the customer IDS connector

The demonstrator will explain the usage policy and will show its implication.

Figure 63 gives an overview of the different components involved in the demonstrator and how they communicate and interact with each other. The data flow of the demonstrator starts with a machine with a temperature sensor attached publishing the sensor data via MQTT. In the demonstrator, this part is simulated using the *MQTT Generator* component. The DT/AAS of the machine now subscribes to this data via the *MQTT Asset Connection*. Within the DT there is a StreamPipes runtime which allows definition and execution of pipelines (including machine learning, neural nets, and other typical

³⁴ https://www.dataspaces.fraunhofer.de/de/software/connector/opc_ua_connector.html

algorithms). Within such a pipeline the *DT Source* and *DT Sink* components are used to easily connect the visual editor of StreamPipes with the HTTP API of the DT making it easy for non-expert users to define custom logic within a DT. The newly calculated values can easily be accessed via the DT HTTP API from within the own company (here: Company A). To provide external companies (here: Company B) access to the DT while ensuring access and usage control, we need IDS Connectors on both sides of the connection, i.e. one for company A and one for company B.

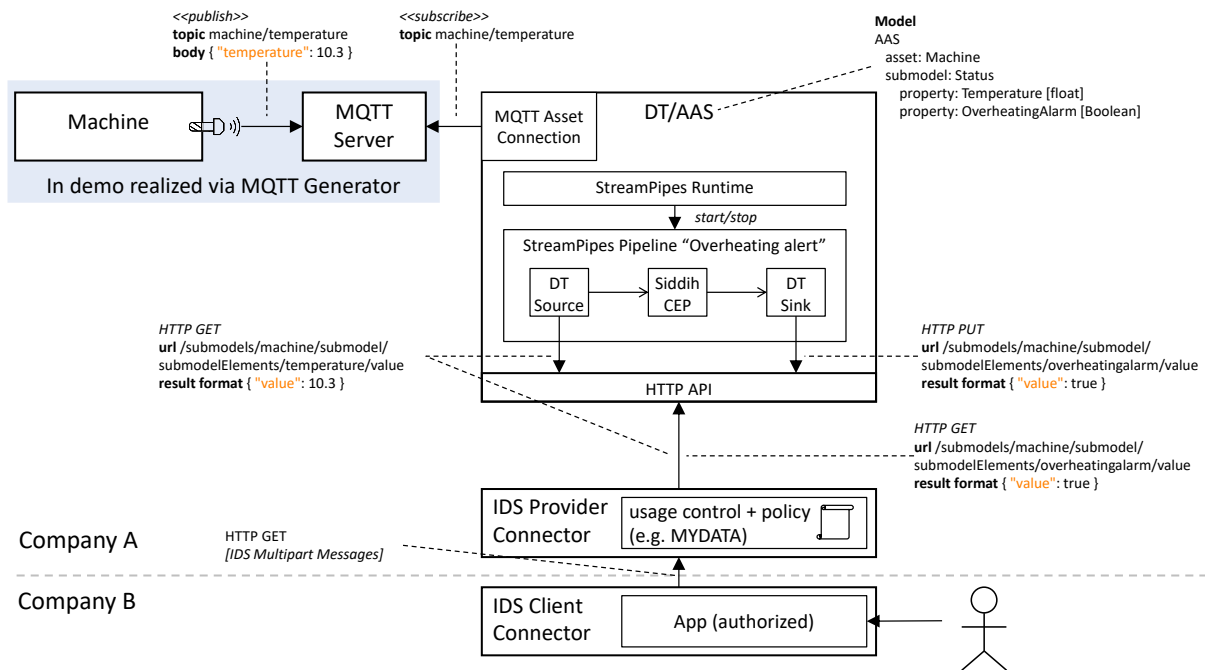


Figure 63: Overview and software architecture of the demonstrator

8.4 Demonstrator of TIA IoT and TIA DATA

Both TIA IoT and TIA Data are designed to develop IoT platforms. Within TIA IoT there are TIA SENSOR, TIA PLC and TIA CONTROL tools. TIA DATA contains TIA STORAGE, TIA STREAM and TIA CONTROL.

More on these tools are available on: <https://www.tia-platform.com/>

The link to the demonstrators is available on the following link: <https://www.youtube.com/watch?v=pvjewnb28lw>

8.5 Demonstrator of Streampipes (integration of numerical model)

This demonstrator explains how the numerical model can be embedded in the Streampipes.

The link to the demonstrators is available on:

[Numerical Model Integration Demonstrator from NISSATECH - YouTube](#)

8.6 Demonstrator of the COGNITWIN Toolbox

This is a demonstration of the COGNITWIN Toolbox and the components and pipelines that it contains.

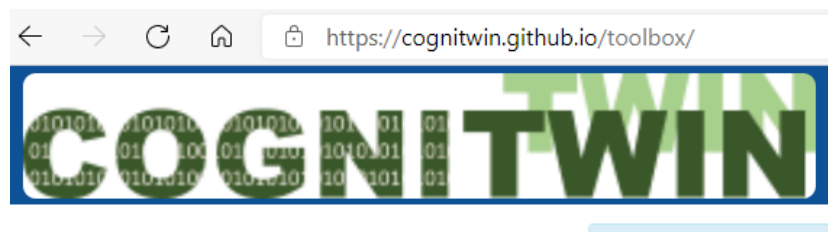


Figure 64: COGNITWIN Toolbox Portal access

The COGNITWIN Toolbox Portal is available through the following link:

<https://cognitwin.github.io/toolbox/>.

This demonstrators (([part 1](#) and [part 2](#))) are available through the COGNITWIN YouTube channel, and goes through the overall structure and content of the Toolbox. It is also convenient to go through this directly online, by browsing the various components and related links.

A video following the focus on Digital Twin pipelines in the second period introduces the access to these through the COGNITWIN Toolbox portal.

Digital Twin Pipelines

<p>Emission (Gas Treatment Center) temperature and fan control</p> <p>Demonstrated through a pipeline implementation in a Hydro Aluminum plant.</p>	<p>Temperature in tapping stream and Slag in refining ladle</p> <p>Demonstrated through a pipeline implementation in an Elkem Silicon plant.</p>	<p>Predictive Maintenance of Machinery</p> <p>Demonstrated through a pipeline implementation in a Noksel Steel plant.</p>
<p>Rolling Mill Tracking System</p> <p>Demonstrated through a pipeline implementation in a Saarstahl Steel plant.</p>	<p>Hybrid model for ladle refractory wear</p> <p>Demonstrated through a pipeline implementation in a Sidenor Steel plant.</p>	<p>Boiler fouling management</p> <p>Demonstrated through a pipeline implementation in a Sumitomo Boiler installation.</p>

Access to various toolbox components is presented in the video from the first period.

Digital Twin Data Acquisition Tools & Services

Communication

<p>MAI</p> <p>Weather data from yr.no for application in model of industrial process.</p> license n/a TRL n/a	<p>Cybernetica OPC-UA Server</p> <p>The Cybernetica OPC-UA Server is a general purpose OPC-UA server supporting the Data Access (DA) interface.</p> license Copyright © TRL 8	<p>FUSE OPC-UA</p> <p>FUSE OPC-UA is a tool enabling the necessary data transfer during on-line operation of FUSE state estimation tool.</p> license Copyright © TRL 5
<p>TOPC-UA</p> <p>OPC-UA and data connectors from Teknopar.</p> license Copyright © TRL n/a		

Orchestration

<p>Adapt_ST</p> <p>Set of adapters (for data sources and components) for StreamPipes-based Toolbox.</p> license n/a TRL n/a	<p>Big Data Pipelines Deployment Framework (BDPDF)</p> <p>A framework to allow high-level design/specification of scalability aspects of Big Data processing pipelines and their deployment on the continuum computing infrastructure.</p> license Apache2.0 TRL 4
--	---

FPGA-AI

<p>Honir</p> <p>This tool allows to perform machine learning / deep learning algorithm inference in real time on images data source.</p> license Copyright © TRL 5	<p>Lodur</p> <p>Lodur is a frame grabber. It is responsible for connecting a camera to an FPGA platform (Honir).</p> license Copyright © TRL 5	<p>TStreamPipes-Adapters</p> <p>TStreamPipes-Adapters enable non-experts to create data stream pipes that end with Cassandra and Fiware.</p> license Copyright © TRL 4
---	---	---

Platforms

<p>Bedrock</p> <p>The BEDROCK Toolbox is a flexible and easily deployable software bundle of modules developed as a platform to be deployed on various servers – with one instance running on local machines and servers at SINTEF Industry.</p> license Copyright © TRL 5	<p>STEEL 4.0 IoT</p> <p>Steel 4.0 IoT is a platform that is used to acquire, collect and store sensor and PLC data.</p> license Copyright © TRL 4
---	--

Figure 65: Digital Twin Data Acquisition Components

Figure 65 shows the COGNITWIN Toolbox components for the area of Digital Twin Data Acquisition, and the demonstrator goes through some example components like Cybernetica and FUSE OPC-UA

servers and the FPGA-AI components Lodur and Honir, as well as platform pipeline examples from Bedrock and TIA IOT.

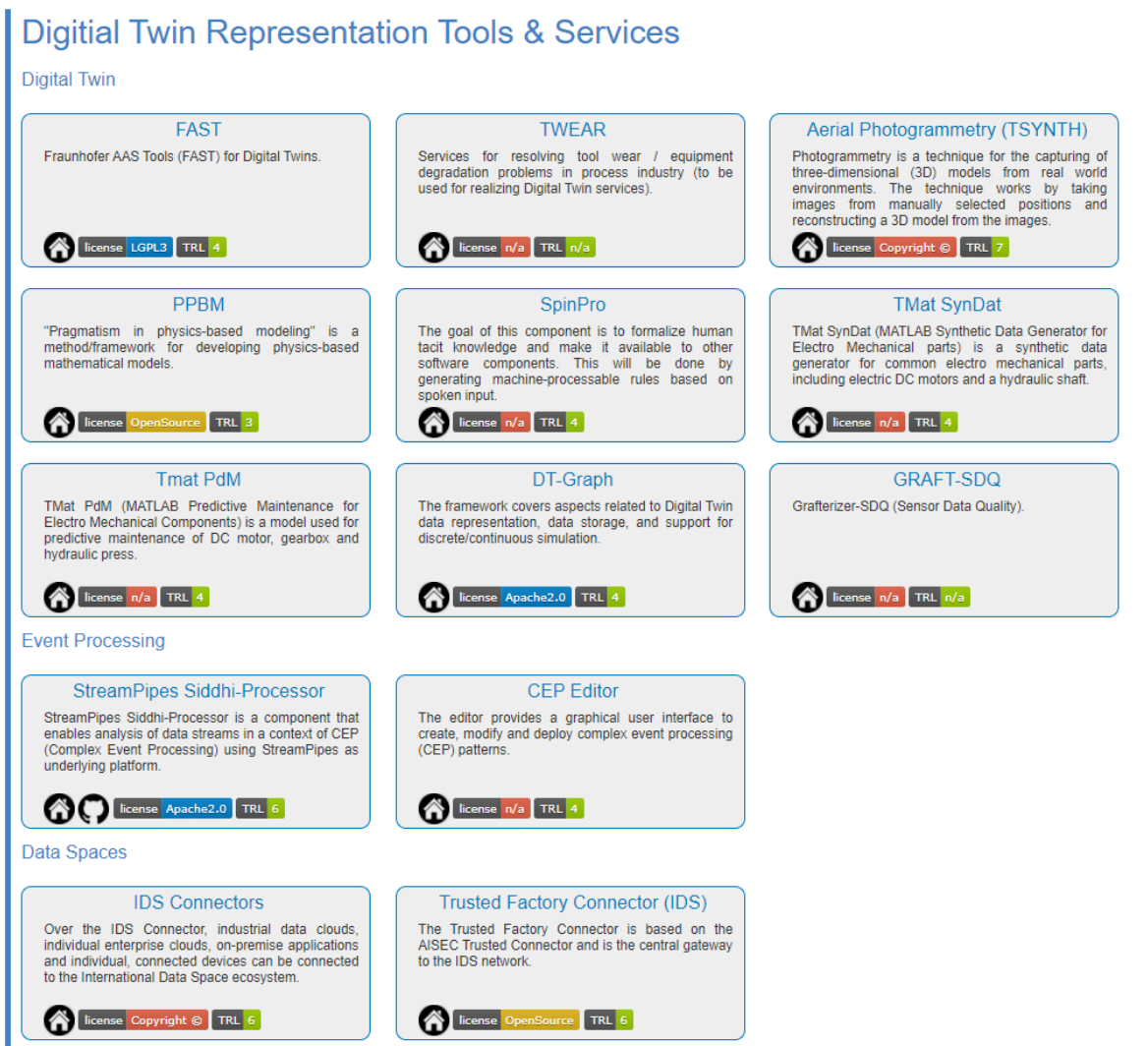


Figure 66: Digital Twin and Data Space Components

Figure 67 shows the COGNITWIN Toolbox components for the area of Digital Twin Analytics with both Machine Learning and First Principles Models - and the demonstrator goes through some example components like TIA AI (previously named as Steel 4.0 TMLL) Bonza, Keras2RTL and Neuroscope, as well as first principles models. All of these are further described in the deliverable document and demonstrators for D5.2.

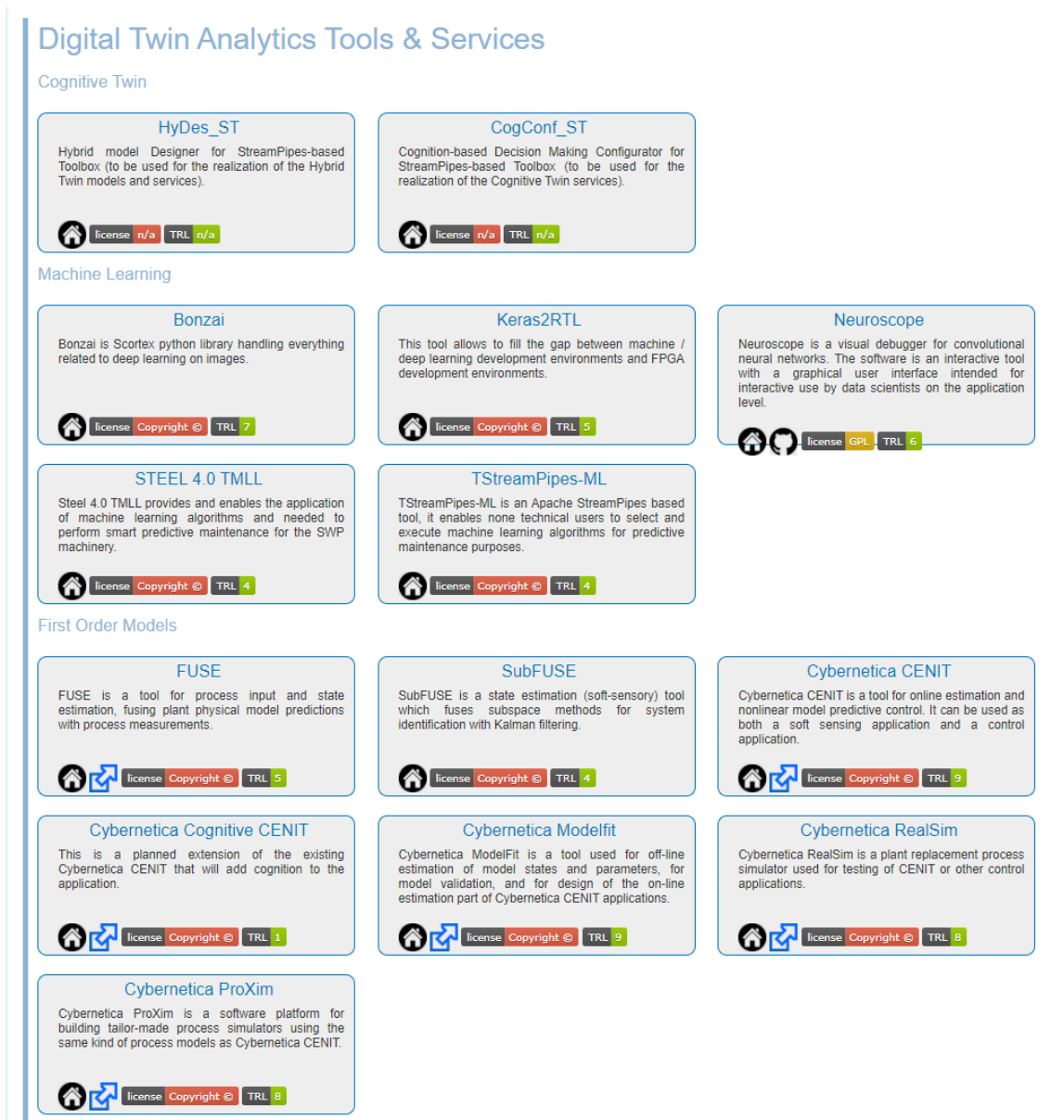


Figure 67: Digital Twin Analytics – for Machine Learning and First Principles Models

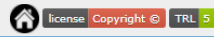
Figure 68 shows the COGNITWIN Toolbox components for the area of Digital Twin Visualisation and Control - and the demonstrator goes through some Steel ICP and Cybernetica Viewer visualisation component. Both of these are further described in the deliverable document and demonstrators for D5.2.

Digital Twin Visualization and Control Tools & Services

2D Visualization

STEEL 4.0 ICPV

STEEL 4.0 ICPV supports the digital twin by means of visual components presenting the generated data which is retrieved from, and processed within other STEEL 4.0 components



3D Visualization

Cybernetica Viewer

Cybernetica Viewer is a tool for creating user interfaces to display and manipulate data from an OPC server in various ways.

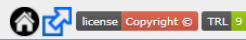


Figure 68: Digital Twin Visualisation and Control Components

9 References

- [ABB+20] Abburu, S., Berre, J.A., Jacoby, M., Roman, D., Stojanovic, L., Stojanovic, N. (2020, November). "Cognitive Digital Twins for the Process Industry", Cognitive Technologies, the Twelfth International Conference on Advanced Cognitive Technologies and Applications (COGNITIVE 2020).
- [AB2+20] Abburu, S., Berre, J. A., Jacoby, M., Roman, D., Stojanovic, L., Stojanovic, N. (2020) COGNITWIN – Hybrid and Cognitive Digital Twins for the Process Industry, in Proceedings of the 2020 IEEE International Conference on Engineering, Technology and Innovation (ICE/ITMC), Cardiff, UK, 15–17 June 2020; pp. 1–8.
- [ALU20] Albayrak, Ö., Unal, P. (2020). Smart Steel Pipe Production Plant via Cognitive Digital Twins: A Case Study on Digitalization of Spiral Welded Pipe Machinery, in the Proceedings of the ESTEP Workshop on Impact and opportunities of Artificial Intelligence in the Steel Industry.
- [Ban+17] Banerjee, Agniva & Mittal, Sudip & Dalal, Raka & Joshi, Karuna. (2017). Generating Digital Twin Models using Knowledge Graphs for Industrial Production Lines.
- [Bar+19] M. Barika, et al., Orchestrating Big Data Analysis Workflows in the Cloud: Research Challenges, Survey, and Future Directions, ACM Computing Surveys 52 (2019)
- [BER+21] Berre, A.J., Tsalgatidou, A., and C. Francalanci, Ivanov, T., Lobo, T.P., Saiz, R.R., Novalija, I., M. Grobelnik, Big Data and AI Pipeline Framework – Technology analysis from a Benchmarking perspective, in TABDV book, Springer Open Access, Spring 2021
- [Bra+21] A. Brandstetter, Machine-Learning & Complex-Event-Processing: Hybrid concepts for data stream analysis, Master's degree of the Karlsruhe University of Applied Sciences - Technology and Economics, to appear 2021
- [Car+19] Thyago P. Carvalho u. a. „A systematic literature review of machine learning methods applied to predictive maintenance“. In: Computers & Industrial Engineering 137 (2019), S. 106024. issn: 03608352. doi: 10.1016/j.cie.2019.106024.
- [Chr+18] Maximilian Christ u. a. „Time Series FeatuRe Extraction on basis of Scalable Hypothesis tests (tsfresh – A Python package)“. In: Neurocomputing 307 (2018), S. 72–77. issn: 09252312. doi: 10.1016/j.neucom.2018.03. 067.
- [Döb+18] Inga Döbel u. a. Maschinelles Lernen: Eine Analyse zu Kompetenzen Forschung und Anwendung. Hrsg. von Fraunhofer-Gesellschaft zur Förderung der angewandten Forschung e.V. München, 2018. url: [https:// www.bigdata.fraunhofer.de/content/dam/bigdata/de/documents/Publikationen/Fraunhofer_Studie_ML_201809.pdf](https://www.bigdata.fraunhofer.de/content/dam/bigdata/de/documents/Publikationen/Fraunhofer_Studie_ML_201809.pdf).
- [GAIX20] GAIA-X Technical Architecture, 2020, https://www.data-infrastructure.eu/GAIX/Redaktion/EN/Publications/gaia-x-technical-architecture.pdf?_blob=publicationFile&v=5
- [Gar+18] Garofalo, Martina & Pellegrino, Maria & Altabba, Abdulrahman & Cochez, Michael. (2018). Leveraging Knowledge Graph Embedding Techniques for Industry 4.0 Use Cases.
- [GUO+16] GUO, Kaiyuan, SUI, Lingzhi, QIU, Jiantao, et al. Angel-eye: A complete design flow for mapping cnn onto customized hardware. In: 2016 IEEE Computer Society Annual Symposium on VLSI (ISVLSI). IEEE, 2016. p. 24-29.
- [IEE90] IEEE, "A Compilation of IEEE Standard Computer Glossaries," New York, Standard 1990

- [INT21] <https://en.wikipedia.org/wiki/Interoperability>, accessed February 2021
- [ISO20] ISO/IEC 20547-3:2020, Information technology — Big data reference architecture — Part 3: Reference architecture, <https://www.iso.org/standard/71277.html>
- [Jau+20] Jacoby, M., Usländer, T., Digital Twin and Internet of Things - Current Standards Landscape, Applied Sciences, 10, retrieved from URL: <https://www.mdpi.com/2076-3417/10/18/6519>, 2020
- [Jac+21] Jacoby, M.; Jovicic, B.; Stojanovic, L.; Stojanovic, N. An Approach for Realizing Hybrid Digital Twins Using Asset Administration Shells and Apache StreamPipes. Information 2021, 12, 217. <https://doi.org/10.3390/info12060217>
- [Jac+22] M. Jacoby, F. Volz, C. Weißenbacher and J. Müller, "FA3ST Service – An Open Source Implementation of the Reactive Asset Administration Shell," 2022 IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA), Stuttgart, Germany, 2022, pp. 1-8, doi: 10.1109/ETFA52439.2022.9921584.
- [Lam+19] R. Lamberti and L. Stojanovic, "Complex Event Processing as an Approach for real-time Analytics in industrial Environments," 2019 IEEE 17th International Conference on Industrial Informatics (INDIN), Helsinki, Finland, 2019, pp. 220-225, doi: 10.1109/INDIN41052.2019.8972311.
- [PAP-18] Papamichael, J. F. K. O. M., Liu, T. M. M., Haselman, D. L. S. A. M., Ghandi, L. A. M., Sapek, S. H. P. P. A., & Woods, G. W. L. (2018). A configurable cloud-scale dnn processor for real-time ai. In Proceedings of the 45th Annual International Symposium on Computer Architecture, ser. ISCA (Vol. 18)
- [PER+05] PERRI, Stefania, LANUZZA, Marco, CORSONELLO, Pasquale, et al. A high-performance fully reconfigurable FPGA-based 2D convolution processor. Microprocessors and Microsystems, 2005, vol. 29, no 8-9, p. 381-391.
- [Ran+17] R. Ranjan, et al., Orchestrating Big Data Analysis Workflows, IEEE Cloud Computing 4 (2017) 20–28.
- [Rol+20] José Roldán u. a. „Integrating complex event processing and machine learning: An intelligent architecture for detecting IoT security attacks“. In: Expert Systems with Applications 149 (2020), S. 113251. issn: 09574174. doi: 10.1016/j.eswa.2020.113251.
- [Sha+18] Shawahna, Ahmad, Sadiq M. Sait, and Aiman El-Maleh. "FPGA-based accelerators of deep learning networks for learning and classification: A review." IEEE Access 7 (2018): 7823-7859.
- [SHS17] Tizian Schneider, Nikolai Helwig und Andreas Schütze. „Automatic feature extraction and selection for classification of cyclical time series data“. In: tm - Technisches Messen 84.3 (2017). issn: 0171-8096. doi: 10.1515/teme2016-0072.
- [SHL19] Michael Schadler, Norbert Hafner und Christian Landschützer. Konzepte und Methoden für prädiktive Instandhaltung in der Intralogistik. 2019. urn:nbn:de:0009-14-49655
- [Shy+16] Shyamala et.al, *Detection of Damage in Beam from Measured Natural Frequencies Using Support Vector Machine Algorithm*, ISBN 978-81-8487-550-8 (2016)
- [Sot+16] José Angel Carvajal Soto u. a. „CEML: Mixing and moving complex event processing and machine learning to the edge of the network for IoT applications“. In: Proceedings of the 6th International Conference on the Internet of Things - IoT'16. Hrsg. von Mareike Kritzler u. a. New York, New York, USA: ACM Press, 2016.

[Sto+21] Stojanovic, L.; Usländer, T.; Volz, F.; Weißenbacher, C.; Müller, J.; Jacoby, M.; Bischoff, T. Methodology and Tools for Digital Twin Management—The FA3ST Approach. *IoT 2021*, 2, 717-740. <https://doi.org/10.3390/iot2040036>

[Wan+16] Wang, Chao, et al. "DLAU: A scalable deep learning accelerator unit on FPGA." *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 36.3 (2016): 513-517.

[ZIL+20] Zillner, S., Bisset, D., Milano, M., Curry, E., García Robles, A., Hahn, T., Irgens, M., Lafrenz, R., Liepert, B., O'Sullivan, B. and Smeulders, A., (eds) (2020) "Strategic Research, Innovation and Deployment Agenda - AI, Data and Robotics Partnership. Third Release." September 2020, Brussels. BDVA, euRobotics, ELLIS, EurAI and CLAIRE"

[ZIL+17] BDVA SRIA, Zillner, S., Curry, E., Metzger, A., Auer, S., & Seidl, R. (Eds.). (2017). European Big Data Value Strategic Research & Innovation Agenda., http://www.bdva.eu/sites/default/files/BDVA_SRIA_v4_Ed1.1.pdf

10 Annex – COGNITWIN Toolbox Components

10.1 Toolbox Components - COGNITWIN Interoperability Toolbox

10.1.1 End-to-End COGNITWIN Toolbox Pipeline Architecture

10.1.1.1 COGNITWIN Toolbox Portal

Component/Tool description
Component/Tool/Method/Framework/Service Name
COGNITWIN Toolbox Portal (GitHub)
Short Description – incl. Purpose
The COGNITWIN Toolbox Portal is providing overview and access to descriptions of all of the components in the COGNITWIN Toolbox, structured according to the Digital Twin pipeline steps.
Progress since last milestone
The COGNITWIN Toolbox Portal has been developed from scratch since last milestone.
Examples of usage / illustrations
<p>The COGNITWIN Toolbox is used to get an overview and access to information about all of the partner components. In addition, the portal contains information about relevant third party tools and standards relevant for the COGNITWIN Digital Twin pipeline.</p>
Interfaces (in/out) – system/user

The COGNITWIN Toolbox Portal is provided as a web portal with a web user interface.
Subordinates and platform dependencies
The COGNITWIN Toolbox Portal is realized through GitHub and associated components with dependencies on this.
Licenses, etc. (free for use in the project)
The implementation of the portal is based on GitHub provided infrastructures
TRL for overall component/tool and any parts/subordinates
TRL6 – Working system for the COGNITWIN Project
References – incl. web etc.
https://cognitwin.github.io/toolbox/
To be considered in particular for the following COGNITWIN pilots
This is relevant in the context of all pilots – and is also being used by all of the COGNITWIN Component and pipeline implementation providers to provide access to relevant information.

10.1.2 Interoperability Orchestration (StreamPipes)

10.1.2.1 Adapt_ST - Nissatech

Component/Tool description
Component/Tool/Method/Framework/Service Name
Adapt_ST - Set of adapters for StreamPipes-based Toolkits
Short Description – incl. Purpose
StreamPipes-based Toolkits provides means of creating custom pipeline elements (based on StreamPipes), further increasing its functionalities and applicability. Creates a path from sensory data to the decision making module, depicting state of the tool, through several pipeline elements. It is intended to run as a part of a bigger pipeline. It is easily modified and extended with other elements, both built-in and custom-built Pipeline elements are encapsulated as standalone microservices, which enables joining elements created by different partners and run on different machines into one pipeline.
Progress since last milestone
These adapters have been developed from scratch since the last milestone.
Examples of usage / illustrations
Main purpose of developed pipeline is to create a path form acyclic sensory data to the Neural Network output, depicting state of the tool, through several pipeline elements. In addition, this pipeline triggers notification when value of particular property goes above certain threshold, displays time between consecutive heats and information about each heat.

These pipeline elements include: element that simulates connection between acyclic sensory data and the rest of the pipeline (Data Stream – marked in yellow), elements for processing, including one with the Neural Network (Data Processors – marked in green) and elements that provide output of the pipeline – visualization, notifications, etc. (Data Sinks – marked in blue).

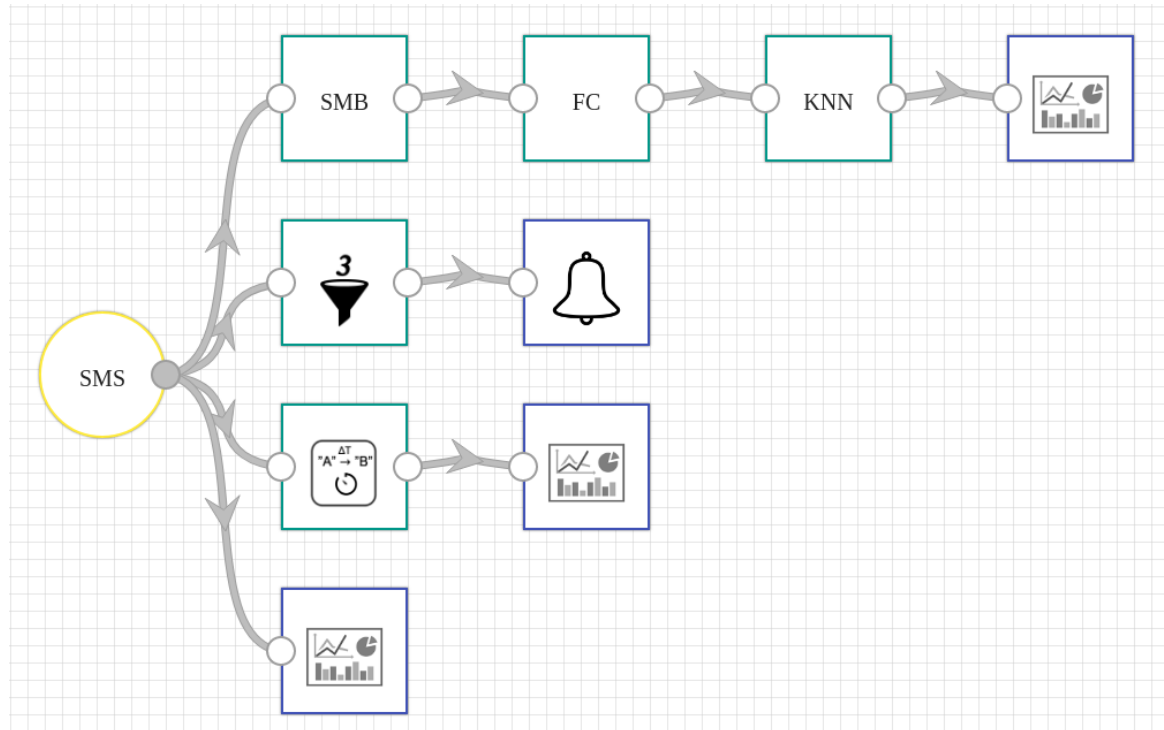


Figure 69: Developed pipeline (arrows represent data flow)

First pipeline element, named SMS for Sidenor Measurements Simulation, simulates connection between acyclic sensory data and the rest of the pipeline, by reading row-by-row of .csv file. Each row represents one heat and contains values of parameters for said heat. Simulates real-world situation in which, after each heat, data measured for said heat would be sent to the pipeline.

First element in the top row, named SMB for Sidenor Measurements Buffer, represents a buffer that orders heats according to ladle, cycle and phase they come from since our Neural Network requires input that is calculated based on all heats from the same ladle, cycle and phase. For each heat that is inputted, this element adds it to the list that holds heats that came from same ladle, cycle and phase and then forwards entire list to the next element.

Following element in the top row, named FC for Factored Contributions, serves as a pre-processing element that prepares forwarded data for inference done by next pipeline element with integrated Neural Network model. This element calculates “contribution” of each parameter to the wear of ladle and outputs a vector where each value represents “contribution” of corresponding parameter.

Said vector represents input to the KNN (Keras Neural Network) element. It represents input of both mentioned pipeline element and Keras Neural Network model loaded within it. After inference, NN outputs class which states condition of ladle (lower/higher degradation than predefined threshold). This output value is forwarded to the visualization element.

First element in the second row, named Numerical Filter, filters events (heats, in this context) based on value of selected numerical property. In this case, it filters heats based on consumed electricity (Kwh_rr) - if heat has value of consumed electricity greater than specified threshold, it gets forwarded to the next element (notification element), otherwise, it gets ignored.

First element in the third row, named Task Duration, computes the time difference between two events (heats, in this context). It forwards measured difference to the visualization element.

Last element in the second row, named Notification and marked with bell, displays a notification in the UI panel of StreamPipes. In this pipeline, it displays a notification regarding heats that have consumed electricity value above certain threshold (Image 2).

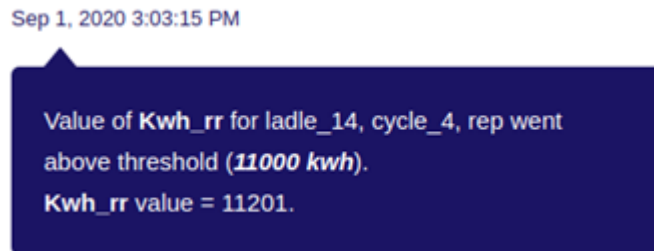


Figure 70: Displayed notification

Rest of the elements colored in blue, named Dashboard Sink, serve as a visualization tool. They visualize data streams in the StreamPipes dashboard.

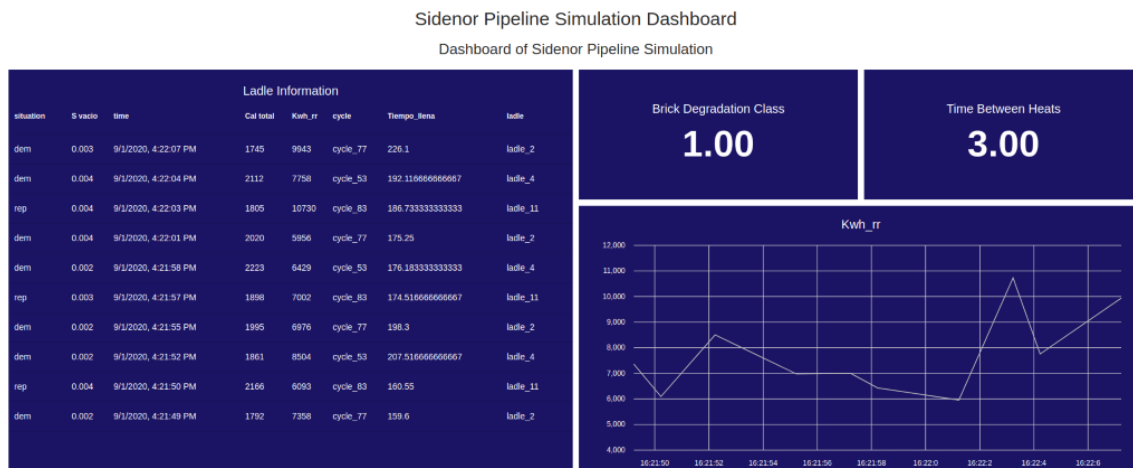


Figure 71: Visualization of outputs of Keras Neural Network (Brick Degradation Class), Task Duration (Time Between Heats) and Sidenor Measurements Simulation (Ladle Information, Kwh_rr)

Pipeline can easily be modified and extended with other elements, both built-in and custom-built, using built-in Editor. Additionally, pipeline elements are encapsulated as standalone microservices, which enables joining elements created by different partners and run on different machines into one pipeline.

Interfaces (in/out) – system/user

This component receives input from any element that provides count of occurrences of interest. In case of demonstration, output from MEWMA is used.

After query execution, it outputs count of inputted values that correspond to the condition.

Since this component is implemented in a way that allows custom values for window length and number of occurrences, a user interaction is required. In essence, when user creates pipeline and employs this component, a corresponding window pops up which prompts user to enter said values.

Subordinates and platform dependencies

StreamPipes (<i>and this component, as well</i>) is available for Linux, Windows and Mac OS X. This component was developed on Linux machine. Docker and Docker Compose are required in order to run pipelines.
Licenses, etc. (free for use in the project)
Proprietary
TRL for overall component/tool and any parts/subordinates
TR5
References – incl. web etc.
StreamPipes - https://github.com/apache/incubator-streampipes/tree/rel/0.67.0 StreamPipes extensions - https://github.com/apache/incubator-streampipes-extensions/tree/rel/0.67.0 StreamPipes Siddhi wrapper - https://github.com/apache/incubator-streampipes/tree/rel/0.67.0/streampipes-wrapper-siddhi
To be considered in particular for the following COGNITWIN pilots
Sidenor – but is also representative examples/templates for StreamPipes adapters that can be used by other StreamPipes adapters in COGNITWIN pipelines.

10.1.2.2 TIA STREAM

Component/Tool description
Component/Tool/Method/Framework/Service Name
TIA STREAM (Teknopar)
Short Description – incl. Purpose
TIA CONNECT is part of TIA DATA. TIA STREAM enables the users to transfer data from data sources to the related destinations, TIA PLATFORM modules and/or different destinations.
Progress since last milestone
This is recently named. Verification and validation have been completed.
Examples of usage / illustrations

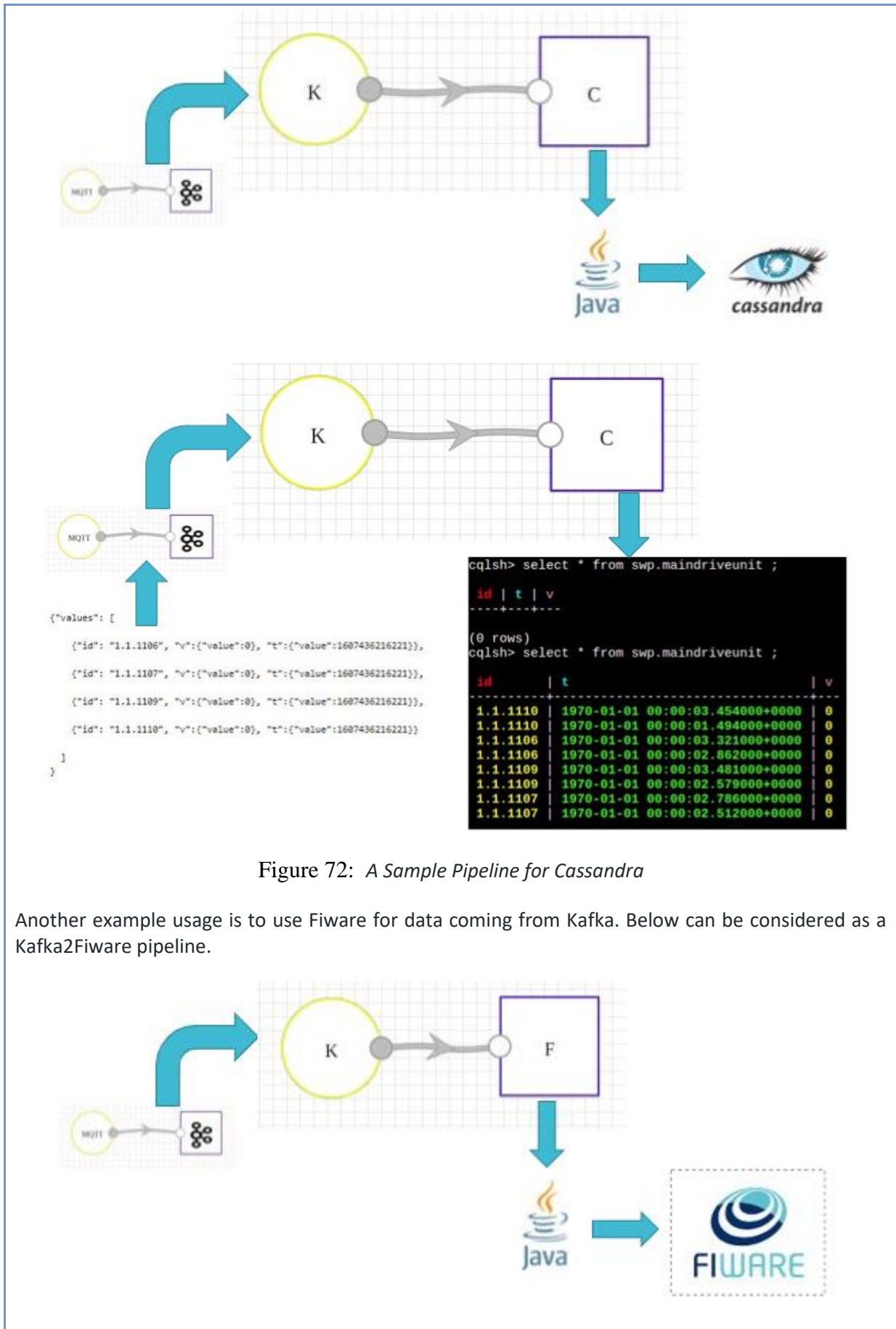
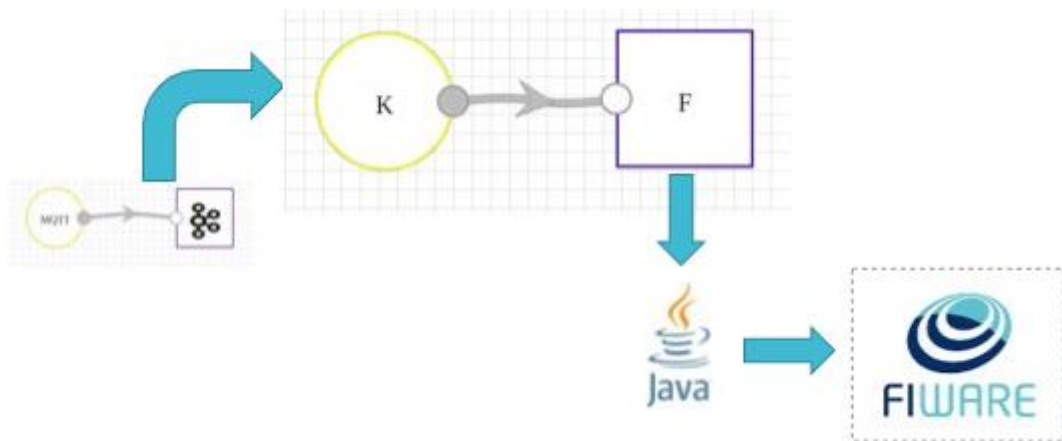


Figure 72: A Sample Pipeline for Cassandra

Another example usage is to use Fiware for data coming from Kafka. Below can be considered as a Kafka2Fiware pipeline.



FUSE OPC-UA

Short Description – incl. Purpose

FUSE OPC-UA is a tool enabling the necessary data transfer during on-line operation of FUSE state estimation tool. This includes transfer of measurement data from plant to the evaluation of FUSE algorithms (Matlab) and propagation of outcomes for further use (e.g. to StreamPipes or company asset management services).

The FUSE OPC-UA communication tool is based on setting up an OPC-UA server (Prosys Simulation Server) for data transfer, supported by OPC-UA client services by the other used software (Matlab, StreamPipes, asset management services).

Progress since last milestone

The tool has been developed (designed, implemented, and verified) after the last milestone in 2/2020.

Examples of usage / illustrations

The tool originates from solving the WP3 pilot problem on fuel characterization, as a part of the heat exchanger fouling monitoring problem. Figure 74 illustrates the designed and tested architecture for data transfer.

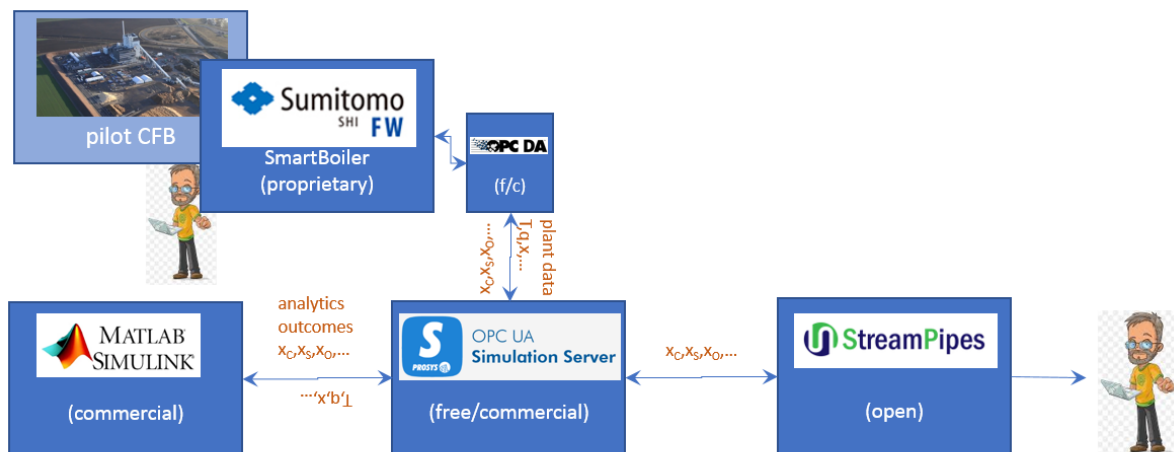


Figure 74: FUSE fuel characterization during one week CFB operation

The links between Matlab – OPC-UA – StreamPipes have been implemented and tested with the FUSE state estimation tool. Currently, the link between pilot and OPC-UA server has been simulated by a Matlab OPC-UA client, the OPC-DA communication has not been tested.

Interfaces (in/out) – system/user

The Prosys OPC-UA Simulation server provides a full user interface for setting up the desired nodes. The Matlab OPC-UA is set up by defining proper function calls. The StreamPipes provides a UI for OPC-UA adapters.

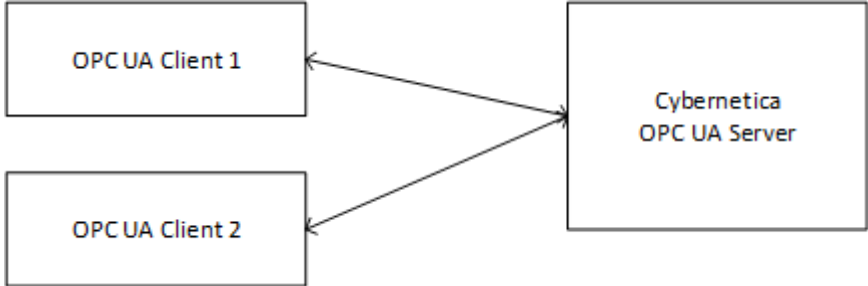
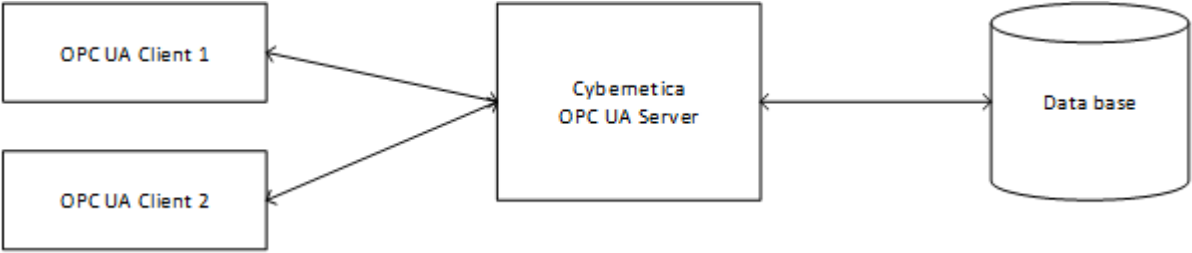
Subordinates and platform dependencies

The FUSE OPC-UA server is implemented on Windows 10. Prosys Simulation server is available for Windows, Linux and macOS.

Licenses, etc. (free for use in the project)
<p>For info on the FUSE OPC-UA tool, contact Markus.Neuvoenen@oulu.fi.</p> <p>Prosys Simulation Server 5.0.2 free edition has full functionality, except importing OPC-UA information models.</p> <p>Matlab OPC-UA functionality requires the Matlab OPC-Toolbox from Mathworks.</p> <p>Apache StreamPipes is open source.</p>
TRL for overall component/tool and any parts/subordinates
<p>Current state is TRL 5 (validated in a relevant environment) currently being raised to TRL 7 (prototype demonstrated in a relevant environment).</p>
References – incl. web etc.
<p>https://www.prosysopc.com/products/opc-ua-simulation-server/</p> <p>https://se.mathworks.com/products/opc.html</p> <p>https://streampipes.apache.org/</p>
To be considered in particular for the following COGNITWIN pilots
<p>Sumitomo SHI FW Energia Oy</p>

10.1.3.2 Cybernetica OPC UA DA Server

Component/Tool description
Component/Tool/Method/Framework/Service Name
<p>Cybernetica OPC UA DA Server</p>
Short Description – incl. Purpose
<p>The Cybernetica OPC UA Server is a general purpose OPC UA server supporting the Data Access (DA) interface. It can be used as a hub for exchanging real-time data from processes with other clients that support OPC UA.</p> <p>The OPC UA server has a plugin API that allows specialized plugins to be developed. These can be used to collect and distribute data from other data sources (like databases, process control systems or simulators).</p>
Progress since last milestone
<p>Some measurements at the Elkem pilot are not available on OPC and are only stored in Elkem’s Oracle database. To have these measurements available on OPC, Cybernetica has created a bespoke extension to the Cybernetica OPC UA Server for the Elkem pilot. This extension queries a set of tables in the Oracle database to extract the relevant measurements and publishes them to OPC. This simplifies the employment of the digital twin, as it allows for using OPC as the single data source.</p> <p>Another extension that allows for publishing data from or storing data to a PostgreSQL has been implemented.</p>
Examples of usage / illustrations

<p>Example 1: Real-time data exchange</p> 	
<p>Example 2: Distributing data from a database (or DCS or some other source)</p> 	
<p>Interfaces (in/out) – system/user</p>	
<p>The interfaces are based on OPC UA</p>	
<p>Subordinates and platform dependencies</p>	
<p>This also integrates with OPC DA.</p>	
<p>Licenses, etc. (free for use in the project)</p>	
<p>Cybernetica OPC UA DA Server licenses are provided free of charge for the duration of the COGNITWIN-project for project partners who need such license to execute their work in the project. Should the project result be taken into permanent use after the end of the project, licenses are provided on fair and reasonable terms as stated in the Grant Agreement.</p>	
<p>TRL for overall component/tool and any parts/subordinates</p>	
<p>8</p>	
<p>References – incl. web etc.</p>	
<p>http://cybernetica.no/technology/model-predictive-control/</p>	
<p>To be considered in particular for the following COGNITWIN pilots</p>	
<p>Hydro, Elkem.</p>	

10.1.3.3 TIA DATA Industrial Big Data Analytics

<p>Component/Tool description</p>
<p>Component/Tool/Method/Framework/Service Name</p>
<p>TIA DATA Industrial Big Data Analytics</p>
<p>Short Description – incl. Purpose</p>
<p>TIA DATA is used in preparation and analysis of sensor data retrieved from PLC.</p>

<p>The purpose of IDBA is to generate meaningful information to support digital twin for state estimation and process control.</p> <p>The sensor data, such as temperature, pressure and vibration, voltage, and current are transmitted to MQTT over OPC, and then to Kafka in JSON format.</p> <p>Being a data streaming platform, Apache Kafka transmits real-time data with a low error margin and short latency. Real time data received by Kafka is then transmitted to the Python-based server, where the attribute extraction process is performed.</p>
<p>Progress since last milestone</p>
<p>Since the last milestone:</p> <ul style="list-style-type: none"> • Data has been preprocessed. • Outliers in the data is identified and eliminated • Descriptive statistics on the collected data is calculated • Standardization, normalization, mixMax Scalar and StandardScaler were applied to data. • Principle Component Analysis (PCA) is performed by making the incoming high-dimensional data low-dimensional, providing more accurate results for machine learning. • PLC data is analyzed to decide whether the PLC is on or off.
<p>Examples of usage / illustrations</p>
<p>The TIA DATA has been made operational for the Noksel pilot in COGNITWIN.</p>
<p>Interfaces (in/out) – system/user</p>
<p>The interfaces area based on the interfaces of the connectors used, in particular by MQTT, OPC and Kafka.</p>
<p>Subordinates and platform dependencies</p>
<p>TIA DATA is used to prepare data to be used by TIA AI, Kafka.</p>
<p>Licenses, etc. (free for use in the project)</p>
<p>Proprietary/ Subject to license</p>
<p>TRL for overall component/tool and any parts/subordinates</p>
<p>The current TRL is 6-7.</p>
<p>References – incl. web etc.</p>
<p>https://tia-platform.com/module/tia-data.html</p>
<p>To be considered in particular for the following COGNITWIN pilots</p>
<p>NOKSEL</p>

10.1.3.4 TIA STORAGE

<p>Component/Tool description</p>
<p>Component/Tool/Method/Framework/Service Name</p>

TIA STORAGE
Defined in Task
T4.1, T4.2
Short Description – incl. Purpose
TIA STORAGE is part of TIA DATA. TIA STORAGE enables the users to store data received from components such as sensors, PLCs, actuators, devices, MES, or machines.
Progress since delivery D4.3
Verification and validation have been completed.
Examples of usage / illustrations
TIA STORAGE is used to store data in time series and/or relational databases, including Cassandra and PostgreSQL.
Interfaces (in/out) – system/user
TIA STORAGE uses TIA IOT data as input, and TIA STORAGE enables TIA APPS and TIA UX to utilize data stored by TIA STORAGE.
Subordinates and platform dependencies
Being a web application, TIA STORAGE is platform independent, it can run on many different types of browsers including Google Chrome, Safari, Microsoft Edge, Mozilla, Opera, etc.
Licenses, etc. (free for use in the project)
Proprietary/ Subject to license
TRL for overall component/tool and any parts/subordinates
The current TRL is 7 as targeted.
References – incl. web etc.
https://tia-platform.com/module/tia-data.html
To be considered in particular for the following COGNITWIN pilots
NOKSEL

10.1.3.5 Grafterizer-SDQ – for Sensor Data Curation

Component/Tool description
Component/Tool/Method/Framework/Service Name

Grafterizer-SDQ – for Sensor Data Curation and Quality check (SDQ)

Short Description – incl. Purpose

DataGraft is a general-purpose data management platform focused on supporting the creation and provisioning of Knowledge Graphs. It consists of a set of cloud-based tools and services for data transformation and access.

DataGraft aims to offer a complete package for transformation of raw data into meaningful data assets and reliable delivery of data assets by providing a solution that outsources various data operations to the cloud and that eliminates upfront costs on data infrastructure and ongoing investment of time and resources in managing the data infrastructure.

DataGraft was developed to allow data workers to manage their data in a simple, effective, and efficient way.

Grafterizer is a component in the DataGraft platform focusing on data cleaning and data curation.

Progress since last milestone

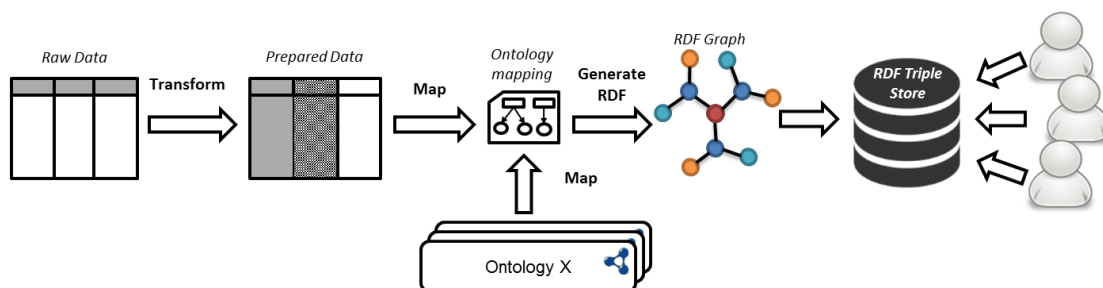
DataGraft has been maintained as a tool for data management, however no major features have been introduced since the previous milestone. DataGraft with Grafterizer was part of the initial COGNITWIN Toolbox baseline. Even if it has not been used in the COGNITWIN project during the first phase, it can be a relevant tool in the subsequent phases where sensor data curation and quality checking is one of the focus areas. It is thus retained for consideration for the next phase of the project.

Examples of usage / illustrations

DataGraft implements a set of capabilities for managing data. Key capabilities include:

- Interactive design of data transformations, including tabular data cleaning and data enrichment
- Repeatable data transformations
- Reuse/share data transformations (user-based access)
- Cloud-based deployment of data transformations

The figure below depicts a typical process supported by DataGraft: from raw data to knowledge graph data.



The following figure is a screenshot of DataGraft with the Grafterizer part that shows the functionality for cleaning and transforming tabular data.

baseregisteredAddressmunicipality	baseregisteredAddressprovince	baseregisteredAddressregion	baseregisteredAddressstate	basicUkcode
Rugby	Warwickshire	Herefordshire, Worcestershire and Warwickshire	United Kingdom	82990
Sunderland	Sunderland	Northumberland and Tyne and Wear	United Kingdom	47240
Bristol, City of	Bristol, City of	Gloucestershire, Wiltshire and Bath/Bristol area	United Kingdom	86220
Tower Hamlets	Tower Hamlets	Inner London - East	United Kingdom	62020
Hillingdon	Harrow and Hillingdon	Outer London - West and North West	United Kingdom	64191
Richmond upon Thames	Hounslow and Richmond upon Thames	Outer London - West and North West	United Kingdom	63990
Wakefield	Wakefield	West Yorkshire	United Kingdom	41100
Solihull	Solihull	West Midlands	United Kingdom	1430
Westminster	Westminster	Inner London - West	United Kingdom	63110
Rotherham	Barnsley, Doncaster and Rotherham	South Yorkshire	United Kingdom	42120
Stafford	Staffordshire CC	Shropshire and Staffordshire	United Kingdom	96090
Stockport	Greater Manchester South East	Greater Manchester	United Kingdom	81299
Stockport	Greater Manchester South East	Greater Manchester	United Kingdom	81221
Stockport	Greater Manchester South East	Greater Manchester	United Kingdom	81210
Stockport	Greater Manchester South East	Greater Manchester	United Kingdom	81222
City of London	Camden and City of London	Inner London - West	United Kingdom	86220
Knowsley	East Merseyside	Merseyside	United Kingdom	46130
Knowsley	East Merseyside	Merseyside	United Kingdom	96090
Leeds	Leeds	West Yorkshire	United Kingdom	56102
Islington	Haringey and Islington	Inner London - East	United Kingdom	70229
Kirklees	Calderdale and Kirklees	West Yorkshire	United Kingdom	43320
SOUTHAMPTON			ENGLAND	87200
Great Yarmouth	Norwich and East Norfolk	East Anglia	United Kingdom	
Wiltshire	Wiltshire	Gloucestershire, Wiltshire and Bath/Bristol area	United Kingdom	78109
South Kesteven	Lincolnshire	Lincolnshire	United Kingdom	70229

Interfaces (in/out) – system/user

DataGraft comes with graphical user interfaces for end users.

DataGraft components expose REST APIs.

Subordinates and platform dependencies

DataGraft is based on a microservices architecture with loosely coupled components.

Licenses, etc. (free for use in the project)

Eclipse Public License (v1.0)

TRL for overall component/tool and any parts/subordinates

DataGraft and its components are at TRL 4-5.

References – incl. web etc.

DataGraft online service: <https://datagraft.io>

Grafteizer 2.0: <https://www.eubusinessgraph.eu/grafteizer-2-0/>

DataGraft software:

- Installation guide: <https://github.com/datagraft/datagraft-portal>
- User guide: <https://github.com/datagraft/datagraft-reference/blob/master/documentation.md>
- API documentation: <https://datagraft.github.io/datagraft-API/dist/index.html?url=https://datagraft.github.io/datagraft-API/swagger.yaml>
- Source code repository: <https://github.com/datagraft/datagraft-portal>

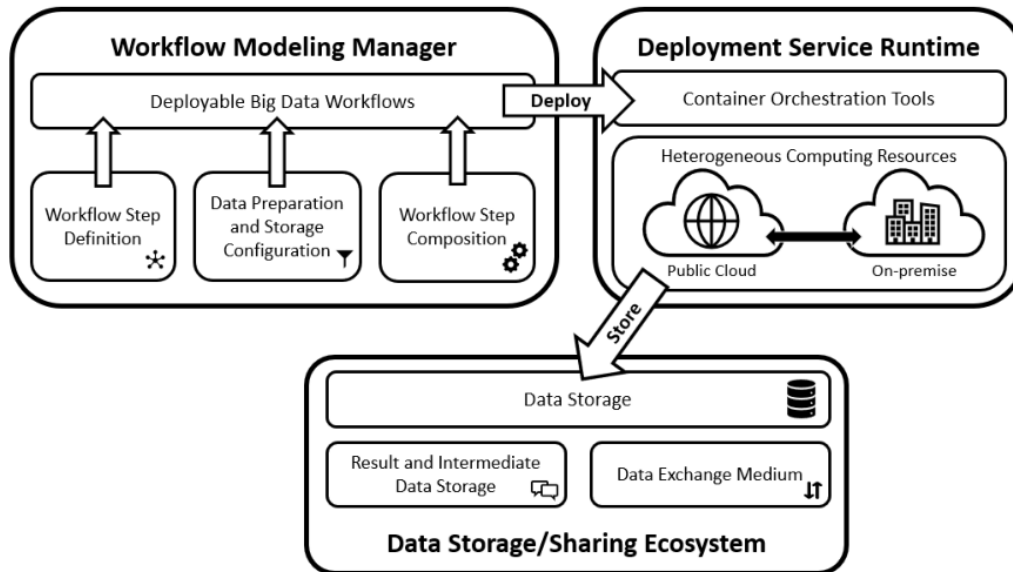
To be considered in particular for the following COGNITWIN pilots

For COGNITWIN, a particular relevant aspect is the management of sensor data quality. For this purpose the Grafterizer component of DataGraft could be relevant and can be considered for further investigation in the pilots.

10.1.4 Big Data Pipelines Deployment

10.1.4.1 Big Data Pipelines Deployment Framework

Component/Tool/Method/Framework/Service Name
Big Data Pipelines Deployment Framework
Short Description – incl. Purpose
<p>A framework to allow high-level design/specification of scalability aspects of Big Data processing pipelines and their effective and efficient deployment and execution on the continuum computing infrastructure (heterogenous Cloud/Fog/Edge infrastructure).</p> <p>The purpose of such a framework is to lower the technological barriers of entry to the incorporation of Big Data pipelines in organizations' business processes regardless of the hardware infrastructure. The framework requires new languages, methods, infrastructures, and software for managing Big Data pipelines such that Big Data pipelines can be easily set up in a manner which is trace-able, manageable, analyzable and optimizable and separates the design- from the run-time aspects of their deployment, thus empowering domain experts to take an active part in their definition.</p> <p>The purpose of such a framework is to allow specification of data processing pipelines by non-IT experts at an abstraction level suitable for pure data processing, in which pipeline specifications are realized (deployed and executed) using instances of a pre-defined set of scalable and composable software container templates (corresponding to step types in pipelines).</p>
Progress since last milestone
<p>Progress since the last milestone include design and prototype implementation of the framework that makes use of software container technologies, message-oriented middleware (MOM), and a domain-specific language (DSL). Furthermore, a set of experiments were performed that show the practical applicability of the proposed approach for the specification and scalable execution of Big Data workflows.</p>
Examples of usage / illustrations
<p>The figure below depicts a typical workflow in the use of the framework. Users typically specify high-level descriptions of workflow steps and their dependencies using the Workflow Modeling Manager. This component handles storage configurations, data preparation, and step-level data processing and transformation operations. The output of the component is a deployable data workflow that feeds into the Deployment Service Runtime – a component representing the collection of hybrid computing resources where workflows steps are deployed. Data Storage/Sharing Ecosystem is responsible for storing intermediate and output data and the data exchange mechanism during workflow execution.</p>



An example of workflow specification depicted below and use a DSL specifically designed to handle Big Data pipelines.

```

workflow prototypeWorkflow {
  communicationMedium: medium MESSAGE_QUEUE
  parameters: MQ_HOST = kubemq
  steps:
    - step unzip
      triggers: external-event
      implementation:
        docker-implementation image: '/ebw-prototype-00-unzip'
      environment:
        STEP_NAME='00-unzip'

    - step tsv2csv
      triggers: external-event
      implementation:
        docker-implementation image: '/ebw-prototype-01-tsv2csv'
      environment:
        STEP_NAME='01-tsv2csv'

    - step split
      triggers: external-event
      implementation:
        docker-implementation image: '/ebw-prototype-02-split'
      environment:
        STEP_NAME='02-split'

    - step transform
      triggers: external-event
      implementation:
        docker-implementation image: 'ebw-prototype-03-transform'
      parameters: transformationJar = '/transformation/transformation.jar'
      environment:
        STEP_NAME='03-transform'

    - step toarango
      triggers: external-event
      implementation:
        docker-implementation image: '/ebw-prototype-04-toarango'
      parameters: transformationJson = '/transformation/transformation.json'
      environment:
        STEP_NAME='04-toarango' }

```

Interfaces (in/out) – system/user
The framework receives as input the pipeline to be executed and the resources available, then it deploys and execute the pipeline on the available resources.
Subordinates and platform dependencies
Sub-components are architected using microservices.
Licenses, etc. (free for use in the project)
Apache-2.0 License
TRL for overall component/tool and any parts/subordinates
TRL4 – a protype has been implemented at this stage and a set of experiments were carried out to test scalability of the solution.
References – incl. web etc.
Software: https://github.com/SINTEF-9012/ebw-prototype
Paper: a paper is currently under review at Internet of Things Journal.
To be considered in particular for the following COGNITWIN pilots
<p>The purpose of this framework is to be able to effectively and efficiently model, deply, and execute Big Data pipelines on heterogenous infrastructures (Cloud-Fog-Edge). For COGNITWIN, this framework is relevant for the pilots that require processing of large amounts of data.</p> <p>The Big Data Workflow framework was part of the initial COGNITWIN Toolbox baseline and has evolved into a software protype to date. Even if it has not been directly applied in the COGNITWIN pilots during the first phase, it can be a relevant framework in the subsequent phases for scalable deployment and execution of data pipelines. It is thus retained for consideration for the next phase of the project.</p>

10.2 Toolbox - Cloud Platform, Data Space, Security, Digital Twin

10.2.1 Cloud Platform

10.2.1.1 Bedrock Platform / Toolbox – Pipeline and Components

Component/Tool/Method/Framework/Service Name
SINTEF Bedrock Platform / Toolbox – Pipeline and Components
Short Description – incl. Purpose
<p>The BEDROCK Toolbox is a flexible and easily deployable software bundle of modules developed as a platform to be deployed on various servers – with one instance running on local machines and servers at SINTEF Industry. The list of available toolbox modules consists of open-source components and SINTEF in-house developed code. The framework is applied as the foundation for digital twinning R&D activities, process control and data analytics.</p> <p>The purpose of the toolbox is to enable a framework for collaborative development of various applications with easy access to the tools needed. The toolbox enables easy access to data through shared databases and components with flexible web-based dashboard solutions for visualization.</p>

The framework is the underlying toolbox ("the bedrock") for several SINTEF activities on advanced process control, digital twin, pilot plant operation, process data analytics and research data management for process plants. It is applied for workflow management in SINTEFs pilot plants and as a sandbox for development of software modules that later can be distilled into contributions to commercial solutions.

Progress since last milestone

The Bedrock Toolbox code repository has since the last milestone undergone restructuring to allowing for improved configuration and remote deployment.

The underlying components in the bundle of the updated repository are still containerized, but the deployments are now centrally configured from a hierarchy of Ansible playbooks. This enables deployment and administration of multiple remotely installed instances on different servers and projects and allows for flexible selection of modules in customized deployments based on the needs. It also enables options for cloud deployment.

The updated code structure allows for easy addition of new components to the toolbox. The Bedrock Toolbox has during 2020 been extended with more components and currently allows for bundled deployments for the number of components* shown in the table below.

* Only selected modules needed for a specific case are deployed in a customized Bedrock deployment.

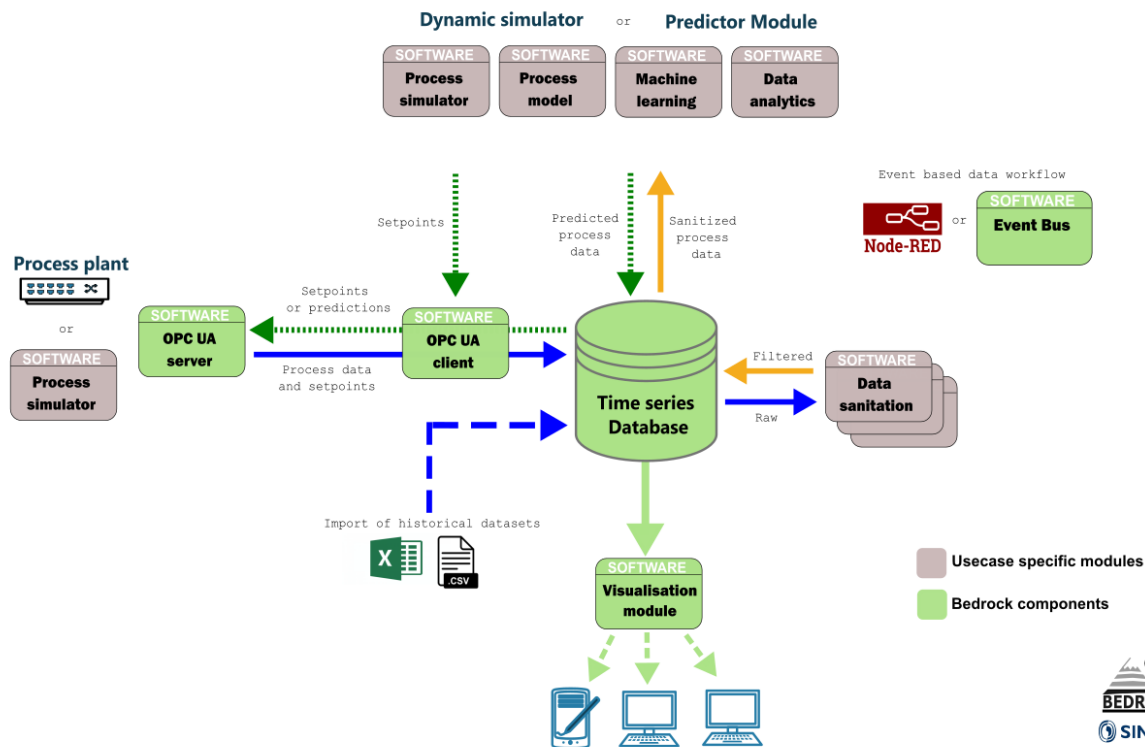
Table 16: Bedrock Toolbox components per 1 Jan 2021

Component	Purpose
InfluxDB database server	Time series database
TICK-stack (servers)	Optional full Influx TICK-stack with Telegraf, Chronograf and Kapacitor in addition to Influx for data workflow management, handling of metrics and events.
TimescaleDB database server	Time series database allowing for relational PostgreSQL databases.
Mosquitto MQTT broker	Allowing for self-serving of MQTT as a shared resource.
Kafka	Event streaming platform allowing for various data workflows. With Zookeeper, Kafka brokers (scaled based on needs), Schema-register and Kafka ksql-server, ksql-cli, Manager and Rest-proxy as a shared resource.
Node-RED server	Platform for low-code GUI based workflow designs and dashboarding and wiring of software modules and hardware.
Flexible REST-APIs	Based on Python and OpenAPI with Swagger auto-documentation of deployed API endpoints. Enabling easy integration with visualization tools and interoperability through open API (REST).
Python OPC UA server and client	For OPC UA communication and testing for easy code integration and no black boxes.
NodeJS OPC UA logger	For asynchronous logging of subscribed OPC UA process tags to database.

SINTEF Python Event Bus	Event bus/state machine for handling event- based workflows. Maintains control of states for input/output data. Can be customized for executing events which listens to the states and responds based on transitions in time or state.
Prometheus	For (administrative) monitoring remote deployment and alert management.
Grafana server	For flexible dashboarding and visualization of databases.
Visualisation module (web server)	Serving of Python Django based webpages with embedded dashboarding with Grafana, Node-RED and Python/Javascript visualization tools. Allows for interactive sharing of process information, data, code and results.
Jupyter Hub server	For embedding of interactive notebooks as part of visualization module. Brings project notebooks to users. Gives users access to computational environments and resources without burdening the users with installation and maintenance tasks. Notebooks are used for sharing of interactive code or for personal use.

Examples of usage / illustrations

Illustration of an example Bedrock Toolbox deployment:



Interfaces (in/out) – system/user
<ul style="list-style-type: none"> ▪ Python, Matlab or similar ▪ InfluxDB API ▪ PostgreSQL API - with TimeScaleDB extension ▪ MQTT brokers ▪ Kafka brokers ▪ REST-API ▪ OPC UA ▪ Web application - with flexible dashboards as part of a customizable website framework. ▪ SharePoint – import or export of files
Subordinates and platform dependencies
Based on several open-source platforms.
Licenses, etc. (free for use in the project)
Various open source licenses.
TRL for overall component/tool and any parts/subordinates
5-6 for the overall combined framework
References – incl. web etc.
https://stash.code.sintef.no/projects/DIG/repos/bedrock_toolbox/browse
To be considered in particular for the following COGNITWIN pilots
Sidenor and Sumitomo SHI FW

10.2.1.2 TIA IOT: Industrial Internet of Things Platform

Component/Tool/Method/Framework/Service Name
TIA IOT: Industrial Internet of Things Platform
Short Description – incl. Purpose
<p>TIA IOT is a platform that is used to acquire, collect and store sensor and PLC data. The platform enables real-time stream processing as well as batch processing. TIA IoT is composed of TIA SENSOR, TIA PLC and TIA CONTROL.</p> <p>TIA IOT platform provides a drag-and-drop easy-to-use interface and enables none technical users to easily create stream pipeline elements and pipelines.</p> <p>PLC data is collected by OPC and later via MQTT is passed to Kafka. Data in Kafka is consumed by Cassandra and PostgreSQL. The data are stored in the Cassandra database with three columns: id, time, and value. The id column shows the component to which the data belong. The JSON format streams of the data transferred to the Cassandra database are presented to users as a log file.</p> <p>In the context of COGNITWIN, the IoTP output will be useful both in real-time condition monitoring and conducting the predictive maintenance.</p> <p>A fully asynchronous communication structure with the event-bus method is used for the transmission of data collected from the source with OPC. Data transmission is provided in the JSON format. In the architecture managed on the basis of Microservice, Cassandra is used as the NoSQL</p>

database, and PostgreSQL, a relational database (RDBMS), is used by the interface program that provides user interaction.

Progress since last milestone

Since the last milestone:

- Two PLCs were connected by PN/PN Coupling.
- Sensor data was reorganized by means of a new coding system.
- OPC gateways and tags were updated and finalized with respect to the new coding system introduced.
- Kafka topics have been redefined for optimized performance.
- Cassandra database tables were updated so as to store component-based data per table.
- Scripts to create OPC IoT gateways, database tables, Kafka topics, etc. were updated.
- A new tag was introduced and utilized to check whether the PLC is on or off.
- Verification and validation have been completed.

Examples of usage / illustrations

PN/PN Coupler module is used for PLCs to communicate. Below figure presents coupling of the PROFINET subnets with the PN/PN Coupler:

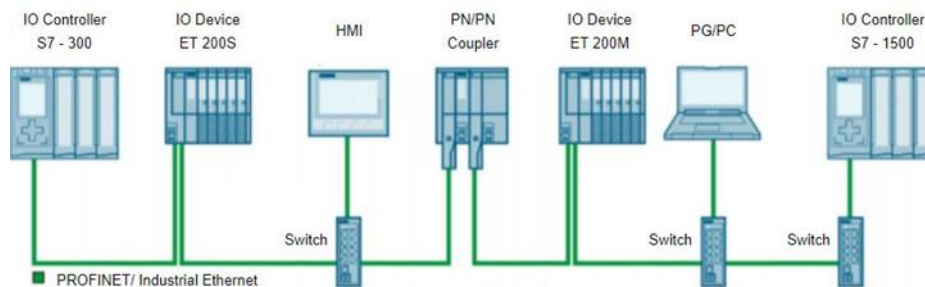


Figure 75: Coupling of the PROFINET subnets with the PN/PN Coupler

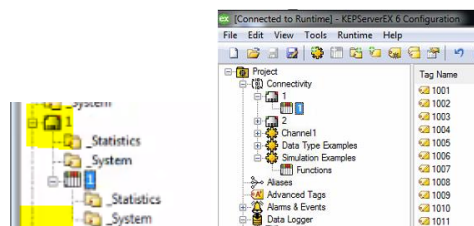
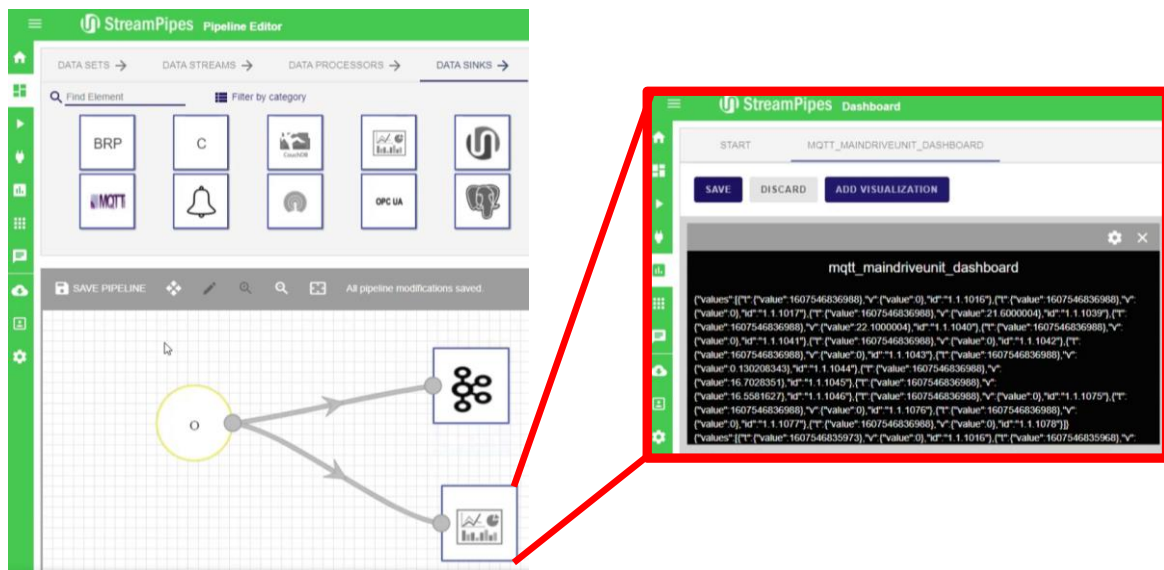


Figure 76: PLC definition on OPC, and Sample Tag List Defined

The pipeline used by the platform to acquire and distribute sensor data from OPC to Kafka is presented as in the following Figure:



Figure

Figure 77: Pipeline for OPC -> MQTT -> Kafka in JSON

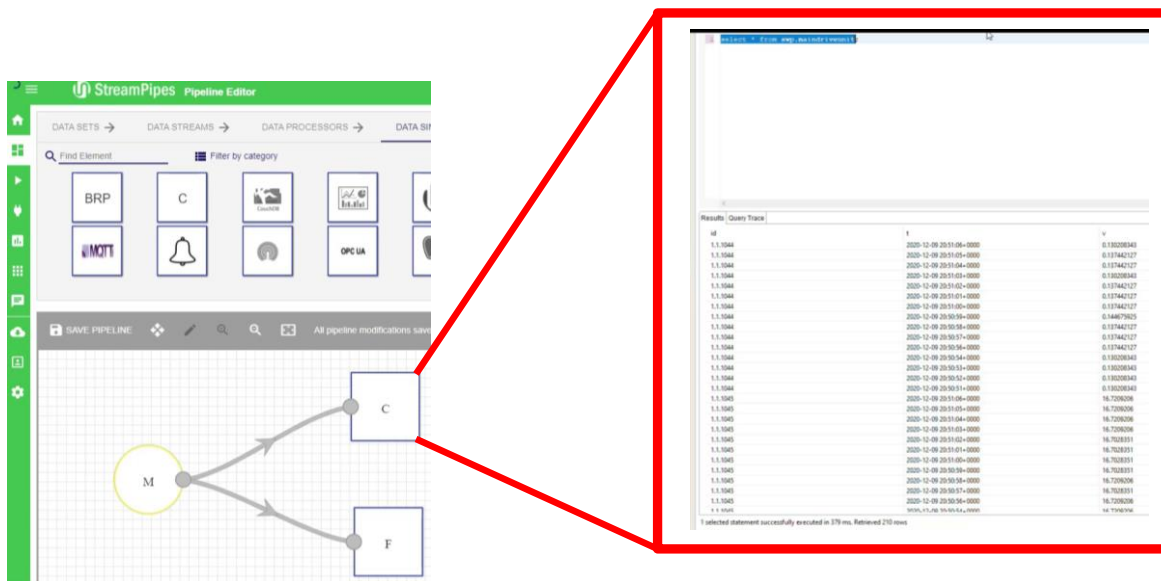


Figure 78: Pipeline to demonstrate Kafka data saved on to Cassandra database

Interfaces (in/out) – system/user

MQTT version 5, 3.1 and 3.1.1 are used, data in JSON is generated and stored in database, which is used by TIA DATA, and TIA AI, and also displayed by TIA UX

Subordinates and platform dependencies

TIA IOT is based on Apache StreamPipes and uses OPC data.

Licenses, etc. (free for use in the project)

Partially open

TRL for overall component/tool and any parts/subordinates
The current TRL is 6.
References – incl. web etc.
https://tia-platform.com/module/tia-iot.html
https://hub.docker.com/_/eclipse-mosquitto
To be considered in particular for the following COGNITWIN pilots
NOKSEL

10.2.1.3 TIA SENSOR

Component/Tool/Method/Framework/Service Name
TIA SENSOR
Short Description – incl. Purpose
TIA SENSOR is one of the applications of TIA IOT. TIA SENSOR services enable the users to select the optimum set of sensors. The service implements the optimum sensor set for the machines.
Progress since last milestone
Verification and validation have been completed.
Examples of usage / illustrations
Interfaces (in/out) – system/user
TIA SENSOR enables users to install sensors and collects data from the installed sensors. Inputs are sensor ID and type, output is sensor data.
Subordinates and platform dependencies
TIA SENSOR enables users to install sensors and collects data from the installed sensors. Inputs are sensor ID and type, output is sensor data.
Licenses, etc. (free for use in the project)
Proprietary/ Subject to license
TRL for overall component/tool and any parts/subordinates
The current TRL is 7.
References – incl. web etc.
https://tia-platform.com/module/tia-iot.html
To be considered in particular for the following COGNITWIN pilots

NOKSEL

10.2.1.4 TIA PLC

Component/Tool/Method/Framework/Service Name
TIA PLC
Short Description – incl. Purpose
TIA PLC is part of the TIA IOT. TIA PLC collects data from the PLCs. TIA PLC services enable the users to select, program, and install PLCs to read from the PLC.
Progress since last milestone
Verification and validation have been completed.
Examples of usage / illustrations
PLC to PLC coupling has been realised.
Interfaces (in/out) – system/user
TIA PLC uses PLC selected fields as input and gather their data to be used by other applications.
Subordinates and platform dependencies
Platform independent
Licenses, etc. (free for use in the project)
Proprietary/ Subject to license
TRL for overall component/tool and any parts/subordinates
The current TRL is 7.
References – incl. web etc.
https://tia-platform.com/module/tia-iot.html
To be considered in particular for the following COGNITWIN pilots
NOKSEL

10.2.1.5 TIA CONTROL

Component/Tool/Method/Framework/Service Name
TIA CONTROL
Short Description – incl. Purpose
TIA CONTROL is part of TIA IOT. TIA CONTROL module enables the users to create for remote management and control of a machine or process.
Progress since last milestone
This is a new component, it has been designed and implemented after D4.3 delivery.

Examples of usage / illustrations
Temperature data is data is taken as an input variable to be controlled and a control system using ontology is designed and implemented.
Interfaces (in/out) – system/user
TIA CONTROL uses variables to be controlled as input, and as output provide intended control on them. It may utilize interfaces with TIA SENSOR and TIA PLC.
Subordinates and platform dependencies
Platform independent
Licenses, etc. (free for use in the project)
Proprietary/ Subject to license
TRL for overall component/tool and any parts/subordinates
The current TRL is 7.
References – incl. web etc.
https://tia-platform.com/module/tia-iot.html
To be considered in particular for the following COGNITWIN pilots
NOKSEL

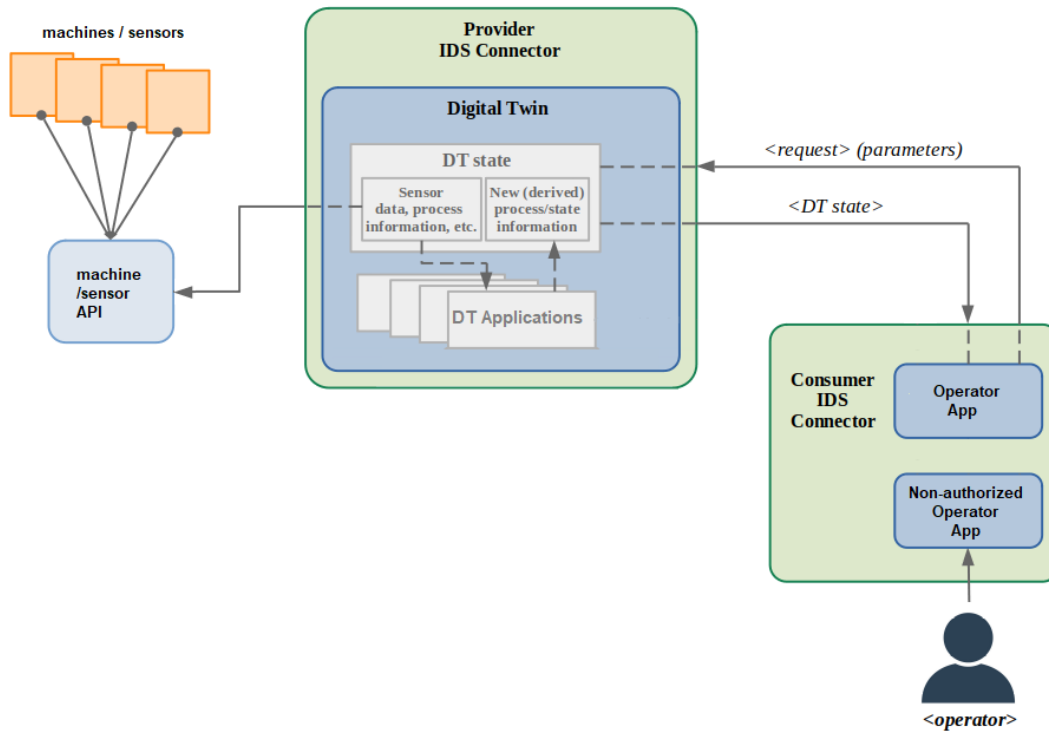
10.2.2 Security and IDS – International Data Spaces

10.2.2.1 Trusted Factory Connector (IDS)

Component/Tool description
Component/Tool/Method/Framework/Service Name
Trusted Factory Connector (IDS)
Short Description – incl. Purpose
The Trusted Factory Connector is based on the AISEC Trusted Connector and is the central gateway to the IDS network. It is based on the German standard, DIN SPEC 27070, which in general describes security gateways. The IDS enable companies to share data across company borders without losing data control. Our use-case includes sharing of critical factory data to mediation platforms in order to realize new business ideas.
Progress since last milestone
Notable technical extensions include the replacement of LUCON with MYDATA ³⁵ Usage Control Framework and the support for digital twins in form of Asset Administration Shells (AAS). A visualization app is used to pull data from digital twins. All data can be protected by Usage Control Policies, so that sharing between companies becomes data sovereign. This is also shown in a new demonstrator to present different standards seamlessly working with each other.

³⁵ <https://www.mydata-control.de/>

Examples of usage / illustrations



Interfaces (in/out) – system/user

IDS connectors usually only communicate with other IDS connectors. For this, the IDS Information Model is used. To connect Apps to the Connector, REST is used, in our case the AAS REST API. Users are usually interacting with graphical interfaces in those apps.

Subordinates and platform dependencies

The Connector is deployed with Docker. Natively the connector is written in Java.

Licenses, etc. (free for use in the project)

The Connector is open-source and free for use. The integrated Usage Control solution MYDATA however is only free for research. Production users will have to license this solution.

TRL for overall component/tool and any parts/subordinates

TRL 6 – prototypes have been used in demo factories

References – incl. web etc.

<https://github.com/c3di/neuroscope> <https://www.mydata-control.de/>

<https://www.beuth.de/de/technische-regel/din-spec-27070/319111044>

<https://www.iosb.fraunhofer.de/de/projekte-produkte/indaspaceplus-industrial-data-spaces-plus.html>

To be considered in particular for the following COGNITWIN pilots

Sidenor

10.2.2.2 IDS Connectors - SINTEF

Component/Tool description

Component/Tool/Method/Framework/Service Name

IDS Connector - SINTEF

Short Description – incl. Purpose

The IDS Reference Architecture Model (IDS-1) positions itself as an architecture that links different cloud platforms through policies and mechanisms for secure data exchange and trusted data sharing (through the principle of data sovereignty). Over the IDS Connector, industrial data clouds, individual enterprise clouds, on-premise applications and individual, connected devices can be connected to the International Data Space ecosystem.

Key participants (actors in the system) in an IDS Data Space system would be the Data Owner, Data Provider, Data Consumer, Data User or Broker Service provider. The complete landscape of roles, their functionalities and relationships result in a model depicted in the following figure.

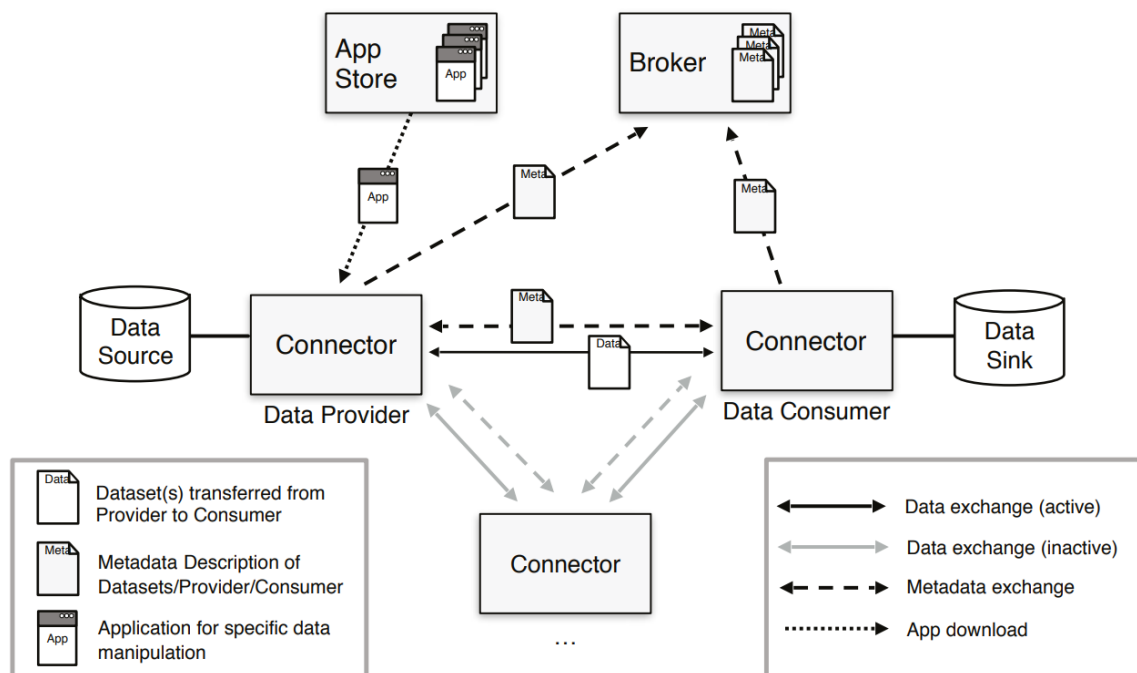


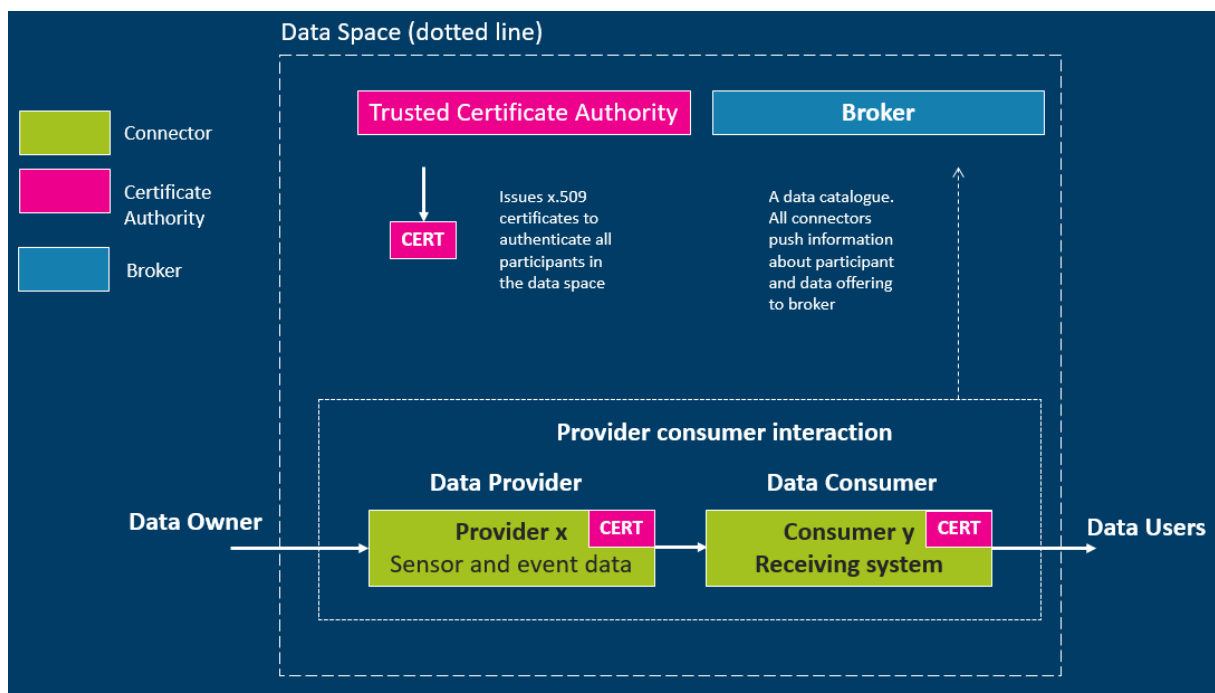
Figure 79: Interaction between technical components of IDS Reference Architecture Model

The Connector is the central technological building block of IDS. It is a dedicated software component allowing Participants to exchange, share and process digital content. At the same time, the Connector ensures that the data sovereignty of the Data Owner is always guaranteed. The Broker Service Provider is an intermediary that stores and manages information about the data sources available in IDS. The activities of the Broker Service Provider mainly focus on receiving and providing metadata that allow provider and consumer connectors to exchange data. The App Provider role is optional in IDS, and its

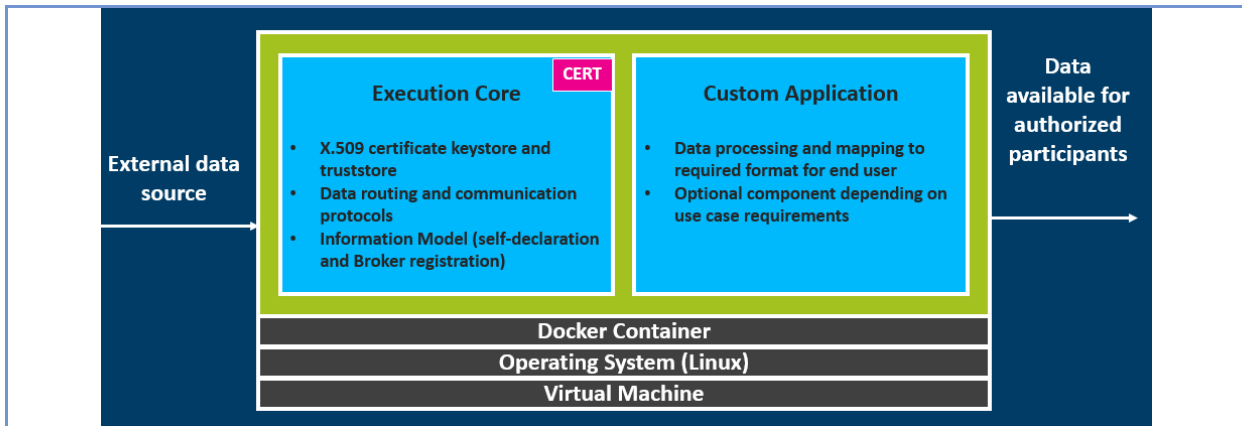
main role is to develop applications that can be used by both data providers and consumers in the data space. Applications are typically downloaded from the remote app store, and run inside the containerized connector.

Establishing trust for data sharing and data exchange is a fundamental requirement in IDS. The IDS-RAM defines two basic types of trust: 1) Static Trust, based on the certification of participants and core technical components, and 2) Dynamic Trust, based on active monitoring of participants and core technical components. For data sharing and data exchange in the IDS, some preliminary actions and interactions are required. These are necessary for every participant, and involve a Certification Body, Evaluation Facilities, and the Dynamic Attribute Provisioning Service (DAPS). The following illustrates the roles and interactions required for issuing a digital identity in IDS:

1. Certification request: This is a direct interaction between a participant and an evaluation facility to trigger an evaluation process based on IDS certification criteria.
2. Notification of successful certification: The Certification Body notifies the Certification Authority of the successful certification of the participant and the core component. Validity of both certifications must be provided.
3. Generating the IDS-ID: The Certification Authority generates a unique ID for the pair (participant and component) and issues a digital certificate (X.509).
4. Provisioning of X.509 Certificate: The Certification Authority sends a digital certificate (X.509) to the participant in a secure and trustworthy way and notifies the DAPS.
5. Register: After the digital certificate (X.509) is deployed inside the component, the component registers at the DAPS.
6. DTM Interaction: The Dynamic Trust Monitoring (DTM) implements a monitoring function for every IDS Component, and DTM and DAPS then exchange information on the behaviour of the component, e.g. about security issues (vulnerabilities) or attempted attacks.



The figures shows the patterns in the framework of IDS connector setup existing from SINTEF, (IDS-3) for secure data transfer by the support of a Trusted Certificate Authority.



Connectors and broker use mTLS (mutual Transport Layer Security) to communicate

Secure communication in all business cases based on two-way authentication with CyCIMS certificates All Data Space participants are assigned a custom domain. Each certificate is again linked to this unique domain, and this forms the basis of the Data Space trust ecosystem. Authorization of user access to data offerings: We use the unique thumbprint of each certificate to allow and restrict access

Progress since last milestone

No work has been done on this component since the last milestone. The offering here is the same as presented for the previous deliverable D4.1. Further work on IDS connectors is pending any identified needs by the pilots to establish an IDS ecosystem with connectors. The first step for this will be related to the AAS IDS connectors through the **Trusted Factory Connector (IDS)** component provided by Fraunhofer for the AAS case. Further Data Space support will be analysed in the context of the pilot needs and priorities.

Examples of usage / illustrations

The IDS Connector from SINTEF has been implemented and realised in the context of a Maritime Data Space involving data exchange between ships and harbour in the national Maritime Data Space project. The operational framework for this is a suitable starting point for the development of additional IDS Connectors also in the context of the COGNITWIN project

Interfaces (in/out) – system/user

The interfaces of the Connectors following the X.509 standard and the protocols of the IDS reference architecture.

Subordinates and platform dependencies

IDS architecture and solutions.

Licenses, etc. (free for use in the project)

Proprietary/ Subject to license

TRL for overall component/tool and any parts/subordinates

The current TRL is 6

References – incl. web etc.

IDS references:

(IDS 1) <https://www.internationaldataspaces.org/>

(IDS 2) <https://www.internationaldataspaces.org/wp-content/uploads/2019/03/IDS-Reference-Architecture-Model-3.0.pdf>

(IDS 3) <https://www.sintef.no/projectweb/maritime-data-space-mds/>

To be considered in particular for the following COGNITWIN pilots

TBD

10.2.3 Digital Twin API – AAS

10.2.3.1 FA³ST – Fraunhofer Advanced AAS Tools for Digital Twins

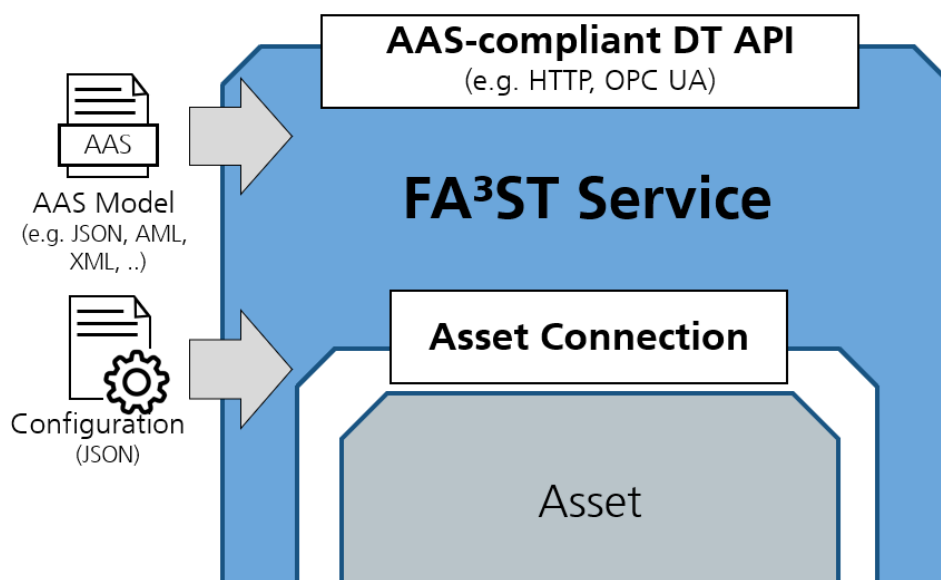
Component/Tool description

Component/Tool/Method/Framework/Service Name

FA³ST (Fraunhofer Advanced AAS Tools for Digital Twins) Service

Short Description – incl. Purpose

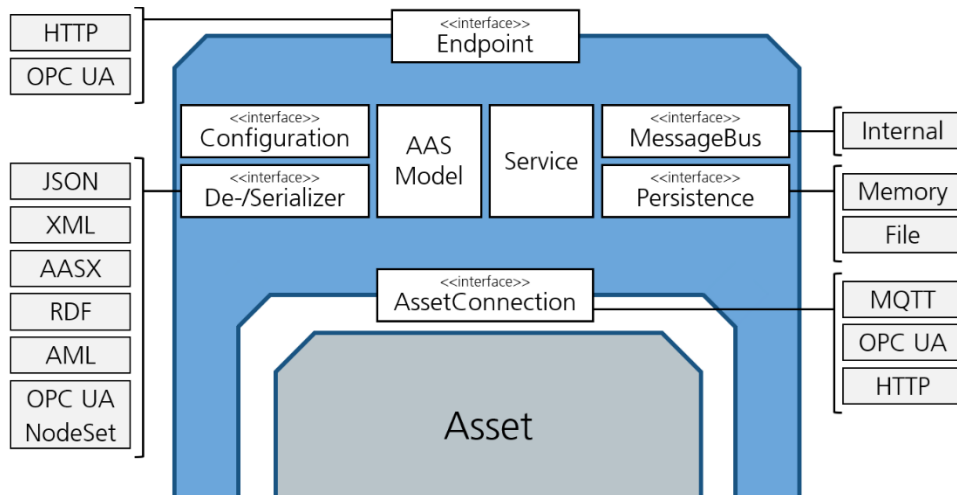
A³ST Service is an open source Java-based software for creating and managing digital twins (DTs), so-called Asset Administration Shells (AASs). It is based on the standard document(s) published by the Plattform Industrie 4.0 [1] and has been used in the project to develop standard-conform executable DTs in the pilots. Main features are that it is designed to be easy to use and extend and that it can be connected to assets using arbitrary communication protocols. To create a reactive DT, all you need to do is start FA³ST Service with an AAS model and a configuration file and in return you get a DT with an AAS-compliant DT API for interaction of the DT with the outside world that can synchronize itself with the underlying asset(s).



FA³ST service is designed to be easily extendible and offers a variety of extension points via interfaces, e.g. for de-/serialization formats and persistence implementation as well as different protocols for endpoints and asset connections.

The following figure shows a high-level architectural view of the FA³ST Service components (white boxes). Most of them are designed as interfaces which enables supporting different concrete implementations in parallel via loose coupling and allows for easy extension in the future. The concrete

implementations of these interface that are already shipped with FA³ST Service are represented by the light grey boxes on the sides.



The basic functionality of each function block is as follows:

Endpoint (interface) - A network endpoint that provides connectivity from external applications to the DT via a standardized API, e.g. via HTTP or OPC UA.

Configuration (interface) - The Configuration interface allows for easy integration and configuration of any kind of implementation of one of the other interfaces via a single configuration file.

AAS Model - This is a code representation of the AAS meta model as defined in the standard document [1]. FA³ST Service uses the adminshell.io java-model open-source library [2] for this which is currently in the process of moving to Eclipse AAS4j [3].

De-/serializer (interface) - The two interfaces allow implementing different de-/serialization algorithms. This is important because the standard defines multiple data formats (JSON, XML, RDF, AutomationML) that should be supported.

MessageBus (interface) - The message bus enables communication and synchronization between the different components in FA³ST Service. This is crucial to keep a consistent state, e.g. when having both an HTTP and OPC UA endpoint.

Persistence (interface) - This interface abstracts from concrete implementations of data storage so that it is easy to integrate any kind of data storage into a DT. Current implementation include in-memory storage as well as file-based storage.

Asset Connection (interface) - This interface defines how the DT interacts with the actual (physical) asset/device. As assets can communicate via lots of different protocols, e.g. HTTP, OPC UA, Profibus, CAN bus, etc., it is crucial that the AAS service library allows custom implementations of this connection logic. FA³ST Service ships with implementation for HTTP, OPC UA and MQTT.

FA³ST Service tries to be flexible and user-friendly but providing clear interfaces for custom extensions as well as a sophisticated configuration mechanism that allows to create your own DT without writing any code. Instead, FA³ST Service is used by providing a configuration file together with your AAS model file on startup. It can be used via command-line interface, as docker container or as embedded library.

As keeping the DT in sync with the real world is crucial, FA³ST Service introduced the concept of asset connection which goes beyond what is defined in the AAS specification. The AssetConnection interface

defines the three conceptual interaction patterns between DT and its asset(s) in a protocol-agnostic way: read/write values, subscribe to values, and execute operations. This protocol-agnostic definition allows FA³ST Service to be integrated with any kind of communication protocol.

FA³ST Service also provides experimental support for integrating time series data into a DT, either stored in the DT itself or from external databases. The implementation is based on a pre-release version of the *SubmodelTemplate Time Series Data* (SMT Time Series Data) specification; an extension/companion specification to the AAS specification

Progress since last milestone

FA³ST Service has been published as open source [4] and continuously enhanced and improved over time. Major progress is that FA³ST Service now supports all of the defined AAS functionality and data formats. It also provides a full-fledge OPC UA endpoint and supports HTTP and MQTT asset connections. Additionally, many smaller improvements and extensions have been made, e.g. all asset connection now support basic authentication and automatic reconnection upon connection loss, support for generic submodel template processors as well as support for concrete submodel template Time Series Data with data being stored in external an InfluxDB. You can find a detailed overview of improvements and fixes over time in the changelog [5].

Overall, FA³ST Service has been established as an open source AAS implementation that is recognized and used in the AAS community.

Examples of usage / illustrations

Usage via command-line interface (CLI)

```
> java -jar starter-{version}.jar --model myModel.json -config myConfig.json
```

Full list of available CLI parameters:

Name (short)	Name (long)	Description
-c	--config	The config file to use.
	--emptyModel	Starts the FAST service with an empty Asset Administration Shell Environment.
	--endpoint	Additional endpoints that should be started.
-h	--help	Print help message and exit.
	--loglevel-external	Sets the log level for external packages. This overrides the log level defined by other commands such as -q or -v.
	--loglevel-faaast	Sets the log level for FA ³ ST packages. This overrides the log level defined by other commands such as -q or -v.
-m	--model	The model file to load.
	--[no-]autoCompleteConfig	Autocompletes the configuration with default values for required configuration sections.
	--[no-]modelValidation	Validates the AAS Environment.
-q	--quite	Reduces log output (ERROR for FAST packages, ERROR for all other packages). Default information about the starting process will still be printed.

-v	--verbose	Enables verbose logging (INFO for FAST packages, WARN for all other packages).
-V	--version	Print version information and exit.
-vv		Enables very verbose logging (DEBUG for FAST packages, INFO for all other packages).
-vvv		Enables very very verbose logging (TRACE for FAST packages, DEBUG for all other packages).
	<key=value>	Additional properties to override values of configuration using JSONPath notation without starting '\$.' (see https://goessner.net/articles/JsonPath/)

Usage as docker container

Pre-built docker images for FA³ST Service are published on Docker Hub [6] and can easily be run via

```
> docker run fraunhoferiosb/faaast-service
```

To configure FA³ST Service in a docker container, you can either use the CLI commands listed above or use environment variables.

Usage from code

Starting a FA³ST Service from code can be done in one command using the fluent API:

```
new Service(ServiceConfig.builder()
    .core(CoreConfig.builder()
        .requestHandlerThreadPoolSize(2)
        .build())
    .persistence(PersistenceInMemoryConfig.builder()
        .environment(EnvironmentSerializationManager
            .deserialize(new File("myModel.json"))
            .getEnvironment())
        .build())
    .endpoint(HttpEndpointConfig.builder()
        .port(8080)
        .build())
    .messageBus(MessageBusInternalConfig.builder()
        .build())
    .build())
    .start();
```

When executed via CLI the output looks similar to this

Interfaces (in/out) – system/user

Ease-of-use has been a primary objective in designing FA³ST Service. It has been achieved by offering different usage modes (CLI, docker, embedded) and use of a single easy-to-maintain configuration file resulting in a no-code usage approach.

FA³ST Service also offers lots of software interface to extend or adjust the software to the users need. These interfaces are documented on code-level and also described in a human-focused documentation.

For communication with the outside world, DTs realized with FA³ST Service offer a standardized AAS-compliant DT API which currently can be used based on HTTP, OPC UA, or both at the same time. This interface can be used to register, read, write, update, delete, and search for AAS and their sub-elements.

Subordinates and platform dependencies

The software is based on Java 11. The code has dependencies on multiple open source libraries, e.g. Eclipse Jetty [7] or Jackson [8]. For the AAS metamodel and de-/serialization we use the open source library java-model [2] resp. java-serializer [9] from adminshell.io, which is currently moving to Eclipse AAS4j [3].

All required libraries are licensed under either Apache 2.0, LGPL or MIT.

Licenses, etc. (free for use in the project)

The software is open source under Apache 2.0 License.

TRL for overall component/tool and any parts/subordinates

TRL 5/6

References – incl. web etc.

- [1] https://www.plattform-i40.de/PI40/Redaktion/EN/Downloads/Publikation/Details_of_the_Asset_Administration_Shell_Part1_V3.html
- [2] <https://github.com/admin-shell-io/java-model/>
- [3] <https://github.com/eclipse-aas4j/aas4j>
- [4] <https://github.com/FraunhoferIOSB/FAAAST-Service>
- [5] <https://faaast-service.readthedocs.io/en/latest/changelog/changelog/>
- [6] <https://hub.docker.com/r/fraunhoferiosb/faaast-service>
- [7] <https://www.eclipse.org/jetty/>
- [8] <https://github.com/FasterXML/jackson>
- [9] <https://github.com/admin-shell-io/java-serializer>

To be considered in particular for the following COGNITWIN pilots

FA³ST Service has been applied in the Sidenor and Hydro use cases.

10.2.4 Digital Twin Graph support for Simulation and Cognition

Component/Tool description

Component/Tool/Method/Framework/Service Name

Digital Twin Graph support for Simulation and Cognition

Short Description – incl. Purpose

The framework covers aspects related to Digital Twin data representation (graph-based data structures for assets data and processes), data storage (assets and time-series data), and support for discrete/continuous simulation (e.g., estimation of production capacity).

The purpose of the framework is to provide the mechanisms to represent and store data in a way that can capture assets and time-series data, while at the same time can offer efficient access to the data and ability to use the stored data for various types of simulations (discrete/flow).

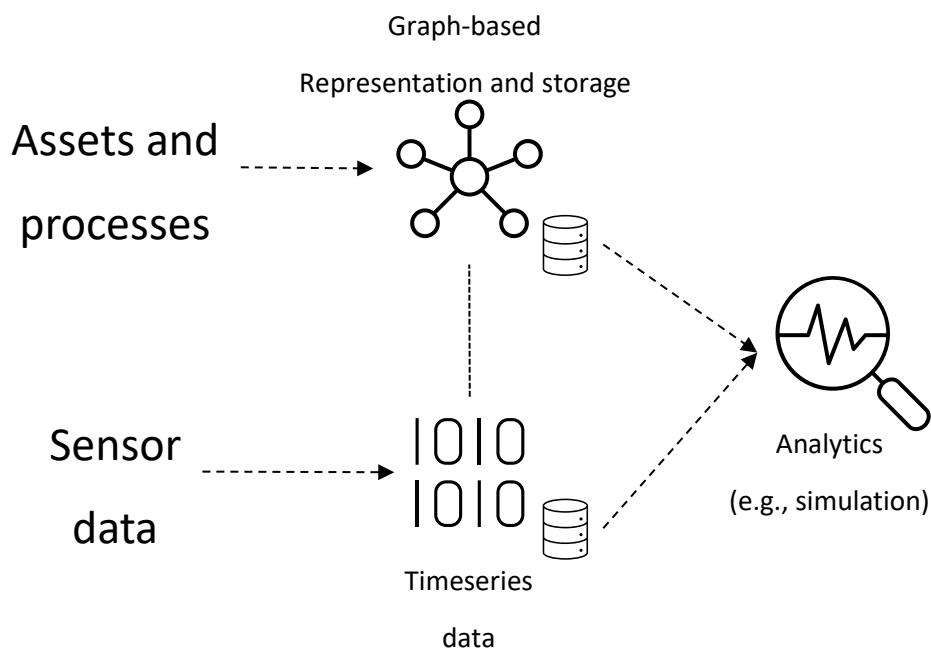
The framework is meant to help factories IT personnel in the process of representing assets and processes within factories and storing data about them, and combining the information with real-time sensor data, to facilitate analytics tasks such as simulations.

Progress since last milestone

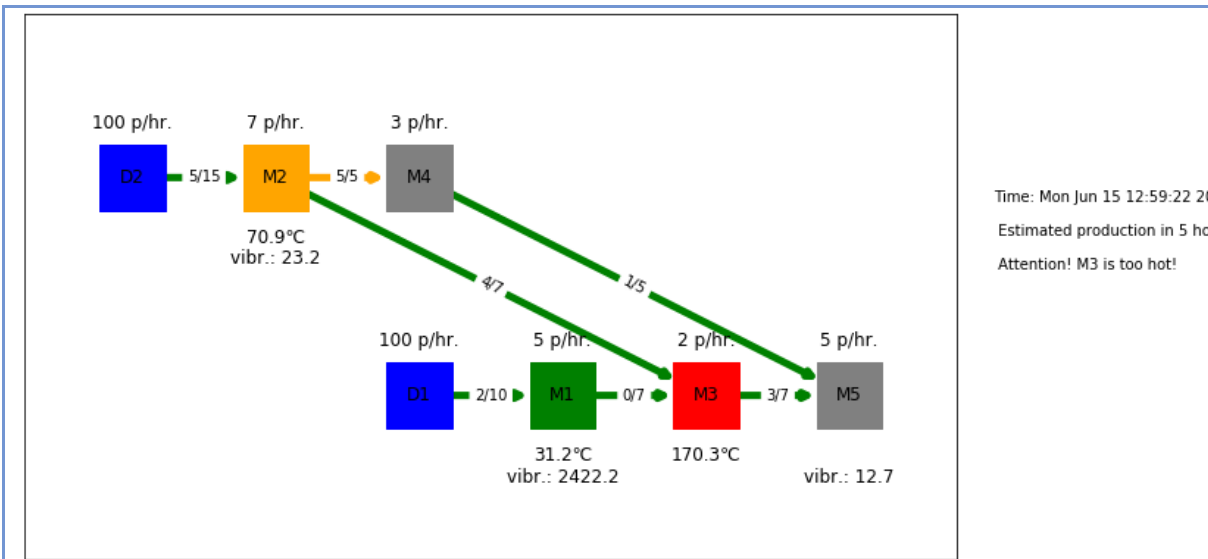
The framework has been designed and a proof of concept has been implemented since the last milestone.

Examples of usage / illustrations

A typical usage of the framework is depicted below, where information about assets and processes is captured in a flexible graph data model (nodes representing assets, connections representing flows of parts between assets), and sensor data is connected to graph data, that in turn is used for analytical tasks such as simulations.



The illustration below depicts a UI of the implemented prototype, where a set of processing machines (nodes) and the flow of parts across the machines (arrows) is depicted, together with production capacity estimation based on discrete simulation of production at a certain point in time (colours depict various types of information based on sensor data).



Interfaces (in/out) – system/user

The prototype has API-level interfaces for data manipulation (CRUD operations on graph and timeseries data). In addition a simple GUI is provided for visualization of analytics tasks (as depicted above).

Subordinates and platform dependencies

The prototype was implemented on a Python-based stack and 3rd party databases were used (e.g., InfluxDB for timeseries data, Neo4j for graph data).

Licenses, etc. (free for use in the project)

The implemented prototype is planned to be released as open source under a flexible license (e.g., Apache-2.0 License – to be finally decided when the prototype is publicly released).

TRL for overall component/tool and any parts/subordinates

TRL4 – a prototype based on a Python stack is implemented at this stage and validated in lab setting.

References – incl. web etc.

<https://github.com/SINTEF-9012/dt-prototype> (The prototype is not yet publicly released).

To be considered in particular for the following COGNITWIN pilots

The framework is to be checked for fit-for-purpose in all the pilots. It is expected that at least the data representation/storage/simulation is of high relevance in the pilots.

10.2.5 Outlier detection OD-tool

Component/Tool description
Component/Tool/Method/Framework/Service Name
Outlier detection tool (OD-tool)
Short Description – incl. Purpose

The FouMon component collects tools for solving the fouling monitoring problem. It includes the ODtool application independent tool for outlier detection and data quality improvement, to support data-driven modeling of system dynamics

Creation of a purely data-driven model requires good quality data. Plant measurements tend to have occasional errors that must be taken into consideration for the self-learning digital twin to be robust. Therefore, in addition to standard measurement verification by detecting sensor faults (1) and frozen values etc. (2), an additional outlier detection (3) method is implemented for this application. Identifying the nontrivial outliers from a batch of multidimensional measurement vectors supports the process of finding a valid digital twin model, thus enhancing the cognitive capabilities of the approach. The tool is available for download.

Progress since last milestone

The work has been conducted in the period M30-M42.

Examples of usage / illustrations

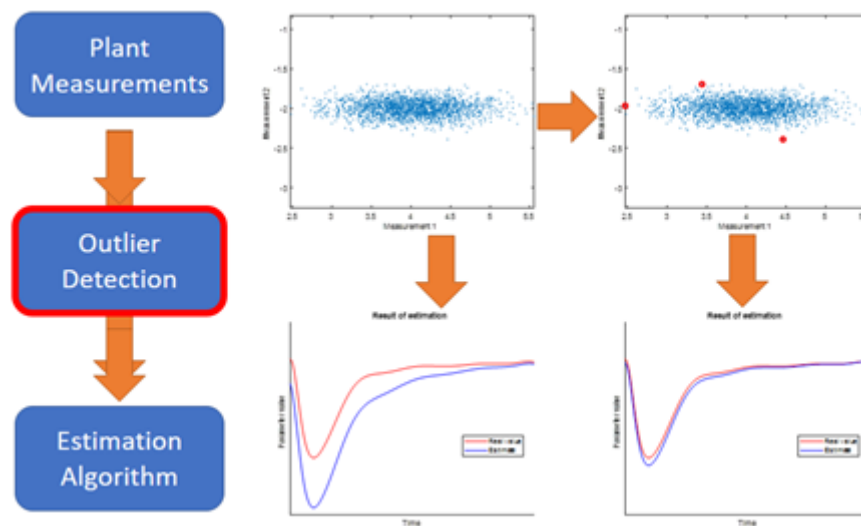


Figure 80: The basic principle of ellipsoidal peeling. Removal of few outliers can improve the performance of identification significantly

Interfaces (in/out) – system/user

The physical model and measurements are set up in the Matlab m-files. Input data (measurements) are provided as numerical vectors. Interactive tuning is enabled by Matlab interface/graphics. Estimation outcomes are provided as numerical vectors.

Subordinates and platform dependencies

The tool is implemented using Matlab language (m-files). Matlab from the Mathworks is required (OD-tool/FouMon has been tested on Matlab 2020b).

Matlab (2020b) is available on all major operating systems, including Windows 7, Ubuntu 16, Debian 9, MacOS 10 and newer. No particular Matlab Toolboxes are required. Open software Octave is known to be able to interpret m-files, but FUSE-codes have not been tested with Octave.

Licenses, etc. (free for use in the project)

ODtool is free for use (contact Markus.Neuvoenen@oulu.fi)
TRL for overall component/tool and any parts/subordinates
TRL4 – a Matlab algorithm is implemented, the approach has been tested using real plant data from the Sumitomo SHI FW pilot problem.
References – incl. web etc.
COGNITWIN Toolbox portal.
To be considered in particular for the following COGNITWIN pilots
Approach is general-purpose. It has been developed and tested in view of the needs for solving the Sumitomo SHI FW pilot problem.

10.3 Toolbox Components- Realtime sensor/data processing

10.3.1 Real-time data preprocessing with complex event processing:

10.3.1.1 StreamPipes Siddhi-Processor

Component/Tool description
Component/Tool/Method/Framework/Service Name
StreamPipes Siddhi-Processor
Short Description – incl. Purpose
<p>StreamPipes Siddhi-Processor is a component that enables analysis of data streams in a context of CEP (<i>Complex Event Processing</i>) using StreamPipes as underlying platform.</p> <p>Siddhi was used for its implementation, as it is a tool for building fully-fledged event-driven applications with possibility of Complex Event Processing.</p> <p>In current version (0.67.0), StreamPipes (SP) offers a Siddhi wrapper which wraps standard library for SP pipeline element creation, Figure 81. It provides mapping of SP pipeline element’s input/output to the Siddhi application’s input/output, as well as means of writing and executing queries that can be customized with user provided parameters during element configuration.</p>
<p style="text-align: center;">Siddhi Wrapper</p>

Figure 81: Siddhi wrapper that enables users to utilize CEP inside StreamPipes element

Currently, SP Siddhi wrapper supports part of commonly-used Siddhi EPL functionalities, with the rest of them, as well as other features and extensions, yet to be implemented. In addition, there are a few problems with SP UI and API when Siddhi processor is employed.

This implementation of Siddhi wrapper requires from users to write Siddhi queries as plain *String* objects. In following version (0.68.0), StreamPipes (SP) will provide object model representation of Siddhi EPL (*Event Processing Language*) for writing Siddhi queries implemented as a part of SP Siddhi wrapper. Significant changes to the Siddhi wrapper implementation are also expected.

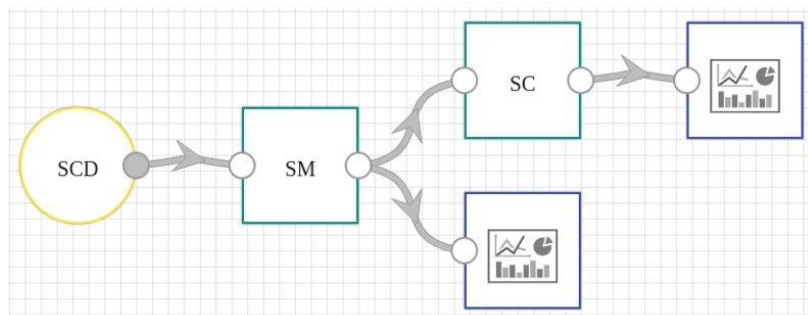
SP Siddhi-Processor's purpose is to extract information and identify meaningful events (*opportunities and threats*) such as patterns, relationship between events, etc. It would receive its input from SP element(s), execute written query on received data, and forward execution result to other SP element(s).

Progress since last milestone

Created StreamPipes pipeline element that utilizes CEP using StreamPipes wrapper for Siddhi.

Wrote Siddhi query that counts number of occurrences of interest in a given window of inputs and outputs this value.

Tested query and pipeline element on Sidenor pilot, where we counted in how many inputs outlier count was above a predefined threshold value (occurrence of interest). SP Siddhi-Processor was tested with following SP pipeline:



SP Siddhi-Processor element is labelled SC, for *Sidenor CEP*. It holds query that performs above-mentioned task. During pipeline creation, user is able to parametrize it, e.g., define window size. It receives output from element labelled SM, for *Sidenor MEWMA (Multivariate Exponentially Weighted Moving Average)*. MEWMA is used to detect outliers in an input data and outputs said detections to the element with Siddhi processor.

Examples of usage / illustrations

The goal of this element is to count number of occurrences of interest in a given set of consecutive inputs. This use-case corresponds to the CEP notion of a *window*, i.e., we want to detect a specific event in a window of inputs. Hence, the resulting query looks like this, with the window size being 5:

```
define stream InputStream(numerical_parameter int);
from InputStream#window.length(5)[numerical_parameter >= 50]
select count() as count
insert into OutputStream;
```

This query counts how many inputs had numerical_parameter value greater than 50. Therefore, in this case, numerical_parameter value greater than 50 represents an occurrence of interest.

For this element, window size, parameters of interest and conditions, are fully customizable and can be changed during pipeline editing phase.

Output of this component can be used by other SP pipeline elements to raise an alert, for further processing, visualization, etc.

Interfaces (in/out) – system/user

This component receives input from any element that provides numerical output that is of interest. In the case of test, output from MEWMA is used (count of outliers).

After query execution, it outputs count of inputted values that correspond to the condition.

Since this component is implemented in a way that allows custom values for window length and number of occurrences, a user interaction is required. In essence, when user creates pipeline and employs this component, a corresponding window pops up which prompts user to enter said values.

Subordinates and platform dependencies

StreamPipes (and this component, as well) is available for Linux, Windows and Mac OS X.

This component was developed on Linux machine. Docker and Docker Compose are required in order to run pipelines.

Licenses, etc. (free for use in the project)

Licence: proprietary

TRL for overall component/tool and any parts/subordinates

TRL6

References – incl. web etc.

StreamPipes - <https://github.com/apache/incubator-streampipes/tree/rel/0.67.0>

StreamPipes extensions - <https://github.com/apache/incubator-streampipes-extensions/tree/rel/0.67.0>

StreamPipes Siddhi wrapper - <https://github.com/apache/incubator-streampipes/tree/rel/0.67.0/streampipes-wrapper-siddhi>

To be considered in particular for the following COGNITWIN pilots

The component is currently only used for the Sidenor pilot. However, it is intended to be used by other pilots where there is a need e.g., to count all occurrences of interest, for trend detection, etc.

10.3.1.2 Editor for CEP patterns

Component/Tool description

Component/Tool/Method/Framework/Service Name

Editor for CEP Patterns

Short Description – incl. Purpose

The editor provides a graphical user interface to create, modify and deploy complex event processing (CEP) patterns. Creating CEP patterns graphically enables domain experts to model their knowledge in such an environment, without requiring additional skills, such as query languages. This editor will replace an Android application, which was limited to Android devices. The new editor will be web based and thus available for a broader range of devices. Additionally, it will be easier to extend and more lightweight.

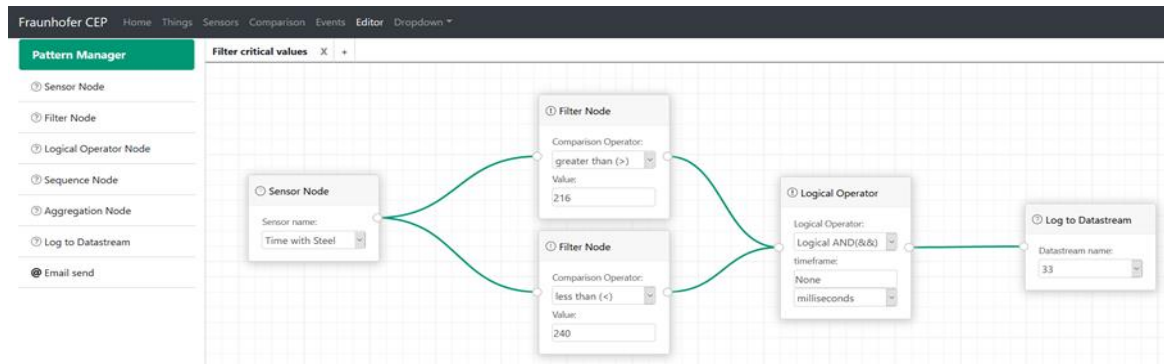
Progress since last milestone

The web-based editor will replace the Android application. During the reporting period the editor was designed and an initial implementation was done. This is still work in progress and will be further developed during the next project phase.

The important extension will be to extend the data sources to be considered. Currently, only the sensor nodes are considered. However, in order to be able to combine the results of ML methods with CEP, there is a need to consider the results of the previous steps in a pipeline (e.g. the output of an ML method) as a data source. This would mean, that the list of sensor nodes will also include the dynamic entities that exist only when a pipeline is active.

Examples of usage / illustrations

Sample Event Pattern for Variable 'Time with steel':



Variable		Q3	Q90
Average Electrical Consumption[kWh]	< 9090	9090 - 10546	> 10546
Average S after vacuum [%]	< 0,006	0,006 - 0,009	> 0,009
Average Time with steel [min]	< 216	216 - 240	> 240
Average Cal Tapping + Sec Met[kg]	< 2021	2021 - 2350	> 2350

Interfaces (in/out) – system/user

There is a GUI to develop and manage a CEP pattern.

The output is a Siddhi-query that can be deployed and executed to any Siddhi engine.

Subordinates and platform dependencies

Web based

Licenses, etc. (free for use in the project)

None - (Editor is based on drawflow (MIT License))

TRL for overall component/tool and any parts/subordinates
TR4
References – incl. web etc.
Drawflow Github
To be considered in particular for the following COGNITWIN pilots
The editor will be used in all use cases where the CEP patterns will be declaratively modelled and not hard-coded. It could be also used to manage (e.g. modify or delete) the patterns.

10.3.2 Video and image sensor data & processing

10.3.2.1 Honir

Component/Tool description
Component/Tool/Method/Framework/Service Name
FPGA Compute Platform : machine learning inference (codename : Honir)
Defined in Task
Task 4.4: Realtime sensor/data processing
Short Description – incl. Purpose
<p>This tool allows to perform machine learning / deep learning algorithm inference in real time on images data source.</p> <p>This tool is a piece of software design that have to be loaded on a board containing an FPGA (Xilinx VU9P for example) and a QSFP+ network interface.</p> <p>Honir is an IP core responsible of performing machine learning inference. Thanks to a loaded quantized machine learning model converted with keras2tool (produced in task 5.3), this module consumes images coming from Lodur tool (produced in task 4.4 too) and produces the tensor results in a stream enhanced with metadata.</p> <p>The advantages of this tool are:</p> <ul style="list-style-type: none"> Real time computing thanks to FPGA platform that allows high parallelism computing Performance / Watt better than usual machine learning platform (CPUs, GPUs) Industrial components thanks to FPGA and electronic boards build for constraints environments
Progress since last milestone
<p>Machine learning inference engine : Scortex implemented the common following deep neural networks layers in VHDL :</p> <ul style="list-style-type: none"> - Convolution - Convolution 2D transpose - Batch Normalization - Add - Merge - Split

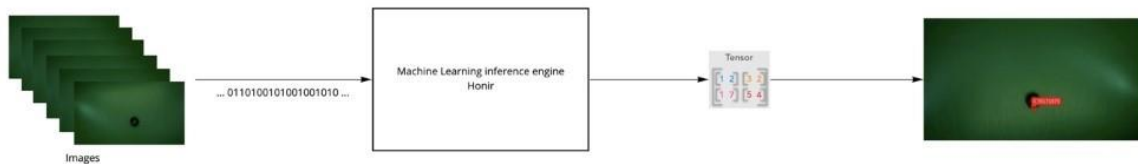
- ReLU activation

Those VHDL blocks can be assembled in modular ways to allow the build of different machine learning topology and then serve different purposes.

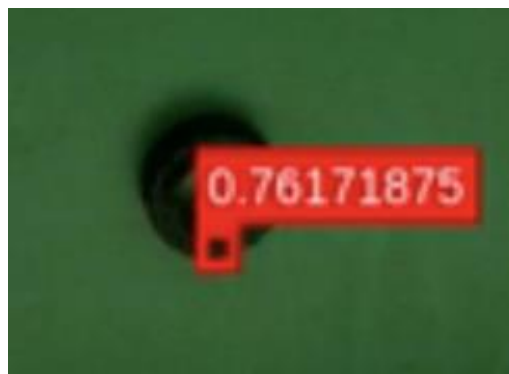
Examples of usage / illustrations

A machine learning model was generated with the VHDL implementation of the layers. This machine learning model correspond to a demonstration that is used for Scortex demonstration purposes that allows defect detection on parts fundable in the market, like bricks parts, electrical switch parts and some handles.

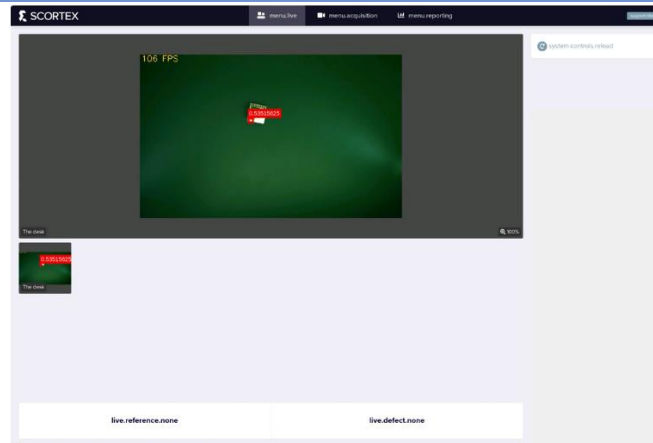
This topology allows the inference of up to 100 images per seconds with a resolution of 1920x1200. That represent a huge amount of compute that is done in an efficient way with less than 10 milliseconds of latency



Images are provided in a convenient format by the Lodur tool (“frame grabber”, produced in task 4.4). Inference is ran on images by Honir and the results produced allow to draw a red box around the defect and also provide a confidence of detection in the source image. The following image shows a zoom on the part and the localization of the defect.



To ease the use of Honir, we included it in our demonstrator to allow live stream of images and live detections.



This image represents the Scortex interface that shows a part moving under the camera with a Machine learning detection at 100 FPS.

Interfaces (in/out) – system/user

At a system level:

IN : video stream (provided by Lodur)

OUT : prediction matrix / tensor

At a user level:

IN : video stream

OUT : results of machine learning inference in a matrix or shown in live on images

Subordinates and platform dependencies

This module takes input images that are sent by the frame grabber Lodur (see dedicated component).

This module is configured based on a keras2RTL tool conversion. Indeed, the user first design a keras model, then use the keras2RTLconvertor, and then can use Honir for inference with this model.

Licenses, etc.

Honir is in development and remains the property of Scortex. For now, it will be used and integrated by Scortex exclusively.

TRL for overall component/tool and any parts/subordinates

TRL5

References – incl. web etc.

- <https://hadrienj.github.io/posts/Deep-Learning-Book-Series-2.1-Scalars-Vectors-Matrices-and-Tensors/>
- https://determined.ai/product/?gclid=Cj0KCQiAmfmABhCHARIsACwPRABr1O5Ob3NjcgABjrEkczdy7HI_3BevgmIDJqYAnRj9dTYR75nDWJcaAiZQEALw_wcB
- <https://www.xilinx.com/products/silicon-devices/fpga/virtex-ultrascale-plus.html>

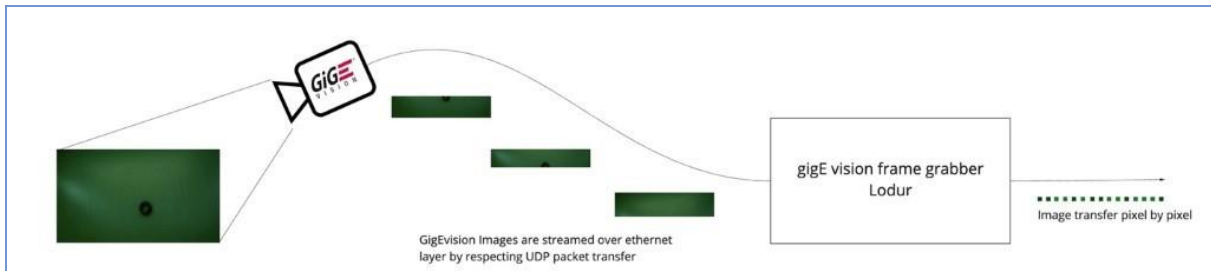
To be considered in particular for the following COGNITWIN pilots

The platform will be considered for optimization of processing speeds for image analytics – it will be benchmarked related to the image analytics involved in the Saarstahl pilot case.

But it could be extended to any other pilots running deep learning on images

10.3.2.2 Lodur

Component/Tool description
Component/Tool/Method/Framework/Service Name
Lodur. FPGA Compute Platform: Frame grabber
Short Description – incl. Purpose
<p>In summary: Lodur is a frame grabber. It is responsible for connecting a camera to an FPGA platform (Honir).</p> <p>This tool is a piece of software design that has to be loaded on a board containing an FPGA (Xilinx VU9P for example) and a QSFP+ network interface.</p> <p>Lodur is an IP core responsible of grabbing images in gigEvision standard usually supported by industrial cameras. It is also responsible to send the received stream into another stream understandable by the machine learning inference engine Honir (done in task 4.4 too).</p> <p>Lodur performs data health checks while receiving the images such as ensuring that the image is complete, or that the format is respected. It can also perform a pre-processing step on the image if required, such as pixel format conversion from YUV to RGB.</p> <p>The advantages of this tool are:</p> <ul style="list-style-type: none"> - Support of gigEvision frame grabbing on FPGA - Support pixel format conversion YUV to RGB - Possibility to support 10gigEvision protocol (10x gigEvision) in the next versions - Possibility to support multi camera stream inputs in the next versions - Smooth integration with Honir
Progress since last milestone
<p>GigEvision (GVSP) image grabbing progress: Scortex implemented a first version that allows the reception of images coming from an industrial camera at the maximum speed of 1gbps (limitation of the camera ethernet interface) and production of the stream computable by the machine learning inference engine. Scortex started a feasibility study on the support of multiple camera inputs and 10gigEvision support.</p> <p>This tool was tested with real cameras as well as with computer simulating a camera. It allows the grab of up to 100 images per seconds with a resolution of 1920x1200. This represent a stream at ~ 1.8 gbps.</p>
Examples of usage / illustrations



Above: An industrial camera takes picture and send it over the gigEvision link. The gigEvision stream rely on UDP packet transfer. It means that the image is cut in a number of packets (sub part of the image) that are sent in sequence to Lodur.

Lodur receive the package, perform health checks on the image, perform pre-processing if activated and cut the image in pixels and then send it to the next module: Honir (inference engine)

Lodur is able to receive images from a camera at maximum camera streaming speed: 1gpbs. We tested it with a computer simulating the camera with a higher throughput and it is able to support up to 1.8 gpbs. This means it is possible to support more powerful cameras that stream at more than 1 gpbs.

Interfaces (in/out) – system/user

At a system level:
 IN: gigEvision video stream
 OUT: pixel stream

Subordinates and platform dependencies

This module can work in standalone. It is, however, necessary for Honir (inference engine) to work properly.

Licenses, etc.

In development, remains the property of Scortex. Will be used by Scortex exclusively

TRL for overall component/tool and any parts/subordinates

TRL5

References – incl. web etc.

- <https://www.visiononline.org/vision-standards-details.cfm?type=5>

To be considered in particular for the following COGNITWIN pilots

The Honir platform will be considered as a way to run the tracking system for the Saerstahl use case. In which case, Lodur will be used to maintain the best FPS (frame per second) performances possible.
 But it could be extended to any other pilots running deep learning on images.

10.3.3 Data preprocessing

Component/Tool description	
Component/Tool/Method/Framework/Service Name	
Outlier detection tool (OD-tool)	
Short Description – incl. Purpose	
<p>The FouMon component collects tools for solving the fouling monitoring problem. It includes the ODtool application independent tool for outlier detection and data quality improvement, to support data-driven modeling of system dynamics</p> <p>Creation of a purely data-driven model requires good quality data. Plant measurements tend to have occasional errors that must be taken into consideration for the self-learning digital twin to be robust. Therefore, in addition to standard measurement verification by detecting sensor faults (1) and frozen values etc. (2), an additional outlier detection (3) method is implemented for this application. Identifying the nontrivial outliers from a batch of multidimensional measurement vectors supports the process of finding a valid digital twin model, thus enhancing the cognitive capabilities of the approach. The tool is available for download.</p>	
Progress since last milestone	
The work has been conducted in the period M30-M42.	
Examples of usage / illustrations	
<p>Figure 82: The basic principle of ellipsoidal peeling. Removal of few outliers can improve the performance of identification significantly</p>	
Interfaces (in/out) – system/user	
<p>The physical model and measurements are set up in the Matlab m-files. Input data (measurements) are provided as numerical vectors. Interactive tuning is enabled by Matlab interface/graphics. Estimation outcomes are provided as numerical vectors.</p>	
Subordinates and platform dependencies	

The tool is implemented using Matlab language (m-files). Matlab from the Mathworks is required (OD-tool/FouMon has been tested on Matlab 2020b).

Matlab (2020b) is available on all major operating systems, including Windows 7, Ubuntu 16, Debian 9, MacOS 10 and newer. No particular Matlab Toolboxes are required. Open software Octave is known to be able to interpret m-files, but FUSE-codes have not been tested with Octave.

Licenses, etc. (free for use in the project)

ODtool is free for use (contact Markus.Neuvonen@oulu.fi)

TRL for overall component/tool and any parts/subordinates

TRL4 – a Matlab algorithm is implemented, the approach has been tested using real plant data from the Sumitomo SHI FW pilot problem.

References – incl. web etc.

COGNITWIN Toolbox portal.

Used in the following COGNITWIN pilots and beyond

Approach is general-purpose. It has been developed and tested in view of the needs for solving the Sumitomo SHI FW pilot problem, and many similar cases in process monitoring and control.