



# COGNITWIN

Cognitive plants through proactive self-learning hybrid digital twins

DT-SPIRE-06-2019 (870130)

## Deliverable Report

<b>Deliverable ID</b>	D5.2	<b>Version</b>	1.0
<b>Deliverable name</b>	Initial Hybrid AI and Cognitive Twin Toolbox		
<b>Lead beneficiary</b>	DFKI		
<b>Editor</b>	Tim Dahmen (DFKI)		
<b>Contributors</b>	Enso Ikonen(UOULU), Istvan Selek (UOULU), Signe Mo (SINTEF), Nenad Stojanovic (NST), Tim Dahmen (DFKI), Pierre Gutierrez (Scortex), Sailesh Abburu (SINTEF), Michael Jacoby (Fraunhofer), Ljiljana Stojanovic (Fraunhofer), Jan Gunnar Dyrset (CYB), Bjørn Tore Løvfall (SINTEF), Aslak Einbu (SINTEF), Stein Tore Johansen (SINTEF), Özlem Albayrak (TEKNOPAR), Perin Ünal (TEKNOPAR)		
<b>Report review</b>	Sudi Jawahery (Cybernetica), Are Dyrøy (Hydro)		
<b>Due date</b>	28.02.2021		
<b>Date of final version</b>	28.02.2021		
<b>Dissemination level</b>	PU		
<b>Document approval</b>	Frode Brakstad	28.02.2021	



The COGNITWIN project has received funding from the European Union's Horizon 2020 research and innovation program 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.

## 1 Executive Summary

This D5.2 report is a report on the technologies and methods that are being developed in the "COGNITWIN Hybrid and Cognitive Twin Toolbox" and which will be applied in the developments for the COGNITWIN industrial pilots. The information provided in the report will further be useful for aligning the concepts and available tools among the COGNITWIN partners but should also give external readers ideas about new Industry 4.0 possibilities.

The COGNITWIN projects aims toward supporting the digitalization of the European heavy industries. A main ambition of the COGNITWIN project is to develop cognitive digital twins that can support a significant improvement in industrial operation. In order to do so, COGNITWIN will work with combining data, physics-based models, machine learning (ML), and Artificial Intelligence (AI) in the best possible manner in order to solve the industrial challenges. Cognition is introduced into the models through self-learning and AI. In COGNITWIN WP5, where this report belongs, the aim is to identify which ML/AI methods are suited for such problems and extend and/or develop new algorithms to further improve performances of the control systems. By developing a Cognitive Twin Toolbox, comprising methods to analyse data, exploit the information from physics-based models, combine information from data and numerical models, and demonstrating applications to process control, this can be applied more generally to support many different process industries. A Cognitive Twin Toolbox is currently being built out from the needs of 6 different industrial pilots, all with their specific and different challenges.

In this report we discuss an intermediate state of the progress on technologies and methods that are being developed for the Hybrid AI/Analytics and Cognitive Toolbox, and which we already started to apply to the pilots. A specifically important part of the report is thus "reflections on the pilots", where technological insights and findings in ML/AI and software architecture are taken from the first applications of the technology to the pilots and should guide development for the remainder of the project.

As technical details below some level are best presented on a per-component-basis, our presentation on the individual tasks focusses on more abstract insights. The bulk of technical details and development status of the individual toolbox components is additionally reported as Appendix 1 and referred to frequently in this document.

Separate reports on the industrial pilots (D1.2, D2.2, D3.2), the "Platform, Sensor, and Data Interoperability Toolbox" (D4.2), and the Key Performance Indicators (D6.2) as well as the Data Management Plan (D8.2) are issued together with this report, giving a more complete overview on the COGNITWIN challenges and Toolbox.

## Table of Contents

1	Executive Summary .....	2
2	Introduction to COGNITWIN Initial Hybrid AI and Cognitive Twin Toolbox .....	7
2.1	Scope and purpose .....	7
2.2	Structure of the deliverable .....	7
3	High-level overview .....	8
4	Plant Digital Twins with ML/AI .....	8
4.1	Objectives, challenges, and components .....	9
4.2	Detailed description of activities performed .....	10
4.2.1	Fuel Characteristic Estimator .....	10
4.2.2	Teknopar Machine Learning Library.....	11
4.2.3	Predicting the Slag Generated in the Furnace and Measured After Tapping .....	12
4.2.4	Estimation Technique for Parameters of the First-Principles Models .....	12
4.3	Progress beyond State of the Art or State of the Practice .....	13
4.4	Summary of the key achievements .....	13
4.5	Next steps.....	13
4.6	Demonstrator Scenario/Description/Video .....	14
5	Multi-variate Sensor analytics with Deep Learning .....	14
5.1	Objectives, challenges and components.....	14
5.2	Detailed description of the activities performed .....	14
5.2.1	Neuroscope & Aerial Photogrammetric.....	14
5.2.2	LSTM deep learning algorithm .....	14
5.3	Progress beyond State of the Art or State of the Practice .....	14
5.4	Summary of the key achievements .....	15
5.5	Next steps.....	15
5.6	Demonstrator Scenario/Description/Video .....	15
6	Deep Learning Performance.....	15
6.1	Objectives, challenges and components.....	15
6.2	Detailed description of the activities performed .....	16
6.3	Progress beyond State of the Art or State of the Practice .....	17
6.4	Summary of the key achievements .....	17
6.5	Next steps.....	17
6.6	Demonstrator Scenario/Description/Video .....	18

7	Hybrid Digital Twins.....	18
7.1	Objectives, challenges and components.....	18
7.2	Detailed description of the activities performed .....	19
7.2.1	Building Physics-based Models .....	19
7.2.2	Bedrock Toolbox.....	19
7.2.3	Pragmatism in physics-based modelling .....	21
7.2.4	FUSE tool .....	22
7.2.5	Hybrid model designer for StreamPipes .....	22
7.2.6	TMat-SynDat.....	23
7.2.7	Hydraulic/Motor/Gearbox Models.....	23
7.3	Progress beyond State of the Art or State of the Practice .....	24
7.4	SIDENOR Pilot.....	24
7.5	Summary of the key achievements .....	25
7.6	Next steps.....	25
7.7	Demonstrator Scenario/Description/Video .....	25
8	Cognitive Digital Twins .....	25
8.1	Objectives, challenges and components.....	25
8.2	Detailed description of the activities performed .....	29
8.2.1	Cognitive CENIT .....	29
8.2.2	State Estimation as Part of Plant Monitoring.....	30
8.2.3	TMat-PdM Predictive Maintenance .....	30
8.2.4	Cognition using StreamPipes Toolbox.....	31
8.3	Progress beyond State of the Art or State of the Practice .....	32
8.4	Summary of the key achievements .....	33
8.5	Next steps.....	33
9	Reflection on the pilots .....	33
9.1	HYDRO Pilot.....	33
9.2	ELKEM Pilot.....	34
9.3	SUMITOMO Pilot .....	34
9.4	NOKSEL Pilot.....	35
9.5	SAARSTAHL Pilot.....	35
10	References.....	36
11	Appendix 1. Toolbox components.....	37

## Table of figures

Figure 1. High-level overview of the COGNITWIN Toolbox.....	8
Figure 2. The PMFIR approach. ....	10
Figure 4. Tech Blog entry of Scortex.....	17
Figure 5 . Data workflow between physical process plant and compute. ....	20
Figure 6. Integration of two data driven models. ....	22
Figure 7. The role of Knowledge in the Cognitive Twin.....	26
Figure 8. Cognition extending Hybrid Twin solution. ....	26
Figure 9. Cognitive CENIT. ....	29
Figure 11 Cognition using StreamPipes Toolbox.....	31
Figure 12. Complex Event Processing (CEP). ....	31

## Acronyms

CEP	Complex Event Processing
CFB	Circulating Fluidized Bed
CFD	Computational Fluid Dynamics
CSTR	Continuous Stirred-Tank Reactor
DCS	Distributed Control System
GTC	Gas Treatment Center
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
SWP	Spiral Weld Pipes
TMat PdM	MATLAB Predictive Maintenance for Electro Mechanical Components
TMLL	Teknopar Machine Learning Library
UKF	Unscented Kalman filter
WER	Word Error Rate

Other acronyms are found at <https://www.allacronyms.com/data/abbreviations>.

## 2 Introduction to COGNITWIN Initial Hybrid AI and Cognitive Twin Toolbox

### 2.1 Scope and purpose

A purpose of the COGNITWIN Initial Hybrid AI and Cognitive Twin Toolbox is to provide the technical foundation for the realization of hybrid AI and cognitive twins in the process industry. This technical foundation takes the form of software components, methods, and processes. An important consideration is that while hybrid AI and cognitive twin concepts are not entirely unique to the process industry, certain aspects *are* specific to the industry. The toolbox developed in this task therefore comprises: (1) custom software components that are developed, extended, or adapted for the specific needs of the Use-Cases in the process industry, (2) existing components that are technically integrated to facilitate interoperability along workflows specific to the Use-Cases, and (3) workflows that are realized entirely by existing software, but which is combined and used in patterns and ways specific to the Use-Cases in the process industry. The initial state of the toolbox is therefore the form of individual, mostly unrelated software tools and processes which are potentially applicable to the Use-Cases in the process industry. During the course of the project, these unrelated components will continually be adapted, integrated, and combined to support the specific needs of the Use-Cases. At the end of the process, the toolbox will provide a technical foundation capable to support the requirements of the Use-Cases in known and tested usage patterns, with specifically adapted software components that are interoperable where required. The current deliverable shows an intermediate progress towards that goal.

### 2.2 Structure of the deliverable

This deliverable is structured as follows. We will start by giving a **high-level overview** of the toolbox in its entirety, specifically design decisions and the conceptual architecture in the intended, *final state* of the toolbox. Going back from the final state, we will show how we intend the toolbox to evolve from its initial state to the final state and where we currently (as of M18) are in this process.

The second part will contain status **descriptions on a task-level**, focusing on progress, challenges, and deviations from original planning. These descriptions will still be kept relatively short, as the bulk of the technical documentation is provided on a component level.

The third and most detailed part of this deliverable consists of detailed descriptions of the individual toolbox components – including detailed descriptions of progress made during the project so far – examples of how to use them, and their relation to the Use-Cases.

### 3 High-level overview

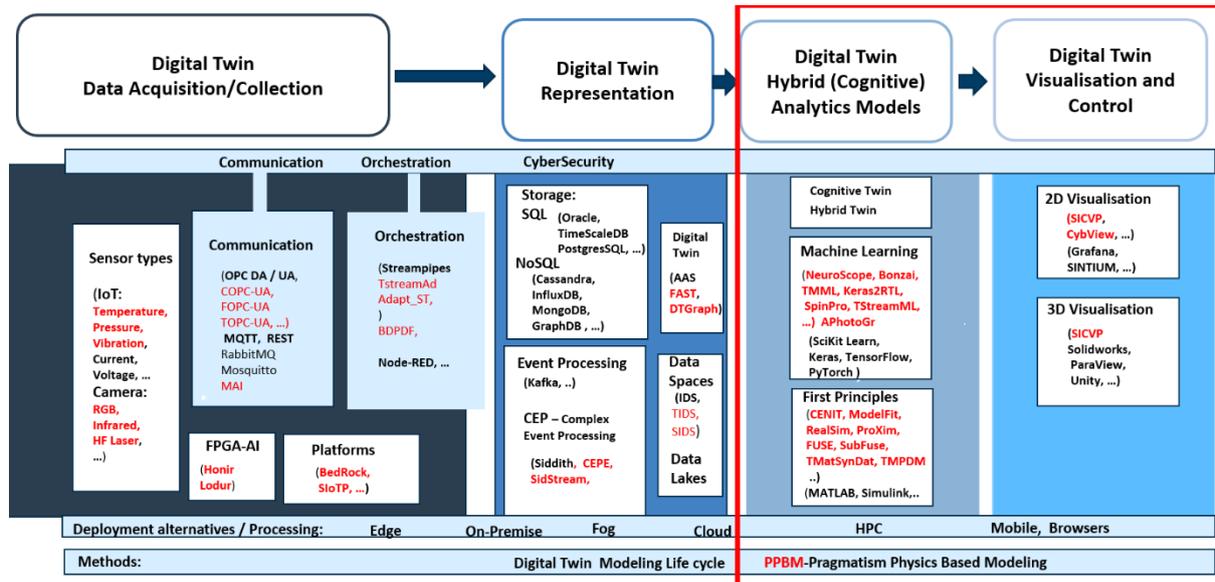


Figure 1. High-level overview of the COGNITWIN Toolbox.

The goal of WP5 is to provide methods and tools for Hybrid AI and Cognitive Twins. Figure 1 shows the structure of the COGNITWIN Toolbox as reflected in the COGNITWIN Toolbox Portal, available at <https://cognitwin.github.io/toolbox/>. This report, D5.2 focuses on the areas in the red box on Digital Twin Analytics Models and Visualisation, while the report D4.2 introduces the areas in the Toolbox and the areas of to the left in the figure on Digital Twin Data Acquisition and Representation.

In the context of the COGNITWIN project, the Hybrid AI and Cognitive Twins should not be developed from scratch, but rather the already existing components/systems should be considered. Additionally, the new services to model the behaviour of a Hybrid AI or Cognitive Twins will be developed or even 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 WP5 deliverable (D5.1) we have already identified a list of the components which will be reused or extended.

In this midterm report, we are reporting progress on the extension and development of components, and reflecting on additional requirements that emerge from applying the toolbox components to the pilots.

### 4 Plant Digital Twins with ML/AI

A **plant digital twin** is a digital replica of the real plant. A modern digital twin gets its essential behavioural contents from a **plant model**, perhaps linking simulations with on-line data and various services such as data visualization and analytics, what-if analyses, or 3D plant animations. Currently, the huge potential of the digital twin technology is reflected in a better design of an asset, based on extensive simulations in various conditions. A model can be used for improving system dynamic behaviour, designing a model-based controller, or a state-estimator. Model-based fault detection and isolation, operator training simulators, or plant analysis as well as integrated plant and control design are other examples of **applications** requiring a proper dynamic process model.

The models can be very detailed and enable powerful simulations. In the heavy process industry, the models are often built based on **physical** considerations, tuned and complemented in various way by **data** from experimental tests or normal operation. **Task 5.1 aims** to promote use of ML/AI methods suited for such problems and extend and/or develop new algorithms to further improve the performance of the control, monitoring, and maintenance systems.

#### 4.1 Objectives, challenges, and components

Task 5.1 (T5.1) examines the role of simulation models and data to effectively model real-world assets. **T5.1 identifies, selects, and extends/develops further ML/AI methods particularly suited to challenges of the process industry.** Eventually, it will provide rules-of-thumb for selection and parameterization of ML methods, as justified and illustrated by pilot case experimentation.

The **state-of-the-art** of the T5.1 methodology was described in the COGNITWIN WP5 M6 deliverable (D5.1) Sec. 4.2. The methods of analytics are developing in a fast pace, and much of the required infrastructure for sensor, data, and automation exists. However, due to the nature of the heavy process industry, the changes in the industrial practice can be slow. Discussion of these topics is emerging in the scientific community, including recent articles such as:

[QIN+19] S. Joe Qin & Leo H. Chiang (2019) Advances and opportunities in machine learning for process data analytics. *Computers and Chemical Engineering* **126**, 465–473.

[BIK+20] Timur Bismukhametov & Johannes Jäschke (2020) Combining machine learning and process engineering physics towards enhanced accuracy and explainability of data-driven models. *Computers and Chemical Engineering* **138**, 1–27.

The heart of most ML/AI approaches is in **learning** from data. This poses serious problems in the heavy process industry, which is typically characterized by slow processes, large size and complexity of plant installations, and heavy safety requirements, which prohibit approaches based heavily on exploratory data generation and testing. Plants are typically operated in production mode, which leads to the operation data probably not being rich in information. Therefore, the role of physical models is particularly pronounced. A significant challenge is to **assess the feasibility** of various proposed approaches in the process engineering context.

The case pilot problems were briefly described in D5.1 Sec. 4.4, more extensive descriptions were given in D5.1 Sec. 2 as well as the pilot work package deliverables (D1.1, D2.1 and D3.1).

T5.1 participates in providing components to the **COGNITWIN toolbox**, focusing on the full exploitation of physical plant models and ML/AI approaches in data-analytics and data-driven model tuning. The work in COGNITWIN associated with T5.1 largely originates from the development of solutions to pilot case problems. The role of T5.1 is to focus on the exploitation of physical modelling aspects on one side, and on data-driven techniques – such as ML/AI – on the other. The hybridization aspects are partially considered in T5.4 on hybrid digital twins. The work in T5.1 is pilot-driven, in that the methods are examined in solving the COGNITWIN pilot case problems, and promising solutions are generalized for extended use via the T5.1, to be cross-utilized in other COGNITWIN pilot cases or other process industry applications.

## 4.2 Detailed description of activities performed

### 4.2.1 Fuel Characteristic Estimator

The UOULU (University of Oulu, Intelligent Machines and Systems research unit) has focused on the Sumitomo **power plant pilot case** problem (WP3). This case considers the monitoring and control/maintenance of heat exchange surface fouling and slugging. As a component for solving the problem, a fuel characteristic estimator has been developed.

The estimator is based on exploitation of a physical model for the circulating fluidized bed (CFB) boiler furnace. This hotloop model consists of more than 1k states, and well over 100 I/O. Based on a Sumitomo in-house model, the **simulator** was developed further for providing one-step-ahead simulations from arbitrary initial states. The simulator parameters were adjusted for the local pilot plant conditions.

A novel approach for **tuning** physical models was proposed (Ikonen and Selek, 2020):

[IKO+20] Enso Ikonen & Istvan Selek (2020) Calibration of physical models with process data using FIR filtering. Australian and New Zealand Control Conference, Gold Coast 26-27 Nov 2020, 143–148.

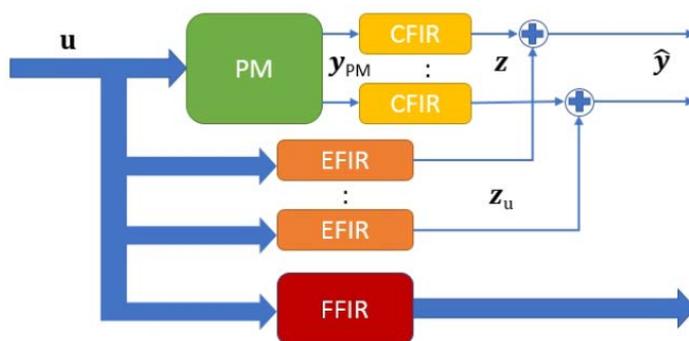


Figure 2. The PMFIR approach.

The main suggestion is to use dynamic tuning elements at the outputs of a physical model (PM), which enables both the application of robust parameter estimation ML techniques and availability of the original physical model results (Figure 2).

Ikonen & Selek (2020) discuss the PMFIR via an application to a nonlinear simulated continuous stirred-tank reactor (CSTR). The incorrect (worst-case) models were successfully calibrated using nominal plant data, even a simple steady-state physical model could be successfully applied. The approach was compared with alternative data-driven methods, including static affine correction at the physical model output, and data-driven modelling of the dynamic process. The proposed approach provided improved performance and was shown to be much less sensitive to the amount of data than corresponding data-driven approaches. Also, the application of the calibrated model in physical model-based state estimation using unscented Kalman filter (UKF) was successfully demonstrated. UKF belongs to the class of ML/AI-inspired population-based nonlinear state estimation techniques.

The principle of tuning physical model outputs was applied for the Sumitomo case pilot problem. With the given model, the experiences suggested a static affine mapping as a feasible and robust tuning. The hotloop physical model was then further applied in fuel characteristics state estimation (FUSE-tool). The main challenge in using a physical model is the **computation time** needed in evaluation of

the model equations. At the moment, with a handful of unknown states/inputs, the simulations take ~1 second/20 seconds of real-time simulation. This presents challenges in the formulation of a state estimation problem, and reducing the number of uncertain states was found to be a feasible formulation. The current tool can be run in real time. The approach was tested on real plant data, as illustrated in the Appendix. See **FUSE** component/tool template in the Appendix.

The FUSE-tool and the examined method/procedure is expected to be applicable for other input/parameter/state estimation problems as well, provided that a suitable plant model is available. A **generalized** FUSE tool (based on the pilot problem solution) is provided for the COGNITWIN toolbox, providing a flexible implementation for the UKF state estimation using a physical plant model. The development will continue in 2021.

The **SubFUSE** tool has been developed in view of the Sumitomo pilot fuel characterization problem. The tool focuses on data-driven process identification using subspace techniques. The estimated linear model can then be used in state estimation, e.g., Kalman filtering. The approach – identification and state estimation – has been tested in simulations, but not with real data as the available data has not been sufficient for subspace modelling. The tool will not be provided for COGNITWIN toolbox until validated in the real pilot.

The SubFUSE tool is described in the component/tool template in Appendix.

The FUSE tool **fuses process physical model with on-line measurement data** from the plant. The bayesian state estimation approach uses the deterministic non-linear plant model to predict the plant behaviour one sampling time ahead in time. The states with uncertainties to be examined are estimated by providing initial, state, and measurement noise characteristics (covariances). An optimal correction using the innovation signal (difference between prediction and measurement) is provided by the algorithm. The physical model tuning method, including an application of the model in UKF state estimation in a benchmark CSTR simulation, was reported in Ikonen & Selek (2020).

Among the bayesian state estimation techniques (extended Kalman filter, **unscented Kalman filter** (UKF), particle filter, grid filter, ... ), the UKF was found to be most appropriate for the Sumitomo pilot case as i) it can handle nonlinear plant models, ii) it does not require the Jacobians to be explicitly evaluated, and iii) the amount of required model predictions / computational load can be kept feasible. It can be expected that similar conditions are found in other heavy process industry applications. A generalized **FUSE** tool (based on the pilot problem solution) is provided for the COGNITWIN toolbox, the development will continue in 2021.

The FUSE tool is described in the component/tool template in Appendix. The application for solving pilot case problem is described in more detail in D3.2.

#### 4.2.2 Teknopar Machine Learning Library

TEKNOPAR has focused on the NOKSEL pilot case of a metal sheet roller system of spiral welded pipes (SWP). The goal is to monitor the production process machine malfunctioning and enable predictive maintenance.

STEEL 4.0 Teknopar Machine Learning Library (TMLL) is being continuously developed for Task 5.1 and T5.2. A library of **ML/DL** (machine learning/ deep learning) algorithms has been created and tested. In TMLL, different machine learning algorithms are applied through the incremental PCA stage to detect anomalies. Prediction results are produced using different machine learning libraries. Both Spark MLLib

and Keras were used. Spark MLlib is produced entirely by Spark, and uses Spark's engine optimized for largescale data processing. Keras library, that uses TensorFlow, is used for deep learning purposes. The Long Short-Term Memory (LSTM) algorithm of this library is utilized. This open-source neural network library makes it simpler to work with artificial neural networks through its user interface facilities and modular structure. The Scikit-Learn library is another open-source machine learning library that contains several algorithms for regression, classification, and clustering. We used algorithms like RF, GBT, LSTM, SVM, KNN, and multi-layer perceptron (MLP) from the Scikit-Learn library for data modelling and prediction. The trained algorithms have been compared in predictive maintenance.

The TMLL and ICPV tools are described in the component/tool template in Appendix.

#### **4.2.3 Predicting the Slag Generated in the Furnace and Measured After Tapping**

SINTEF Digital used machine learning models for predicting the slag generated in the furnace and measured after tapping, in the Elkem pilot in WP1. The input time series data are measurements from the furnace, materials added, and post tap hole measurements from previous tappings. Several machine learning models were evaluated and compared, including support vector machines, a k-nearest neighbour algorithm, and ensembles of decision trees. The results from testing these methods have proved useful in getting a better understanding of the underlying process, e.g. with respect to parameter importance. Still, neural network-based models outperformed the other models. The best results have been obtained from models which take sequences of data as input. A model consisting of one convolution layer followed by three LSTM layers have been trained on filtered data to predict the trend in slag measurements. Many sequence models have been tested, but all give similar or worse results compared to LSTM models. Even if the current models show some predictive abilities, their results are still quite unreliable. This makes it difficult to tune models and compare similar results. Further improvements of the models rely on a cooperative and iterative process; by analyzing and discussing the results of the current best models, more insight into the process might help in the tuning of the models.

The status of BedRock tool is given in Chapter "Hybrid Digital Twins", and is explained in the Appendix.

#### **4.2.4 Estimation Technique for Parameters of the First-Principles Models**

Cybernetica is planning on applying a recursive estimation technique to estimate parameters and states of the first-principles models of the Elkem ferrosilicon refining pilot and the Hydro Gas Treatment Centre (GTC) pilot. The physical models are formulated as a nonlinear state-space model, which typically includes an integration from one sample time to the next. Some of the parameters are quite uncertain and vary with time in a manner that is difficult to model. The interfacial area between slag and metal is one such parameter in the Elkem model, as it varies due to complex fluid dynamics which is unsuitable for modelling in a real-time application. This parameter is important for calculating the rate of the reactions taking place at the slag metal interface and thus affects the composition of the metal. In order to estimate this parameter using the available measurements online, the state vector is augmented with the parameter vector and the model is extended with a data-driven noise model. Applying a recursive estimation algorithm to this extended model, such as a Kalman Filter or a Moving Horizon Estimator, allows us to estimate uncertain parameters online. Great care has to be taken when choosing which parameters to estimate online, and combining the knowledge of which parameters are uncertain from a physical point of view, together with the knowledge of which parameters the model is most sensitive to has proven powerful in previous applications.

### 4.3 Progress beyond State of the Art or State of the Practice

A number of physical models and ML/AI methods have been developed/applied in solving the COGNITWIN pilot case problems. The LSTM recurrent neural network approach has been found applicable in several pilots. The novel methods of model-based state estimation have been considered as very promising approaches in the fusion of physical models with plant data in solving the pilot case problems.

Development and execution of detailed physical models require a specialized software. Typical data storage/communication platforms do not support established ways of representing and solving standard physical model forms, such as ordinary/partial differential equations. Highly sophisticated physical modelling tools do exist, however, such as Matlab (including Simulink and Simscape), CENIT, and others, used in the COGNITWIN. A digital twin requires a means for connecting to external tools. Following the basic principle of WP4 interoperating toolbox, this connection has been developed using e.g. OPC-UA and MQTT services for data transfer.

The significant lesson learned from the applied work is the importance of fusing physical modelling with plant data. This has been experienced in several (if not all) pilots, where it has been observed that the operation data history is of limited range. In the development of plant operation optimization and control, the models need to cover areas not typically visited by the plant (as the optimization is likely to change the operation point). A similar problem appears in maintenance/fault detection and isolation in data-driven modelling of rare events. Therefore, data-driven approaches alone are not sufficient. Digital twin provides a tool for a fusion of data and models. A general approach for physical process model tuning based on plant data has been proposed. In an alternative approach, the physical model has been used to generate complementing data for ML/AI learning.

### 4.4 Summary of the key achievements

A paper on the COGNITWIN T5.1 outcome was presented in a scientific conference in November 2020 and published in the IEEE Xplore:

[IKOI+20] Enso Ikonen & Istvan Selek (2020) Calibration of physical models with process data using FIR filtering. *Australian and New Zealand Control Conference*, Gold Coast 26-27 Nov 2020, 143–148.

The following tools in the COGNITWIN toolbox, were developed:

- FUSE-tool (Fuel state estimation),
- Steel 4.0 TMLL machine learning library,
- Steel 4.0 ICPV industrial control panel and visualization (3D modelling and digital twin),
- Bedrock tool.

These tools/methods have been validated in at least one COGNITWIN pilot and are available for the development of applications in other pilots in the COGNITWIN toolbox. Video demonstrations are available, the tools are described in more detail in the Appendix.

### 4.5 Next steps

T5.1 will continue monitoring and collecting the tools, methods and experiences related to physical modelling and data-driven analytics and ML/AI. As more experience is collected via the pilots, more general conclusions and guidelines can be drawn on ML/AI methods suitable for the process industry in the wealth of approaches available.

## 4.6 Demonstrator Scenario/Description/Video

A video demonstration of the FUSE-tool is available at the COGNITWIN Youtube channel:

<https://www.youtube.com/channel/UCgHunz1V68YGOxaqVkkyN1A>

# 5 Multi-variate Sensor analytics with Deep Learning

## 5.1 Objectives, challenges and components

The possibility to process multi-variate sensor data using Deep Learning will be crucial for the realization of hybrid AI and cognitive twins. While the extraordinary performance of Deep Learning systems to many classification and regression problems has been demonstrated in many contexts, key challenges remain. Their challenges are generally recognized to be: (1) high demand in compute power, (2) high demand in training data, and (3) difficulties in obtaining trustworthy results or even finding good technical explanations in cases when unintended system behaviour occurs. The generally high demand in compute power is less problematic for many applications in the process industry as high-performance hardware is readily available. For situations where latency is critical, we dedicate a specific task (Deep Learning Performance) to finding solutions. This task will therefore focus in the two remaining core problems of Deep Learning: the availability of training data and trustworthiness concerning explainability.

For the availability of training data, we will focus on the generation of training data using parametric models and sensor simulations, specifically on capturing of 3D information using **photogrammetry**. For improved explainability, we will adapt a visual debugger for neural networks called **Neuroscope** for the specific needs of the Use-Cases.

## 5.2 Detailed description of the activities performed

### 5.2.1 Neuroscope & Aerial Photogrammetric

DFKI have improved two key components: a visual debugger for neural networks called Neuroscope and a process for the photogrammetric capturing of plant sites using aerial photogrammetry. A detailed description of the activities can be found in the component descriptions for the toolbox components **photogrammetry** and **Neuroscope**, which are detailed in the Appendix.

Working closely with DFKI, Scortex has prepared an evaluation of the synthetic data processing approach using photogrammetry under high performance conditions using our technology.

### 5.2.2 LSTM deep learning algorithm

Focused on NOKSEL pilot, TEKNOPAR has conducted a LSTM deep learning algorithm. Multiple sources of sensor data (vibration, temperature, pressure, etc.) have been used and DL (deep learning) models have been applied. The Keras library, that uses TensorFlow, is used for deep learning purposes. The LSTM algorithm of this library is utilized. This open-source neural network library makes it simpler to work with artificial neural networks through its user interface facilities and modular structure.

STEEL4.0 TMLL of TEKNOPAR, which is under continuous development, is detailed in the Appendix.

## 5.3 Progress beyond State of the Art or State of the Practice

For Neuroscope, we have built on the published state of the art in the visualization of neural networks by implementing a number of methods in an interactive, useable software. While such methods have

been published, the majority of these methods is designed for, and immediately applicable, to classification (i.e., output of the network is a label per image) networks only. We initially assumed our Use-Case would require semantic segmentation networks (i.e., output of the network is a class label per pixel). We have therefore adapted the methods for applicability to semantic segmentation problems. In the following, we give a list with methods that were implemented in Neuroscope, with an indication whether the methods work for classification (C) or segmentation (S) problems. The following methods were implemented: Activation maps (C,S), Saliency map (C,S), Guided back propagation (C,S), Grad-CAM (C,S), Guided Grad-CAM (C,S), Grad-CAM plus (C), Score map (C,S), Segmented score map SSM (S), Guided segmented score map GSSM (S), Similarity map (S), Fusion score map FSM (S), Confusion Matrix (S, in conjunction with provided ground truth).

## 5.4 Summary of the key achievements

We have published a useable version of the software Neuroscope in version 1.0, available at: <https://github.com/c3di/neuroscope>.

We have prepared a manuscript describing the software. The manuscript has the following coordinates:

[SCO+21] Schorr, C., Godarzi, P., Chen, F., Dahmen, T. (2021) Neuroscope - An explainable AI toolbox for semantic segmentation and classification of deep neural nets, in: Applied Sciences, Special issue on explainable artificial intelligence (accepted for publication).

## 5.5 Next steps

In deeper consideration of the Saarstahl Use-Case, it became apparent that even semantic segmentation (i.e. output of the network is a label of object-class per pixel) will be insufficient for the Use-Case of tracking milled bars, but instead will require either multi-object detection and localization (output of the network is a list of labels with bounding boxes), or instance segmentation networks (i.e. output of the network is a different label for multiple objects of the same class, assigned per pixel). We have already begun to implement support for multi-object detection and localization networks in Neuroscope, so will likely address instance segmentation support during the course of the project.

## 5.6 Demonstrator Scenario/Description/Video

A video demonstrator on both Neuroscope and the Photogrammetry workflow are available on the COGNITWIN YouTube channel:

<https://www.youtube.com/channel/UCgHunz1V68YGOxaqVkkyN1A>.

# 6 Deep Learning Performance

## 6.1 Objectives, challenges and components

The previous tasks “Multi-variate sensors analytics with deep learning” is about the ability to train deep learning algorithms and achieve a good enough accuracy for a task. In order to be able to deploy such models in production, a specific focus has to be set on the pipeline performance in terms of inference time, memory used, compute resources needed, etc. This is especially true when it comes to Industry 4.0, where systems are deployed in the 3D world and have real time. For example, in the field of quality inspection, dozens of images of high resolution may be used to take a decision in real time. This means that standard architectures from the literature designed on the ImageNet dataset

may be too slow to be used in practice. This task is about finding solutions which allow engineers to deploy state of the art deep learning algorithms in practice, in factories.

## 6.2 Detailed description of the activities performed

Scortex has experimented on several ways to solve the deployment issues. Because Scortex is a quality inspection company, it encounters the issue more often than other companies. Indeed, high resolution images are often required to detect very small defects.

Scortex followed several leads related to:

**Network architecture design:** Architectures were specially designed to enable inference on high resolution images (from 1000x1000 pixels up to 2000x2500 pixels) using only one GPU or FPGA. These networks were trained and evaluated on Scortex datasets. The tasks of interest were supervised semantic segmentation / detection as well as anomaly detection.

**Pruning of the network:** Pruning strategies were implemented. Though this method helps lowering disk space and RAM/GPU memory constraints, it does not provide faster inference. We believe this will be the case as long as TensorFlow/Keras does not provide a better sparse tensor support.

**Distillation:** We successfully managed to transfer knowledge from a large network to a smaller one. The performance is not as good as the large model, but better than the performance of the smaller model trained on its own.

**Inference graph optimization:** We investigated “folding” Batch Normalization which provided 30% speed in inference time. Details can be found in Scortex blog:

<https://scortex.io/batch-norm-folding-an-easy-way-to-improve-your-network-speed/>

**Quantization of networks:** This is a necessary step in order to deploy networks on FPGA hardware (see task 4.4). We also work on a Keras2RTL component which enables converting a Keras model into something usable by the FPGA platform (see Honir, task 4.4)

**Efficient inference pipeline:** Scortex is currently working on an end-to-end library called “sensei” to acquire images, apply preprocessing and deep learning networks as well as post processing on top of them. This library supports asynchronous and parallel inference on GPU so that real time capabilities can be achieved in a robust fashion.

Scortex now uses its “bonzai” library to train models and its “sensei” library to deploy such models at its customers. Scortex is ready to scale the methodology to the use case partners who have need for fast deep learning technology.

TMLL is used by TEKNOPAR for comparing the different machine learning models. While setting up a machine learning model, it is difficult to predict which model architecture will provide the best result. The parameters which affect the model architecture are called hyper-parameters. For each machine learning algorithm utilized, hyper-parameter tuning has been performed by first comparing the previously determined success criteria, and then selecting the best result combination by examining the results obtained through testing possible combinations of the hyper-parameters' values in a certain range. For each ML algorithm used, additionally, various parameters - such as precision, recall, F1 score, error detection rate, total training time, total test time, average training time, Type 1 error, Type II error - were calculated and displayed to the user.

The user is offered a voting option to decide on the algorithm to use. The application enables users to select the machine learning model for a given set of data, and then compares the output using graphical elements. For developing and testing purposes, the AML Workshop dataset from Microsoft (AML data set) is used in TMLL module. The Scikit-learn library has been used for Random Forest, Gradient Boosted Tree, MultiLayer Perceptron, Support Vector Machine, and K-Nearest Neighbors. For the LSTM, Keras has been used.

STEEL4.0 TMLL of TEKNOPAR, which is under continuous development, is briefed in Appendix.

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

The work done by Scortex on “Batch Norm Folding” has been published on its blog:

<https://scortex.io/batch-norm-folding-an-easy-way-to-improve-your-network-speed/> (Figure 3).

Figure 3 shows a Tech Blog entry from Scortex. The left side is a thumbnail for a blog post titled "Batch Norm Folding: An easy way to improve your network speed" dated June 30, 2020. The right side displays the mathematical derivation of Batch Normalization folding. The equations are as follows:

$$y_i = \frac{\gamma}{\sqrt{\sigma^2 + \epsilon}} x_i - \frac{\gamma}{\sqrt{\sigma^2 + \epsilon}} \mu$$

$$x_i = W \cdot z_{ci} + b$$

$$y_i = \frac{\gamma}{\sqrt{\sigma^2 + \epsilon}} (W \cdot z_{ci} + b) - \frac{\gamma}{\sqrt{\sigma^2 + \epsilon}} \mu + \beta$$

$$y_i = W' \cdot z_{ci} + b'$$

$$W' = W \frac{\gamma}{\sqrt{\sigma^2 + \epsilon}}$$

$$b' = \frac{\gamma}{\sqrt{\sigma^2 + \epsilon}} (b - \mu) + \beta$$

Figure 3. Tech Blog entry of Scortex.

It is now one of the most read articles on the subject. As of 2021/01/19, it has been read by 1683 different people according to Scortex website statistics. Readers came from recognized industry companies such as: Sony, Nokia, Daimler, AMD, Intel, Thales, Huawei, Zoom, ETH Zurich, Panasonic, etc.

By combining several ideas described above, Scortex managed to deploy a station able to perform a complex inspection of rotating parts. The Scortex box handles the inspection of 3 parts per second which requires inference of 300 (3 x 100) 1280x640 grayscale images per second. To the best of our knowledge, Scortex is the only company able to achieve such performances in a real-life deployment.

### 6.4 Summary of the key achievements

Using Batch Norm Folding, we were able to reduce the inference time of light architectures by 30%.

The work is summed up on Scortex blog:

<https://scortex.io/batch-norm-folding-an-easy-way-to-improve-your-network-speed/>.

We managed to design and quantize efficient architectures. Once deployed on FPGA, we can achieve 120 FPS (frames per second) on 1920x1200 RGB images. Deployed on 2 GPUS, we achieved 300 FPS on 1280x640 grayscale images.

### 6.5 Next steps

In the following months, Scortex plan to continue the FPGA library to make it more versatile and more robust so that it could be easily used internally and externally. This work is mostly related to task 4.4

but does need some inputs from the task 5.3. The automatization of the conversion process of a topology and the weights from a neural network into a hardware designed platform will be done in the next steps (improvement of the Keras2RTL component). This will ease the use to the process and reduce the time needed to transition from a functional neural network design on computer to one running on an FPGA.

The use of very light architectures can be an issue because of the lack of expressiveness and capacity they lead to. This is something that we have observed in practice. We plan to rework our architectures, typically using recent architecture blocks from the literature such as Efficient Net / efficient det MBconv, SE and SK blocks, or more generally the mechanism of attention (since we anticipate some transformers breakthrough computer vision in 2021). Part of the work detailed above is still experimental. Packaging and standardization work will be necessary to make the methods more “production-ready”. Finally, these methods have been proven on internal datasets and now need to be tested and benchmarked on the pilot data, deep learning tasks, and network architectures.

## 6.6 Demonstrator Scenario/Description/Video

Scortex released a video of the first iteration of the FPGA platform on the COGNITWIN YouTube channel:

<https://www.youtube.com/channel/UCgHunz1V68YGOxaqVkkyN1A>

## 7 Hybrid Digital Twins

A Hybrid Digital Twin is an extension to the plant Digital Twin where the physics-based model is combined with data from the real process and one or more data driven models that adapts and corrects the Digital Twin such that it better represents its real-world counterpart.

A definition for a Hybrid Digital Twin as the second layer in a three-layered twin definition was given in:

[ABB+20] Sailesh Abburu et al. (2020) COGNITWIN – Hybrid and Cognitive Digital Twins for the Process Industry. *2020 IEEE International Conference on Engineering, Technology and Innovation*.

“An extension of Digital Twin in which the isolated Digital Twin models are intertwined to recognize, forecast and communicate less optimal (but predictable) behaviour of the physical counterpart well before such behaviour occurs. A Hybrid Digital Twin integrates data from various sources (e.g., sensors, databases, simulations etc.) with the Digital Twin models, and applies AI analytics techniques to achieve higher predictive capabilities, while at the same time optimizing, monitoring, and controlling the behaviour of the physical asset. A Hybrid Digital Twin is typically materialized as a set of interconnected models, achieving symbiosis among the Digital Twin models.”

### 7.1 Objectives, challenges and components

The main goal of Task 5.4 is to contribute to the CogniTwin toolbox with tools which support the creation, use, and exploitation of Hybrid Digital Twins. The tools will originate from the development of twins in the pilots. Certain, suitable elements will be extracted from the pilot twins, generalized, and made available as tools for the other pilots and eventually other processes.

The focus of the task is on the following aspects:

- Enhancement of the digital twin technology by combining physics-based models with data driven models, including AI and machine learning functionality
- Development of soft sensing applications based on digital twins
- Development of advanced, predictive, and self-learning control applications based on digital twins

So far, the main development in the pilot cases has been on the development and tuning of different process models, so hybridization of the models has not yet been addressed in all of them.

## 7.2 Detailed description of the activities performed

### 7.2.1 Building Physics-based Models

Cybernetica's work has been focusing on the Elkem and Hydro pilot cases. A large part of the work effort has gone into building physics-based models for these two processes and to tune the model parameters such that the models replicate the physical processes as correctly as possible. The physics-based models for both pilots have been implemented as extensions to the tools Cybernetica CENIT, Modelfit, and RealSim. The extensions are in the form of application-specific modules and have been implemented in C/ C++ using a pre-made application component template. Cybernetica Modelfit has been used together with logged data from the processes for offline tuning of the model parameters, and the models now represent the processes quite well.

Cybernetica CENIT will be used to run the models online for both the Elkem and Hydro pilots as soon as the required IT infrastructure is in place. The plan is to extend the physical models with data driven models which continuously adapt the twins to the real processes. Extended Kalman Filter or Moving Horizon Estimator will be considered for updating these models.

Soft sensing of unmeasurable variables in the processes will then be possible, and later also extensions for model predictive control will be considered. In the meantime, Cybernetica RealSim has been used as a plant replacement simulator for testing different scenarios with the twins.

In the Hydro pilot, the process model will receive meteorological input data in addition to process measurements. The weather data is fetched from a public API by a specific component developed by SINTEF (described in deliverable D4.2). In the Elkem pilot, it is planned that the process model will receive estimated parameters as inputs from a data-driven model (AI) of the up-stream furnace process section.

The generic part of Cybernetica CENIT has been extended with functionality for evaluation of the quality of both the input signals and its own calculation results. In the case that invalid input or calculation results are detected, the application will send a notification of this via an OPC connection. Thus, the plant operators can be notified, and a proper fallback solution can automatically be activated. This extension forms an important foundation for adding more sophisticated error detection and self-examination algorithms, like the proposed Cognitive CENIT extension, described in the Appendix.

### 7.2.2 Bedrock Toolbox

The Bedrock Toolbox code repository by SINTEF has since the last milestone undergone restructuring to allow for improved configuration and remote 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. 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 while allowing for flexible selection of modules in customized deployments based on the needs. It also enables options for cloud deployment for work with external partners. A deployed central Bedrock Toolbox server allows for collaborative work on joint datasets with various compute modules interacting with a shared database. Figure 4 shows an example of a BedRock deployment.

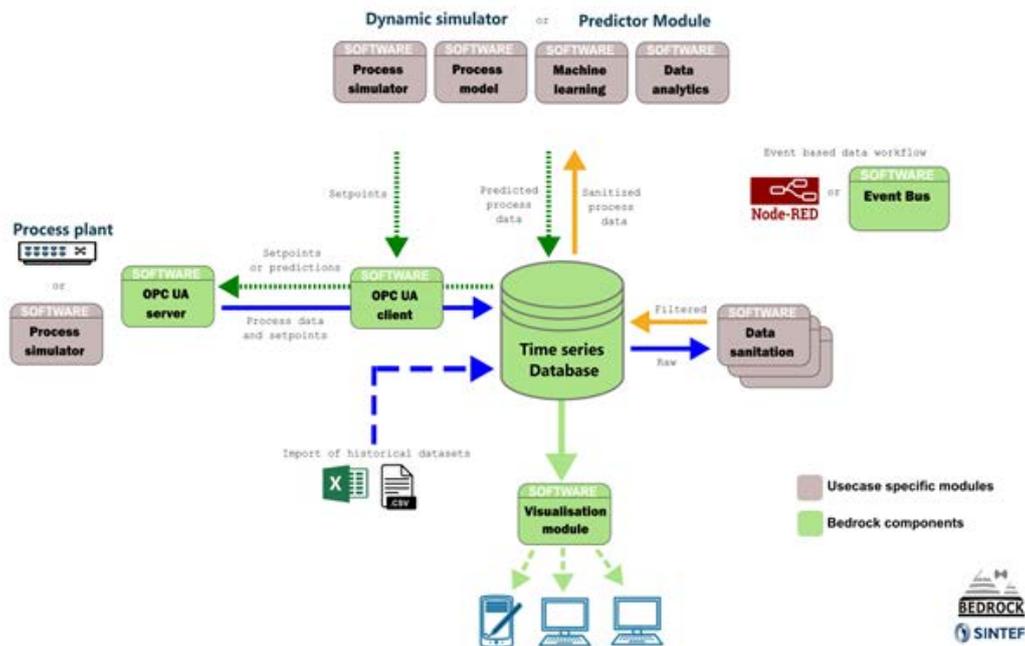


Figure 4. Data workflow between physical process plant and compute.

Application of a central time series database for data workflow on treatment of process data allows for structuring and visualization of data from sensors and models/simulators. Process data inputs to the database can be from real-time collection of process plant time series or from historical datasets. This approach enables easy access and structuring of time series for iterative data analytics and machine learning of raw data, filtered data, or calculated/simulated/predicted datasets. Two-way communication between the physical plant or simulator and the compute modules enables return of setpoints or decision support data to the process control. Advanced process control can be explored by replacing the physical plant with a dynamic simulator when available (software-in-the-loop). The application of OPC UA historian and the establishment of additional OPC server tags for calculated parameters is an option for a direct link between process plant and compute modules, however the application of dedicated databases in the compute framework allows for flexibility and is beneficial in the development phase of digital twin modules.

Data from the Sidenor pilot, currently present as Excel export files from the plant, will be imported and structured in a relational time series database. This will enable easy structuring of the data set for application in hybridization between physics-based models and machine learning. Implementing the developed models in the above framework is coming up.

### 7.2.3 Pragmatism in physics-based modelling

*Pragmatism in physics-based modelling (PPBM)* has been applied and further developed in the Sidenor pilot case. The industrial task is related to the lifetime of a ladle in the steel industry. A ladle is a container for liquid steel, typically filled with 100 – 180 tons of liquid steel. For each use of a ladle the refractory will be eroded and after N uses the ladle must be taken out of use in order to avoid serious accidents (more than 100 tons of extremely hot liquid steel, flowing out and onto the floor in the plant). The task is to find a method to extend the number N, without compromising with safety.

The PPBM process was applied to the above-mentioned industrial case. Based on access to offline data from the plant and multiple discussions with the pilot owner, a quite clear picture of the challenge could be formed. A physics-based model was proposed and implemented. The numerical implementation, written from scratch, has been done using Python 3. The model is aiming to predict the temperature evolution in the system (steel, slag, refractory) and the erosion evolution of the wear bricks of the refractory. A number of challenges have to be faced due to the complexity of this task:

i) The model must be fast. Therefore, it is designed locally (at each height) to be one-dimensional through the refractory. Conservation equations for energy of metal, slag, and refractory (quasi 2D) are included, ii) A ladle is going through multiple operations for each use. This requires specific model boundary conditions for each part of the sequence, iii) The boundary conditions (energy and composition of dissolved species) are complex due to applications of gas-induced stirring, natural convection, radiation (only heat), and use of submerged electrodes in the slag, iv) The addition of alloying material and slag formers consume considerable heat. This necessitates an enthalpy-based description of the slag and metal that can handle heats of phase transition. Thermodynamic data is not readily available for such systems, v) The erosion of the refractory is driven by thermal shocks and dissolution of refractory when contacted by slag and metal. The solubility of refractory components into the slag and metal can only be obtained from thermodynamics software, built on lab experiments, vi) The mass transfer is depending on natural convection and forced convection due to application of gas (bubble) stirring. Both this and the additional impact of surface waves must be represented, and vii) In addition, there are processes that must be considered (waiting times between uses, use of burners and lids) and mistakes in the data input given by operators.

The model should be fast enough to predict the thermal evolution and erosion losses during the lifetime of the ladle, and this to happen in an acceptable time (<1 h). From the learnings, indicated by i) to vii) above, several general recommendations for improvements of the *PPBM* can be extracted and documented for future application.

The model is currently predicting temperature but is still under development. The erosion model has not been activated. This will be implemented and planned to be in operation during 2021. Due to the complexity of the modelling tasks, still significant simplifications have to be introduced. By going through this process, the learning will be generalized and included into the *PPBM tool*.

**Publication of results.** The details of the learnings and recommendations for an improved PPBM tool will be published in a paper, dealing mainly with the methodology and the process for developing pragmatism-based physics models. In a second paper we will describe the physics-based model in full detail and show validation of the model against data.

After this, papers on hybrid and cognitive twins, where the focus will be on methods to exploit the interaction between physics-based and data-based (ML/AI) methods, will be produced.

The results from the model will be shared at github.com at the end of the project. This will include source code and documentation. This code will be available under an open-source licence and can be reused for similar projects in the metals industry and other related industries.

**7.2.4 FUSE tool**

The FUSE tool fuses a physical process model with process data in stochastic nonlinear state estimation. The tool is developed for the Sumitomo pilot (see D3.2), using a dynamic physical model for a boiler furnace, data from the Sumitomo pilot, and UKF.

The tool is described in more detail in the Appendix, in D5.2 Task 5.1 (“Plant DT with ML/AI”). The physical model tuning results were published in Ikonen & Selek (2020). The applied view to the tool is reported in D3.2. A generalized FUSE-tool is provided for the COGNITWIN toolbox, providing an implementation of UKF, with a possibility to flexibly use a physical model and select input/measurement signals.

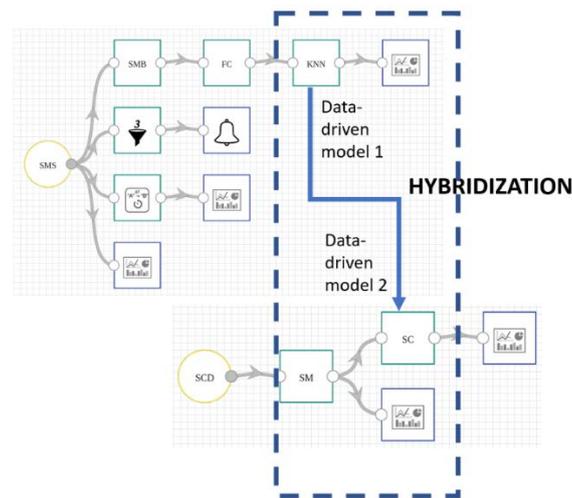
The SubFUSE tool is an alternative approach for solving the Sumitomo pilot. The SubFUSE uses subspace identification for data-driven construction of a plant model. As the outcome is a linear state-space model, a Kalman filter can be used for state estimation. The approach has been tested in simulations, see Task 5.1 for details.

**7.2.5 Hybrid model designer for StreamPipes**

One of the most challenging tasks in the hybrid modelling is enabling an efficient creation of hybrid models, since it requires an efficient orchestration. We argue that StreamPipes is a very suitable framework for such a hybridization due to its pipeline-oriented nature.

This component uses these functionalities to support the creation of hybrid models.

In the following figure we illustrate the integration of two data-driven models (developed for the Sidenor pilot), whereas the output of one model is used as an input for the other model.



*Figure 5. Integration of two data driven models.*

As shown in Figure 5, the output of one model is used as an input for another model. There can be different ways of connecting and StreamPipes orchestration seems to be suitable for any meaningful combination of models.

### 7.2.6 TMat-SynDat

By combining two related models - a data driven-model and a (physics)-driven 1<sup>st</sup> order principal model - a hybrid digital twin has been generated. 1<sup>st</sup> order physics-driven models can be beneficial to the data-driven ones in many ways including but not limited to:

- Synthetic data generation in case of poor data: An example is training a machine learning pipeline for the predictive maintenance. Generally, when a machine is new, it does not have historical sensor data that can be used to train a data-driven approach. When carefully designed, the virtual physics-based twin can generate the needed supervised training dataset.

- Data-Driven Digital Twin quality control: When operating a critical infrastructure or asset, it is seen as a risky approach to fully rely on data-driven approaches in taking real-time decisions. To mitigate these risks, it is possible to build a control pipeline in which the physics-based model will be used as a controller to the data-driven predictor. A broker needs to be designed to integrate the two approaches in a seamless way.

Data-driven models can be used to continuously calibrate physics-based models. Machine degradation, wearing of parts, environment, and other factors impacts the overall process performance over time. The state of the practice is that an operator will manually recalibrate the control system when a deviation is identified. Such manual operation can be replaced by setting a data-driven model to identify and calibrate critical process variables that will be fed into a physics-based model, which in turn will optimize the control system of the process.

A hybrid digital twin for predictive maintenance of a component which is composed of electrical and mechanical elements has been generated. The hybrid twin includes the sensor installed on the machinery.

TMat-SynDat is a synthetic data generator. Developed in MATLAB, TMat-SynDat enables synthetic data generation for common electro-mechanical parts (electric DC motor and hydraulic shaft). First, the physical hydraulic/motor/gearbox models for the plant have been developed. They have been developed/implemented using Matlab SimScape with the aim of predictive maintenance purposes. Model parameters have been calibrated and simulated. Potential failure scenarios have been identified and used in the generation of synthetic data, used by ML/AI algorithms for predictive maintenance. The model is enriched with data retrieved from the experts.

### 7.2.7 Hydraulic/Motor/Gearbox Models

Physical hydraulic/motor/gearbox models for the plant have been developed. The models are developed/implemented using Matlab SimScape with the aim of predictive maintenance purposes. Model parameters have been calibrated and simulated. Potential failure scenarios have been identified and used in generation of synthetic data, used by ML/AI algorithms for predictive maintenance. The model is enriched with data retrieved from the experts.

Since the last milestone, sensors have been added to the 1st order principal models. Current and temperature sensors have been added to the motor and gearbox model, while a hydraulic press sensor has been added to the previously developed hydraulic press model.

Following the sensor implementations for the models, model parameters have been calibrated and the random error sources have been introduced to the model. Thus, the model has been updated to be ready for predictive maintenance algorithms by introducing sources of random errors to be used in

predictive maintenance, adding appropriate sensors to observe the effect of these error sources on important variables of the models and calibrating the geometrical, electrical, and hydraulic parameters of components to make the model as applicable and as realistic as possible.

The TMat-SynDat is described in the component/tool template in Appendix.

### **7.3 Progress beyond State of the Art or State of the Practice**

Several physical and data-driven models have been developed for the pilot cases.

TEKNOPAR has created a hybrid digital twin for predictive maintenance, combining a physical model of the process with sensor data and ML/AI algorithms.

Nissatech has developed a hybrid model designer for StreamPipes. It has been applied in the Sidenor pilot to combine two data-driven models.

Important extensions and modifications for the SINTEF Bedrock, SINTEF Pragmatism, and Cybernetica CENIT tools/ platforms have been made that will be used by Task 5.4 later in the project.

### **7.4 SIDENOR Pilot**

The pilot is addressing the wear of refractories in steel ladles and focusing on increasing the ladle lifetime and thereby cutting the costs of the ladle operations. The pilot goal is to be able to predict if a ladle can be used safely at least one more time before relining.

Operational data have been supplied to the development team, including both static (acyclic) and dynamic (cyclic) data from the ladle operations. Sidenor plans to set up a mirror of their internal data, which also can make real time data available to the development team.

The toolbox elements, being developed and used in the pilot, are "Pragmatism in physics-based modelling" (PPBM), the " Hybrid model Designer for StreamPipes-based Toolbox ", " Set of adapters for StreamPipes-based Toolbox " and " Services for resolving tool wear / equipment degradation problems in process industry". In addition, a number of classical machine-learning methods (Python libraries) are being applied.

The work has demonstrated the importance of the project development team having a very good understanding of the process. This is a prerequisite for building the best possible model. The Sidenor data is typical for this type of heavy industry and cannot be used directly without qualified pre-processing. It is therefore critical to have a close interaction with the pilot owner.

The Sidenor pilot case is progressing as planned and first model predictions have been demonstrated. The development team builds one physics-based model of the erosion process. The thermal part of this model has shown that it is possible to obtain a mapping of the internal erosion state of the ladle by combining a physics based model and thermal images of the outside wall of the ladle. This possibility was however not included in the scope and plans for the pilot, but may be pursued at a later stage. The physics-based model is now being extended with erosion predictions.

In parallel, ML-based methods on the provided data indicate that it is possible to predict the probability for a safe next heat for the ladle, based on data.

Next steps are to combine the methods mentioned above, to arrive at a hybrid twin that exploits the best of the different approaches. Hybridization methods, based on StreamPipes, are under development, and are planned to orchestrate the hybridization, and later, the cognitive twin.

## 7.5 Summary of the key achievements

Task 5.4 has been involved in the development or extension of the following tools:

- several physics-based process models for the pilots
- improvements to the Bedrock platform
- improvements to the Pragmatism methodology
- input signal and calculation results validation (Cybernetica CENIT extension)
- FUSE/ SubFUSE
- Hybrid model designer for StreamPipes
- Synthetic data generator for DC motor and hydraulic shaft

## 7.6 Next steps

As the model development in the pilot processes goes forward, Task 5.4 will continue to monitor how the different models are combined and used together as Hybrid Digital Twins in the pilots. Suitable tool candidates will be identified from the pilot tasks, and generic elements will be extracted, generalized, and finally implemented as tools in the toolbox.

The possibility to use the tools in other pilots will also be investigated.

## 7.7 Demonstrator Scenario/Description/Video

The Pragmatism-based model for the Sidenor pilot is currently predicting temperature. The erosion model has not been activated. Videos demonstrating the use of the model and how CENIT is used to build a hybrid digital twin in the Hydro pilot case are available on the COGNITWIN YouTube channel: <https://www.youtube.com/channel/UCgHunz1V68YGOxaqVvkyN1A>.

# 8 Cognitive Digital Twins

## 8.1 Objectives, challenges and components

The main objective of this task is to design the cognition process as a part of industrial decision making and its realization in Cognitive Twins. During the 1st year of the project, we reviewed the relevant definitions in the literature and documented the results of this analysis in our paper “Cognitive Digital Twins for the Process Industry” accepted for the Twelfth International Conference on Advanced Cognitive Technologies and Applications (COGNITIVE 2020). We defined the cognitive digital twins as an extension of digital twins with cognitive capabilities in the context of the process industry.

For realizing cognitive digital twins in the process industry, an essential aspect is to devise the architectural building blocks that can serve as a foundation for cognitive systems in this domain. We provided our architectural perspective on the type of cognitive services needed for Cognitive Twins in the context of process industry. The proposed architecture provides a blueprint, supporting a wide range of abilities similarly to human capabilities. In the following figure (Figure 6) we illustrate the role of knowledge in the Cognitive Twin (Layer in the Toolbox).

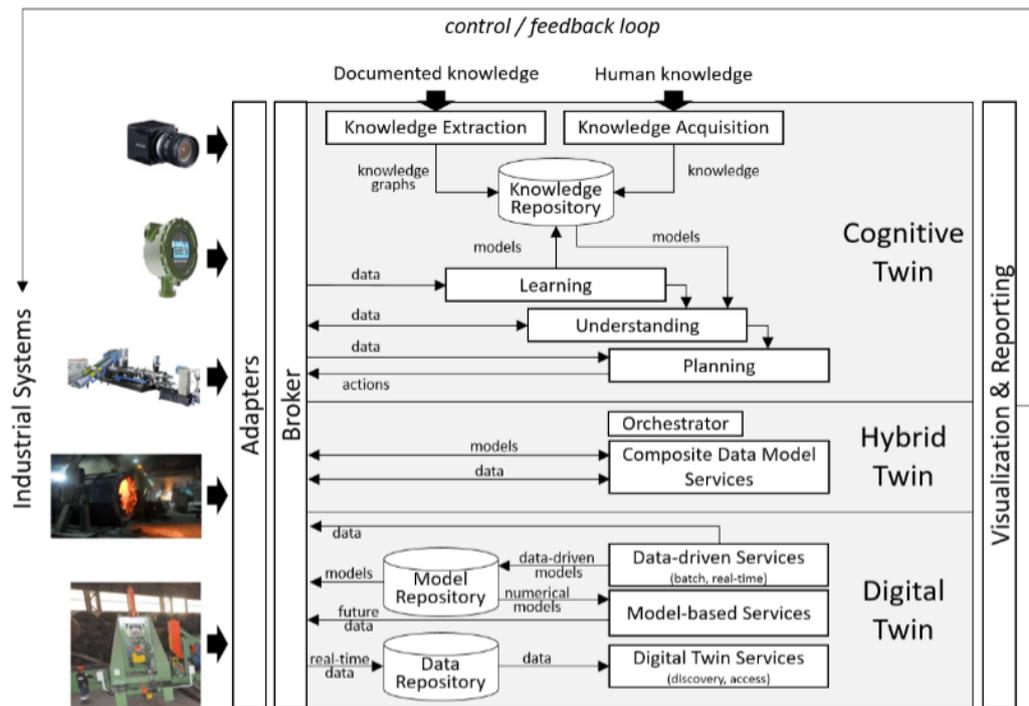


Figure 6. The role of Knowledge in the Cognitive Twin.

There are many definitions of the cognition, but for this paper we focus on that derived from the cognitive computing domain, which are related to reasoning and understanding at a higher level, in a manner that is analogous to human cognition. We specialize this view for the complex cases where there is a lot of uncertainties inherent in the available data and models. We expect that a human-cognition-like approach will enable a broader, as well as a more connected view on the data and models. The key advantage is the introduction of new knowledge that should provide missing insights for resolving original cases, as illustrated in Figure 7.

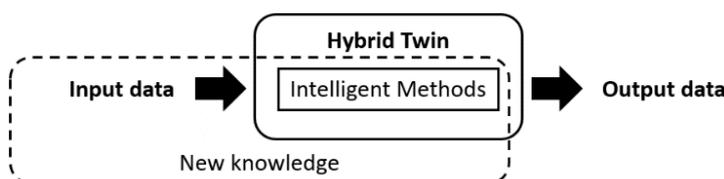


Figure 7. Cognition extending Hybrid Twin solution.

As presented in the figure, we assume that “intelligent methods”, which can be a part of Hybrid Twin, support the development of a solution that “maps” inputs into outputs. However, the solution might be missing a high accuracy, due to not having enough data in the training set. As illustrated, cognition supports augmenting the input data, as well as intelligent methods with new knowledge (gathered directly from the expert or some other sources), with the goal to generate new outputs (with a higher accuracy). Therefore, we argue that the uncertainty inherited in the problem (e.g., missing data, models) can be resolved by augmenting data and intelligent methods to compensate missing information. Notice that this process is not about getting new data, but rather new insights about existing data (through cognition).

Therefore, the main role of cognition services is to enable understanding the monitored system's behavior under various types of uncertainties/unknowns, to support reliable decision making (by human experts) or control (in autonomous systems). Uncertainties can be of different types, but we focus on the two most important types from the DT point of view: lack of data and unavailability of models regarding the current system behavior. It means that the current system behavior cannot be understood neither a) by analyzing past data, since the relevant data is missing nor b) by the simulations of numerical models, since these do not exist (or are not accurate enough). In such cases, it is important to compensate these unknowns by introducing new processing steps that will gradually improve the understanding of the system behavior, until this understanding is enough for the desired/requested action. This process we consider as cognition, where the processing steps are part of cognition services.

The main challenge is that the real time data is not enough for understanding the current situation (regarding the underlying problem). The main goal of the cognition service is to enable resolution of the original problem by introducing new knowledge which provides new insights for the model learning/creation processes, e.g. introduction of some constraints in the interpretation of originally collected data. Therefore, cognition is working on top of existing models, which can be derived using AI methods, extending the intelligence with the with the deep understanding, and reasoning strategies.

We can materialize this general process by following four steps (cf. Figure 6): (1) Inserting new knowledge (relevant for the problem) (2) Learning more accurate models, by applying new knowledge (3) Better situational understanding (e.g. lower interpretation uncertainty), by applying new models (4) Planning actions for resolving the problem, based on improved situational understanding.

We describe these steps in the following: **Firstly**, by knowledge extraction and knowledge acquisition, for gathering knowledge from the existing data sources (e.g. unstructured and semi-structured content) and from experts, respectively. The goal is to collect knowledge related to the uncertainties in data and models. Since the process is related to supporting human-like understanding, it is important that the process is driven by the well-defined knowledge structures (like knowledge graphs) which provide a general description of the domain. Indeed, one of the main characteristics of the human cognition is a very fast discovery of hidden connections between arbitrary information items, which is based on large memory maps. **Secondly**, by learning, which encompasses applying new knowledge to the existing data, models, and methods, with the goal of learning more accurate models (from existing datasets). There are three main activities: transforming existing datasets in the anomaly-free ones, which can be used for learning more accurate models, improving used learning methods by introducing some knowledge-driven constraints in the learning process, and adding new methods which can complement existing ones in the context of the above-mentioned uncertainties. **Thirdly**, understanding, which is related to applying new models on real-time data to get a better interpretation of the situations of interest (e.g. problem/anomaly detection). We assume that, as in the human-like cognition, this process can be iterative, i.e. understanding processes can generate data which can be used for improving the learning process. **Finally**, planning, for defining optimal actions based on system behavior understanding.

There are many challenges to be addressed to realize the vision of the cognitive digital twins. The most important ones are discussed below:

**Knowledge representation challenge.** The first question to be clarified is how knowledge can be formally represented to enable a digital twin to learn from experience and behave intelligently like a human. All cognitive services mentioned above are heavily dependent on this decision.

The more complex the representation of knowledge is, the more difficult it is to acquire this knowledge automatically. However, more advanced reasoning services can be offered. Our goal is not only to support the decision-making process, but also to increase its accuracy and human-acceptance. Thus, both declarative and procedural knowledge is needed, as questions like ‘what?’, ‘how?’, ‘when?’, ‘in what context?’, ‘what-if?’ etc. should be answered.

Several knowledge representation formalisms seem to be suitable for cognitive digital twins. To clearly separate the general knowledge from the specific knowledge, it makes sense to structure the knowledge into two parts: ontologies for representing the domain knowledge and rules for representing the problem-solving knowledge.

To better understand a current situation (i.e. the asset itself, the context in which it is used, its environment, etc.), we consider using ontologies. They are a knowledge representation method that is on one hand expressive enough and on the other hand extensible. They could be used:

- to represent the domain knowledge which includes the vocabulary domain-experts apply (e.g. brick wall: types of bricks - e.g. red shale, clay bricks, etc. - the features of bricks - thermal shock resistance, mechanical strength, etc. - and so on) as well as the constraints (e.g. temperature threshold at which the stone is unusable)
- to take into account existing standards for the domain
- to support collaboration between digital twins, e.g. for cooperative execution of complex tasks.

Although simple constraints (e.g. temperature of a ladle must not exceed a certain threshold) can be modeled by using ontologies, there are many scenarios where complex (functional or behavioral) constraints should be considered (e.g. calculations including results of different physics-based, AI, statistic-based, etc. models). To mimic the reasoning of a human expert in solving knowledge intensive problems, there is a need to use rules (e.g. event condition action rules). Rules should be used even in the presence of incomplete and/or uncertain information to (i) focus the attention to the most important aspects and (ii) collect additional, goal-oriented information relevant for a given context. This can be done by mapping raw sensor data and/or outputs of different digital twin models into actions (such as control decisions or recommendations for human operators).

**Knowledge acquisition challenge.** The second challenge is to collect knowledge which is not only spread in different documents (e.g. excel tables) and software systems (e.g. error reports in MES systems), but could be also implicit as it is based on personal experience which is even more difficult to express. To make the tacit knowledge explicit and machine-understandable as well as -processable, different cognitive technologies could be used, such as NLP, speech recognition, etc. For example, one possibility is to apply a speech-to-knowledge approach, as speech is relevant for the shop floor workers for short information interchange allowing hands-free conversations. Since the multilingual speech functionality in recent years became a commodity available on smart speakers, mobile phones, and computers, the pre-existing solutions could be reused and added to the cognitive digital twin to enable speech communication channels with human operators. Ontologies can help achieving higher accuracy

of resulting rules, as synonyms, multilingual aspects, context, etc. can be taken into account. In this way, the domain and problem-solving knowledge will be connected.

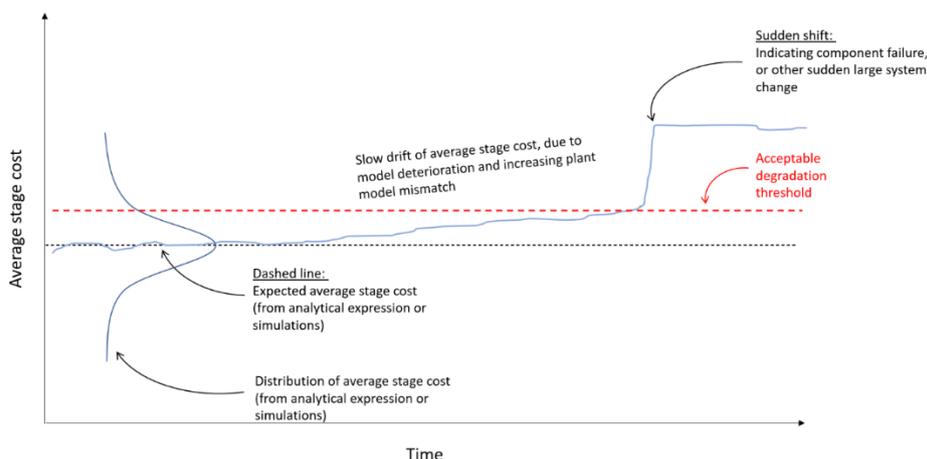
**Knowledge update challenge.** In addition to collecting knowledge, the ability to learn, to unlearn, and to continuously update knowledge is crucial for cognitive digital twins to create competitive advantage. Knowledge update is however a complex process, which includes knowledge extension (e.g. adding a new entity in the ontology for new types of bricks), knowledge forgetting (removing an ontology entity representing material not used anymore for bricks), and knowledge evolution (e.g. changing the maximum temperature of a ladle). Similar strategies can be applied to the problem-solving rules. The challenge lies not only in ensuring the consistency after applying a change, but more importantly in discovering the need for a change. This can be done by applying usage-driven strategies (e.g. by monitoring whether the proposed decisions were accepted by domain experts) or by using structure-driven methods (e.g. by using ontology-based reasoning to discover conflicting rules or generalized/specialized rules).

## 8.2 Detailed description of the activities performed

### 8.2.1 Cognitive CENIT

Cybernetica has started the development of an extension to its software product Cybernetica CENIT called Cognitive CENIT, which enables self-diagnosing. A framework for self-monitoring of the model predictive control application via stage cost monitoring has been developed and is currently being evaluated on a simplified and simulated test process. The framework consists of the following steps:

- Estimate the measurement error distribution
- Propagate that noise distribution through the closed-loop model predictive controller via Monte Carlo simulations
- Compare the resulting distribution of the average stage cost from the actual plant. If the average stage cost is significantly off from the theoretical distribution, this indicates an error in the closed-loop model. This is illustrated in Figure 8:



*Figure 8. Cognitive CENIT.*

As shown in Figure 8, analysing the average stage cost distribution in closed loop applications can detect anomalies like component failure or model mismatch

### 8.2.2 State Estimation as Part of Plant Monitoring

The University of Oulu has developed the physical model tuning approaches and model-based state estimation tools, reported in T5.1. These tools are to be used for the development of cognitive digital twin behaviors. In the considered framework, state estimation will be a significant part of the plant monitoring tool, from where further decision support as well as optimization tools for plant operation and maintenance are envisioned. The tool is developed in view of experiences and requirements of the Sumitomo pilot fouling monitoring and control problem in particular.

### 8.2.3 TMat-PdM Predictive Maintenance

Focused on the NOKSEL pilot case of a metal sheet roller system of spiral welded pipes (SWP), TEKNOPAR has worked on related to T5.5. TStreamPipes-ML is developed by TEKNOPAR to apply ML algorithms on the stream and to compare the results of the algorithms. The developed data processor enables users to select the algorithms to be used/compared and the output is displayed on a dashboard.

TEKNOPAR's TMat-PdM component which is under continuous development, can be used for predictive maintenance of the DC motor, gearbox, and hydraulic press. Different ML models have been used by TMat-PdM. TMat-PdM uses MATLAB's Predictive Maintenance toolbox and Classification Learner app. TMat-PdM enables visualization of a confusion matrix for the selected algorithms to present the difference between simulation outputs.

In the related pilot case (the NOKSEL pilot), the cognitive twin will introduce improved decision making by integrating human knowledge into the decision-making process. The anomalies, alarms, and early warnings of machine and system problems will be tackled by the cognitive twin. The decision-making process will emulate the experienced human operator with embedded knowledge base. The cognitive twin will augment expert knowledge for unpredictable cases on the digital and hybrid twins. The human operator's knowledge is reflected to process knowledge and physics-based models with parametric values as well as thresholds and causality relations. Expert knowledge on the causes of breakdowns is collected with the series of the problematic operations and the initial causes which trigger the successive reactions. Cognition will be further integrated by making use of the machine learning algorithms, ontologies, and knowledge graphs to capture background knowledge, entities, and their relationships. Reacting to early warnings, cognitive twins will bring life cycle optimization, and suggesting optimized predictive actions will improve operational performance by optimized operational parameters and it will also decrease energy usage (Figure 9).

TSteampipes-ML and TMat-PdM are detailed in Appendix.

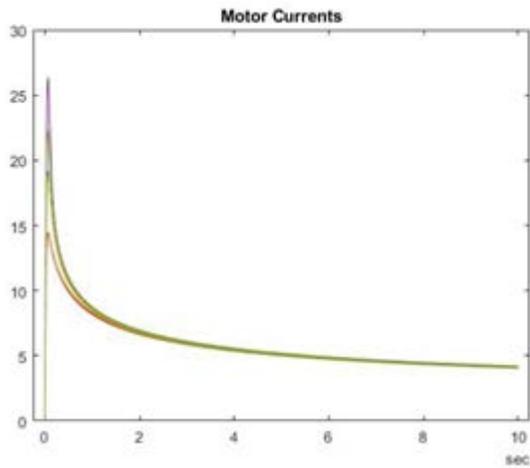


Figure 9 Cognition using StreamPipes Toolbox.

### 8.2.4 Cognition using StreamPipes Toolbox.

StreamPipes Siddhi-Processor (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)**.

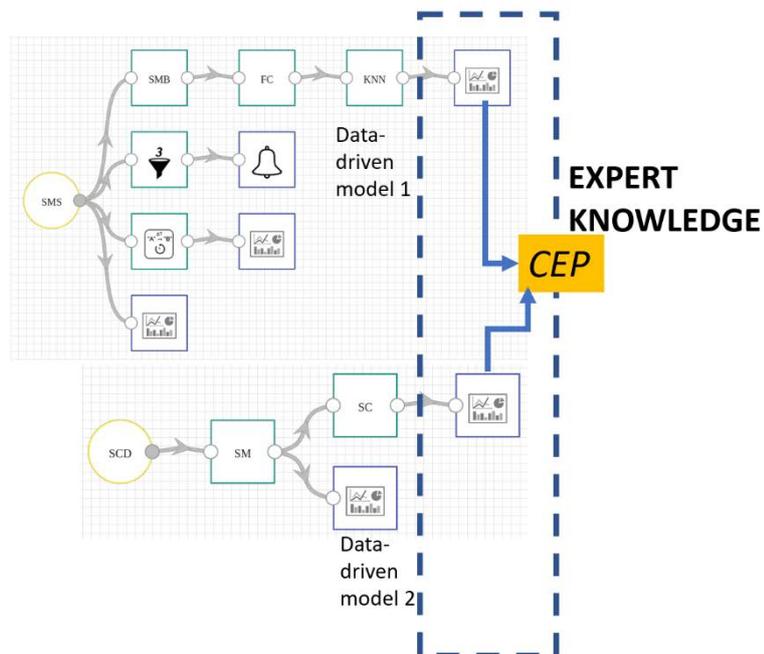


Figure 10. Complex Event Processing (CEP).

The role of Complex Event Processing (CEP) in codifying expert knowledge. Siddhi CEP performs Complex Event Processing using the Siddhi engine. It provides application of complex logic to the “main” outputs of this pipeline (results of various analytical methods). In addition, this element provides points of connection for this and previous pipelines – CEP can be applied on outputs of multiple pipelines connecting them into one complex pipeline (Figure 10).

**Knowledge/Patterns.** Regarding outputs from data-driven models, MEWMA, and KNN elements, we singled out the following queries (patterns) that can be applied:

- Test whether there were more than  $M$  anomalies in a single heat during a 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 element*)
- Test whether there were too many anomalies in single heat with the same root-cause. For example, if there were more than 100 anomalies with the same root-cause parameter, this can indicate some sensor or part of a machine is faulty and should be checked. (*uses output from MEWMA element*)
- Test whether during last  $N$  heats, a 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.
- 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.

### 8.3 Progress beyond State of the Art or State of the Practice

Most related work is dealing with the self-awareness of digital twins. There are several systems which we briefly analyze in the following text.

In [Kap+20], a DDDAS system is extended to a self-aware digital twin to support real-time path planning of an unmanned aerial vehicle according to its structural integrity. In this work, stimulus awareness and goal awareness are implicitly involved.

In [Rok+20], a dynamic data-driven approach is applied to the digital twin model for 3D printer products. The digital twin is a machine learning model that predicts the surface texture and dimension of the product to be printed by using environmental parameters from sensors as input. Stimulus awareness and time awareness are implicitly involved.

In [ROß+14], a 3D simulation model is used as the mental model for the path planning of a mobile robot. The robot simulates all the possible future paths resulting from different initial parameters. Stimulus- and time-awareness are implicitly involved, the latter to predict future path trajectories.

In [ZHA+20] twins that can exhibit a high level of intelligence are described. They can replicate human cognitive processes and execute conscious actions autonomously. The paper brings together the concepts of digital twins and self-awareness and discusses how the different levels of self-awareness can be harnessed for the design of cognitive digital twins.

[Kap+20] M. G. Kapteyn, D. J. Knezevic, and K. Willcox, "Toward predictive Digital Twins via component-based reduced-order models and interpretable machine learning," AIAA Scitech 2020 Forum, pp. 1–19, 2020.

[Rok+20] S. Rokka Chhetri and M. A. Al Faruque, "Dynamic data-driven Digital Twin modeling," in Data-Driven Modeling of Cyber-Physical Systems using Side-Channel Analysis, Cham: Springer International Publishing, 2020, pp. 129–153.

- [ROß+14] J. Roßmann, E. Guiffo Kaigom, L. Atorf, M. Rast, G. Grinshpun, and C. Schlette, “Mental models for intelligent systems: eRobotics enables new approaches to simulation-based AI,” *KI - Künstliche Intelligenz*, vol. 28, no. 2, pp. 101–110, Jun. 2014
- [ZHA+20] Nan Zhang, R. Bahsoon; G., Theodoropoulos Towards Engineering Cognitive Digital Twins with Self-Awareness, 2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC), 2020

## 8.4 Summary of the key achievements

There are three key achievements: (1) Initial conceptualization of the cognition, driven by the role of knowledge in the cognitive twin layer (Toolbox) (2) Initial realization of the cognition in two pilots. (3) Conceptual model for the cognition in the StreamPipes toolbox.

## 8.5 Next steps

Since this task is responsible for the realization of the cognitive twins, which are one of the main outcomes of the project, there are several dependencies to other tasks, i.e. the results from these tasks should be fed into this one.

In addition, initial concepts and realizations (as mentioned in previous section) will be further developed and validated.

# 9 Reflection on the pilots

## 9.1 HYDRO Pilot

There has been good progress in the Hydro case towards developing both physics- and AI-based models that can be incorporated into a cognitive digital twin aimed at regulating operation of the Gas Treatment Centre (GTC).

A dynamic physics-based model has been developed and implemented as a Cybernetica Application and Model Component. The model has been tuned using Cybernetica Modelfit to best reflect fluoride emissions measurements from the GTC and has been shown to accurately follow trends in emissions data caused by changes in weather. The model uses input from several sources, including Weather API developed by SINTEF which allows searching and collecting weather data made available online by the Norwegian Meteorological Institute (MET Norway). Other sources of input data include alumina certificates, electrolysis cells, and the GTC itself. A digital twin is ready to be implemented online using Cybernetica CENIT once the case can overcome COVID-19 limitations and obtain a virtual machine. Process measurements for continuous adaption of the physical model will be taken into consideration.

A purely data-driven AI model for the process was developed in parallel using the weather data collected through the Weather Api developed by SINTEF but has so far been difficult to use to replicate process behavior independently, as the measurements recorded in the GTC are influenced by many other process quantities apart from just weather.

An ongoing challenge to the digital twin technology is better understanding the sources of error in process measurements, inputs, and the physical model itself. To address this, another ML model is under development for analyzing the deviation between process measurements and estimated measurements from the physics-based model (residuals). By using both weather data and process quantities not included in the physics-based model, an additional ML layer will be put on top of the

existing physics-based model and trained to estimate the residuals, to create a hybrid physics-ML model boosting the total accuracy.

## 9.2 ELKEM Pilot

There has been good progress in the Elkem Pilot case. A physics-based model has been developed and implemented as a Cybernetica Application and Model Component. The model has been tuned against data logged from the process using Cybernetica Modelfit and it reflects the process well.

Not all measurements were available on OPC, therefore a bespoke Cybernetica OPC UA server was created to query relevant databases and publish the measurements to OPC. The digital twin is soon ready to be run online using Cybernetica CENIT. Process measurements for continuous adaption.

A data-driven AI model is being developed to describe the slag amount from the upstream furnace. The input time series data are measurements from the furnace and the materials added. Even if the current models show some predictive abilities, their results are still quite unreliable. Further improvements of the models rely on a better understanding of the process and incorporating more input data, like post tap hole measurements from previous tappings.

Infrared cameras will utilize a machine vision tool to provide additional measurements; most importantly temperature of the metal during tapping, refining, and casting. Additional information may be extracted from the thermal cameras, for example ladle slag coverage, metal/slag ratio in tapping, and chemistry/dynamics during refining. A set of machine vision tools will be developed for this purpose, alongside the necessary adapters.

## 9.3 SUMITOMO Pilot

The Engineering pilot (Sumitomo SHI FW) in WP3 considers monitoring and control of heat exchange surfaces in biofuel combustion. On-line characterization of the incoming fuel feed is important information in fouling monitoring. A state estimation tool was constructed to estimate the uncertain input fuel fragments in the fuel. The estimator was based on applying a detailed circulating fluidized bed (CFB) furnace model in conjunction with nonlinear bayesian state estimation tool. A generalized version of the fuel characterization tool – enabling application of other prediction models and setups of plant measurements – was provided for the COGNITWIN toolbox as a set of Matlab code. An example of a setup of data communication was demonstrated via an OPC-UA tool, consisting of free software (Prosys) and existing properties of Matlab (Mathworks) and StreamPipes (Apache). The tools are described in more detail in the Appendices of component descriptions in D5.2 and D4.2.

The tool promotes the fusion of first principle based physical models with on-line plant measurement data. A procedure for fine-tuning the physical model for local plant conditions was suggested, keeping in mind the value of the physical model predictions. The state estimation was based on UKF, a modern realization of the bayesian state estimation in the spirit of population-based machine learning paradigms. The approach provides hybridization of physical knowledge with data, as a service of a plant digital twin platform. Among the lessons learned was that the estimation of highly data-driven dynamics based on plant operating data is complex in the industrial environment, so ensuring robustness of approaches is highly valuable. The fusion of a number of models – including physical, grey-box and data-driven, partial and complementary – is foreseen in solving the problems in the next phases of the WP3, looking at monitoring of fouling and slagging at the heat exchange surfaces. It is expected that the cognitive features of a digital twin supporting integration and decision making will play an important role when deriving approaches for the control of the fouling phenomena. This phase

eventually expects to look at improvements to automatic control and/or prescriptive maintenance procedures.

#### 9.4 NOKSEL Pilot

The NOKSEL use case has been progressed as planned. Sensor installations on the SWP machinery and related components have been completed. Following the sensor installations, by means of two PLCs multi-sensor data were acquired over OPC and passed in MQTT in JSON format. Kafka uses the MQTT data. Data consumed by Kafka is stored in Cassandra database.

Machine learning and deep learning algorithms need big data to be collected. The cognitive digital twin for the use case is related to predictive maintenance. The collected data does not have many machine breakdowns. Labelling a dataset for failed data is a challenge, not only does it require real data for machine failures, but also should expert knowledge be inserted into the models. While working, a huge amount of data is collected. Missing failure related real data has been overridden by means of a tool used to generate synthetic data as close as possible to the real cases. To cope with the missing data challenge, a model-driven twin is developed and integrated together with the data-driven model, as a result a hybrid digital twin has been generated.

#### 9.5 SAARSTAHL Pilot

There has been **substantial progress in the Saarstahl use case** 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 allowing 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 bars 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 bars, 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 – 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 by using either a network from the class of multi-object detection and localization networks, or instance segmentation networks. Multi-object detection and localization networks means that the output of the network is a list of objects, each specified by a label and a bounding box. The technology is well understood and mature, but is likely to encounter problems with the very elongated shape of the rolled bars, which will lead to a very high degree of overlap between the bounding boxes. Instance segmentation means that the output of the network is a label per pixel (as for semantic segmentation), but different instances of the same object class are recognized and receive separate labels. The technology is more promising for very elongated objects, but in general is less mature and less understood, leading to a higher development risk and effort. **A consequence for our toolbox components is that Neuroscope will be extended for support of the respective network types.**

## 10 References

- S. Joe Qin & Leo H. Chiang (2019) Advances and opportunities in machine learning for process data analytics. *Computers and Chemical Engineering* **126**, 465–473.
- Timur Bismukhametov & Johannes Jäschke (2020) Combining machine learning and process engineering physics towards enhanced accuracy and explainability of data-driven models. *Computers and Chemical Engineering* **138**, 1–27.
- Enso Ikonen & Istvan Selek (2020) Calibration of physical models with process data using FIR filtering. *Australian and New Zealand Control Conference*, Gold Coast 26-27 Nov 2020, 143–148.
- Enso Ikonen & Istvan Selek (2020) Calibration of physical models with process data using FIR filtering. *Australian and New Zealand Control Conference*, Gold Coast 26-27 Nov 2020, 143–148.
- Schorr, C., Godarzi, P., Chen, F., Dahmen, T. (2021) Neuroscope - An explainable AI toolbox for semantic segmentation and classification of deep neural nets, in: Applied Sciences, Special issue on explainable artificial intelligence (accepted for publication).
- Sailesh Abburu et al. (2020) COGNITWIN – Hybrid and Cognitive Digital Twins for the Process Industry. *2020 IEEE International Conference on Engineering, Technology and Innovation*.
- M. G. Kapteyn, D. J. Knezevic, and K. Willcox, “Toward predictive Digital Twins via component-based reduced-order models and interpretable machine learning,” AIAA Scitech 2020 Forum, pp. 1–19, 2020.
- S. Rokka Chhetri and M. A. Al Faruque, “Dynamic data-driven Digital Twin modeling,” in *Data-Driven Modeling of Cyber-Physical Systems using Side-Channel Analysis*, Cham: Springer International Publishing, 2020, pp. 129–153.
- J. Roßmann, E. Guiffo Kaigom, L. Atorf, M. Rast, G. Grinshpun, and C. Schlette, “Mental models for intelligent systems: eRobotics enables new approaches to simulation-based AI,” *KI - Künstliche Intelligenz*, vol. 28, no. 2, pp. 101–110, Jun. 2014
- Nan Zhang, R. Bahsoon; G., Theodoropoulos Towards Engineering Cognitive Digital Twins with Self-Awareness, 2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC), 2020

## 11 Appendix 1. Toolbox components

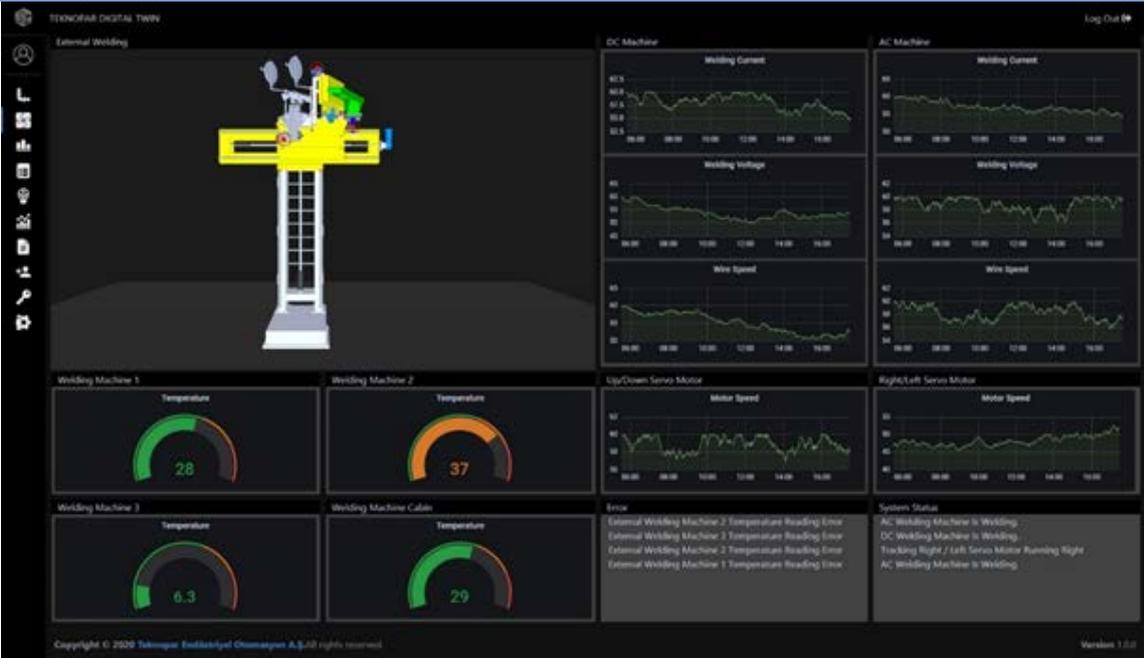
Component/Tool description
<b>Component/Tool/Method/Framework/Service Name</b>
STEEL4.0- ICPV: Industrial Control and Visualization Panel
<b>Defined in Task</b>
T5.1
<b>Short Description – incl. Purpose</b>
<p>STEEL4.0-ICPV supports the digital twin by means of visual components presenting the generated data which is retrieved from, and processed within other STEEL4.0 components.</p> <p>The purpose of STEEL4.0-ICPV is to visualize the digital twin. ICPV visualizes the historical, real time data as trend graphs and status reports by means of different types of graphical elements. Both real-time and processed information that are used for condition monitoring and predictive maintenance of SWP are visualized by ICPV.</p>
<b>Progress since last milestone</b>
<p>Visual components of all of the SWP machinery parts were prepared.</p> <p>GUI designs (including the icons, graph types, dashboard elements, etc.) for the display screens were updated, graphical elements to display real-time and calculated fields have been changed. AI/ML related elements are visualized on GUIs of ICPV.</p> <p>Accepted Proceeding Paper including STEEL4.0 Digital Twin (related to ICPV elements):</p> <p>Albayrak, Ö., P. Unal “Smart Steel Pipe Production Plant via Cognitive Digital Twins: A Case Study on Digitalization of Spiral Welded Pipe Machinery” has been accepted for publication in the Proceedings of the ESTEP Workshop on Impact and opportunities of AI in the Steel Industry.</p>
<b>Examples of usage / illustrations</b>

<p><i>Figure: ICPV Sample GUI for External Welding of SWP displaying Real-Time sensor Data.</i></p>



Figure: Real-Time ML Sample GUI visualized by ICPV.

**Interfaces (in/out) – system/user**

ICPV uses real time sensor data and predictions as input and displays them in the forms of visual elements to the users.

**Subordinates and platform dependencies**

Being a web application, ICPV 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**

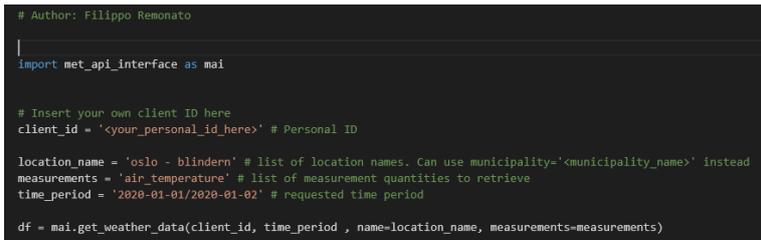
The current TRL is 5-6 running to be TRL 7.

**References – incl. web etc.**

none

**To be considered in particular for the following COGNITWIN pilots**

NOKSEL

Component/Tool description
<b>Component/Tool/Method/Framework/Service Name</b>
MAI - Collection of weather data
<b>Defined in Task</b>
Task 5.1: Plant Digital Twins with ML/AI
<b>Short Description – incl. Purpose</b>
<p>The MET API Interface (MAI for short) is a user interface to FROST, the API developed by the Norwegian Meteorological Institute (MET), which enables public access to weather data. MAI is a software bundle written in Python and is intended to be used as a standard Python module.</p> <p>The purpose of MAI is to simplify data retrieval from FROST by providing a streamlined user interface. A user can select a location, a time interval, and a series of measurements, then MAI takes care of contacting the correct access point in FROST, submitting a properly composed request, as well as receiving and handling the response. MAI allows querying FROST for three main purposes: 1) Retrieve all available weather-station names in a given area (at municipality level); 2) retrieve a list of all available measurements at a selected location or municipality; 3) retrieve all data available for the selected measurements at a chosen location and time interval. The weather data is collected in a properly formatted Pandas DataFrame for ease of use.</p> <p>The modularity of MAI allows for flexible development and extension of its features.</p>
<b>Progress since last milestone</b>
<b>Examples of usage / illustrations</b>
<p>To allow MAI to access FROST, a user must first register and receive its client ID. This is done by visiting <a href="https://frost.met.no/auth/requestCredentials.html">https://frost.met.no/auth/requestCredentials.html</a> and registering with an email address. MET's API <a href="#">terms of use</a> as well the <a href="#">privacy statement</a> hold in this step. After registration, weather data can be retrieved by the user in one simple call to MAI. The figure below illustrates a minimal usage example.</p>
 <pre># Author: Filippo Remonato   import met_api_interface as mai  # Insert your own client ID here client_id = '&lt;your_personal_id_here&gt;' # Personal ID  location_name = 'oslo - blindern' # list of location names. Can use municipality='&lt;municipality_name&gt;' instead measurements = 'air_temperature' # list of measurement quantities to retrieve time_period = '2020-01-01/2020-01-02' # requested time period  df = mai.get_weather_data(client_id, time_period, name=location_name, measurements=measurements)</pre>
<p><i>Figure: Minimal usage example for retrieving weather data using MAI.</i></p>
<b>Interfaces (in/out) – system/user</b>
<p>The software is intended to be run as a standard Python module, imported in a script and run either in terminal or in a notebook. Most functions contained in MAI accept as input location names, list of measurements, and time intervals (or a combination of those). The returned output can be either messages on the screen containing the requested information, or a Pandas DataFrame containing the weather data, formatted with timestamps in rows and the different measurements in columns. Errors are handled through descriptive explanations and suggestions to the user.</p>
<b>Subordinates and platform dependencies</b>

MAI is available on any platform that can run Python 3.x.
<b>Licenses, etc. (free for use in the project)</b>
none
<b>TRL for overall component/tool and any parts/subordinates</b>
none
<b>References – incl. web etc.</b>
none
<b>To be considered in particular for the following COGNITWIN pilots</b>
Hydro.

<b>Component/Tool description</b>
<b>Component/Tool/Method/Framework/Service Name</b>
FUSE
<b>Defined in task</b>
5.1 (5.4)
<b>Short Description – incl. Purpose</b>
<p>FUSE is a tool for process input and state estimation, fusing plant physical model predictions with process measurements. The model and measurements are fused in nonlinear state estimation using an unscented Kalman filtering (UKF)-based approach.</p> <p>In particular, the tool is developed for estimation of combustion boiler input fuel composition characteristics. The tool uses a physical model of the CFB boiler hotloop (fluidization and combustion) as well as on-line measurements from the process (flue gas, furnace temperatures, etc). The approach can be applied for alternative state estimation purposes, given that a suitable plant model and measurements are provided. A generalized version is under development.</p> <p>The UKF algorithm is well known and many implementations are available (e.g. in Matlab Control System Toolbox and Matlab Central open exchange). The FUSE tool focuses on practical aspects: enabling the selection of states/inputs to estimate, measurement selection, data validation and reconciliation, physical model tuning, UKF tuning, and reduction of computational load, so as to support exploitation of computationally heavy physical models in plant operation and control.</p>
<b>Progress since last milestone</b>
<p>The tool has been developed (designed, implemented, and verified) after the last milestone in 2/2020. A paper has been published on the physical model tuning, available in IEEE Xplore (see References).</p>
<b>Examples of usage / illustrations</b>
<p>The tool originates from solving the WP3 pilot problem on fuel characterization, as a part of the heat exchanger fouling monitoring problem. The tool was tuned and tested using real full scale boiler plant design and measurement data.</p> <p>Figure 1 illustrates the estimation filter outcomes during fuel test experiments.</p>

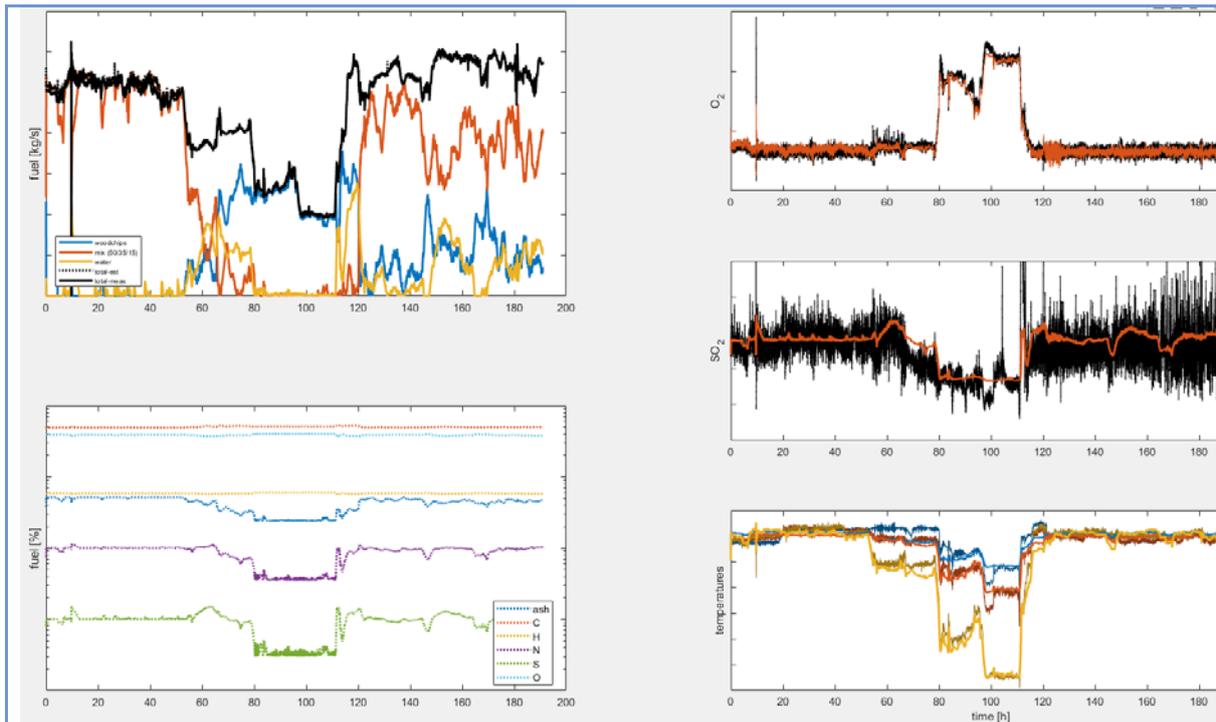


Figure: FUSE fuel characterization during one-week CFB operation.

The top left picture shows the estimated fuel characteristics as a function of time. The coloured lines show the mass flow feed for pure forest wood chips, design fuel mix, and moisture. The black lines show the estimated and measured total fuel feed (indistinguishable). The bottom left picture illustrates the elementary composition of the estimated fuel feed. The right plots show the measured and predicted flue gas oxygen and sulfur dioxide concentrations and furnace temperatures during an eight-day plant operation period. The estimated fuel feeds match with the feeds during five known test setup periods (3 to 8 hours), also performance outside of test periods appears feasible.

**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.

A link with StreamPipes is enabled by an OPC-UA client/server component (see FUSE OPC-UA tool).

**Subordinates and platform dependencies**

The tool is implemented using Matlab language (m-files). Matlab from the Mathworks is required (FUSE 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 such as 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)**

The FUSE code is free for use in the project (contact Enso.Ikonen@oulu.fi). The CFB hotloop physical model is Sumitomo SHI FW Energia Oy proprietary. The plant measurement data in its unprocessed form is proprietary of the pilot plant.

**TRL for overall component/tool and any parts/subordinates**

Current state is TRL 5 (validated in a relevant environment) currently being raised to TRL 6 (demonstrated in a relevant environment).

**References – incl. web etc.**

Ikonen & Selek (2020) Calibration of Physical Models with Process Data Using FIR filtering. Australian and New Zealand Control Conference, Gold Coast, pp-143-148.

The generalized Matlab-tool is available at <http://cc.oulu.fi/~iko/COGNITWIN/>

**To be considered in particular for the following COGNITWIN pilots**

Sumitomo SHI FW Energia Oy

<b>Component/Tool description</b>
<b>Component/Tool/Method/Framework/Service Name</b>
SubFUSE
<b>Defined in Task</b>
5.1 / 5.4
<b>Short Description – incl. Purpose</b>
<p>SubFUSE is a state estimation (soft-sensory) tool which fuses subspace methods for system identification with Kalman filtering. The tool is solely data-driven, it requires IO data (measurements) as input.</p> <p>In particular, the tool is developed for the estimation of input fuel characteristics of combustion-thermal power plants. Relying on regular process data including flue gas composition measurements, an estimate of the chemical structure of the fuel fed to the furnace of a CFB boiler is provided. The approach can be applied for alternative state estimation purposes as well, given that a suitable plant model can be generated and proper measurements are provided.</p> <p>The tool has been tested in a simulated environment which aims to replicate the dynamics of the pilot problem. Validation of the tool using data from pilot will be conducted in 2021, a generalized version will be developed based on validation results.</p> <p>The tool has been implemented in MATLAB, and is available in script format.</p>
<b>Progress since last milestone</b>
The tool has been developed (designed, implemented, and verified) after the last milestone in 2/2020.
<b>Examples of usage / illustrations</b>
<p>The tool target is solving the WP3 pilot problem on fuel characterization, as a part of the heat exchanger fouling monitoring problem. Using IO data pairs of the process of interest, the tool proceeds in two steps: first, a sufficient Linear Time-Invariant approximation of the governing dynamics is conducted utilizing subspace identification. Once the approximate dynamics is available, a standard Kalman filter is used for state estimation.</p> <p>For example, Figures 1 and 2 illustrate the performance of the tool in soft-sensing the chemical composition of the fuel fed to a CFB boiler. In the learning phase (figure 1) the tool learns to mimic the dynamics of combustion.</p>

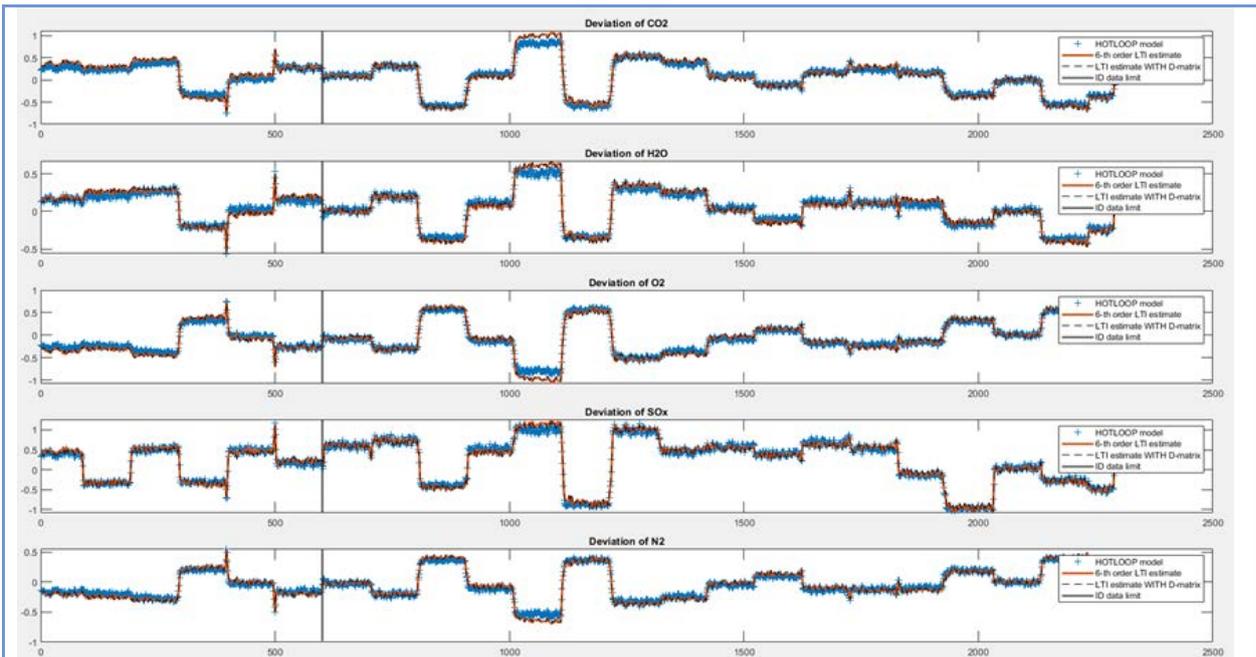


Figure: The identified Linear Time Invariant model (red) of a nonlinear combustion dynamic using IO measurement pairs (blue). Training data (left) are separated from the validation data (right) by a black line located at timestep 600.

Based on the IO relationship identified from data, the tool estimates (soft-senses) the chemical composition of the fuel using standard flue gas measurement data available at the power plant of interest (figure 2).

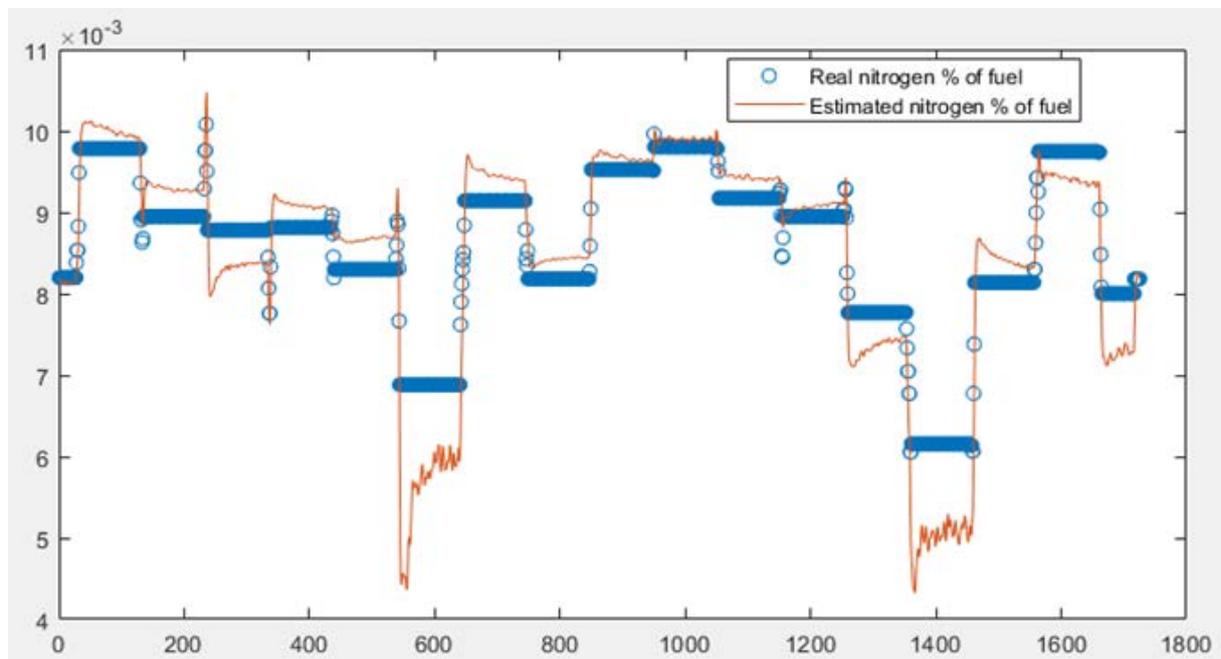


Figure: Actual (blue) and estimated (red) nitrogen content of the fuel used for heat generation in the combustion-thermal power plant of interest.

**Interfaces (in/out) – system/user**

The tool is implemented in MATLAB script (.m file). Input data (measurements) are provided as numerical vectors (matrices). Estimation outcomes are provided as numerical vectors (matrices).

A link with StreamPipes is enabled by an OPC-UA client/server component (see FUSE OPC-UA tool).

#### **Subordinates and platform dependencies**

The tool is implemented in MATLAB script (.m file). MATLAB, a product Mathworks is required to run the application. (SubFUSE has been tested on MATLAB version 2020b).

MATLAB (2020b) is available on all major operating systems, including Windows, Unix/Linux and MacOS. The tool uses the MATLAB core, additional toolboxes are not required. Open software such as 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)**

The SubFUSE code is free for use in the project (contact Istvan.Selek@oulu.fi).

#### **TRL for overall component/tool and any parts/subordinates**

Current state is TRL 4 (validated in lab) currently being raised to TRL 5 (validated in a relevant environment).

#### **References – incl. web etc.**

Istvan Selek (Istvan.Selek@oulu.fi)

Markus Neuvonen (Markus.Neuvonen@oulu.fi)

#### **To be considered in particular for the following COGNITWIN pilots**

Sumitomo SHI FW Energia Oy

<b>Component/Tool description</b>
<b>Component/Tool/Method/Framework/Service Name</b>
STEEL 4.0- TMLL Teknopar Machine Learning Library
<b>Defined in task</b>
T5.1, T5.2
<b>Short Description – incl. Purpose</b>
<p>STEEL 4.0- TMLL provides and enables the application of machine learning algorithms and needed to perform smart predictive maintenance for the SWP machinery. TMLL utilizes supervised, unsupervised, multidimensional scaling, and reinforcement learning algorithms as needed.</p> <p>To compare ML models used for predictive analysis, GUIs have been developed.</p>
<b>Progress since last milestone</b>
<p>Since the last milestone, frontend and backend software of TMLL were developed.</p> <p>Multiple machine learning algorithms have been applied to the data passing through the incremental PCA stage to detect anomalies. RF, Gradient boosted tree, LSTM, SVM, KNN, and MLP algorithms have been used.</p> <p>Unal, P., et.al. (2021) <i>“Data-driven Artificial Intelligence and Predictive Analytics for the Maintenance of Industrial Machinery Based on an Event Processing Platform”</i></p>
<b>Examples of usage / illustrations</b>
<b>Interfaces (in/out) – system/user</b>
IDBA data is used by TMLL as input. TMLL output is visualized by ICPV.
<b>Subordinates and platform dependencies</b>
None (platform independent web application)
<b>Licenses, etc. (free for use in the project)</b>
Proprietary/ Subject to license
<b>TRL for overall component/tool and any parts/subordinates</b>
The current TRL is 4 (validated in laboratory environment) running to be TRL 6.
<b>References – incl. web etc.</b>
none
<b>To be considered in particular for the following COGNITWIN pilots</b>
NOKSEL

<b>Component/Tool description</b>
<b>Component/Tool/Method/Framework/Service Name</b>
Neuroscope
<b>Defined in Task</b>
Task 5.2: Multi-variate Sensor analytics with Deep Learning
<b>Short Description – incl. Purpose</b>
<p>Neuroscope is a visual debugger for convolutional neural networks. The software is an interactive tool with a graphical user interface intended for interactive use by data scientists on the application level.</p> <p>The purpose of the Neuroscope software is to allow data scientists to gain insight into the inner workings of a neural network, in the case of a system malfunction or misbehavior. The approach taken by Neuroscope is a network visualization approach, which means that the weights of the inner layers are visualized in a human-interpretable way, which is helpful for understanding why a certain misbehavior occurred. In the context of COGNITWIN, the Neuroscope software will be useful as part of a software toolbox to make machine learning technology practically useable.</p> <p>The software supports the following major features: (1) visualization of network architectures loaded from PyTorch or TensorFlow files as graph representation, visualization of trained weights by means of (2) activation maps, (3) saliency map, (4) guided back propagation, (5) grad-CAM, (6) guided Grad-CAM, and (7) grad-CAM plus.</p> <p>The software supports arbitrary networks architectures for classification and semantic segmentation of image-like data and is currently being extended to multi-object detection and localization architectures.</p> <p>Neuroscope is comparable to systems like Tensorboard. The major difference is the support of exchangeable machine learning backends (TensorFlow and PyTorch), and the interactive use via a graphical user interface.</p>
<b>Progress since last milestone</b>
<p>Since the last milestone, we added the following features to the project: (1) support for semantic segmentation network architectures, and (2) support for guided Grad-CAM. We are currently raising the technology readiness level to TL 7 by bug fixing and implementing smaller improvements. We are currently implementing (3) support for multi-object detection and localization architectures.</p> <p>We are currently writing a publication concerning the software:</p> <p>Schorr, C., Godarzi, P., Chen, F., Dahmen, T. (2021) <i>Neuroscope - An explainable AI toolbox for semantic segmentation and classification of deep neural nets</i>, in: Applied Sciences, Special issue on explainable artificial intelligence (manuscript under review).</p>
<b>Examples of usage / illustrations</b>
<p>The example synthetic image of billet rolling process illustrates the capability of Neuroscope to analyze a given neural network (top left). The task is to segment the image regarding the class “billet” as a component of automatic optical detection and tracking process. Using a saliency metric, a diffuse image of pixels sensitive to the class “billet” is computed (top right). The second visualization method called guided Grad-CAM shows clear regions of high activation in places of billets, as well as some localized areas of low activation (bottom left). The activation map computed by Guided Backpropagation method (bottom right) highlights pixels of high activation</p>

exactly at billet locations. The analysis of these 3 types of visualization maps raises the question of why the activation maps do not show high activated pixels monotonously within the bounds of billets. This observation could be the indicator of a poorly trained deep learning model or scarce training data.

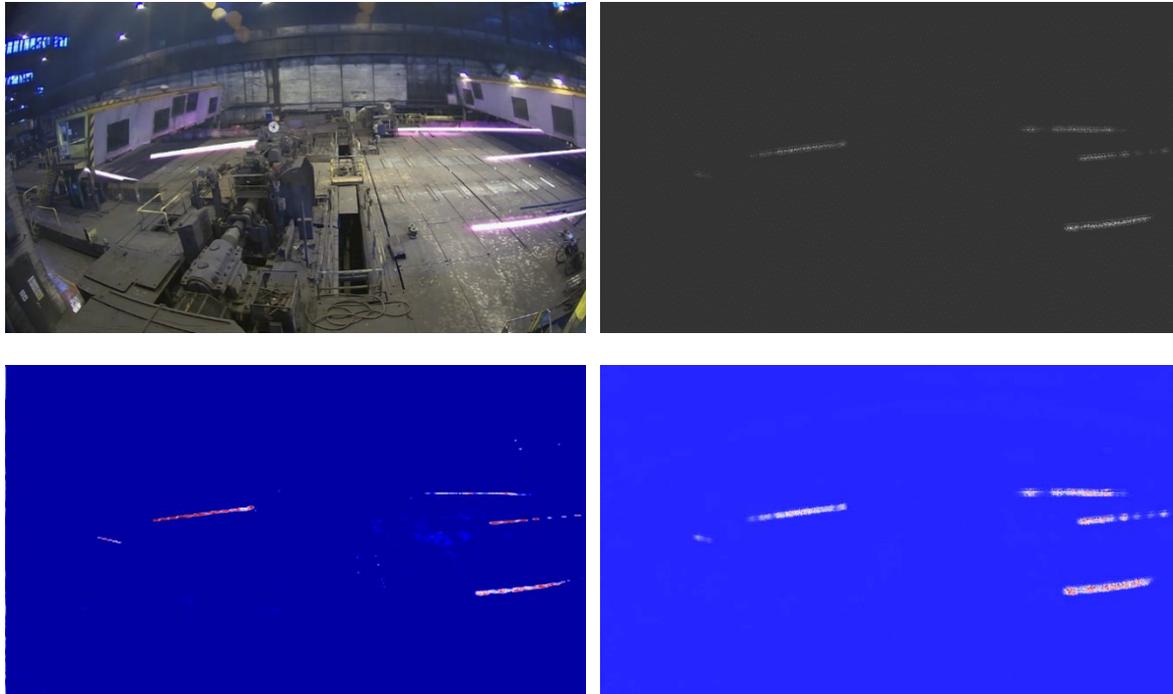


Figure: Visualization of the class “billet” using Neuroscope: top left: original image, top right: saliency map, bottom left: guided Grad-CAM, bottom right: guided backpropagation.

**Interfaces (in/out) – system/user**

The system features a graphical user interface and is intended for interactive use only. The system can load network architectures and weights in TensorFlow and PyTorch format. It is able to export visualization results in common image formats.

**Subordinates and platform dependencies**

Neuroscope is available for Linux and Windows. It supports PyTorch and TensorFlow as backends.

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

We provide a community version under GPL. For commercial licenses, contact DFKI directly (Tim.Dahmen@dfki.de).

**TRL for overall component/tool and any parts/subordinates**

6, currently being raised to 7

**References – incl. web etc.**

<https://github.com/c3di/neuroscope>

**To be considered in particular for the following COGNITWIN pilots**

Saarstahl.

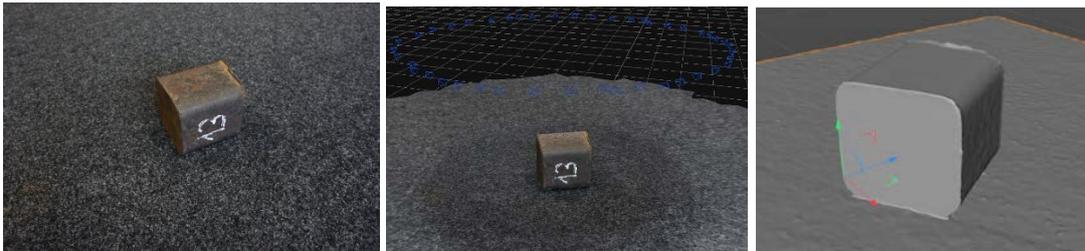
<b>Component/Tool description</b>
<b>Component/Tool/Method/Framework/Service Name</b>
Aerial Photogrammetry
<b>Defined in Task</b>
Task 5.2: Multi-variate Sensor analytics with Deep Learning
<b>Short Description – incl. Purpose</b>
<p>Photogrammetry is a technique for the capturing of three-dimensional (3D) models from real world environments. The technique works by taking images from manually selected positions and reconstructing a 3D model from the images. As the camera parameters (position and orientation) are unknown, the most important step is to computationally determine position and orientation of each image.</p> <p>Photogrammetry has a reduced surface precision compared to laser-based surface scanners, but has the immense advantage that it generates textured models in a single step, i.e. the generated models can immediately be used for photorealistic rendering. Another advantage is that photogrammetry is applicable to a wide range of length scales. If combined with macro photography, photogrammetry can capture details on the micrometer scale, if combined with handheld cameras it can produce centimeter scale objects, and combined with drones (aerial photogrammetry), the capture area can extend over several square kilometers.</p> <p>In the course of this project, we will provide a process for aerial photogrammetry which allows the capturing of entire sections of production plants, such as the Saarstahl milling plant. This workflow will be assembled from commercially available components but adapted to the specific needs if being used in a large indoor-environment with harsh production conditions. The purpose of the captured 3D models is the generation of training data for machine learning applications.</p>
<b>Progress since last milestone</b>
<p>A number of datasets was captured using different flight patterns from a drone and using a handheld and tripod-mounted camera system. Reconstruction results were compared to determine the optimal capturing mode.</p> <p>A commercially available photogrammetry software (Agisoft Metashape) was purchased, installed on suitable server hardware, and reconstruction settings were optimized to work with the available datasets.</p>
<b>Examples of usage / illustrations</b>
<p>The first example depicts a photogrammetric model reconstruction of Saarstahl’s steel billet chunk with dimensions 10x10x10 cm. The chunk was shot from different viewpoints by a handheld camera. 44 unprocessed photographs were used as an input for Agisoft Metashape software.</p>


Figure: Photogrammetric reconstruction of Saarstahl’s billet chunk. Left: one of 44 original photographs of the billet chunk. Middle: a viewport of Agisoft Metashape with indicated camera positions. Right: a reconstructed untextured 3D model.

The second example is a usage of photogrammetry for synthesizing training data for deep learning applications. In particular, in the course of the project we explore the ability of using synthetically generated images for training billet tracking models. To begin the simulation process, a large number of high-resolution overlapping photos was taken over the area of Saarstahl’s blooming train. Here, we conducted image capturing by a drone as well as by a tripod-mounted camera. Using Agisoft Metashape software, we reconstructed the 3D shape of the scenery, which replicates the real environment in correct dimension proportions (Figure top). Next, the 3D model of the blooming train was repaired and cleaned up. After 3D parametric models of billets were inserted into the scene, the wide-angle lens distortion was applied (Figure, bottom left). To simulate changes in scenery and lightning conditions, the 3D setting was composed with original footage captured by surveillance cameras (Figure, bottom middle). Finally, we rendered images paired with billet segmentation masks (Figure, bottom right) which are used as a training data for billet detection and tracking models.

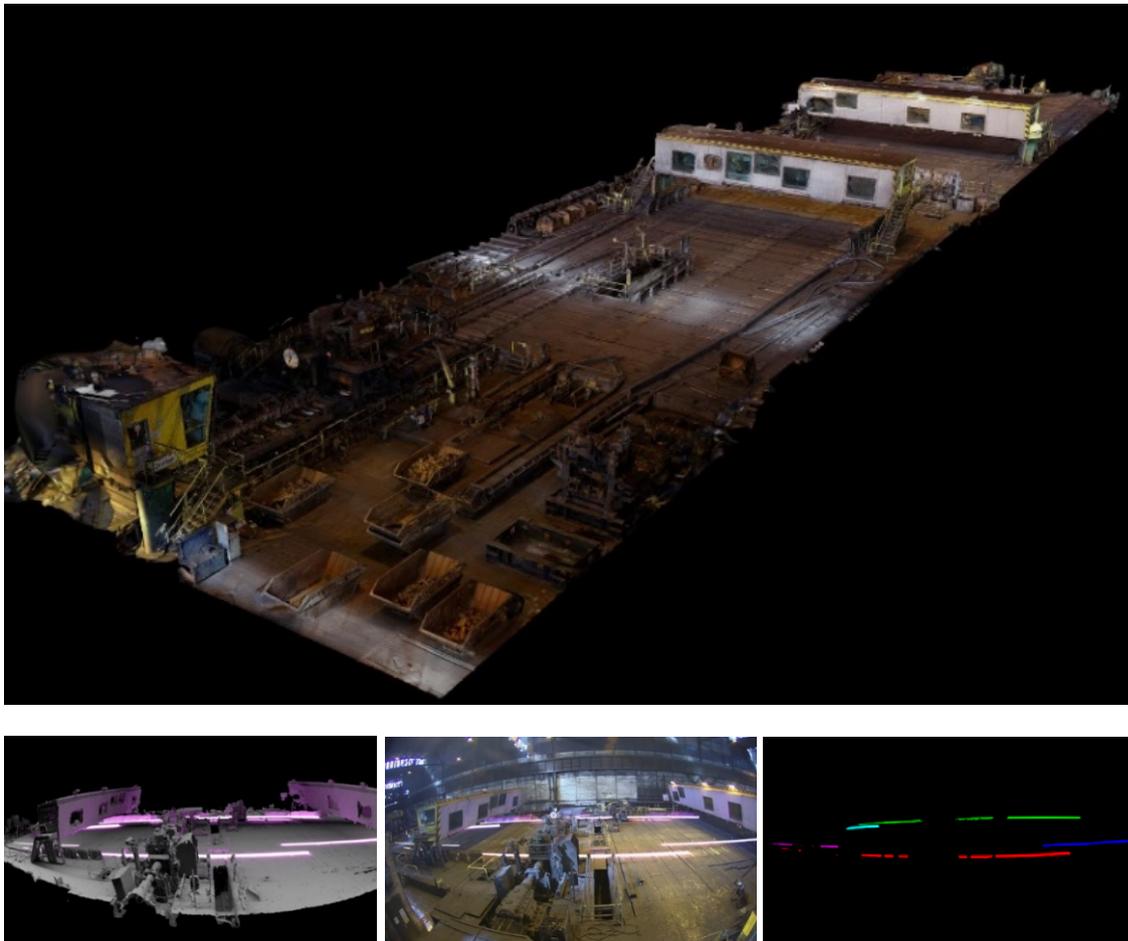


Figure: Process of synthesizing training data. Top: a textured 3D model of Saarstahl’s blooming train. Bottom left: a 3D setting of billet rolling process. Bottom middle: a generated image. Bottom right: a generated instance segmentation mask.

**Interfaces (in/out) – system/user**

Agisoft Metashape software is a stand-alone software that performs photogrammetric processing of digital photographs. It loads arbitrary number of unprocessed images in common formats and generates a 3D model of a scenery. Here it is important to capture high-quality and highly overlapping images of the setting to compute the best quality model. It is recommended to employ a camera with 5Mpx resolution at least and to shoot photographs with 80% overlap.

**Subordinates and platform dependencies**

Recommended configurations for Agisoft Metashape:

- Windows 7 SP 1 or later (64 bit), Mac OS X Mountain Lion or later, Debian/Ubuntu with GLIBC 2.13+(64 bit)
- Intel Core i7 or AMD Ryzen 7 processor
- Discrete NVIDIA or AMD GPU
- 32 GB of RAM

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

The component is realized as a non-disclosed inhouse workflow.

**TRL for overall component/tool and any parts/subordinates**

7

**References – incl. web etc.**

<https://www.software3d.de/agisoft-metashape-pro>

**To be considered in particular for the following COGNITWIN pilots**

Saarstahl.

Component/Tool description
<b>Component/Tool/Method/Framework/Service Name</b>
Bonzai
<b>Defined in Task</b>
Task 5.3: Deep Learning Performance
<b>Short Description – incl. Purpose</b>
<p>Bonzai is Scortex python library handling everything related to deep learning on images.</p> <p>Bonzai is built on top of Keras / Tensorflow. It uses as input a connection to a mongo database for annotations and meta-information (dates, part reference, acquisition system version, ...), as well as an azure filesystem for image storage. The main output is the production of deep learning model in tf.keras format (topology in .json and weights in .h5).</p> <p>With this library, Scortex engineers manipulate and clean images and their metadata. They use it to train deep learning models and properly evaluate these models.</p>
<b>Progress since last milestone</b>
<p>Scortex has greatly improved its machine learning library.</p> <p>A large focus was put on the maintainability and traceability of the deployed systems. From any deployed model, Scortex is able to retrace which images, metadata, and preprocessing was used to train it.</p> <p>Scortex extended its library to user use cases as well. Previously, only defect detection was supported. Now, the library can handle part detection/segmentation as well as anomaly detection.</p> <p>Related to inference speed:</p> <p>Scortex worked on a way to improve all models inference time and published a blog post about it: <a href="https://scortex.io/batch-norm-folding-an-easy-way-to-improve-your-network-speed/">https://scortex.io/batch-norm-folding-an-easy-way-to-improve-your-network-speed/</a>. This technology is now deployed at some of Scortex' clients.</p> <p>Scortex worked on improving the architecture speed of the network they are using at their clients. Typically using a smaller / shallower network. The difficulty of this is to maintain good robustness (mostly for repeatability. See this other blog post: <a href="https://scortex.io/robustness-and-repeatability-of-modern-deep-neural-networks-a-review/">https://scortex.io/robustness-and-repeatability-of-modern-deep-neural-networks-a-review/</a>).</p> <p>Scortex devised real time / light architectures for the task of detection / semantic segmentation as well as anomaly detection.</p> <p>Scortex has investigated pruning networks but that did not provide good results for inference time, as most software / hardware (example: GPU + TensorFlow) do not support leverage sparsity. It is our hope FPGA technology will be able to do so.</p> <p>“Distillation”: Scortex successfully managed to transfer knowledge from a large network to a smaller one. The performance is not as good as the large model but better than the performance of the smaller model trained on its own.</p> <p>By combining, Scortex managed to deploy a station capable of a complex inspection of rotating parts. The Scortex box handles the inspection of 3 parts per second, which requires inference of</p>

300 (3 x 100) 1280x640 grayscale images per second. To the best of our knowledge, Scortex is the only company able to achieve such performances in a real-life deployment.

Some experiments such as “Pruning” or “distillation” are not yet in production as the benefits did not outweigh the implementation cost as of today.

Scortex is also working on improving training speed as shown by this blog post:

<https://scortex.io/extending-selective-back-propagation-to-segmentation-focus-biggest-losers/>.

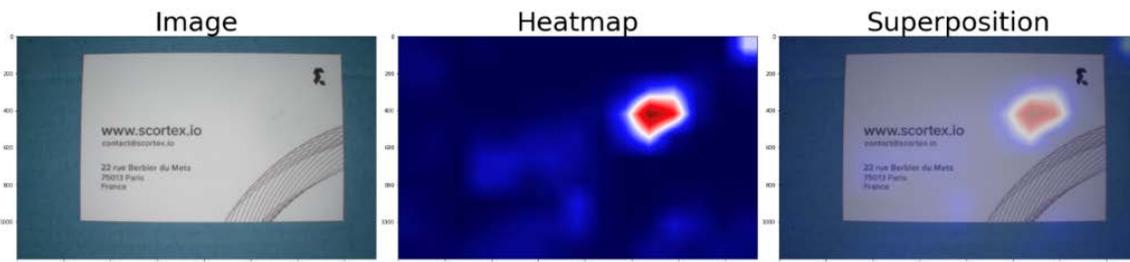
Its new unsupervised demonstrator allows one second training for a very constrained set up.

### Examples of usage / illustrations

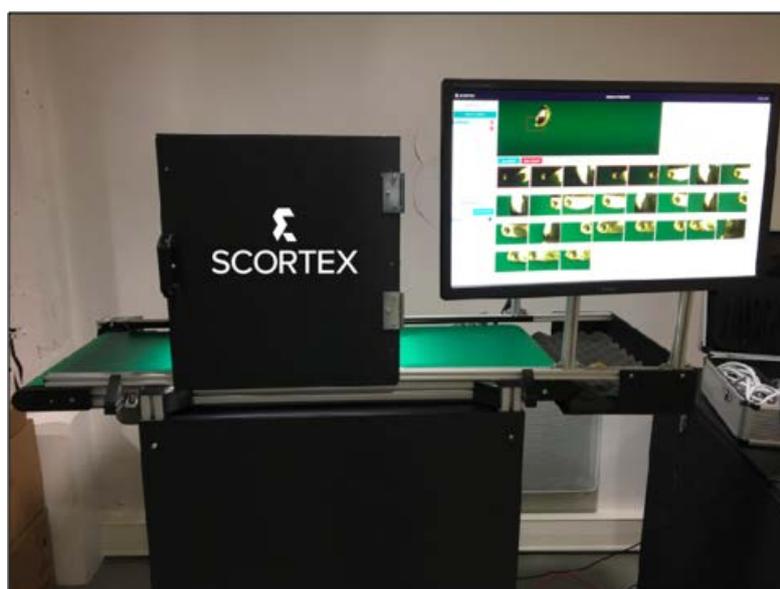
Below is an example image from our unsupervised anomaly detection demonstrator.

The user can train a model with a few un-annotated images and the model will detect anomalies.

The result shows the original image with a defect score and a defect localization. Here it detects a tiny pen mark on the business card the model was trained on.

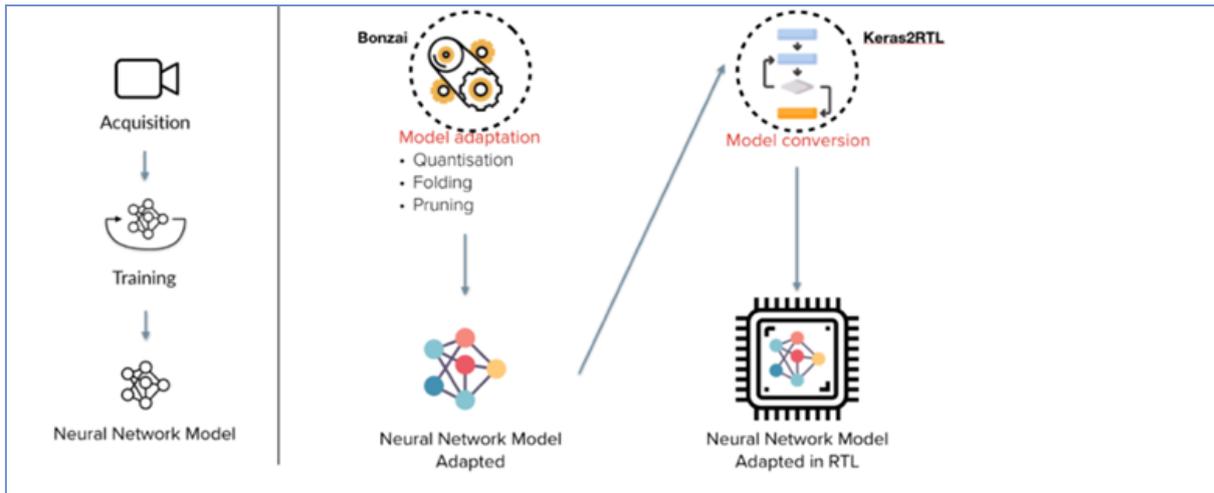


Example of detections on our supervised demonstrator (less constrained). The part goes on the conveyor belt. Inside the Scortex dark “box” there are 2 cameras filming continuously 1920x1200 colored images. One report is created per part. The (defect) detection is shown with closeups on the bottom of the screen.



There is currently no frontend to train supervised models using only the mouse but we are hoping to be able to work on this in 2021.
<b>Interfaces (in/out) – system/user</b>
Bonzai connects to mongodb and other database systems to get images and meta-data (e.g.: annotations). Outputs are deep learning Keras/Tensorflow models and pipelines to be used in productions.
<b>Subordinates and platform dependencies</b>
keras/tensorflow, mongodb.
<b>Licenses, etc. (free for use in the project)</b>
Proprietary. In development, remains the property of Scortex. Will be used by Scortex exclusively.
<b>TRL for overall component/tool and any parts/subordinates</b>
TRL 7
<b>References – incl. web etc.</b>
<a href="https://scortex.io/batch-norm-folding-an-easy-way-to-improve-your-network-speed/">https://scortex.io/batch-norm-folding-an-easy-way-to-improve-your-network-speed/</a>
<a href="https://scortex.io/robustness-and-repeatability-of-modern-deep-neural-networks-a-review/">https://scortex.io/robustness-and-repeatability-of-modern-deep-neural-networks-a-review/</a>
<a href="https://scortex.io/extending-selective-back-propagation-to-segmentation-focus-biggest-losers/">https://scortex.io/extending-selective-back-propagation-to-segmentation-focus-biggest-losers/</a>
<b>To be considered in particular for the following COGNITWIN pilots</b>
Saarstahl, Sumitomo

Component/Tool description
Component/Tool/Method/Framework/Service Name
Machine learning to FPGA conversion: Keras2RTL
Defined in Task
Task 5.3: Deep learning performance
Short Description – incl. Purpose
<p>This tool allows to fill the gap between machine / deep learning development environments and FPGA development environments.</p> <p>Machine learning development environments are typically based on the python programming language and libraries such as Keras and Tensorflow (or pytorch) relying on the Nvidia low level computing library named CUDA.</p> <p>Meanwhile, FPGA developments are based on VHDL programming language and integrated development environments (IDE) provided by the company providing the FPGA component. For a Xilinx Virtex Ultrascale+ VU9P, the IDE is named <a href="#">Vivado</a>.</p> <p>Keras2RTL takes as input a tf.keras model that is a file with .h5 extension which contains the topology of the machine learning neural network and the weights. At Scortex, the Keras model is generated using the bonzai library (see other 5.3 components).</p> <p>Keras2RTL converts the topology from the .h5 file and generates the VHDL configuration files. These files will be used by Vivado to generate Honir.</p>
Progress since last milestone
<p>This task is currently handled manually by following a defined process to verify the corner cases and technically cover 100% of the scope to unlock the automation of this process.</p> <p>Its 100% automation will be part of the next steps.</p>
Examples of usage / illustrations
<p>The overall usage flow is:</p> <ul style="list-style-type: none"> <li>- Train a tf.keras / keras model (typically using bonzai).</li> <li>- Quantize the model (or train it in a quantized way immediately).</li> <li>- Convert it using keras2RTL.</li> </ul> <p>Use this model to run inference on images using the Honir component.</p>



As a demonstrator, Scortex trained a Keras model on data from one of its demonstrators. The model allows defect detection on common goods parts, such as Lego bricks, electrical switches and door handles.

This deep learning model was quantized using the bonzai tool (see other component) and then processed by Keras2RTL process manually. The VHDL files produced was used for the configuration of the inference engine (Honir: in task 4.4).

In a more general manner, the tool should be used to allow automatic and fast conversion from a Keras machine learning model to a VHDL config file that can be used for Honir creation.

**Interfaces (in/out) – system/user**

At a user level (a command line tool)  
 IN : keras .h5 file (topology + weights)  
 OUT : VHDL config file (for Honir tool build in T4.4)

**Subordinates and platform dependencies**

This module can work in standalone. It is, however, necessary for Honir (inference engine) to work properly. Today the tool will support only keras models.

**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.h5py.org/>
- [https://www.tensorflow.org/guide/keras/save\\_and\\_serialize](https://www.tensorflow.org/guide/keras/save_and_serialize)
- <https://github.com/keras-team/keras>

**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 Saarstahl use case. In which case, keras2TL will be used to generate Honir configuration.
- But it could be extended to any other pilots running deep learning on images.

<b>Component/Tool description</b>
<b>Component/Tool/Method/Framework/Service Name</b>
TMat SynDat MATLAB Synthetic Data Generator
<b>Defined in task</b>
T5.4
<b>Short Description – incl. Purpose</b>
<p>TMat SynDat (MATLAB Synthetic Data Generator for Electro Mechanical parts) is a synthetic data generator for common electro-mechanical parts, including electric DC motors and a hydraulic shaft.</p> <p>The purpose of TMat SynDat is to generate synthetic of fault data from the model elements. Hence, the generation of such data enables users to have a model-driven digital twin for a common DC motor, gearbox, and hydraulic shaft and their associated components.</p> <p>TMat SynDat utilizes several 1<sup>st</sup> order models: DC motor and gearbox models, and a hydraulic press model. For both of these models, random sources of errors (degradation of the components) are introduced. A load representative to what they may experience in the real world is then applied. Virtual sensors will collect data for several specific degradation scenarios. The outcome is a supervised and annotated dataset. The latter will be used in training a ML classifier. The classifier will be used to monitor the condition of the machine in operation and provide early warning for potential fault.</p> <p>In the context of COGNITWIN, the TMat SynDat output will be useful in conducting the predictive maintenance of the modelled elements.</p>
<b>Progress since last milestone</b>
<p>Since the last milestone, sensors have been added to the 1<sup>st</sup> order principal models. Current and temperature sensors have been added to the motor and gearbox model, while a hydraulic press sensor has been added to the previously developed hydraulic press model.</p> <p>Following the sensor implementations for the models, model parameters have been calibrated and the random error sources have been introduced to the model. Thus, the model has been updated to be ready for predictive maintenance algorithms by introducing sources of random errors to be used in predictive maintenance, adding appropriate sensors to observe the effect of these error sources on important variables of the models and calibrating the geometrical, electrical, and hydraulic parameters of components to make the model as applicable and as realistic as possible.</p>
<b>Examples of usage / illustrations</b>
MATLAB Simulink is used to develop and update the models.
<b>Interfaces (in/out) – system/user</b>
TMat SynDat generates output data in .mat format and/or .csv format.
<b>Subordinates and platform dependencies</b>
TMat SynDat works with MATLAB Simulink and it converted to operate a standalone executable program.
<b>Licenses, etc. (free for use in the project)</b>

TBD
<b>TRL for overall component/tool and any parts/subordinates</b>
The current TRL is 4 (validated in laboratory environment) running to be TRL 5.
<b>References – incl. web etc.</b>
none
<b>To be considered in particular for the following COGNITWIN pilots</b>
NOKSEL

<b>Component/Tool description</b>
<b>Component/Tool/Method/Framework/Service Name</b>
Pragmatism in physics-based modeling (PPBM)
<b>Defined in Task</b>
Task 5.4: Hybrid Digital Twins
<b>Short Description – incl. Purpose</b>
<p>"Pragmatism in physics-based modeling" (PPBM) is a method/framework for developing physics-based mathematical models. Such models may serve as a digital twin alone or as companion with data-based AI/ML methods, to form hybrid digital twins to exploit the combination of data and physics-based modeling.</p> <p>The purpose of PPBM is to devise a generic methodology for development of physics-based models for application in digital twins. The PPBM provides a recipe for attacking a digital twin development, starting out from problem definition, information collection, including exact definitions of the output requirements for the model, assembling a system architects team, model specification, use of sub-level empirical or computed data, model building and application. The PPBM can only be developed further through application in industrial cases, like the Sidenor pilot case.</p> <p>The physics, chemistry, and numerical methods to be used may differ between applications, but PPBM should help the developers (system architects) to run through a set of well-defined steps on the way from problem definition to final application. For each new application using the PPBM, new learning must be extracted and reported (published).</p> <p>In the Sidenor pilot case offline data is used. The data has multiple challenges. We now explore extending the PPBM to applying a PPBM based application to clean the available data, but also provide additional simulated data. These combined data will be further be explored in a hybrid approach.</p>
<b>Progress since last milestone</b>
<p>Since the last milestone, we extended the method/framework through development of a physics-based model for the Sidenor pilot. Following elements have been in focus:</p> <ul style="list-style-type: none"> <li>• i) Methods to assure the industrial challenge is fully understood, ii) Define the problem in a problem document. Consolidation with the industrial partner. Defining and collecting the needed information, iii) Process to agree on model requirements, iv) Writing model specification, v) Model implementation and adaptation, vi) Model verification</li> </ul> <p>The Sidenor pilot is a good case for further development of the pragmatism method. The models are defined to explain the thermal evolution of a steel ladle during its lifetime and predict the refractory erosion from heat to heat. In order to handle the major and complex physics and thermodynamics (multiphase flows, slag heater, radiation, thermal stress erosion, interface waves, local heat and mass transfer, chemical equilibrium of complex metal-slag systems, dissolution of refractory components) several simplifications are introduced, some based on using Computational Fluid Dynamics (CFD) to create data that can be applied in the simpler and faster model.</p> <p>In this pilot we apply and extend the PPBM to be implemented in the Python framework where many tools available for Python may be explored when hybridization will be introduced.</p>

The work, involving extensions of the PPBM, will be published under "Pragmatism in industrial modeling, applied to ladle life in the steel industry". A particularly important contribution is to show how such a model can reveal the goodness of the industrial data. This is critical as industrial data, for many good reasons, may be completely incorrect, and it may in some cases be impossible to build automated filters that can fix the problem.

**Examples of usage / illustrations**

In the Sidenor pilot, applying and extending the PPBM method, we have developed a model for the ladle refractory erosion. The thermal model is now possible to operate and an example is shown below. The erosion models are under development.

We can see that the steel level was surprisingly low in this case. This was found to be an error in the data. However, the thermal model alone demonstrates that different erosion levels on the inside of the ladle will lead to different dynamics surface temperatures. In this case it supports that combining PPBM, thermal imaging and ML/AI a route to keep track on ladle life is possible.

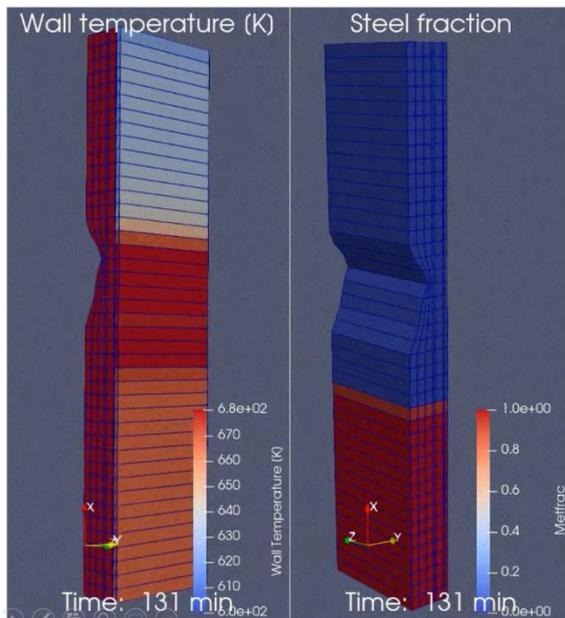


Figure 1: We see the outer surface temperature at time 131 min after metal was filled into the ladle. The ladle was initially eroded.

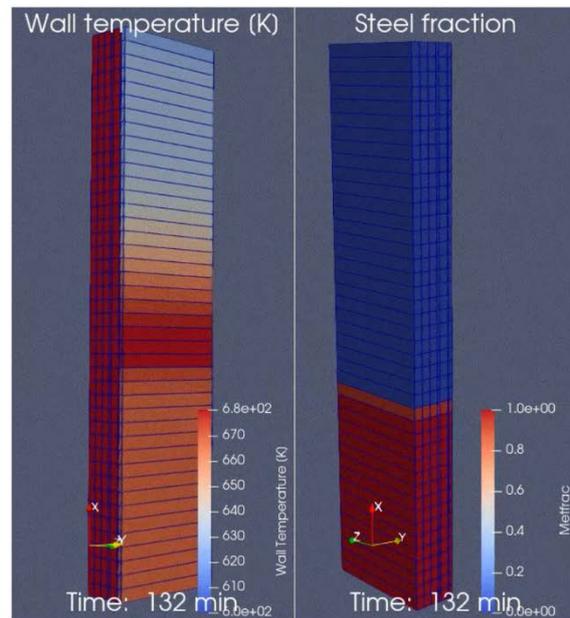


Figure 2: Same case as in Figure A, but ladle has no initial erosion. The outer surface temperature of the ladle is significantly modified.

**Interfaces (in/out) – system/user**

The Sidenor application of the PPBM is currently reading data in ascii format (\*.csv, \*.xls) and output is saved as files in similar formats. In addition, data can be saved in the VTK \*.vts format. These files can be loaded into the Paraview (<https://www.paraview.org/>) tool for 3D visualization of the results.

**Subordinates and platform dependencies**

The Sidenor model is using Python 3 and standard Python libraries. It can be run on both Windows and Linux systems.

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

The model will be open (free use) and at the end of the project to be published at <https://github.com>.

**TRL for overall component/tool and any parts/subordinates**

Currently it is TRL-3. As this belongs to the class of methods it may be developed into a technical standard at a later stage. [https://en.wikipedia.org/wiki/Technical\\_standard](https://en.wikipedia.org/wiki/Technical_standard)

**References – incl. web etc.**

[ZOR+15] J. Zoric, S. T. Johansen, K. E. Einarsrud, and A. Solheim, ‘ON PRAGMATISM IN INDUSTRIAL MODELING’, Progress in Applied CFD, Selected papers from 10th International Conference on Computational Fluid Dynamics in the Oil & Gas, Metallurgical and Process Industries, vol. 1, pp. 9–24, 2015. Available: <https://www.sintefbok.no/book/download/1038>

[ZOR+15b] J. Zoric et al., ‘On Pragmatism in industrial modeling - Part II: Workflows and associated data and metadata’, Melbourne, Australia, 7-9 December, 2015, 2015, p. 7 pages, [Online]. Available: [http://www.cfd.com.au/cfd\\_conf15/PDFs/032JOH.pdf](http://www.cfd.com.au/cfd_conf15/PDFs/032JOH.pdf) .

[JOH+17] S. T. Johansen, E. A. Meese, J. Zoric, A. Islam, and D. W. Martins, ‘On Pragmatism in Industrial Modeling, Part III: Application to Operational Drilling’, in Progress in Applied CFD – CFD2017 Selected papers from 12th International Conference on Computational Fluid Dynamics in the Oil & Gas, Metallurgical and Process Industries, Trondheim, 2017, p. 11, [Online]. Available: <http://hdl.handle.net/11250/2465068> .

**To be considered in particular for the following COGNITWIN pilots**

Sidenor and Sumitomo.

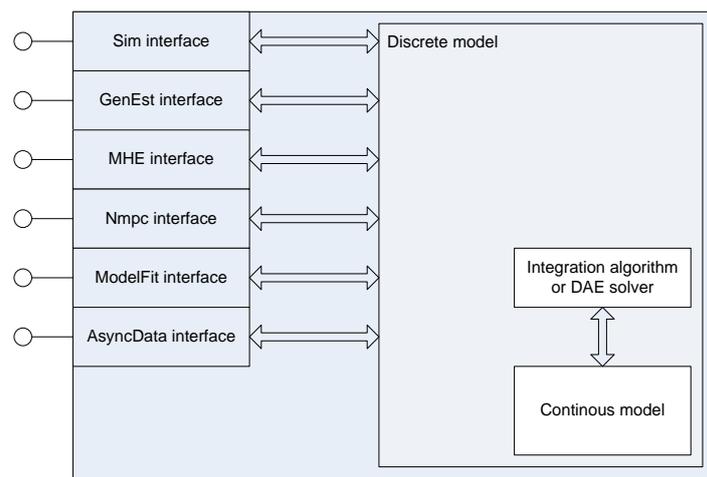
In addition, the Sidenor application may be exploited by COGNITWIN partners Elkem and Saarstahl.

Component/Tool description
<b>Component/Tool/Method/Framework/Service Name</b>
Cybernetica CENIT
<b>Defined in Task</b>
5.4 Hybrid Digital Twins
<b>Short Description – incl. Purpose</b>
<p>Cybernetica CENIT is a tool for online estimation and nonlinear model predictive control. It can be used as both a soft sensing application and a control application.</p> <p>Model Predictive control is an advanced control method where a mathematical model of the process is used to predict future behavior. The predictions from the model are used in a mathematical optimization algorithm that calculates the optimal process inputs in order to achieve optimal future behavior of selected variables in the process. Constraints and setpoints may be imposed both on the manipulated process inputs variables and the controlled process output variables. Model predictive control also has the advantage that couplings between variables in the process are taken into account.</p>
<b>Progress since last milestone</b>
<p>Cybernetica CENIT has been extended with application modules for the Elkem and Hydro pilot processes. These modules contain physics-based models of the processes and make it possible to run online state and parameter estimation, as well as implement soft sensing and nonlinear model predictive control applications.</p> <p>Cybernetica CENIT has further been extended with routines for validation of the input data and its own calculation results. This includes new interface routines for the application components. This extension forms a basis for the development of the proposed “Cognitive CENIT” tool.</p>
<b>Examples of usage / illustrations</b>
<p>Main components of Cybernetica CENIT:</p> <pre> graph TD     PM([Process model]) &lt;--&gt; CK[Cenit Kernel]     CK &lt;--&gt; TCP/IP  CMMI[Cenit MMI]     CK &lt;--&gt; OPC  CS[Control System]     DB[Database] --&gt; CK     CK -.-&gt; DB     DB --&gt; OA[Offline analysis]     </pre> <p>Cybernetica CENIT consists of a generic part and an application-specific part, namely the process model. A Cybernetica CENIT application is defined as Cybernetica CENIT and some process model together.</p> <p>The following table describes the main components of a Cybernetica CENIT application:</p>

Component	Purpose
CenitKernel	This is the main component of Cybernetica CENIT. It implements communication with the process control system and the calculation algorithms (estimator and nonlinear model predictive controller).
CenitMMI	This is an engineering interface used to configure and supervise CenitKernel, mainly during the engineering phase of the project. The operators interface is normally integrated in the existing DCS interface.
Process model	This is the application-specific part of a Cybernetica CENIT application. It implements a mathematical representation of the process that is controlled.
Database	An optional database for logging parameters and calculated data from CenitKernel. The data is used both by CenitMMI and for offline data analysis, and can be used to trend inputs, states and other calculated values.
Control system	This is the process control system (DCS/ PLC), which handles the low-level communication with the process. This system is not a part of Cybernetica CENIT and should implement an OPC server on a standard form to handle the communication with CenitKernel. Both OPC Classic and OPC UA interfaces are supported by Cenit. The communication includes process measurements, manipulated variables and possibly other variables as well.

The model component is implemented as a Microsoft Windows dynamic link library (DLL). One or more model interfaces can be implemented in such a DLL, depending on which calculation modules shall be used. It is not necessary to implement unused interfaces.

The interfaces do not depend on each other, and it is possible to implement different models for each interface, i.e., a complex model for the simulator interface and a simpler model for the controller. However, it is quite common to implement the same model for all the interfaces. The figure below shows how to do this. In this figure, there is a common inner model code base for all the interfaces:



The available interfaces are:

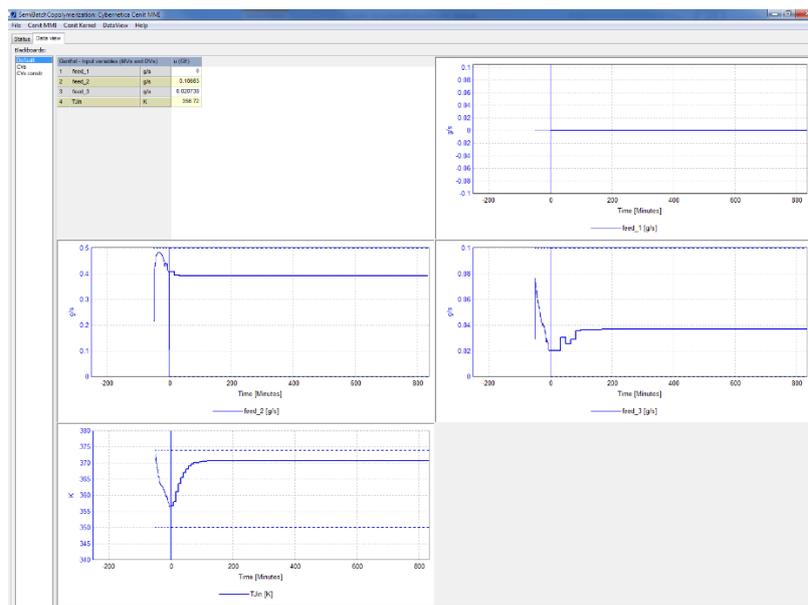
- Sim interface: Used to simulate the process.
- GenEst interface: Used by the Kalman Filter.
- MHE interface: Used by the Moving Horizon Estimator.

- Nmpc interface: Used by the non-linear predictive controller
- ModelFit interface: Used by Cybernetica ModelFit.
- AsyncData interface: Used by Cybernetica Cenit to handle input data that requires special handling; e.g. registration of process event data.

**Interfaces (in/out) – system/user**

Data can be presented to the user by using Cybernetica CenitMMI, or extracted from the database using the included tool getdbdata.

Example of CenitMMI displaying some historical trend and prediction plots for some manipulated variables:



**Subordinates and platform dependencies**

May use PostgreSQL database.

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

Cybernetica Cenit 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.

**TRL for overall component/tool and any parts/subordinates**

9 - Commercial product.

**References – incl. web etc.**

<http://cybernetica.no/technology/model-predictive-control/>

**To be considered in particular for the following COGNITWIN pilots**

Hydro, Elkem.

**Component/Tool description**

**Component/Tool/Method/Framework/Service Name**

Cybernetica ProXim

**Defined in Task**

5.4 Hybrid Digital Twins

**Short Description – incl. Purpose**

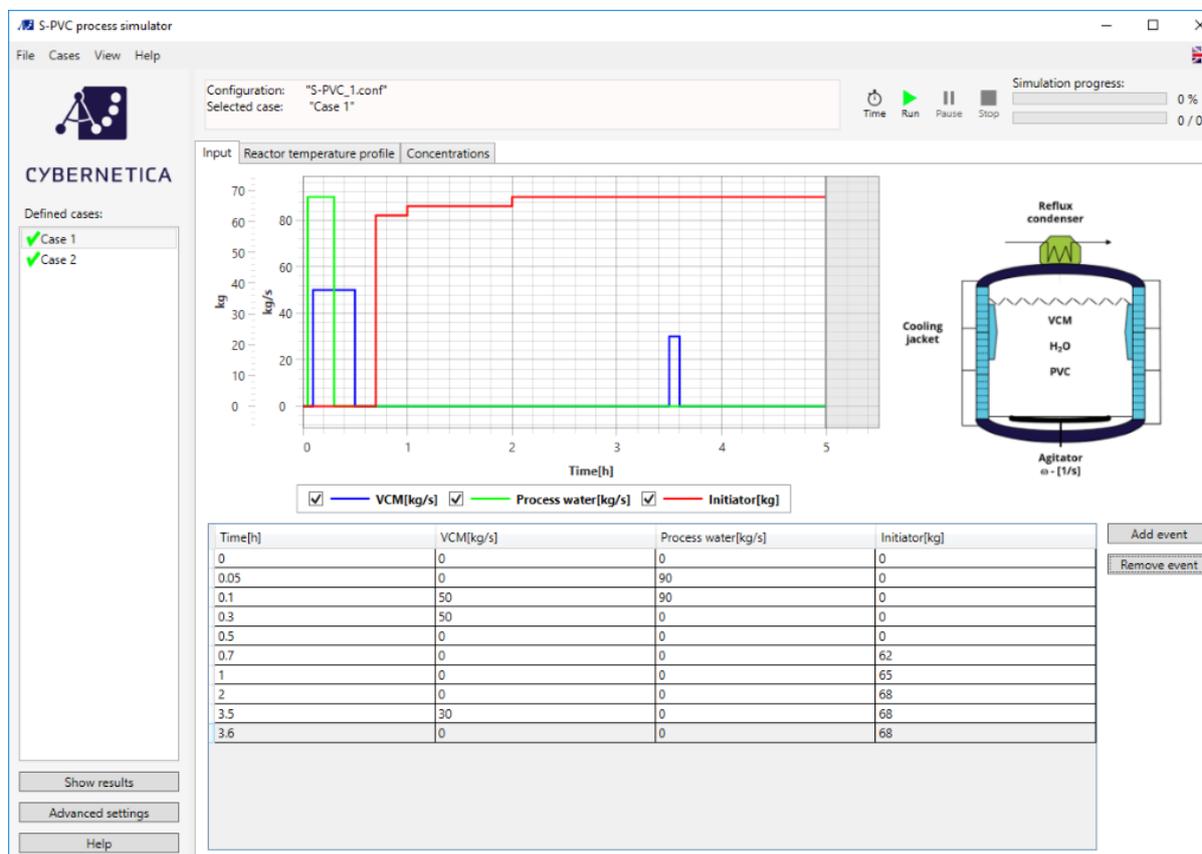
Cybernetica ProXim is a software platform for building tailor-made process simulators using the same kind of process models as Cybernetica CENIT. The platform includes components for simulation and data visualisation.

**Progress since last milestone**

This product has not been updated since last milestone, but the description is included as it will be used at a later stage in the Elkem and Hydro pilots.

**Examples of usage / illustrations**

Example of the user interface of a process simulator:



**Interfaces (in/out) – system/user**

**Subordinates and platform dependencies**

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

Cybernetica ProXim 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.

**TRL for overall component/tool and any parts/subordinates**

8

**References – incl. web etc.**

<http://cybernetica.no/technology/model-predictive-control/>

**To be considered in particular for the following COGNITWIN pilots**

Hydro, Elkem.

<b>Component/Tool description</b>
<b>Component/Tool/Method/Framework/Service Name</b>
Cybernetica RealSim
<b>Defined in Task</b>
5.4 Hybrid Digital Twins
<b>Short Description – incl. Purpose</b>
<p>Cybernetica RealSim is a plant replacement process simulator used for testing of CENIT or other control applications. It communicates over the OPC protocol in order to replicate the interface to the DCS at the plant as closely as possible. It interfaces to Cybernetica Model and Application Components. The plant replacement model might be the same as the model used in CENIT or it might be a different one in order to evaluate how the controller responds to model uncertainty and unknown process disturbances. Cybernetica RealSim is typically used during application development and for factory acceptance tests.</p>
<b>Progress since last milestone</b>
<p>Cybernetica RealSim has been extended with application modules for the Elkem and Hydro pilot processes. These modules contain physics-based models of the processes and make it possible to run online state and parameter estimation, as well as implement soft sensing and nonlinear model predictive control applications.</p> <p>Support for using OPC UA servers for data exchange has been added.</p>
<b>Examples of usage / illustrations</b>
<p>Example of Cybernetica RealSim user interface:</p>

**Simulation control**

Run Pause Step sample Step module Reset Terminate

As fast as possible  
Infinitely (until stopped)

Sim time: 2001-01-01 00:00:00  
Sample no: 0.0  
Status: Idle.

**Simulator modules**

H	Module	Last	Mean	Min	Max
1	Process simulator	0.000	0.000	0.000	0.000
0	CENT	0.000	0.000	0.000	0.000

**Current data**

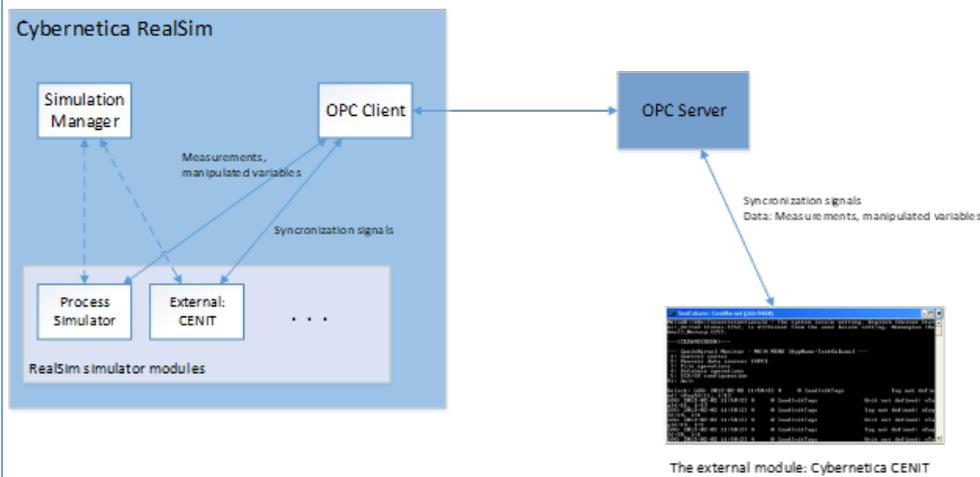
Module: 1 ProcessSim Process simulator, 0 External CENT

Variable: u - Process inputs, ym - Process measurements, z - Calculated outputs, xs - Process states, par - Parameters, con - Constants, extpar - External variable, extflag - External flags

#	Name	Unit	Value
1	Reactor temperature	C	0
2	Reactor temperature measurement	C	0
3	Reactor pressure	barg	0
4	Jacket cooling water flow	m3/h	0
5	Condenser cooling water flow	m3/h	0
6	Jacket inlet water temperature	C	0
7	Jacket outlet water temperature	C	0
8	Condenser outlet water temperature	C	0
9	VCM mass flow	kg/h	0
10	RVCM mass flow	kg/h	0
11	ProcessWater mass flow	kg/h	0
12	Feed/purge water 1 mass flow	kg/h	0
13	Feed/purge water 2 mass flow	kg/h	0
14	Feed/purge water 3 mass flow	kg/h	0
15	Feed/purge water 4 mass flow	kg/h	0

**Overall architecture / pipeline / workflow (incl. figure – elements according to BDVA)**

The following figure shows how Cybernetica RealSim works as a plant replacement tool for Cybernetica CENT:



**Interfaces (in/out) – system/user**

<b>Subordinates and platform dependencies</b>
<b>Licenses, etc. (free for use in the project)</b>
Cybernetica RealSim 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.
<b>TRL for overall component/tool and any parts/subordinates</b>
8
<b>References – incl. web etc.</b>
<a href="http://cybernetica.no/technology/model-predictive-control/">http://cybernetica.no/technology/model-predictive-control/</a>
<b>To be considered in particular for the following COGNITWIN pilots</b>
Hydro, Elkem.

**Component/Tool description**

**Component/Tool/Method/Framework/Service Name**

Cybernetica Modelfit

**Defined in Task**

5.4 Hybrid Digital Twins

**Short Description – incl. Purpose**

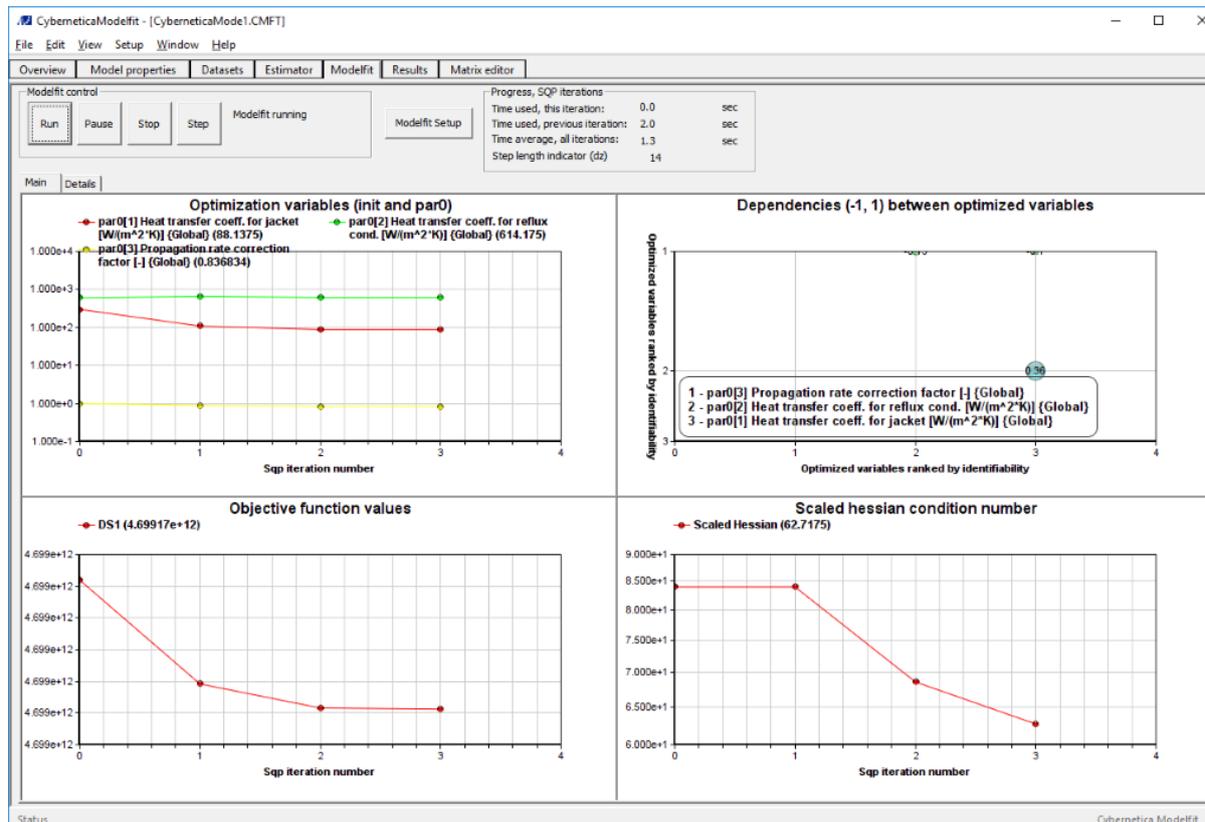
Cybernetica ModelFit is a tool used for off-line estimation of model states and parameters, for model validation, and for design of the on-line estimation part of Cybernetica CENIT applications. ModelFit is used to decide which model parameters should be estimated on-line, to design the on-line estimators, and to estimate the parameters that are considered constant. ModelFit interfaces to Cybernetica Model and Application Components, and it supports the same model formats as CENIT.

**Progress since last milestone**

Cybernetica Modelfit has been extended with application modules for the Elkem and Hydro pilot processes. These modules contain physics-based models of the processes and make it possible to run online state and parameter estimation, as well as implement soft sensing and nonlinear model predictive control applications.

**Examples of usage / illustrations**

*Cybernetica ModelFit user interface:*



*The features of Cybernetica ModelFit include:*

*Design and tuning of on-line estimators in CENIT applications.*

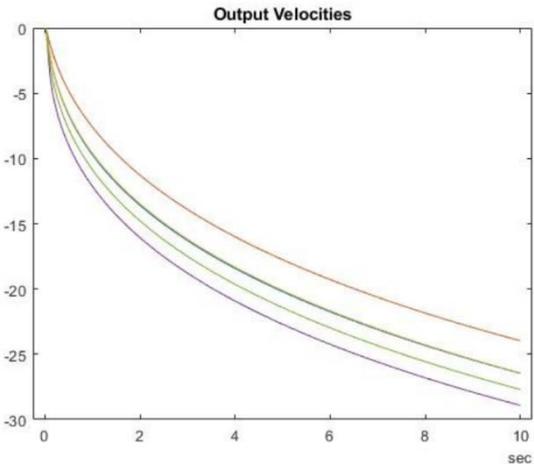
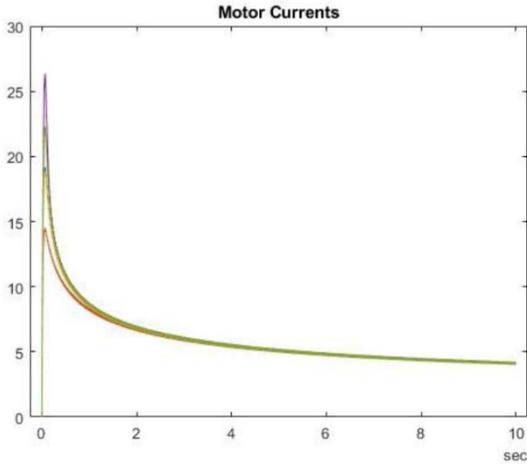
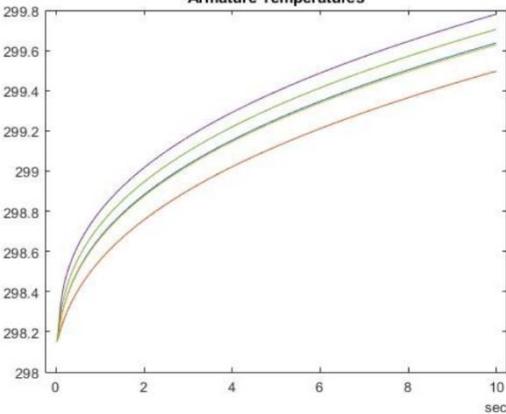
- Estimation of constant or time varying model parameters.
- Estimation of initial states.
- Simultaneous use of multiple data sets.

<ul style="list-style-type: none"> <li>• <i>Parameter identifiability analysis.</i></li> </ul> <p><i>Cybernetica ModelFit is flexible with respect to configuration of the parameter estimation. Parameters can be time varying or constant. Multiple data sets from different operating conditions may be used to find the best parameter fit taken all data sets into account.</i></p>
<b>Interfaces (in/out) – system/user</b>
<b>Subordinates and platform dependencies</b>
<b>Licenses, etc. (free for use in the project)</b>
Cybernetica ModelFit 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.
<b>TRL for overall component/tool and any parts/subordinates</b>
9 – Commercial product.
<b>References – incl. web etc.</b>
<a href="http://cybernetica.no/technology/model-predictive-control/">http://cybernetica.no/technology/model-predictive-control/</a>
<b>To be considered in particular for the following COGNITWIN pilots</b>
Hydro, Elkem.

<b>Component/Tool description</b>
<b>Component/Tool/Method/Framework/Service Name</b>
TStreamPipes-ML Teknopar Machine Learning on StreamPipes as a Data Processor
<b>Defined in task</b>
T5.5
<b>Short Description – incl. Purpose</b>
<p>TStreamPipes-ML is an Apache StreamPipes based tool, it enables none technical users to select and execute machine learning algorithms for predictive maintenance purposes. The performances and the results obtained by the executed Machine Learning algorithms are presented in forms of graphs.</p> <p>The tool has an easy-to-use, drag and drop user interface. Different data sources can be used as inputs to the ML algorithms of TStreamPipes-ML, including data from Kafka, and .csv files. The ML algorithms are MLP, GBT, LSTM, RF, SVM, and KNN.</p>
<b>Progress since last milestone</b>
Since the last milestone a data processor to conduct an ML application has been developed.
<b>Examples of usage / illustrations</b>
Data coming from Kafka or .csv files can be used by the user selected machine learning algorithms, and the results of the predictive maintenance applied by the selected algorithms are displayed on the presented GUI above.
<b>Interfaces (in/out) – system/user</b>
User selected set of ML algorithms are executed on the stream and the results of the algorithms can be compared and graphically presented. Stream data contains sensor data in vector form.
<b>Subordinates and platform dependencies</b>
Apache StreamPipes, Apache Kafka, Fiware
<b>Licenses, etc. (free for use in the project)</b>
Proprietary/ Subject to License
<b>TRL for overall component/tool and any parts/subordinates</b>
The current TRL is 4 running to be TRL 6.
<b>References – incl. web etc.</b>
none
<b>To be considered in particular for the following COGNITWIN pilots</b>
NOKSEL

<b>Component/Tool description</b>
<b>Component/Tool/Method/Framework/Service Name</b>
Cybernetica Viewer
<b>Defined in Task</b>
5.4 Hybrid Digital Twins
<b>Short Description – incl. Purpose</b>
Cybernetica Viewer is a tool for creating user interfaces to display and manipulate data from an OPC server in various ways.
<b>Progress since last milestone</b>
This product has not been updated since last milestone, but the description is included as it will be used at a later stage in the Elkem and Hydro pilots.
<b>Examples of usage / illustrations</b>
The following figure shows an example of Cybernetica Viewer. The user interface is tailor made for the specific application:
<b>Interfaces (in/out) – system/user</b>
OPC classic + OPC UA
<b>Subordinates and platform dependencies</b>
<b>Licenses, etc. (free for use in the project)</b>
Cybernetica Viewer 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.

<b>TRL for overall component/tool and any parts/subordinates</b>
9
<b>References – incl. web etc.</b>
<a href="http://cybernetica.no/technology/model-predictive-control/">http://cybernetica.no/technology/model-predictive-control/</a>
<b>To be considered in particular for the following COGNITWIN pilots</b>
Hydro, Elkem.

<b>Component/Tool description</b>
<b>Component/Tool/Method/Framework/Service Name</b>
TMat- PdM
<b>Defined in task</b>
T5.5
<b>Short Description – incl. Purpose</b>
<p>TMat- PdM (MATLAB Predictive Maintenance for Electro Mechanical Components) is a model used for predictive maintenance of DC motor, gearbox and hydraulic press.</p> <p>By studying random scenarios, due to the changes in gearbox efficiency, resistance and damping coefficient values multiple random scenarios can be generated. Using the data fault code, labeling is performed. TMat-PdM uses different ML Models.</p>
<b>Progress since last milestone</b>
All of the related work related to TMat-PdM is conducted after the last milestone.
<b>Examples of usage / illustrations</b>
<p>TMat-PdM presents some graphs to display the difference between simulation outputs.</p> <div style="display: flex; justify-content: space-around;"> <div style="text-align: center;">  <p><b>Output Velocities</b></p> </div> <div style="text-align: center;">  <p><b>Motor Currents</b></p> </div> </div> <div style="text-align: center; margin-top: 20px;">  <p><b>Armature Temperatures</b></p> </div>
<b>Interfaces (in/out) – system/user</b>
TMat-PdM uses data generated by MATLAB SynData-EM as input.

<p>TMat-PdM enables visualization of confusion matrix for the algorithms used.</p>
<p><b>Subordinates and platform dependencies</b></p>
<p>TMAT-PDM uses Predictive Maintenance Toolbox of MATLAB and Classification Learner App.</p>
<p><b>Licenses, etc. (free for use in the project)</b></p>
<p>TBD</p>
<p><b>TRL for overall component/tool and any parts/subordinates</b></p>
<p>The current TRL is 4 (validated in laboratory environment) running to be TRL 5.</p>
<p><b>References – incl. web etc.</b></p>
<p>none</p>
<p><b>To be considered in particular for the following COGNITWIN pilots</b></p>
<p>NOKSEL</p>

<b>Component/Tool description</b>
<b>Component/Tool/Method/Framework/Service Name</b>
Cybernetica Cognitive CENIT
<b>Defined in Task</b>
5.5 Cognitive Digital Twins
<b>Short Description – incl. Purpose</b>
<p>This is a planned extension of the existing Cybernetica CENIT that will add cognition to the application.</p> <p>The goals of the extension are to:</p> <ul style="list-style-type: none"> <li>• Add self-diagnosing capability to CENIT by use of data analysis</li> <li>• Combine mechanistic modelling of physical processes with machine learning/ AI</li> <li>• Exploit big data sets from the process to improve the model</li> </ul> <p>Both generic functionality and application specific in the form of a new model interface will be added. The cognitive extension may either extend the current estimator (digital twin) or it may replace it entirely.</p> <p>Cybernetica CENIT already implements adaption in the form of parameter estimation. In addition we would like develop and implement methods for real-time and offline analysis of the estimator (digital twin) performance related to process data.</p> <p>In this way it should be possible to automatically classify types of errors: sensor failure, input error or model error. Ultimately, the goal will be to suggest model improvements based on this analysis.</p>
<b>Progress since last milestone</b>
<p>The development of an extension of Cybernetica CENIT that enables self-diagnosing has been started. A framework for self-monitoring of the MPC application via stage-cost monitoring has been developed. The framework consists of the following steps:</p> <ul style="list-style-type: none"> <li>• Estimate the measurement error distribution.</li> <li>• Propagate that noise distribution through the closed-loop MPC model via Monte Carlo simulations</li> <li>• Compare the resulting distribution of the average stage cost to the actual average stage cost from the actual plant. If average stage cost is significantly off from the theoretical distribution, this indicates an error in the closed-loop model.</li> </ul> <p>After monitoring and error detection, the next step is to develop error classification and correction routines.</p>
<b>Examples of usage / illustrations</b>
<p><b>Example 1: Error detection</b></p> <p>The performance of the MPC system will eventually degrade over time due to changing plant conditions, i.e. increased plant-model-mismatch. To prevent poor controller performance, the error/performance degradation first must be detected. There are currently no self-diagnosing capabilities in CENIT. An important part of Cognitive CENIT will be the ability to perform self-diagnosis and detect when the controller performance is unsatisfactory. In the case where unacceptable levels of control performance degradation has been found, further action (such as error classification and error correction) is needed.</p>

**Example 2: Error classification**

Estimators are generally unable to distinguish between prediction deviations resulting from the following errors:

- Faulty input data (requires correction or scepticism)
- Faulty model (suggest adaption)

Being able to distinguish between these errors is important because the required response is very different:

In the case of input error, the appropriate response is some combination of correcting the faulty input signal and minimizing the faulty signal’s impact on the model-predictive control.

This can include:

- Using a default signal instead of the faulty signal,
- Ignoring model state variables that are highly correlated with the faulty signal, and
- Altogether turning off estimation for the affected data points.

In the case of model error, the appropriate response is to try to adapt the model to most accurately reproduce the process data.

An important goal for Cognitive CENIT will be to distinguish between these cases based on an offline training of a classification algorithm.

**Example 3:**

Situations where the model structure is incomplete or wrong may be identified using an automated analysis of the prediction error distributions. Currently Cybernetica CENIT estimators assume that the model structure is correct, and that the prediction error is normally distributed around a mean value, which the estimator tries to centre at zero. In many cases this is not true, and significant deviation from normally distributed error may imply error in the model structure. Identifying this error is non-trivial and may be a well-suited task for an AI extension.

**Interfaces (in/out) – system/user**

TBD

**Subordinates and platform dependencies**

- May use PostgreSQL database.
- May use Python and some of its stat. analysis packages such as pandas, numpy, scipy, pyspark, etc.

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

Cybernetica Cenit 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.

**TRL for overall component/tool and any parts/subordinates**

Cybernetica CENIT: TRL 9  
 Cybernetica Cognitive CENIT: TRL 1-2

**References – incl. web etc.**

**To be considered in particular for the following COGNITWIN pilots**

Hydro, Elkem.

<b>Component/Tool description</b>
<b>Component/Tool/Method/Framework/Service Name</b>
SpinPro – Speech support in Production
<b>Defined in Task</b>
Task 5.2
<b>Short Description – incl. Purpose</b>
<p>The goal of this component is to formalize human tacit knowledge and make it available to other software components. This will be done by generating machine-processable rules based on spoken input. In addition, the knowledge base will be extended by the "discovered" rules, while at the same time ensuring the consistency of the knowledge base (e.g., avoiding contradictory rules).</p> <p>The component will be realized by combining three different types of technologies and by using commodity software:</p> <ul style="list-style-type: none"> <li>- Speech2Text to convert audio to text by using open-source software for speech recognition like cmuSphinx, DeepSpeech, etc.</li> <li>- Shallow NLP to analyse the content by using the background knowledge (e.g. provided in a form of domain-related vocabulary);</li> <li>- Formalisation of extracted information in form of Event-Condition-Action rules which will be evaluated by e.g. Siddhi engine or the VISPAR component (more information is provided in D4.2 deliverable as result of T4.4).</li> </ul>
<b>Progress since last milestone</b>
<p>During the reporting period, we have defined the conceptual architecture (see below) and performed detailed analysis of the existing speech recognition frameworks.</p> <p>Today, many speech recognition frameworks archive good accuracies on given test sets like LibriSpeech (<b>Panayotov, Chen, Povey, &amp; Khudanpur, 2015</b>). However, these results often are not applicable to an industrial and commercial usage. Consequently, we focus on making these techniques more robust to their respective environments. Moreover, domain specific language must be recognized as well. Therefore, possibilities for efficient addition of vocabulary to existing frameworks needs to be found. As a first step, we made an overview on some speech recognition frameworks available now and evaluates a subset of them. Additionally, we discuss the possibility to use multiple speech recognition frameworks and evaluate the margin for improvement by using the proposed technique. Moreover, possibilities to add vocabulary by using the proposed technique are presented.</p> <p>Recently several frameworks for speech recognition were published. Some of them are compared in</p> <p>. The data for the comparison is obtained from the linked repositories and connected websites. The word error rate (WER) is obtained from the papers describing the approaches. Most of the frameworks are written either in C++ or Python. In order to evaluate the performance of a framework, the WER on the LibriSpeech test-clean dataset (<b>Panayotov, Chen, Povey, &amp; Khudanpur, 2015</b>) is used. Comparing the WERs, RETURNN (<b>Zeyer, Alkhouli, &amp; Ney, 2018</b>) and Espresso (<b>Wang, et al., 2019</b>) archive the best performance with 2.3 % and 2.8 % WER respectively. Kaldi (<b>Povey, 2020</b>) archived 3.76 % WER and therefore archives the best performance of the C++ frameworks. Even though Kaldi is only the third best performing framework of the compared ones, it is the most important one. Many frameworks are based on Kaldi, for example Vosk (<b>Inc., 2020</b>) and Espresso. The frameworks differ in their approaches. DeepSpeech, Espresso, Eesen, RETRUNN, wav2letter++ and NVIDIA NeMo utilize machine learning approaches. Vosk uses a database to minimize training</p>

time and utilize a bigger pool of audio transcripts. In the following the project descriptions of some frameworks are gathered.

Framework	Binding for JAVA	Primary Programming Language	Offline	Open-Source	Best model accuracy on LibriSpeech clean-test (WER in %)	Repository
Sphinx-4	Yes	JAVA	Yes	Yes		<a href="https://github.com/cmusphinx/sphinx4">https://github.com/cmusphinx/sphinx4</a>
DeepSpeech	No	C++	Yes	Yes		<a href="https://github.com/mozilla/DeepSpeech">https://github.com/mozilla/DeepSpeech</a>
Kaldi	Unofficial	C++	Yes	Yes	3.76	<a href="https://github.com/kaldi-asr/kaldi">https://github.com/kaldi-asr/kaldi</a>
Vosk	Yes	C++	Yes	Yes		<a href="https://github.com/alphacep/vosk-api">https://github.com/alphacep/vosk-api</a>
Eessen	No	C++	Yes	Yes		<a href="https://github.com/srvk/eessen">https://github.com/srvk/eessen</a>
Wav2letter++	No	C++	Yes	Yes	7.2	<a href="https://github.com/facebookresearch/wav2letter">https://github.com/facebookresearch/wav2letter</a>
HTK3	No	C	Yes	No		-
RETURNN	No	Python	Yes	Yes	2.3	<a href="https://github.com/rwth-i6/returnn">https://github.com/rwth-i6/returnn</a>
Espresso	No	Python	Yes	Yes	2.8	<a href="https://github.com/freewym/espresso">https://github.com/freewym/espresso</a>
Nvidia NeMo	No	Python	Yes	Yes		<a href="https://github.com/NVIDIA/NeMo">https://github.com/NVIDIA/NeMo</a>
Google Speech-To-Text	Yes	-	No	No		-

Table 1: Comparison of speech recognition frameworks

In the following we will focus on:

- Sphinx-4, because it has a native JAVA binding that we needed at the time we first experimented with the frameworks
- DeepSpeech, because it offers one of the best trained open-source models, and
- Vosk, because it uses an alternative approach.

All of them are easy to use, offline and open source and therefore fulfil our most basic requirements. Sphinx-4 slowed down the evaluation process significantly. As seen in Figure, Sphinx-4s performance is significantly worse. The recognition time takes about 1.8 times the length of the audio part to recognize. In contrast DeepSpeech only takes about 0.68 and Vosk only 0.15 times the audio length. Since also the accuracy of Sphinx-4 is twice as bad as the accuracy of the other ones, the usage of Sphinx-4 is impractical.

Another observation out of Figure is, that Vosk (vosk-model-small-en-us-0.3) needs only about half the time to recognize long audio samples than DeepSpeech (deepspeech-0.8.1-models.pbmm). If Vosk and DeepSpeech are used in parallel, this difference is utilizable by reducing the resources for Vosk. Therefore, the recognition is less expensive. Another possibility is using the additional time to improve the accuracy of the result. One facility is to additionally preprocess the audio sample and run the original as well as the preprocessed sample through the recognition process. Thereafter an algorithm combines the two results into one result by utilizing the differences in the recognition results.

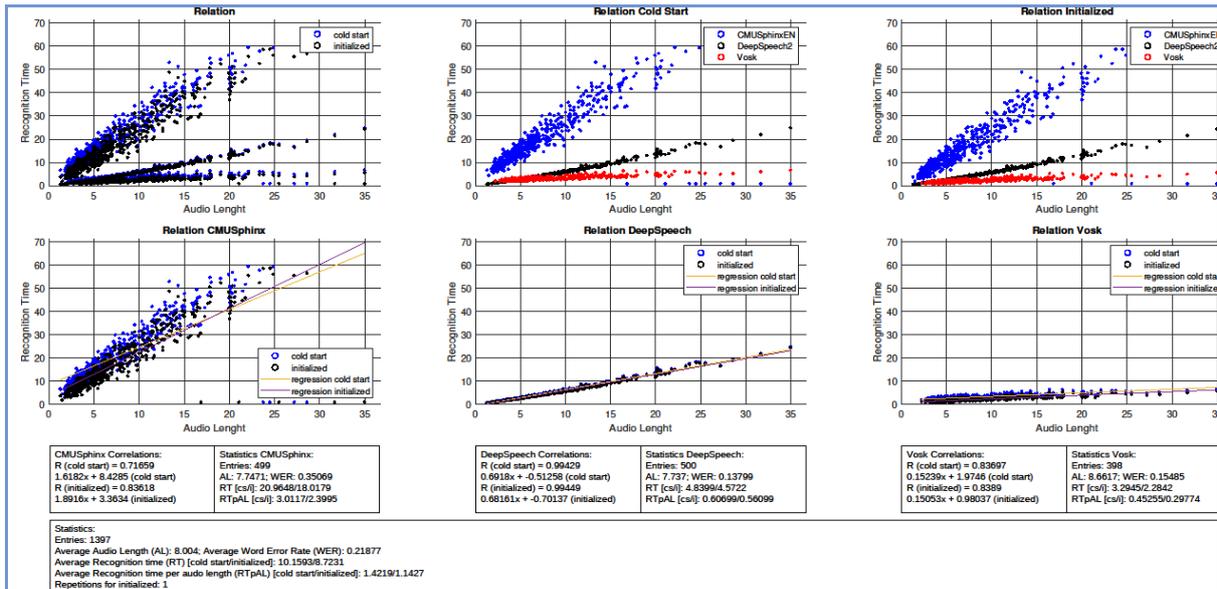


Figure: A performance comparison of CMUSphinx, DeepSpeech (deepspeech-0.8.1-models.pbmm) and Vosk (vosk-model-small-en-us-0.3) on a randomly chosen subset of LibriSpeech’s test-clean. On the x-axis the duration of the given audio segment to recognize is given. On the y-axis the needed time to recognize the audio is described. The results are divided by cold start and initialized. Cold start times include the time needed for preparation when initializing a framework. Initialized times do not include any file independent preparation time, like model loading.

In order to evaluate the potential of such an algorithm, the impact of preprocessing on the recognition results must be estimated first. Therefore, three preprocessings were chosen and the differences in the WERs per audio sample compared. For each of the two LibriSpeech test sets (test-clean/test-other) three additional test sets were created. In the \*-normalized test sets, the volume of the audio samples is raised to a threshold of -1.0 decibel. In the \*-compressed test sets, a compressor was used to reduce dynamic in the audio samples and afterwards normalize them. For the \*-equalized test sets, the intensity of very high and low frequencies was reduced.

In Figure and Error! Reference source not found. the fraction of audio samples with an improvement and a debasement is shown. Files with no change are not shown. On average over all test-other test sets and tested framework model combinations about 28% of the entries changed with a maximum of 54.6% for vosk-model-small-en-us-0.3 at test-other-compressed. About half of them improved the WER. If the algorithm is capable to distinguish between improvements and debasements, the WER on about 14% (average) of the audio samples is improvable. Consequently, these variabilities lead to a margin for improvement of speech recognition frameworks.

Moreover, a combination algorithm could use specialized frameworks for recognizing domain specific language. Those frameworks only need to recognize a few words similar to finding keywords for speech assistants. This task is significantly easier than a complete speech recognition and therefore allows for smaller models that are trained more easily. This method can also be applied to recognize words that are important for the later use of the recognition result. This use may be the control of a machine or similar tasks. Those tasks usually use a rather limited vocabulary, and it may increase the accuracy if the words are recognized by specialized frameworks. It is important to note that the framework is still usable for the general case even if it uses specialized frameworks, because the general tasks are still done by generic speech recognition frameworks. The specialized frameworks only improve accuracy where possible.

For the purpose of developing an algorithm that combines the result of different preprocessing for an audio sample recognized by different frameworks with different models, more research has to

be done. Most likely an evaluation of the results on word level is needed to be able to distinct features, that give hints over the accuracy of a recognized audio sample. Ideally generic relations between the preprocessing framework model combination and the accuracy of the recognition of special kinds of words or auditory events are found. However, it is likely, that such generic rules do not exist or lead to bad results due to the complexity of the underlying recognition process. Nonetheless it may be interesting to try using machine learning approaches for the algorithm. We will try to combine several recognitions of an audio sample into one sentence the reduction of post recognition error (RPRE) and will continue to research in this topic.

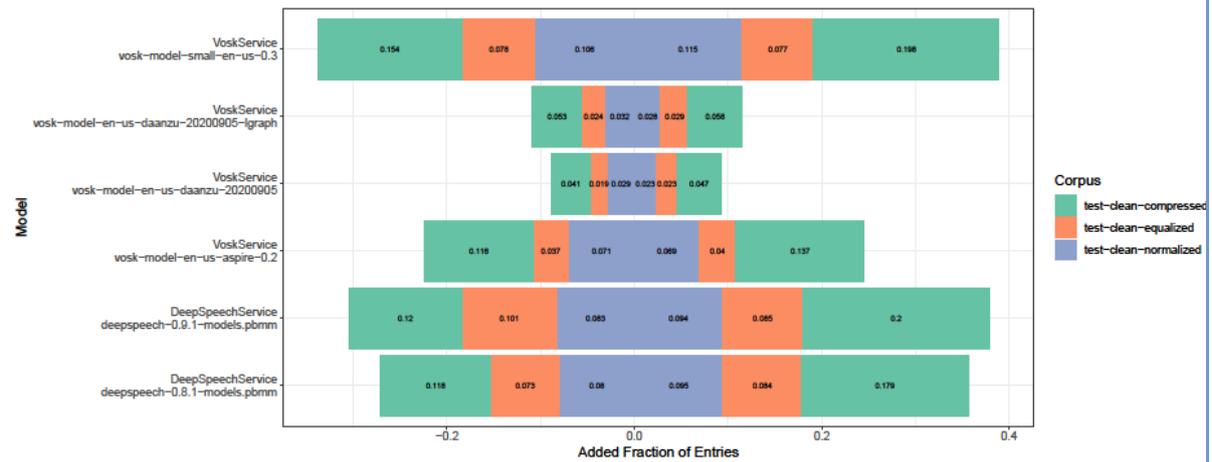
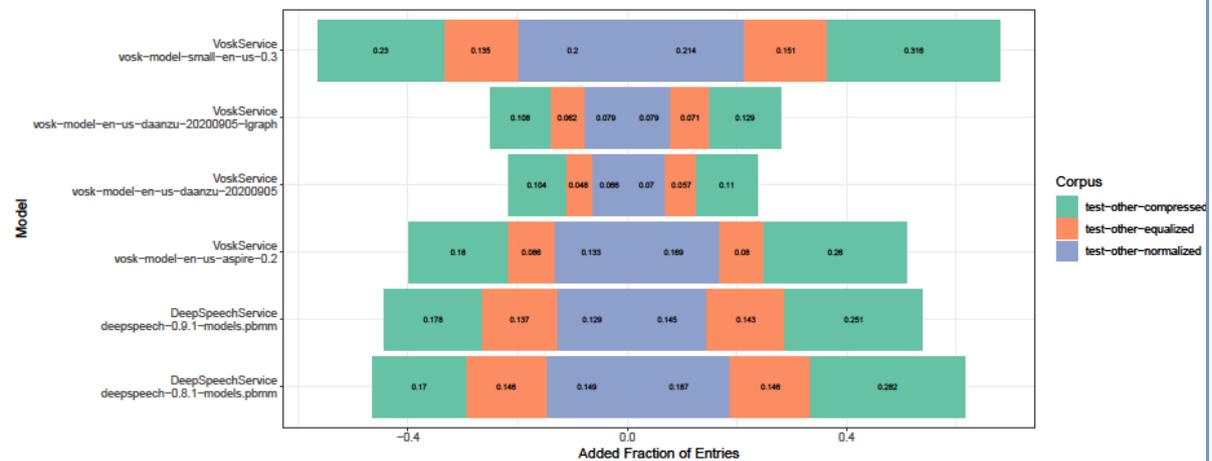
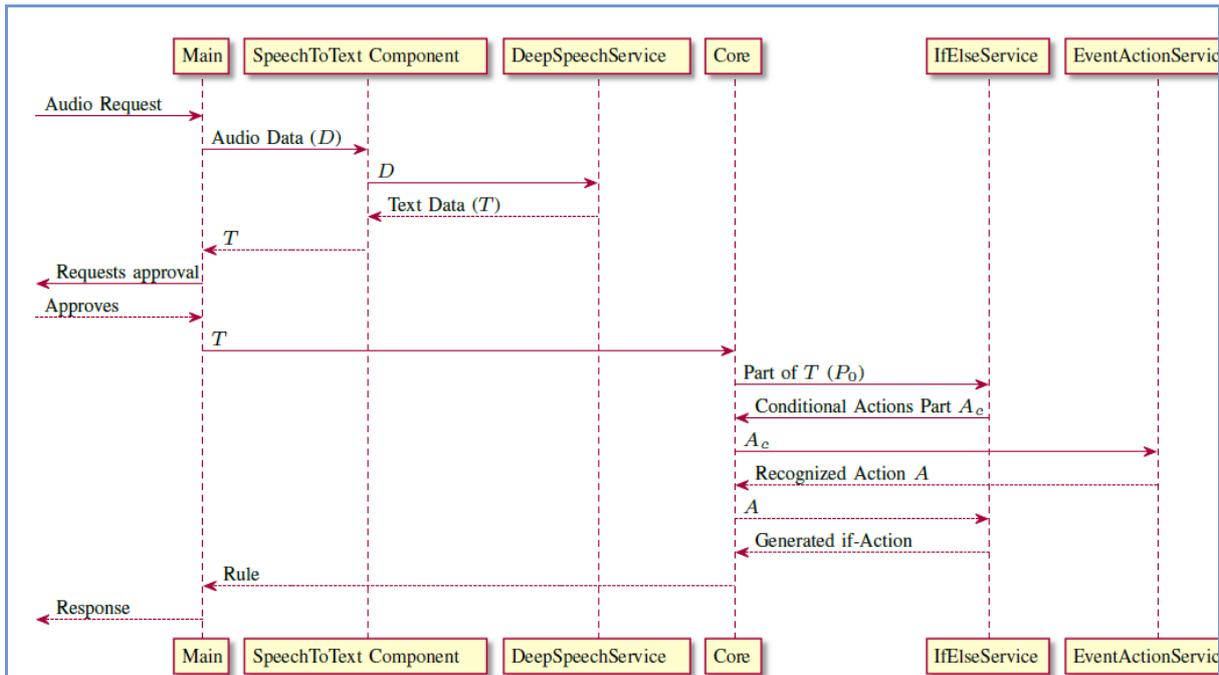


Figure: An illustration of changes in accuracy per preprocessed file with respect to the original file on LibriSpeech test-clean. Positive entries signalize an improvement in WER and negative entries signalize a debasement. The original test-clean LibriSpeech corpus contains 2620 entries. The other corpi were created by applying an audio effect to each file. Consequently, each of the corpi contains 2620 entries.



An illustration of changes in accuracy per preprocessed file with respect to the original file on LibriSpeech test-other. Positive entries signalize an improvement in WER and negative entries signalize a debasement. The original test-other LibriSpeech corpus contains 2939 entries. The other corpi were created by applying an audio effect to each file. Consequently, each of the corpi contains 2939 entries.

Examples of usage / illustrations



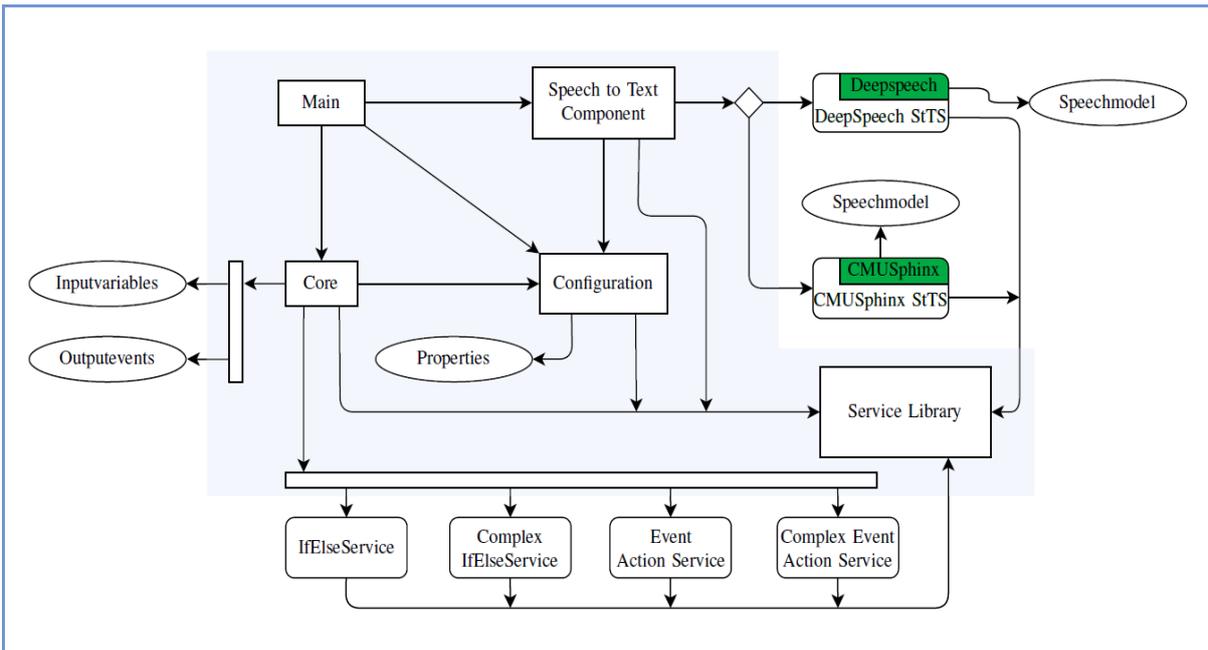
**Interfaces (in/out) – system/user**

In – spoken text

Out -rule(s)

**Subordinates and platform dependencies**

SpinPro is divided in several components as illustrated in Figure below. The process is split into two parts, the speech recognition and the rule creation. Each part is capsulated in one program component. The speech recognition is performed in the Speech to Text component (StTC) and the rule creation in the Core component (Core). The Main component (MC) manages the StTC and the Core. Furthermore, the MC provides the APIs for user interaction and contains the entry point of SpinPro. Those three components use a fourth component called Configuration component (Config). The Config provides a logging framework, the application preferences and messages in multiple languages used as error and logging messages. Moreover, the Config contains a framework for the loading of services. Services are user definable and exchangeable components, that are loaded at the start of the application. The Core and the StTC both use services in order to maximize customizability. All services and all components, besides the MC, use a fifth SpinPro component called Service Library (SL). The SL provides the definitions for all interfaces used in services. Furthermore, it contains definitions of the predefined actions and several parsers for arithmetic and Boolean expressions as well as for parsing single variables and events.



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

License will be defined when the component is ready.

**TRL for overall component/tool and any parts/subordinates**

TR4

**References – incl. web etc.**

Inc., A. C. (2020). *Vosk*. Tratto da Vosk: <https://alphacephei.com/en/>

Panayotov, V., Chen, G., Povey, D., & Khudanpur, S. (2015). Librispeech: An ASR corpus based on public domain audio books. *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, (p. 5206-5210).

Povey, D. (2020). *Kaldi*. Tratto da Kaldi: <https://kaldi-asr.org/>

Wang, Y., Chen, T., Xu, H., Ding, S., Lv, H., Shao, Y., . . . Khudanpur, S. (2019). Espresso: A Fast End-to-end Neural Speech Recognition Toolkit. *Espresso: A Fast End-to-end Neural Speech Recognition Toolkit*.

Zeyer, A., Alkhoul, T., & Ney, H. (2018). RETURNN as a generic flexible neural toolkit with application to translation and speech recognition.

Frameworks

- Sphinx-4 - <https://github.com/cmusphinx/sphinx4>
- DeepSpeech - <https://github.com/mozilla/DeepSpeech>
- Kaldi - <http://kaldi-asr.org/doc/about.html>
- VOSK - <https://github.com/alphacep/vos>

<https://github.com/c3di/neuroscope>

**To be considered in particular for the following COGNITWIN pilots**

TBD - Relevant in consideration for Cognitive services.