Horizon 2020
European Union funding
for Research & Innovation

H2020 - FoF11-2016: Digital Automation

Decentralised architectures for optimised operations via virtualised processes and manufacturing ecosystem collaboration

Deliverable 4.2

Data Analytics Toolkit

Workpackage:	4 – CPS, IoT and analytics
Authors:	Damiano Falcioni, Christopher Gadowski, Wilfrid Utz, Robert Woitsch [BOC] Dietmar Gaertner, Marco Di Pasquale, Walter Waterfeld, Frank Werner [SAG]
Status:	Final
Date:	28/02/2019
Version:	1.0
Classification:	Public

Disclaimer:

The DISRUPT project is co-funded by the European Commission under the Horizon 2020 Framework Programme. This document reflects only authors' views. EC is not liable for any use that may be done of the information contained therein.

DISRUPT Project Profile

Contract No.: Horizon 2020: LEIT – ICT WP2016-17 - 723541

Acronym:	DISRUPT
Title:	Decentralised architectures for optimised operations via virtualised processes and manufacturing ecosystem collaboration
URL:	http://www.disrupt-project.eu/
Start Date:	01/09/2016
Duration:	36 Months

Executive Summary

This deliverable details how production managers gain a detailed insight into current production data to support quality analytics and decision making, via the data analytics toolkit in the scope of the DISRUPT project. The theory-based goal is to generate an understanding of how production data is processed in Industry 4.0, and in DISRUPT in particular. Thus, the data processing initiative covers the lowest, production level from where the data are initially collected, through the data analytics and knowledge layers, to the strategy management level where the semantically lifted data will interact with the use-case organizations' production managers.

The business advantage of this approach is to combat growing customer demands for flexibility and efficiency in production, by offering full transparency between production and strategy management levels via the data analytics toolkit. This is in order to manage changing demands such as time-to-market, as well as optimizing cost and speed holistically throughout the entire production flow. The production flows come from two use-case organizations, one in the automotive industry, and one in the home-appliances industry. The shop-floors in production are mapped, and IoT-technologies are noted in the relevant production steps in order to be understood for the subsequent processes in semantic lifting.

From the production level, the IoT-devices collect production data, which is then processed in the data analytics layer. This environment contains a "speed layer," which undertakes the streaming analytics and predictive model execution, while transferring the analysed data to the "serving layer" directly, along with storing in batches at the "batch layer." In this layer, analysis is done with batch data, creating opportunities among others for machine learning, and sending them to the "serving layer" to be forwarded to smart applications.

In the subsequent knowledge layer, the concept of the domain-specific meta-model is applied in order to take the analysed data from the data analytics layer, and put it into the relevant business context. The principle of the balanced scorecard is used in order to extract measurable goals and develop Key-Performance-Indicators (KPIs). These KPIs are connected to the data from the previous layer, which then can give real-time measurements. The use-case specific models are created in this layer in order to display relevant business aspects, such as cause-and-effect, in production. The KPIs and their success rates are displayed on management dashboards, where the production managers of the organization can observe the data in real-time. This enables quality analytics, since the rate of success along the production line is visualized in order to give the user a holistic view of the organisations' productivity.

Table of Contents

List of Figures

List of Tables

1 Introduction

1.1 Purpose and Scope

The purpose of this deliverable is to present the data analytics and knowledge environment and architecture to be used by the use-case organisations involved in the DISRUPT project via the data analytics toolkit. This document outlines the framework for both the lower layers of data collection and subsequent analytics, as well as the upper, knowledge layer via the IBPM approach in order to provide full production transparency. Both the data analytics and knowledge environments refer to the plant flows in production, where the IoT devices are mounted and data are collected.

1.2 Relation to other Work Packages

This is related to the framework for data processing described in WP2, and the framework for data collection defined in Task 4.1. The architecture defined in this document will also be used as input for further tasks, such as Task 4.3, where the predictive analytics toolkit will be designed in conjunction with this framework.

1.3 Structure of the Document

Firstly, the document will describe the data processing backdrop for Industry 4.0 and its initiatives in data processing, smart data management, and acquisition. Afterwards, DISRUPT specific applications in data processing will be explained, along with the project's use-case scenarios. The production flows for analytics will be visualized, so as to get an understanding of the source environments of data collection. The data analytics layer will be described, including the architecture of how the data will be collected, processed, and sent to the following knowledge layer. In the knowledge layer, the technical architecture is described, along with the IBPM approach, balanced scorecard, and use-case organizations' focus areas. Finally, the management dashboards of the use-case organizations are described in order to monitor production. This gives perspective on how the data are processed from the lowest, production level, to the highest, strategy management level.

1.4 Contribution to the Scientific and Business Objectives

This document provides a scientific framework for collecting and transforming production data to understandable and contextual support for monitoring and strategy management. The data analytics toolkit to be described is scientifically important since it realizes the initiatives of industry 4.0 data processing on a practical and justifiable level within the DISRUPT project. This is an innovative solution since it allows for a holistic insight into the status of an organisations' entire production environment. The toolkit is a functional example of how to collect, process, and transform big data into concrete and contextual production monitoring and decision support, and therefore is important to gain competitive advantage in the current business environment.

2 Data Processing - Industry 4.0

This section is based on the rephrasing of the research paper "Data Processing in Industrie 4.0 - Data Analysis and Knowledge Management in Industrie 4.0" by Frank Werner and Robert Woitsch [43], written also in the context of the DISRUPT project. The paper describes the data processing architecture pursuant to Industry 4.0 from the lowest, IoT and sensor management level, through the data analytics level, subsequent knowledge level, to the highest, strategic production monitoring level. Both authors agree to the rephrasing of this paper specifically for this deliverable, as it is pinnacle for the theoretical understanding of the subsequent sections regarding the DISRUPT framework.

2.1 Basics

This section presents data models to aid the level of understanding between data analysts and the organisations' managers. Specific hurdles here are in present in the arrangement of the complex process stages, vertical transparency of data, as well as calculi that mutually contradict each other (e.g. Quality, Cost, and Speed). Through improved communication between managers of factories, data analysts, and the line-workers, the noted structures can be implemented with more transparency, and therefore with more efficiency.

Nowadays, industrial decision makers are being met with a plethora of data which, in order to be useful, to be efficiently collected, observed, and related. Industrial processes of production are typically complex, since they often involve thousands of modules and an abundance of suppliers. Disturbances in the production related to unexpected events in the manufacturing process (e.g. unscheduled maintenance, software issues, or unexpected orders), the transportation process (e.g. delays due to traffic or incorrect deliveries), or in higher areas of the supply-chain (e.g. quality of components) can possibly cause severe monetary losses on an hourly basis. Additionally, there is the possibility of high variance in the amount of customer orders, changes in orders executed with little time to delivery, or changes in actual order quantities from initial requests. Such situations need a flexible environment for production. Existing works related to the flexibility and swiftness of production are available in [1], [2].

In pursuant to Industry 4.0, circumstances such as the ones previously described can be skilfully managed if the following prerequisites are met: collection of current information, lifting of that information analytically and in the relevant business context, followed by appropriate presentation to the responsible decision makers. (c.f. Figure 1, Modelled in BPMN) Without such prerequisites, the enormous amount of potentially useful information stays away in data silos, and therefore the potential business values arising from that information cannot be taken of advantage. By merging sources of information in production and lifting the information through appropriate contexts regarding the business project and end-user, this information can reveal possible production issues, point to critical situations, and contribute to business value growth.

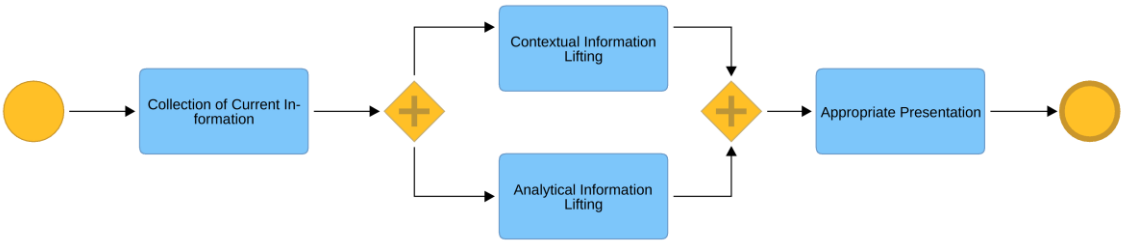

Figure 1: Prerequisites for Data Management in Industry 4.0

In this area, a solution has been drafted to be developed and researched into as part of the DISRUPT project. In production, sensor systems have been installed for data collection, and collected data is linked using complex event processing (CEP). The CEP capabilities of determination are enhanced through models and trained machine learning by structures that permit actual forecasting. Resultant data-driven Key Performance Indicators (KPIs) can be enhanced with knowledge from production (called "semantic lifting") through management dashboards concisely to the responsible decision makers. In this structure, deviances from reference values are not only detectable in real-time, production can be influenced through the modelling perspective, aggregates and compound figures can also be computed. Therefore, the management dashboards can be more specifically customized for the decision makers, working together with the data analysts.

The sensor management output data are used as input for the CEP system in order to calculate data-driven KPIs, which are linked to the knowledge platform to enable dash-boarding. This then illustrates the business context and permits strategic production monitoring (c.f., Figure 2). Source events such as the ratio of rejection or downtime are compiled within the layer of sensor management through IoT devices. On the superordinate layer (Streaming Analytics & Prediction), the events from the base layer are combined and cumulated to compound figures through streaming analysis based on EPL (Event Processing Language) rules put into play on a CEP engine. Certain events, such as machine downtime, could synchronously activate the PMML (Predictive Model Markup Language) Execution Engine to forecast possible results from historical patterns. All outcomes create compound figures which are then guided to further superordinate level through compounds of combined events. The subsequent knowledge management layer brings the compound events into line with a business layer and context (done with a semantic-database and semantic-lifting). Finally, the enhanced data figures are revealed in the relevant Business Context Strategic Production Monitoring level through applications such as dashboards, and other monitoring applications.

2.2 Data Processing and Event Gathering Framework

Networked machines and systems are the basic building blocks of Industrial Internet of Things (IIoT) applications. Added value for the industry, however, is created only by digital products and services, for example in real-time monitoring or data-based optimization of machine utilization.

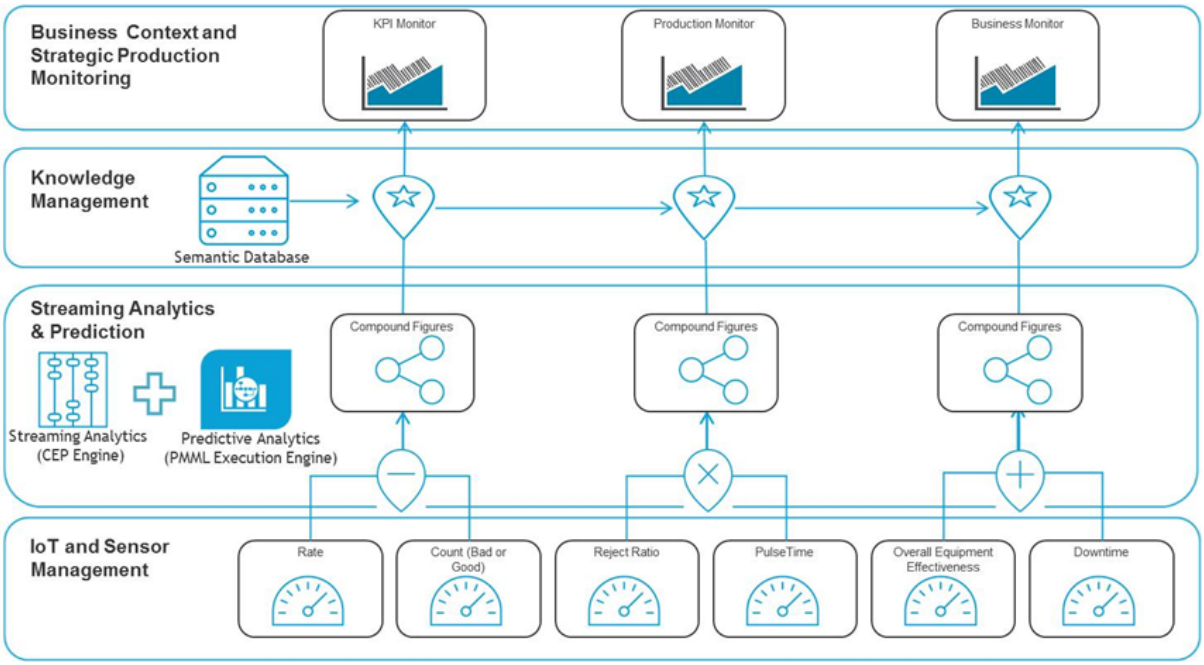

Figure 2: High-level architecture of the data- and knowledge management environment

This section illustrates the DISRUPT data and event processing architecture with its core components like the Messaging Bus (based on Universal Messaging by Software AG, as described in [6]) which interconnects the DISRUPT platform with used IoT sources; it further describes the application of Streaming analytics (mainly based on Apama, the Software AG complex Event Processing engine, as described in [5]), to monitor, aggregate, and correlate real-time event streams.

2.2.1 Streaming Analytics in Real-Time

The "Streaming Analytics & Prediction" level in Figure 2 takes advantage of many different information points collected from the IoT and Sensor Management level and calculates compound statistics by data accumulation, correlation, or analytics on the basis of scientific models. The usage of this layer was motivated by λ-Architecture [4] impressions, which allow for the examination of information in real-time, while at the same time presenting a way to implement prediction models previously trained by data science and algorithms for machine learning.

Streaming Analytics is built on a commercial solution [5] through a specific, in-memory architecture that allows for efficient processing of great volumes of data in real-time (greater volumes than traditional IT-solutions for databases). The rules of the CEP are described in EPL, and do not need to be fully defined during the design-phase. Such rules could, for example, characterize the calculation of production data spikes to discern if the values of individual sensors are within the range of normal operation or analysis of the latest values to evaluate the current level of change from the norm. Figure 3 shows a sample of such a CEP rule (written in EPL).

In pursuant to the work done in [10] and [11], dynamic adaptation of patterns of events is possible through the CEP by means of a model-based approach: The modelling of CEP rules takes place at the service layer (e.g. within the layer of management dashboards illuminating business context), and is integrated into rules in EPL, that can be activated by the CEP during operations. Since the information from sensors is already integrated in the connected sensor management system.

In Figure 3, an example of a script to monitor production is presented that is written in EPL and covers three rules. The first two rules are in place for monitoring temperature and pressure event data streams, and perform the print statement when values from the streams surpass a previously defined threshold. The final rule is executed if, within a three-second window, temperature increases 2% higher than the target temperature and then pressure increases more than 5% of the stated maximum.

```
monitor SensorMonitor {
    float targetTemperature:=100.0;
    float targetPressure:=800.0;

 action onload () {
   //subscribe to event event streams and monitor sensor S001
   monitor.subscribe("TemperatureS001");
   monitor.subscribe("PressureS001");

   // Temperature high rule
   on all Temperature (sensorId="S001", temperature >= targetTemperature * 1.10) as
 temperature {
     print "TEMPERATURE_HIGH: " + temperature.toString();
   }
   // Pressure high rule
   on all Pressure (sensorId="S001", pressure >= targetPressure * 1.10) as pressure
 {
     print "PRESSURE_HIGH: " + pressure.toString();
   }
   // Temporal rule, temperature high followed by pressure high
   on all Temperature (sensorId="S001", temperature >= targetTemperature * 1.02) as
 temperature ->
        Pressure  (sensorId="S001", pressure >= targetPressure * 1.05) as pressure
within (3.0) {
     print "TEMPERATURE_PRESSURE_RISE: " + temperature.toString() + " " + pressure.
toString();
    }
  }
}
```

Figure 3: CEP Monitoring Script in EPL

2.2.2 Predictive Analytics

In conjunction to the defined streaming analytics, base events from the IoT devices are dispersed through a "Message Bus" for storage in an "Event Store", within the superordinate batch layer (see Figure 4). The "Event Store" layer could be composed of either an SQL-based database (such as PostgreSQL), a non-relational database (such as Cassandra[1]), or Big Data tech (such as Apache Hadoop[2]). The specific reason for having an "Event Store" is to gather all relevant data from the event sources (IoT devices) and to make that data available for processing in data analytics. In this scenario, the analytics are integrated in order to enable predictive modelling, and through this medium, machine learning. Tools in this area like KNIME [7], WEKA [8], R [9], and other applications for statistical analysis are used. With the aid of these applications and processes, models on the acquired events are trained to achieve a PMML model. Such a model is then available for execution

[1] Apache, https://cassandra.apache.org/, Last Accessed on 27.02.2019

[2] Apache, https://hadoop.apache.org/, Last Accessed on 27.02.2019

on the Predictive Execution Engine [12] in order to predict effects from the triggering events on the basis of trained models.

2.2.3 Management of Sensors and IoT Devices

The fundamental layer of Sensor Management and IoT Devices allows for holistic connectivity to the platform through the Message Bus by administering search functionalities of all connected devices and their status, including the localization of those devices, software and firmware analytics, and their remote control. This scenario also enables concise troubleshooting and is cloud-connected. This assists in unified M2M communication and defines the bread-and-butter parameters of the devices like their location #, expected data values, and type of sensor. Such an approach is not only applicable on IoT-devices but also enables connections with Web services and other information sources in order to supplement the information available. As for some scenarios a tight integration with the real production environments is not feasible, the DISRUPT Analytics toolkit has been fed with real, historic data. In the CRF use-case, for example, large log files from truck-movements and shop-floor data have been used as input for the platform. On the other side, live data are integrated into the platform to cover the real-time aspects like weather information - on upcoming storms and risks weather conditions – and real-time traffic information by integrating platform external services.

This approach has two advantages; it shows the feasibility to integrate real-time, live information sources that require instantaneous reaction of the underlying systems and algorithms while giving the pilot users a means at hand to experiment, i.e. to replay certain experiments and situations with changes parameters and inputs and observe the outputs provided by the analytics module.

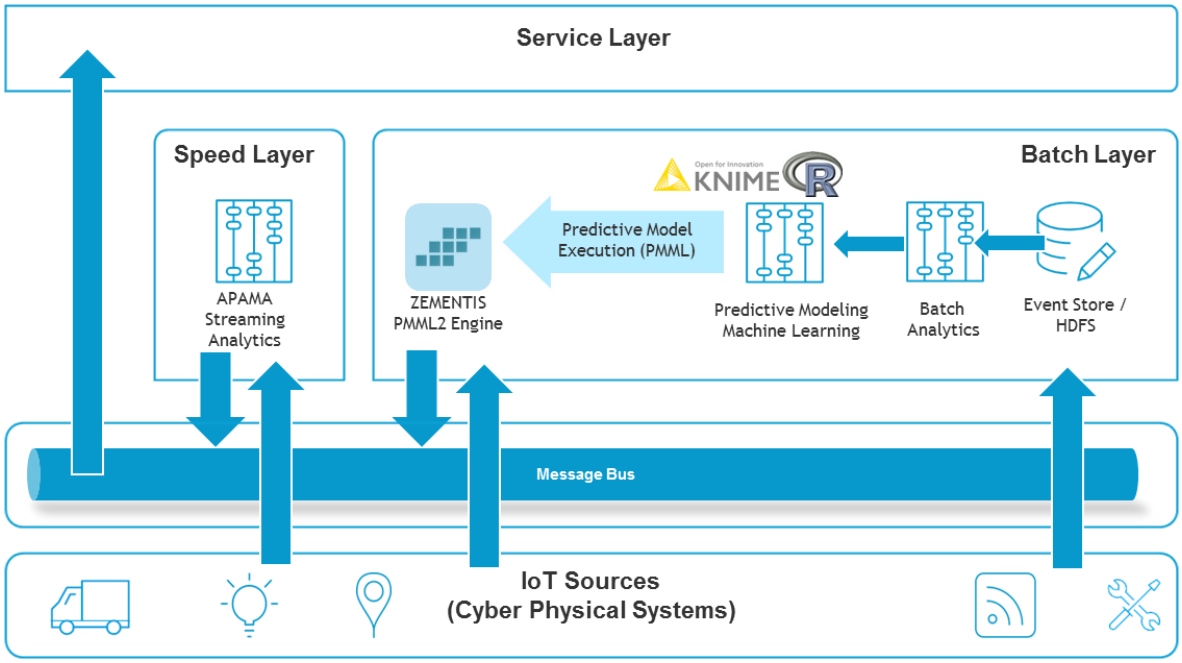

Figure 4: IoT Platform – Streaming Analytics and Data Prediction

This figure illustrates the different instruments in order to gather, process, and influence data. An important addition to this data-handling is the semantic enrichment, which allows for additional analytics and interpretation of that data for cross-domain and – sensor processing. Such an approach is a proposal to achieve the following, "Smart Data Management."

2.3 Smart Data Management

The interpretation of data, information, and knowledge is a key challenge in the IT-sector, and particularly in the area of data management. A holistic aim not only includes collecting data but bringing it into the relevant context to support data interpretation by humans, and/or machines. The management dashboard is therefore observed as a data visualization approach, specifically as a support system for responsible decision making, which aligns to the coinciding knowledge layer. Such a layer can support the responsible decision makers, or even enable machine-based interventions whether the case permits it.

In Figure 2, a knowledge management layer is added to the traditional dashboard to enhance interpretation. We put forth to use conceptual models in order to implement the knowledge layer, since such models can embody knowledge in a full-range medium with semi-formal models that enable interpretation by humans, to strict-formal models that also enable interpretation by machines [14]. An informal model uses graphical process models with text descriptions, while strict-formal can use graphical representations but concentrate on formal connotations by means of specifically defined rules and ontologies. Such a model based-approach to knowledge management can be mapped in a top-down or bottom-up fashion.

The top-down fashion used if humans outline their strategic intentions and construct conforming sensors and data formats in order to suit those intentions. A usual example of top-down mapping is to outline parts-per-hours, and from that configure sensors (e.g. a sensor that tallies the amount of produced parts in a production line – such data output aligns with the strategic purpose). This is, in reference to Figure 4, manifested by the definition of event patterns in models, and through their subsequent conversion to CEP rules. On the contrary, a bottom-up fashion is used when conclusions can be made from collected data, and from that detecting the resulting business impact. In this sense, the data being gathered gives meaningful information from which to draw business conclusions. An example of this could be the detection of an anomaly from patterns of the status of different machines in the production line. Once analysis of possible influences is undertaken, the outcome may rise to a business strategy, for example predictive maintenance.

2.3.1 Model-driven Data Management

This sub-section describes how strategic intentions can be charted to data from sensors, and from sensors back to the intentions. The model-driven support of management of data in the scope of "Smart Data Management" contains (a) model-based methods to aid in the management of data, as well as (b) model-based methods to bring about smartness. The introduction of smart monitoring in industry, the practical application of business intelligence, and the application of data analytics for predictive behavior are well-researched topics , and cover the areas of smart monitoring to multi-agent-based systems [23],[24], smart sensors [25],[26], edge computing [27],[28],[29], and large scale sensor architectures [30].

Regarding model-based methods to bring about smartness, certain Operations Research tools [31] exist such as modelling predictive control designs [32], [33], and their applications in Cyber Physical Systems [34], or regarding data analytics in production [35], [36], [37] or [38]. The unique characteristic in this method is that it combines any of the stated approaches with conceptual

modelling [39]. Conceptual modelling aids in the concept formation of human knowledge and permits integration with formal technological aspects in the areas of data analytics, sensor-usage, or business intelligence. This combination of both human and machine knowledge is the novel characteristic, which is an aid on a continual basis for the extraction of knowledge, and adds an additional layer of understanding (semantic-lifting) on top of the data machine layer.

The idea of "model-driven management" was practically put into use through the commercial management software ADOSCORE® [15], which encompasses a scorecard. Originally, this scorecard was designed for management of financial strategies, but has been extended in order to monitor performance and contribute to knowledge management. Furthermore, this tool has been specifically enhanced to analyse and manage complex datasets. The idea of "smart data management" has been put into practical use through PROMOTE® [16], [17], an approach to practically implement knowledge management through process-orientation. This approach can be improved with machine understandable aspects such as rules, workflows, semantic illuminations, or cases.

2.3.2 Model-based Data Management

Data management based on models expresses the knowledge on how to connect business objectives with the data collected [18].

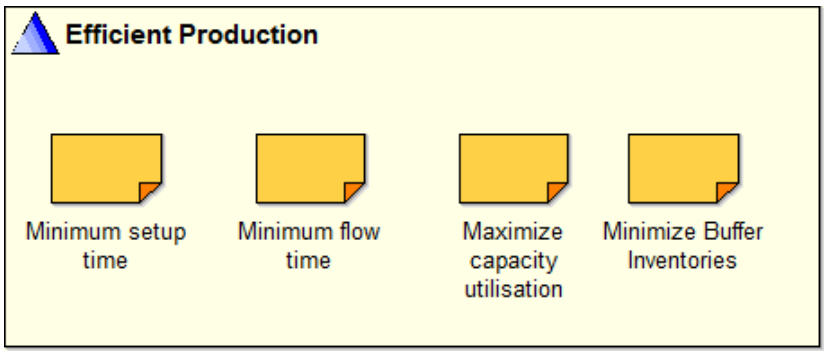

Figure 5: Critical Success Factor Model

Business objectives and intentions can be observed critically through critical success factors. The purpose of a management dashboard is exploited from the collection of critical success factors and their grouping into business goals. This is outlined in the below figure where the superordinate goal of "efficient production" is illustrated with relevant success factors in order to measure if the goal is being reached. The creation of such models is usually undertaken through workshops with the responsible production decision makers, to design the model accurately from domain knowledge.

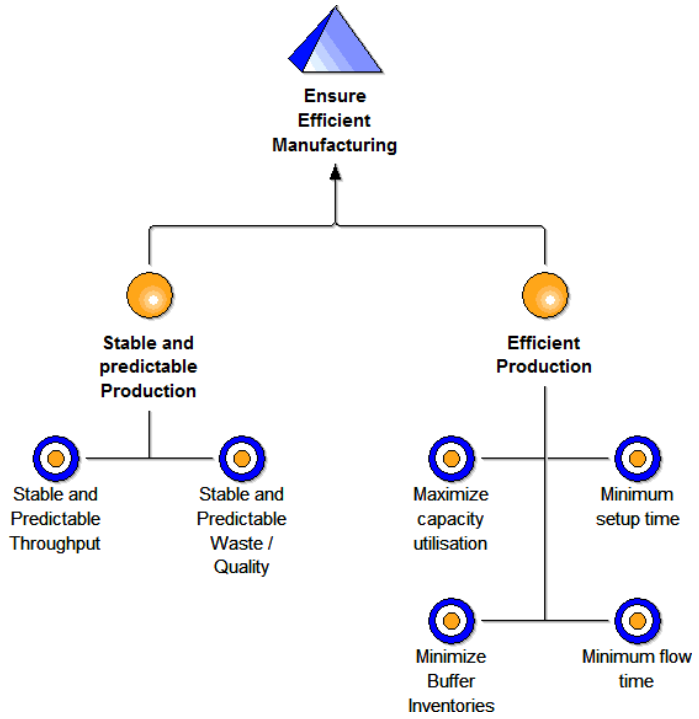

Figure 6: Cause and Effect Model

Cause and effect models take business intentions and create measurable points on their efficacy through the medium of dashboards, with the original critical success factors being expressed as KPIs (Key-Performance-Indicators), and each of those KPIs being allocated to a sub-goal or goal. This type of model is showing in Figure 6, where the goal is first detailed, and success factors are measured through the linked KPIs. The described approach introduces a take on KPIs that is not data-specific, but rather domain-specific. Through this, the KPIs are defined through knowledge that is specific to a certain domain in order to detail exact parts of goals, types of measurement, the business ambition as well as with intervals of time.

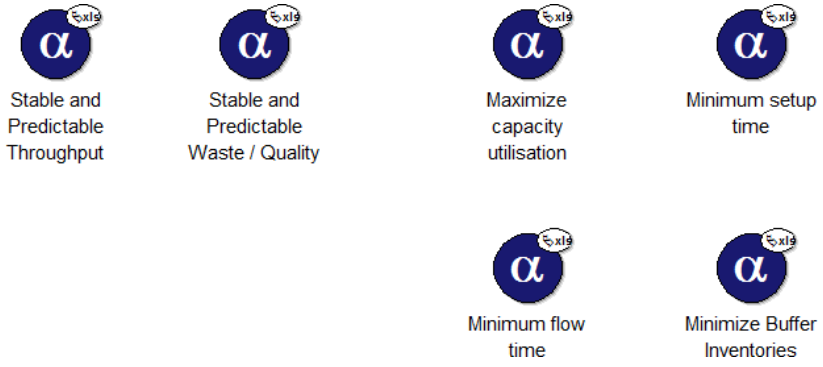

Figure 7: "α"-Indicator Model

The data is assessed by the knowledge layer via the "α"-Indicator Model. Each of the previously described domain-specific KPIs are connected to "data-indicator" models through an algorithm. The simplification of this overall method encompasses the following: Collection of data via sensors, storage of the data through "snapshot" databases, and algorithm-usage to access the data. In Figure

7, a listing of all data indicators is shown, which has the relevant information saved in order to access data points by means of queries or APIs.

The mentioned models are typically constructed by specific experts, due to the fact of the knowledge-intensiveness of the undertaking. For example, generating the cause and effect model usually involves the organisation's plant engineers, and the domain responsible decision makers. In another example, the α-indicator model is usually generated in conjunction with the IT-experts and technicians of that specific domain, in order to integrate knowledge of sensors and their data. A main goal of the referenced model-based approached is the extraction of knowledge with the objective of continuously improving the analysis of the data, and through that, constant learning in the organisation. The name "α"-indicator is specific due to its purpose to access the appropriate data, like the atomic characters that words are made up with. In this sense the "atomic" character delivers physical data points which then are used to create management dashboards.

Typically, a top-down method of modelling is used to analyse the connection of business intentions to the CSFs (Critical-Success-Factors). A sensor for data access is then applied to transform the original data set to a management-oriented output. The advantage of such an approach is that specified business approaches can be concretely analysed with success factors, and then monitored through the KPIs and their data connection.

2.3.3 Smart Data Management

The mentioned approaches can be improved with knowledge-based practices in order to create intelligent solutions. Such model-based approaches to the management of data aim to conquer two current challenges:

1. Currently, assessing KPIs qualitatively is done usually through questionnaires from the responsible actors. Even though this works well to understand opinions, proofs need to be established in the areas of data analysis findings, visual analytics, and detection of anomalies in order to support them.
2. The management of big data has proven to be a serious challenge in its own write. In order to take advantage of the new trends in digitization such as the Industrial-Internet-Of-Things, cloud computing, edge computing, or smart sensing enable the possibility of more complete analytics and reporting, through knowledge-management techniques.

ADONIS® is a platform for managing business processes through models. It is established on a "meta-model," that enables precise individualisation by configuring the "meta-model" for different environments, can a "data-repository" where information for the models, along with technical and semantics information is stored. Through this medium, ADONIS® is the proposed approach to be used, since it offers a very useful BPM (Business Process Management) approach with features allowing one to define, examine, simulate and document the industrial process, in addition to enabling the integration of semantics needed to describe the data. The approach ADOSCORE® was created originally for use as a Balanced Scorecard but has progressed to being a more generic tool for dash-boarding. The strong suit of ADOSCORE® is to support data visualization, while focusing on knowledge design, and therefore knowledge abstraction within the knowledge environment.

3 Data Processing - DISRUPT

3.1 DISRUPT Integration and Manifestation of Data

In the context of the DISRUPT project in creating "Smart Factories," the integration of the systems and environments is undertaken in order to create (a) data collection and aggregation, (b) a prediction framework in order to facilitate event forecasting from historical data, (c) monitor of production through management dashboards, and (d) to create a smart decision support tool in the event of disturbances in production.

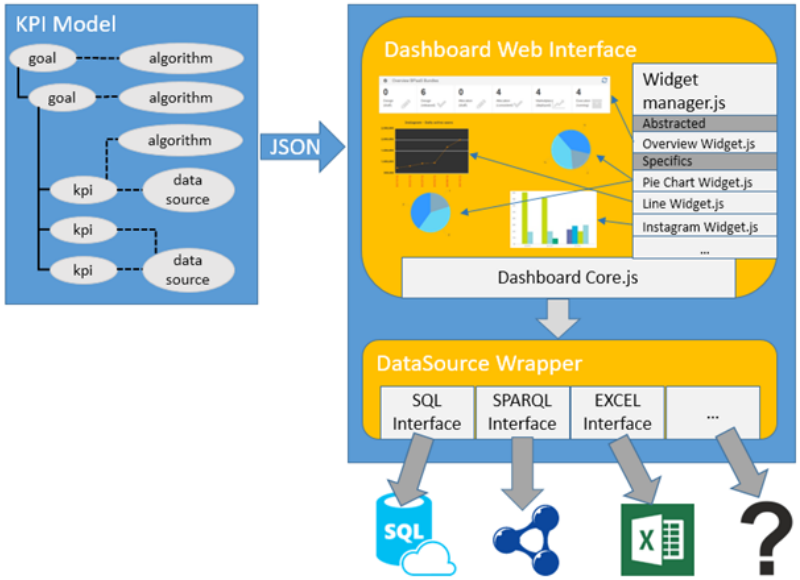

Figure 8: Overview of the Data Dashboard

We are introducing a system to support decision making which uses automated and continuous data analytics and prediction methods. The described messaging bus ensures uniform, dependable, and efficient communication between all parts. It acts as a principal component that receives the information from the IoT devices in production, and at the same time provides the communication through all the other parts of the DISRUPT framework. The messaging bus, therefore, is used to interchange information between the data origins in the IoT devices and the management dashboards via a publish-subscribe archetype to enable real-time analysis [5], streaming of events [6], and tracking for the predictive engine [12] and data delivery.

Such an approach aligns with the environmental obligation of modularity within the system. Through this, every part of the environment can be unplugged from the event messaging bus and replaced with technological solutions with the same characteristics when needed. Therefore, the actual users of the end-product are not constrained to use a specific matrix but can rather choose specific parts, and therefore companies that fit their environment most closely overtime.

The management dashboard of BOC for smart data management, including all data management design component environment and the data sensors, is available via the combined commercial platforms of ADONIS® and ADOSCORE® [15]. In addition, a research prototype variation of the

platform is also available, resulting from European projects[3], which has created a world-wide open innovation community ADOxx.org[4].

The execution of this mentioned scenario makes use of and improves the open-source micro-service structure OLIVE[5], delivered by OMiLAB[6]. The community OMiLAB is a world-wide open-source community for modelling methods [22] that puts forth support for model-based executions. The OLIVE framework is used as a main vessel for the dashboard, ensuring modularity, and integration with other micro-service parts in the environment. The capabilities are direct results from participation in internal projects.

The introduction of data processing and anomaly discovery is taken place in the layers IoT and Sensor Management & Streaming Analytics and Prediction (as observed in Figure 2). The current data collection management environment ADOSCORE®, which typically involves a person entering qualitative data by hand based on a rating scale (e.g. 1-5), is supported by means of the responsible actor getting the data analytics results in the configuration of the pre-determined context. The topic of manual data entering is worked out by reducing the need for manually entered qualitative data as compared to quantitative data, which can be automatically imported, and adapted to import the data analytics results directly into ADOSCORE® for viewing in the dashboards.

In a technical sense, these are data services [20], [21], which are "snapshot databases" that store appropriate data streams and enable a standardised access to the raw data across all mediums. Such data services are implemented as independent REST services and therefore allow for an automatic intercession between ADOSCORE® and the service of Software AG's data analytics for automatic dashboard import.

The purpose of this is to explain that integration of technical components is not a requirement, as long as semantic integration is in place. Semantic integration is the procedure of interlinking information originating from different sources. Such integration can be put into motion in various ways, starting from tagging, and moving to more advanced tactics. Tagging, or in more advanced cases, semantic lifting, is undertaken by a designer, called in this scenario the "Data Service Designer." The raw data sets must be stored in a specific way, usually by means of a time-series snapshot database. The data form of a snapshot database is referred to as the Data Service Engine, indicating needs for data stream storage, calculations, and access.

[3] BOC Group, https://uk.boc-group.com/science-research/, Last Accessed on 27.02.2019

[4] BOC Group, https://www.adoxx.org/, Last Accessed on 27.02.2019

[5] BOC Group, https://www.adoxx.org/live/olive/, Last Accessed on 27.02.2019

[6] Open Models Initiative, http://omilab.org/, Last Accessed on 27.02.2019

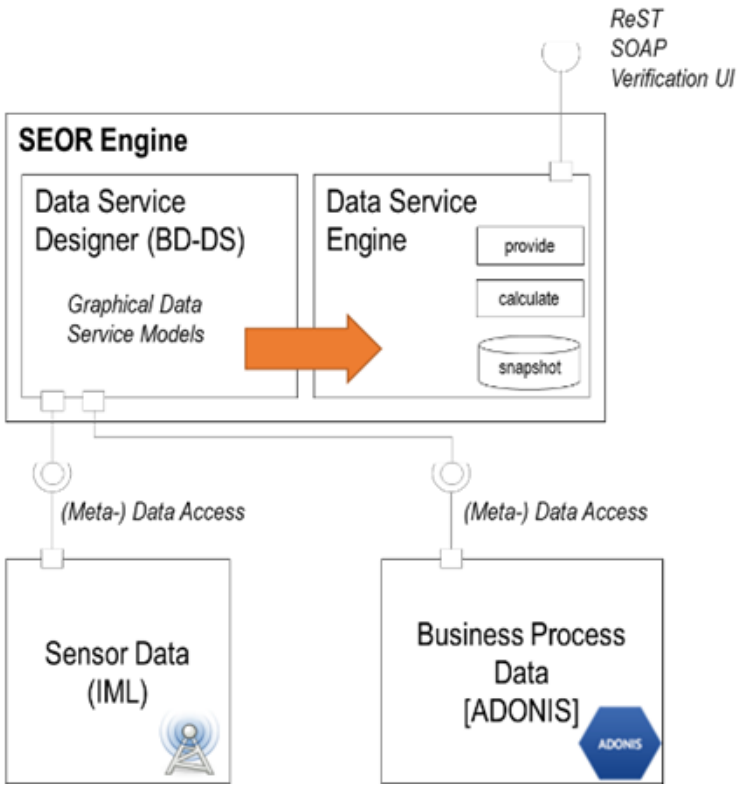

Figure 9: Data Services from OMiLAB

Figure 9 describes the SEOR (Systematic Energy Operational Rating), which is a direct outcome of the European project ORBEET. Through this, data services are applied to access and monitor the data stream. We apply this described environment and enhance it for Industry 4.0 in order to monitor data streams continuously as an output of Software AG's data management environment. Concurrently, semantics are added to the data through descriptions such as vocabulary, classification, or ontology to enrich the original import-data. In the above Figure 9, such an enhancement is observed by ADONIS®. Therefore, data services accumulate the data, which is then semantically enhanced with the relevant business context. The described semantic enhancement can be specifically applied in order to more efficiently manage the incredible amount of data streaming from the sensors, and therefore master the mentioned second hurdle to smart data management.

3.2 DISRUPT Application Sectors

The offered framework can be used in many different areas and scenarios. Nevertheless, through analysis of industry domains, different industry requirements arise.

3.2.1 Automotive Industry

The automotive industry is component-driven and facing high market saturation. In its current state, the process of digitization is pinnacle to further growth, in order to update the industry to being solution-based [13]. Specifically, the process of digitization enables significant advantages such as connected supply-chains. Having connected supply chains, for example, enables the traceability of products in order to aid in the management of inventory. Through the described method, acquiring data in real time from the IoT devices delivers the analysed output data in real time, through the

medium of the data analytics and knowledge management environments. The environments present supply chain level analysis and visibility, along with business stability, collaboration visibility, and dynamic reactions to unexpected events. Through the analysis of [13], investments in this realm are expected to yield up to $1bn per OEM (Original Equipment Manufacturer), by enhancing efficiency, reducing costs, and increasing innovation agility and overall collaboration.

On top of this, this industry is facing an increase in critical aspects of their environment, such as the intricacy of connected products, mass-customization, and meeting KPIs while minimizing resources & impact on the environment. The described framework presents a solution to contest with businesses in emerging economies, which have a competitive advantage in this field in terms of labour rates, regulations on taxes and production costs. Specifically, the Industrial IoT-devices will aid the Automotive OEM in observing and conquering the hurdles to increase competitiveness of current production.

3.2.2 Home Appliances Industry

Manufacturers are implementing more technologies, such as business process management solutions, in order to renovate their processes in manufacturing. The increase of transportation costs, the requirements for increased productivity, the demand for more eco-friendly products, and the rise of volatility are all factors demanding innovation. The FOF (Factory of the Future) is an important viewpoint, described as an interlinked chain of manufacturing, including Information & Communications Technology (ICT) and technologies for automation. Through this, connected software is able to universally manage the assets of the factory, while connected data collection products will enable the interchange of data on a two-way basis, along with full quality control of production. These aspects shall be pushed through the following factors:

a. The necessity to improve consumption levels of resources – through energy- and material-efficient processes and equipment
b. The fact that the increase of processing power for the ICT, and more refined software for analysis, provides performance analysis in real-time
c. The necessity for substantial increases in efficiency, along with safety and resource sustainability in both logistics and in production
d. The necessity to improve time-to-market
e. The opportunity to optimize production processes via digital factory modelling

Through the DISRUPT project, organisations can continuously observe fluctuations in the processes of manufacturing, their suppliers, and other issues across their broad value-chain, as new possibilities arise in the continuous advancement in interconnectivity of the world.

3.3 DISRUPT Production Flows

The following section describes the shop-floor production flow of each use case-organisation that is depicted in the following figures. The production flows are modelled in BPMN (Business Process Model Notation), and encompass the points where sensors and IoT-technologies are implemented for data aggregation. In CRF's case, the flow of the assembly of different vehicle parts is studied and analysed from this base setting. In the case of ARCELIK, the model indicated the machines that are

used in the production. Also, modelling in BPMN shows how the KPIs are specifically connected to the final products of production and their status.

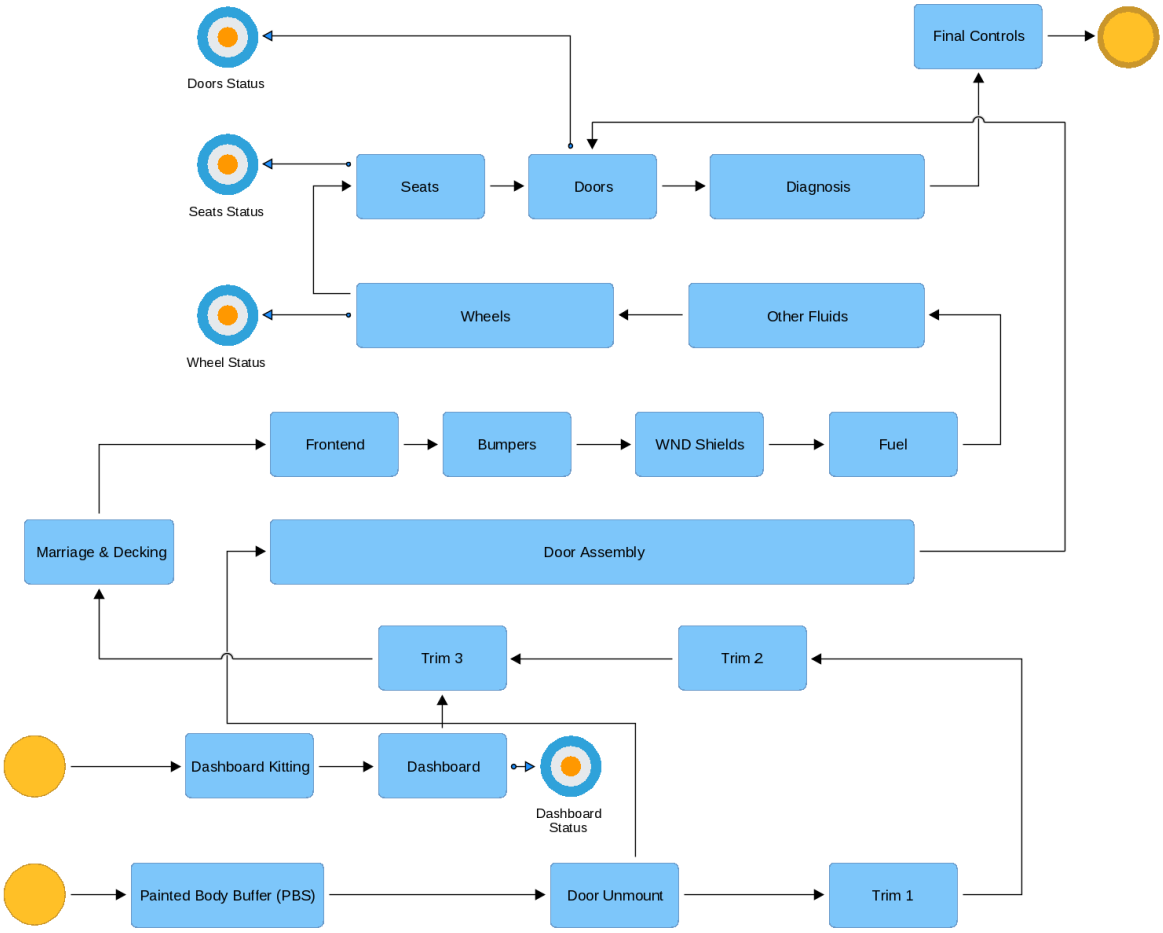

Figure 10: Production Flow of CRF

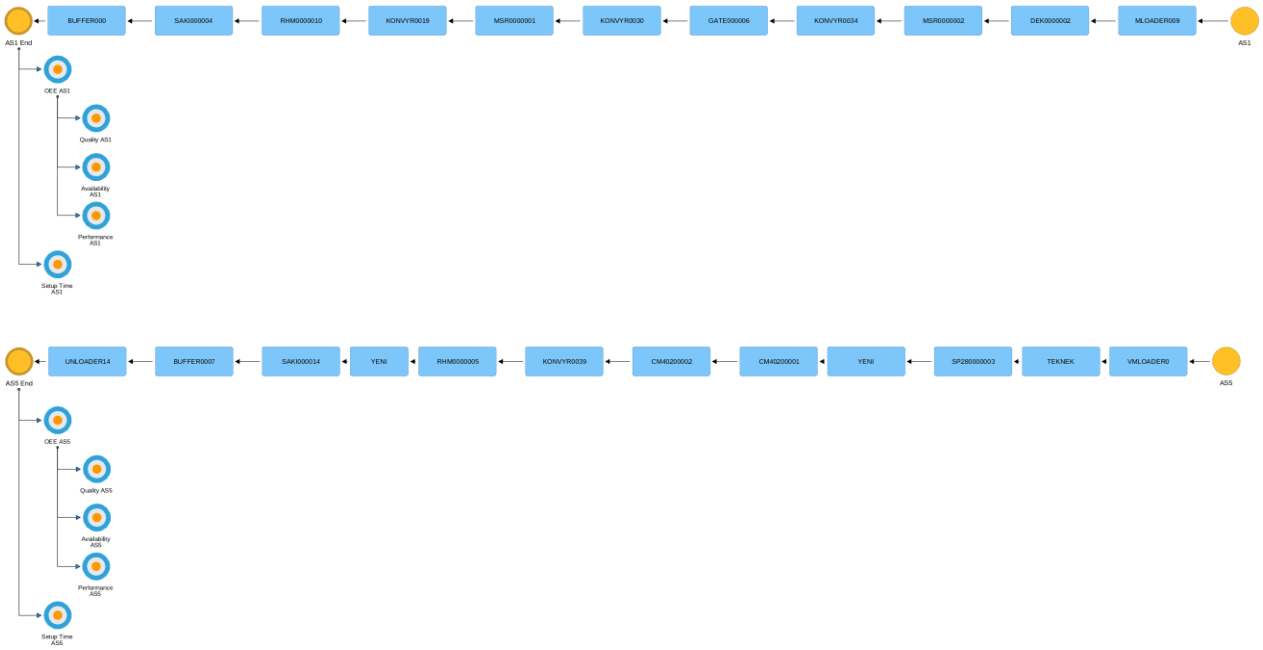

Figure 11: Production Flow of ARCELIK

3.4 DISRUPT Application of Industrial Data Space Reference Architecture

The reference model for the industrial data space, defined by the International Data Spaces Association and the Fraunhofer Society [42], is a standard for how data is to be handled in the Industry 4.0 dataspace. This standard outlines specific roles and aspects in the function of secure industrial data processing and management. Important features such as data sovereignty (the equilibrium between the need for sharing and protecting data), security and interoperability (the unrestricted understanding of linked resources) are described in order to have a holistic architecture structure to manage the ecosystem of data.

The roles that the handlers of the data are defined as such the data owner, data provider, data consumer, data user, identity provider, app store, and app provider. These roles and their interactions will be defined in subsequent sections in how the DISRUPT project data processing architecture (c.f. Figure 2) fits into this framework.

3.4.1 IoT and Sensor Management Level

On the lowest level, the data to be collected from production through the IoT-devices and sensors is provided by each use-case organisation, who takes the pinnacle role of data owner. The use-case organisation entrusts the responsible management of their data, by ensuring the aggregation of relevant data from each source, and management of the secure transport of that data to the cloud database, for decentralised access by the appropriate stakeholders. The communication path from this level is to the cloud-database. Through the cloud database, interaction with all other levels in data processing is realised.

3.4.2 Streaming Analytics & Prediction Level

The architecture described in section 4 of this document takes the roles of a data consumer, identity provider, and app provider. The data is consumed from the sources in the IoT devices, in order to be processed and analysed at this level via the aforementioned architecture. This level also provides identity to the data, by lifting the raw data sets in order to enhance meaning. Finally, the data analytics plugin for the DISRUPT project is a specific app for data processing, hence giving the role of app provider. The communication paths from this level are observed through access to the cloud-database, and the knowledge management and strategic monitoring environments.

3.4.3 Knowledge Management Level

The architecture described in section 5 of this document takes the roles of data consumer, software provider, and vocabulary provider. The enhanced data is, through semantic lifting, humanized and made presentable by being integrated into the concrete industrial business processes of the organisation for strategic production monitoring. This layer also is a data consumer, since the data is linked from the could-database and data analytics parts to the data knowledge management environment. Apart from being a software provider with the IPBM-approach of ADONIS® , this layer is, importantly, the vocabulary provider in order to describe the data sets in the appropriate business context for the responsible decision makers. The communication paths from this level are from the cloud-database, from the streaming data analytics level, and to the strategic monitoring environment.

3.4.4 Strategic Production Monitoring Level

This level returns the broad architecture full-circle with having the data owner (the use-case organisation) monitor their production through observing the enhanced production data in the relevant business context via the medium of the management dashboards. This level also takes the role of data consumer, as well as the data user since the end-user monitors the data trends from this level. Communication paths on this level are to the knowledge management software, which then communicates with the data analytics layer and the cloud database. If sudden changes in production occur, the monitoring level can interpret the data changes in this level and make the relevant strategic change in the organisation and/or production flow.

4 DISRUPT Data Analytics Layer

The following text describes the technical architecture of data collection, aggregation, and transformation at the lower, data analytics layer. The data analytics layer is connected to production via the IoT-devices and sensors, therefore directly interacting with the production flow and source of the data.

4.1 IoT λ-Architecture & Core Components

The core data collection and integration architecture in DISRUPT is explained in detail in D4.1. It follows best-practice from industrial solutions based on IoT architectures, while considering core concepts of the lambda-architecture (with its Speed- and Batch-Layer paradigm), a unified messaging layer that facilitates connectivity between the "outer world" and the DISRUPT platform, and a consistent alignment with principles of an Event-Driven Architecture (EDA) approach. There are three key areas of a typical IoT Lambda architecture (c.f. Figure below):

a. Speed layer: Provides real time processing to compensate for the delays in the batch layer. This is an area where in-memory processing plays a big role. The same data destined for Hadoop and eventual batch processing is sent here to be subject to real-time aggregations, calculations, enrichment, and filtering. Results are sent to the serving layer for immediate consumption.
b. Batch layer: Performs batch computations to create view and aggregations for later ingestion by users or other systems. In a typical situation is to persist data in an event store (e.g. Hadoop or NoSQL based DB systems like Cassandra[7]). It is the dominant batch processing engine for very large unstructured datasets. Map/reduce processing creates outputs that are sent to the serving layer for consumption.
c. Serving layer: Results from batch and speed layers are presented to end users. It may simply provide access to precomputed results or provide ad-hoc querying capability. This is another area where in-memory can play a role by providing caching, high performance distributed search, and additional processing. In DISRUPT this is also the place from which enriched data sources are taken to be used for machine learning algorithms.

[7] Apache, https://cassandra.apache.org/, Last Accessed on 27.02.2019

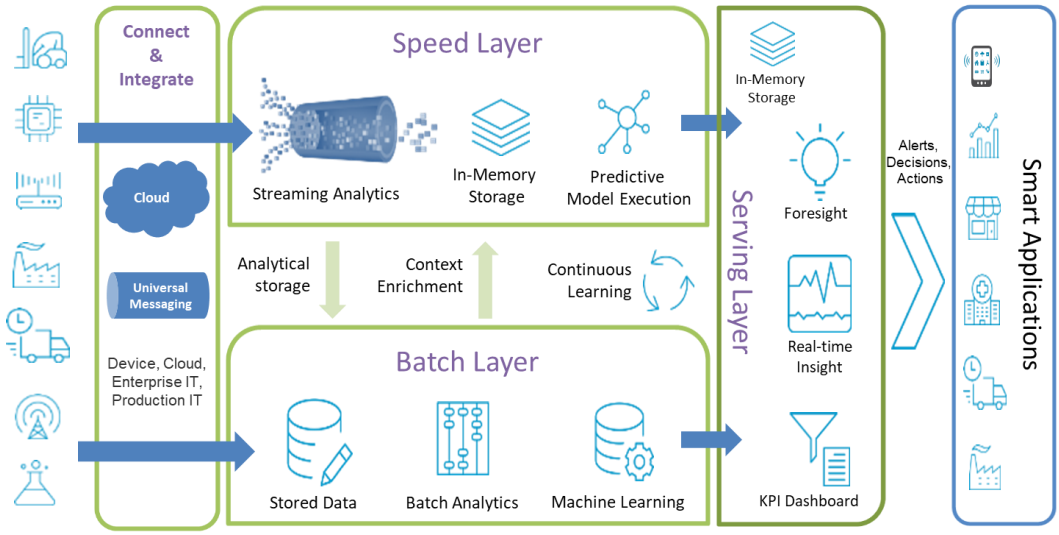

Figure 12: A standard IoT Solution with Elements of the Lambda-Architecture

This section will cover the layer "Connect & Integrate" and "Speed Layer" as shown in Figure 12. The "Batch Layer" providing Batch Analytics as well as parts of the "Serving Layer" – with its Machine Learning capabilities – are described in detail in D4.3 Predictive Analytics Toolkit due in M33.

The last figure gives a most complete overview of the lower layers of the DISRUPT architecture. The Messaging Bus interconnects event sources from IoT devices and cyber physical systems (CPS) with the platform. Directly attached to the message bus is the Speed layer represented by an Apama Streaming Analytics Deployment which correlates, aggregates and monitors incoming event streams in real-time (real-time meaning hundred thousand of single events per second) offering a new way of analysing pattern and helping the business side to gain valuable results from the data streams. The Batch Layer in DISRUPT is represented by a tools chain working on persisted, off-line data. Events from the Message Bus are persisted for further processing using tools like R, KNIME to cleanse the data and use it as input for predictive modelling based on machine learning techniques. Valid models (after being validated on the actual data and obtaining sufficiently high accuracy) in terms of trained predictive results are copied to ZEMENTIS, an execution engine for predictive models using the exchange format Predictive Model Markup Language (PMML). Details of the Machine Learning Process, encompassing the whole tool-chain will be illustrated in detail in D4.3 - Predictive Analytics Toolkit.

4.2 Streaming Analytics and CEP

The use of Complex Event Processing (CEP) technology is a relatively new field which gained more and more popularity within the last decade. Forrester Research describes it as a "software that can filter, aggregate, enrich, and analyse a high throughput of data from multiple, disparate live data sources and in any data format to identify simple and complex patterns to provide applications with context to detect opportune situations, automate immediate actions, and dynamically adapt." [46] Forrester Research, March 2016. CEP technology is sometimes referred as a typical database model

turned upside down: Where databases typically stored data and runs queries against the data the CEP stores queries and runs data against the queries.

Streaming analytics is neither a scheduling technology nor a decision tool but a re-active solution which always needs an external trigger. Respective rules are hereby defined at design time in EPL (c.f. subsection on Apama's EPL), an SQL-like dialect. It doesn't allow for a perpetual analysis and executes observations in a defined and well-prescribed time-window (e.g. last five seconds, last 10,000 observations, etc.).

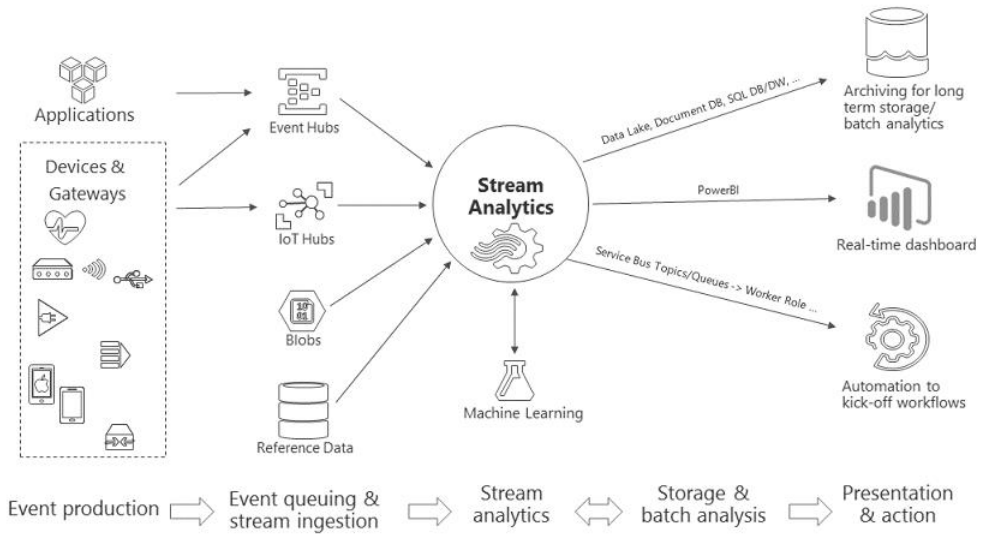

Figure 13: Data Flow used in Industrial Analytics Scenarios

Apama's EPL is an environment in which to express pattern matching logic. Its great flexibility and performance is in part down to an implemented Hypertree. With discrete event matching you are typically looking for individual events occurring within a time period and in a specific order. For example, "*Event A being received, followed by Event B, and this happening within 1 minute*". Streaming aggregation allows a performance analysis on a stream window to search for calculations that work across a window of data points, e.g. "*Average value of all Event As across a window 24 hours*". A set of aggregate functions are provided, that have been substantially extended to the use case scenarios within DISRUPT.

Apama Queries extend the platform capabilities by providing inherent in-memory and scalable analysis. The event data is stored off-machine in a redundant in-memory store providing data reliability. The scalable analysis is provided by multiple instances of Apama correlators that all access the data from live event streams or the redundant in-memory store. The models of Apama EPL and Apama queries can be combined using the common currency of passing events between them. Based on this technology events with historical or context specific data from external sources such as databases can be easily enriched with off-line data from other sources. The following figure (c.f., Figure 14 below) gives a short overview about the different capabilities.

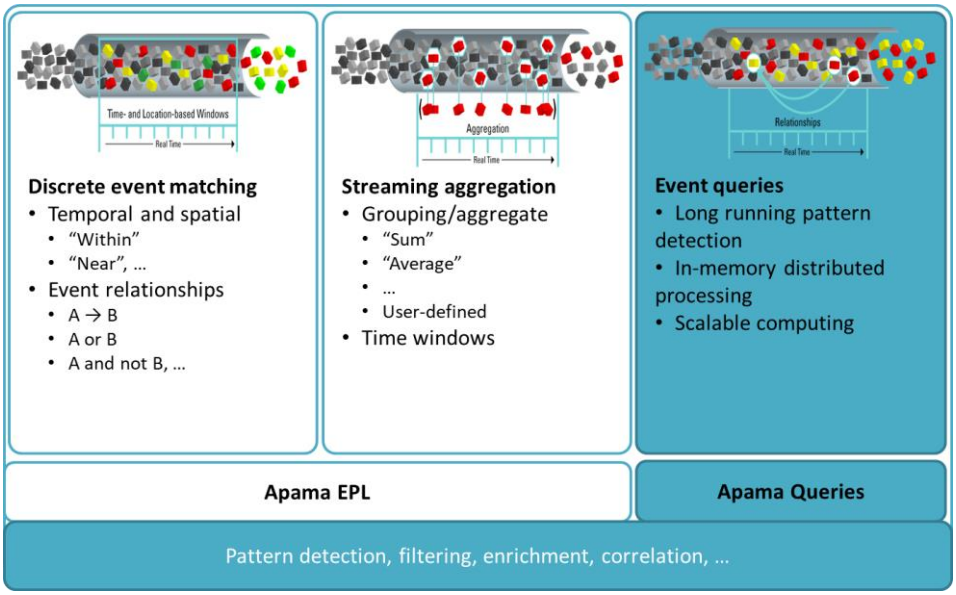

Figure 14: Different Capabilities of the Apama EPL and Apama Query Engine

EPL is a standardized imperative scripting language which is strongly statically typed and used to pass information as events. It enforces loose coupling which facilitates adaptation to changes and extensions. An example in EPL – taken from the Supply Chain Monitoring Scenario in DISRUPT is shown below in Table 1.

Table 1: Sample in EPL on the Supply Chain Monitoring Scenario

```
using com.apama.correlator.jms.JMS;
using DISRUPT.Checkpoint;
using com.industry.analytics.Data;
monitor CalculateTimeWindow {
       JMS jms;
      string CONNECTION_ID := "UniversalMessaging";
      dictionary<string,dictionary<decimal, float> > truckTimeDict:=
       new dictionary<string,dictionary<decimal, float> >;
      dictionary<decimal, float> meanTimeDict:=
       new dictionary <decimal,float>;
      dictionary<string, float> checkDict:=
       new dictionary <string,float>;
      string target;
       action onload() {
[...]
             Checkpoint cp ;
              on all Checkpoint() : cp {
                    //Dict for Analytics KIT
                    dictionary<string, string> emptyDictionary:=
                  new dictionary <string,string>;
                    //Create Dataformat for Analytics Kit
                    //Check if TruckId is already subscribed
                    if (not checkDict.hasKey(cp.truckId)){
                        send com.industry.analytics.Analytic("Delta",
                      [cp.truckId],["DELTA"],new
                      dictionary<string,string> ) to "";
                        checkDict.add(cp.truckId,1.0);
                    }
                    send com.industry.analytics.Data(cp.truckId,
                  "COMPUTED",cp.truckId,cp.getTime().toDecimal(),
                  cp.timestamp.toDecimal(),cp.timestamp.toString(),
                  cp.rta.toFloat(),cp.checkpoint.toFloat(),3.0,
                  emptyDictionary) to cp.truckId;

                    }
              monitor.subscribe("DELTA");
             //Triggers the DataMonitor
              on all Data() as dataIn spawn DataMonitor(dataIn);
             //DataMonitor triggers new monitor which calculates the
               delay
              action DataMonitor(Data dataIn) {
                    from t in all WeatherEvent() within 60.0 where
                  t.truckId = dataIn.sourceId  select t as we{
                            spawn DataWeatherMonitor(dataIn,we);
                            on Data() die;
                            }
                    }
[...]
```

4.3 Event Message Bus

A central component of the architecture is the messaging bus [c.f. D4.1: Data Collection Framework], facilitating message handling and message persistence. Within DISRUPT a messaging and event handling component from Software AG was chosen which is called *Universal Messaging*. Universal Messaging is a Message Orientated Middleware product that guarantees message delivery across public, private and wireless infrastructures. Universal Messaging has been built from the ground up to overcome the challenges of delivering data across different networks. It provides its guaranteed messaging functionality without the use of a web server or modifications to firewall policy. Universal

Messaging design supports both broker-based and Transport-based communication, and thus comprises client and server components. [44]

There are several message delivering strategies like point-to-point, fan-out, request-response, publish/subscribe and Message Queues. In the following the two mostly used messaging paradigms publish/subscribe and Messaging Queues, which are used in the DISRUPT Messaging Bus, are described.

Publish/Subscribe is an asynchronous messaging model where the sender (publisher) of a message and the consumer (subscriber) of a message are decoupled. When using the channels/topics, readers and writers of events are both connected to a common topic or channel. The publisher publishes data to the channel. The channel exists within the Universal Messaging realm server. As messages arrive on a channel, the server automatically sends them on to all consumers subscribed to the channel. Universal Messaging supports multiple publishers and subscribers on a single channel.

Messaging Queues is like the Publish/Subscribe paradigm an asynchronous messaging model where publisher and sender of the message are decoupled. Unlike publishing/subscribing to channels only one consumer can read a message from a queue. In the event and messaging world different modes of operation (*Queues*, *Channels*, *Data Groups*) exist, each with its respective advantage and disadvantage. These will be explained in the following.

Using a delivery model of type "*Queues*", each message delivered to one and only one consumer. Messages are removed on delivery which is ideal for implementing transactional request/reply style interactions. The following figure depicts the behaviour of Queues.

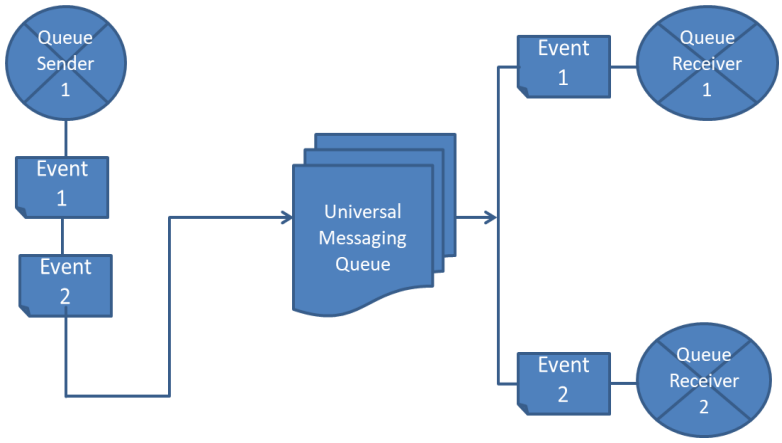

Figure 15: Universal Messaging -- Delivery Mode: Queues

In the delivery model of type "*Channels/Topics*" individual messages delivered to multiple consumers. This basically decouples message producers and consumers which allows deployments ideal for delivering same messages to multiple users. The following figure depicts the behaviour of Channels/Topics. *Channels* are the places where published Data are stored and can be subscribed. [45]

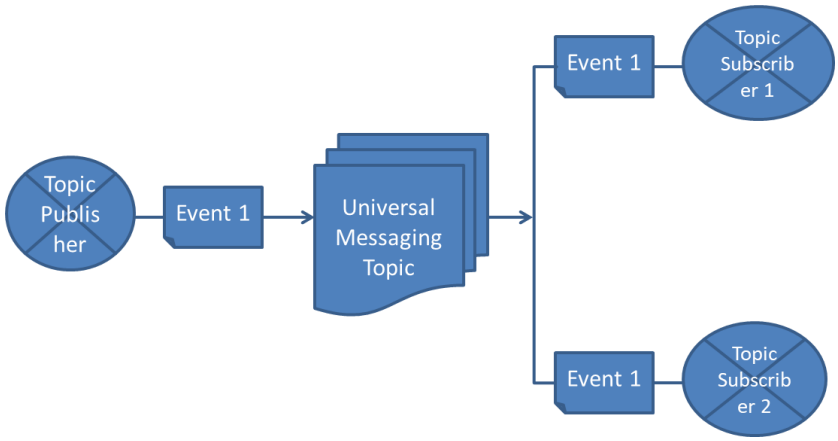

Figure 16: Universal Messaging -- Delivery Mode: Channels/Topics

Using *"Datagroups"* as delivery models, a dynamic structure supporting remote subscription management is enforced on delivery and it permits user membership of one or more groups. Producers of messages are actively in control of group membership which allows scenarios ideal for transparently delivering messages to dynamic categories of users. The following figure depicts the behaviour of *Datagroups*:

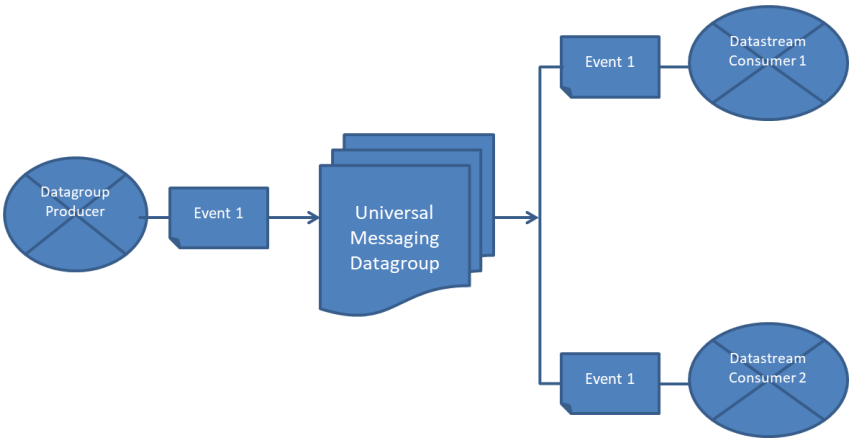

Figure 17: Universal Messaging -- Delivery Mode: Datagroups

Performance of the messaging server can be fine-tuned by setting parameters like *TTL* (time-to-live), *queue capacity* or *channel capacity*. The TTL on a channel specifies how long (in milliseconds) each event is retained on the channel after being published. For example, if you specify a TTL of 10000, the events on the channel will be automatically removed by the server after 10000 milliseconds. However, TTLs need to be carefully defined because just having a simple channel with a TTL of say 1 day (86400000 milliseconds), and you expect to publish a 1k event per second, this channel alone will consume approximately 86.4MB of memory. However, if your data has a very short lifespan defined by setting a low TTL, then the memory consumption would be much less than it would be with a 1 day TTL.

Channel or queue capacity describes the number of events that can be stored in the channel or queue. If the size is set to 0 this means an unlimited number of events can be stored which is dangerous as it can swiftly cause system degradation and performance issues due to memory

restrictions. Hence capacity must be carefully set if the maximum number of events is reached the channel or queue cannot receive any new events till there are events deleted.

In the most cases it is meaningful to set a TTL if a channel or queue capacity will be set, that the channel don't stop receiving event. A rule of thumb would be

*TTL * expected events < channel capacity*

In this case it is like a kill switch that the server does not run full on events if there is an unexpected high event rate which can possibly not handed.

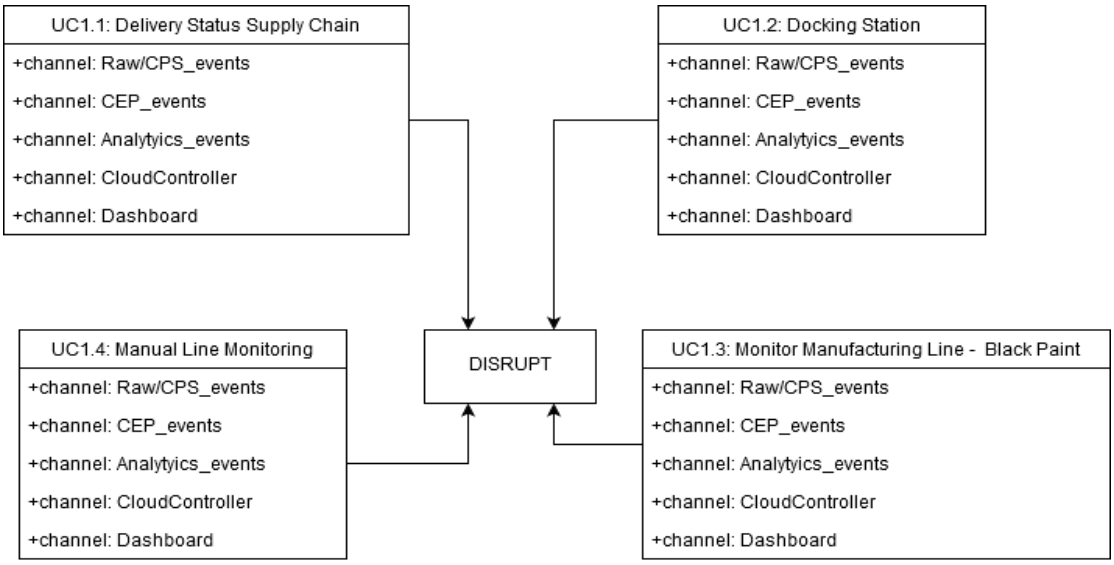

Figure 18: Excerpt from the Channel Definition on the Message Bus

In DISRUPT, the messaging channel operates on TCP/IP enabled networks and is mainly supported by the messaging bus, an intermediary that supports multiple communication interfaces, each one defined by a protocol and a port. Messaging relies on transferring of data through a common channel in an asynchronous manner. Messages are sent from an application without knowing if the other application is up and ready at the same time. In the same way, an application that was not connected to the messaging system for whatever reason, might receive a message, which was sent some time ago right after it successfully reconnects to the messaging system. Asynchronous messaging strengthens the decoupling of the applications and allows integrators to choose between broadcasting messages to multiple receivers, routing a message to one of many receivers, or other topologies.

A messaging event is the object that is published to the messaging bus (channel, queue, peer-to-peer, or data group) and is passed on to subscribed consumers as required. An event may be a simple byte array or contain more complex structures, such as complex data structures, dictionaries, Google Protocol Buffers, JSON, or JMS events. The underlying messaging model is publish/subscribe [39] an asynchronous messaging model where the sender (publisher) of a message and the consumer (subscriber) of a message are decoupled. The two simply share a common channel/topic. The publisher published data to the channel, which exists within the messaging bus. As messages arrive on a channel, the server automatically sends them onto all consumers subscribed to the channel [c.f. D4.1: Data Collection Framework – Section 3.3.4].

Every Scenario is assigned its own Channels: 1 channel where all the raw data from CPS is published. Another one where all the CEP Events are which are based on the CPS raw data. A third channel on which the aggregated Analytics events are stored; and eventually 2 dedicated channels for the CloudController and the KPI Dashboard where events are passed to for visualization and dashboarding.

To consume events from a channel asynchronously you need to supply an object that the session can push the messages into. This object must implement *nEventListener*. The server will send events to the client as soon as they are published to the channel. The session will read events from its connection and push them into the user space via the nEventListener. Once you have called add subscriber, your go call-back will be invoked in a different thread, so you don't need to do anything in the thread which added the subscriber. A sample code is shown in the following table.

Table 2: Messaging Client Subscribing to a Channel for asynchronous messaging

```
public class Subscriber implements nEventListener {
            public Subscriber(nChannel c) {
            // add the subscriber, you can also supply filters
            // e.g. "CCY = 'USD-GBP'" to the addSubscriber method call
                    c.addSubscriber(this);
            }
            // callback for consuming events from a channel
            public void go(nConsumeEvent e) {
                    nEventProperties props = e.getProperties();
                    String ccy = props.getString("CCY");
                    double bid = props.getDouble("Bid");
            // now perform application logic with the data received
            }
          }
```

4.4 Supply Chain Monitoring and Truck Location

This scenario, applied at CRF, monitors truck movements from different suppliers to a central warehouse where containers are reordered according to the actual production plan and eventually shipped to the plant. Truck movements are based on real, recorded data along the checkpoints of the respective routes. Once at the warehouse, the truck monitoring is continued until the plant is reached. For the reloading at the warehouse, the minimum time taken from historic data is taken.

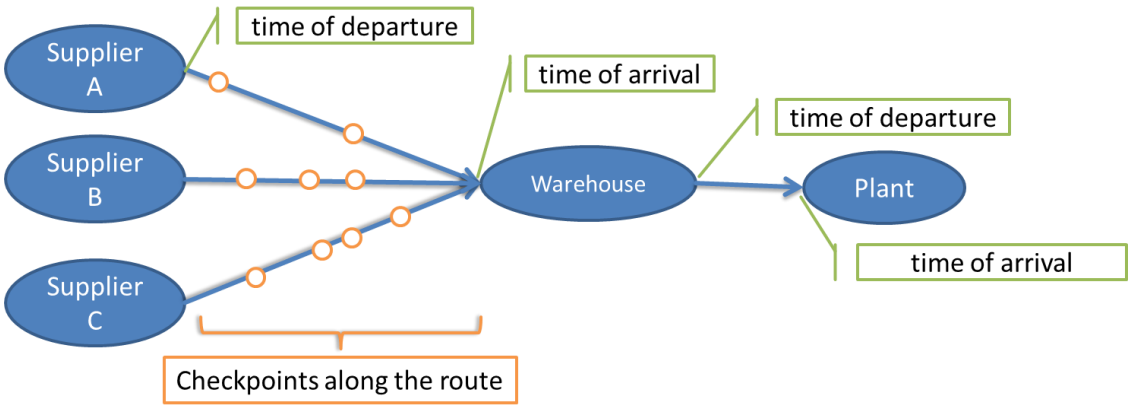

Figure 19: The Supply Chain Monitoring Scenario

The important key figure of interest in this scenario - which is continuously updated - is the estimated time of arrival (ETA) which is continuously checked against the required time of arrival (RTA) directly taken from the production schedule. Whenever ETA is later than RTA an alarm is issued by the CEP to inform the KPI dashboard and CloudBoard about this incident. In addition to the historic data, information from actual and most recent traffic is taken into the equation to provoke delays as well as current weather conditions in terms of weather warnings. The CEP thereby produces 2 kinds of external events on the messaging bus: a continuously updated stream of information which is consumed by the KPI Dashboard as well as Traffic Alarms events only generated in case of serious issues which are consumed by the Cloud Board (c.f. Figure 20 below). The CEP makes use of external services for traffic information and weather conditions (c.f. respective subsections below) to ensure real-time operation on most recent data.

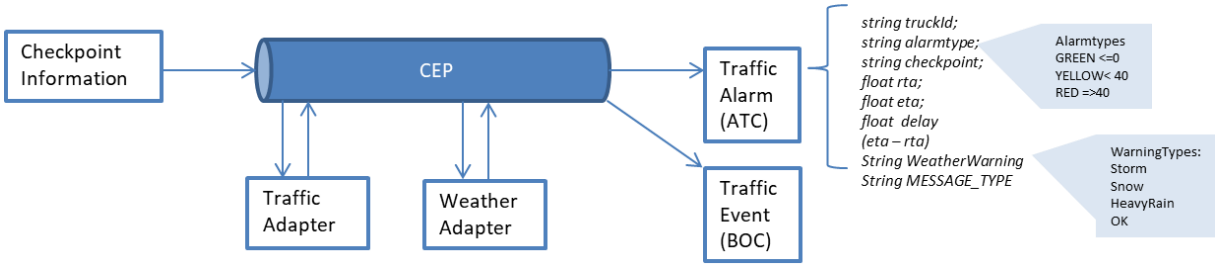

Figure 20: Inputs and Sources for the CEP to react on

4.5 Traffic Information Adapter

To check the current traffic situation for CRF, the HERE routing API[8] is used which returns the normal duration of the route and also the calculated time assuming current traffic conditions. Out of the difference of these two values we get the expected delay. This delay gets added on the mean duration of the past Truck movements on the route.

The traffic and rerouting information is taken from a real-time traffic service [41]. It provides precise instructions to a destination using various transport modes (e.g., car, truck, public transit, bicycle) and leveraging different algorithms (e.g., matrix, isoline routing). The following code sample shows a

[8] HERE, https://developer.here.com/, Last Accessed on 27.02.2019

REST request on the HERE API for the fastest truck routing from GPS coordinate 52.5,13.4 to 52.5,13.45 while considering actual traffic conditions.

Table 3: Sample Call of the REST service provided by HERE

```
https://route.api.here.com/routing/7.2/calculateroute.json
      ?app_id={YOUR_APP_ID}
      &app_code={YOUR_APP_CODE}
      &waypoint0=geo!52.5,13.4
      &waypoint1=geo!52.5,13.45
      &mode=fastest;truck;traffic:enabled
```

This information is frequently fetched for predefined itineraries to provide the best feasible routing along the transport route. The weather service of the here API returns a lot of different weather conditions. Based on the conditions weather warnings are changed, the ETA is adjusted when there are severe storms, snow or other serious weather forecasts. The actual weather forecast is taken from the open service provided by [40]. To indicate weather warnings or upcoming risky road conditions. It gives insights into real-time weather forecasts, alerts, and astronomical info for any location and the services provides a REST interface with parameters according to the following table (c.f. Table below).

Table 4: Parameters on the REST Service from HERE

Element	Value/Example	Description
Base URL	`https://weather.api.here.com`	Production environment only
Path	`/weather/1.0/`	
Resource	`report`	Uses HTTP GET
Format specifier	`.xml` `.json`	We recommend that you specify whether you expect XML or JSON output. The default output format is XML.
Parameters	`product` `name`	For more information on query parameters in general, see the sub-sections under Resources.
Application Id	`&app_id={YOUR_APP_ID}`	Substitute your own unique app_id
Application Code	`&app_code={YOUR_APP_CODE}`	Substitute your own unique app_code

4.6 Machine Emulation and Error Simulation

The focus on this scenario is on machine reliability and failures taken from the ARCELIK case. The situation for CRF is very similar but is based on different machine names and failure rates accordingly. Data is based on specific machines with their MachineID and their respective failure rates taken from the DISRUPT database. The time span of how long a machine runs is based on an exponential function with its failure rate as the threshold. The starting point at time 0 is randomized to ensure that there are different failures rates and times since the last machine breakdown. If a machine is

broken, the time it is not available is generated by the mean time to repair, a predefined figure which depends very much on the type of the machine. Every machine has a momentary throughput of material which is also generated. Thereby emulated events are sent to the messaging bus via a java service. There CEP is consuming them with the event definition in Table 2 via JMS consumer. The following figure depicts the event definition of a machine which will be instantiated during the emulation phase.

Table 5: Event definition of a machine

```
event Machine {
            string MachineID; //According to name DB definition
            float TimeSinceLastBreakdown;
            boolean breakdown;
            float jph;
 }
```

The CEP monitors the machines' throughput and aggregates it over the time window of the last hour therefore a machine status event is published. If there is a breakdown detected a machine failure event is triggered.

Table 6: Abstract of Machine Monitor

```
monitor MachineMonitor {
[...]
on all Machine() as m{
      dictionary<string, string> emptyDictionary:=
      new dictionary <string,string>;
       float breakd ;
              if(m.breakdown) {
                      breakd := 1.0;
              }else {
                      breakd := 0.0;
                     }
       send com.industry.analytics.Data (m.MachineID,
      "COMPUTED",m.MachineID,m.getTime().toDecimal(),
      m.jph.toDecimal(),m.jph.toString(),m.TimeSinceLastBreakdown,
      breakd,3.0, emptyDictionary) to m.MachineID;

              if(not checkDict.hasKey(m.MachineID)) {
                send Analytic("Average",[m.MachineID],["AVERAGE"],
                {"timeWindow":"3600.0"}) to "";
                 checkDict.add(m.MachineID,m.MachineID);
                     }

                     }
[...]
```

An overview of all inputs and respective output channels is shown in the following figure.

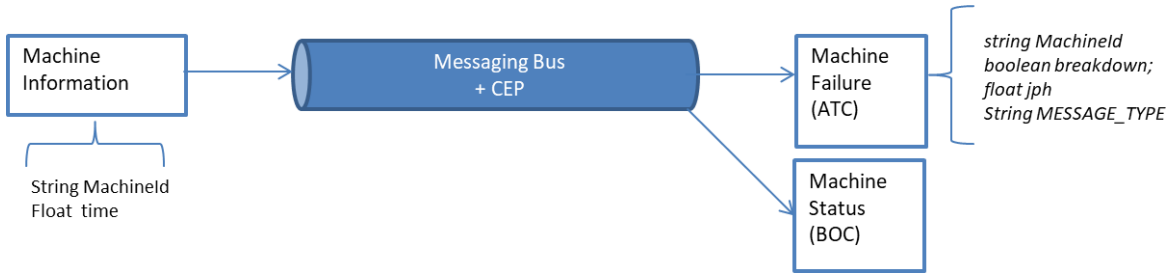

Figure 21: Inputs and Sources for the CEP in the Machine Failure Scenario

5 DISRUPT Knowledge Layer

5.1 Technical Architecture

In this chapter the technical architecture and environment of the DISRUPT Knowledge layer is presented. An overview is provided explaining all involved components. Every component is explained how to install, configure and use it, and in the end the specific component can be extended and customized.

5.1.1 Overview

The Knowledge layer is mainly composed of 3 components working together as shown in Figure 22:

- A specific meta-model named "Cause and Effect" used first to collect all the information on the Goals, KPIs and Metrics of the domain with their relations and then, using the same information, to configure the KPI Dashboard component.
- A micro-service framework named "Olive" used to create services that connect with the data provider and return the measure relative to a specific KPI or Metric.
- A Dashboard that combines the information from the model with the measures from the micro-services and visualize them in a customizable widget-based interface.

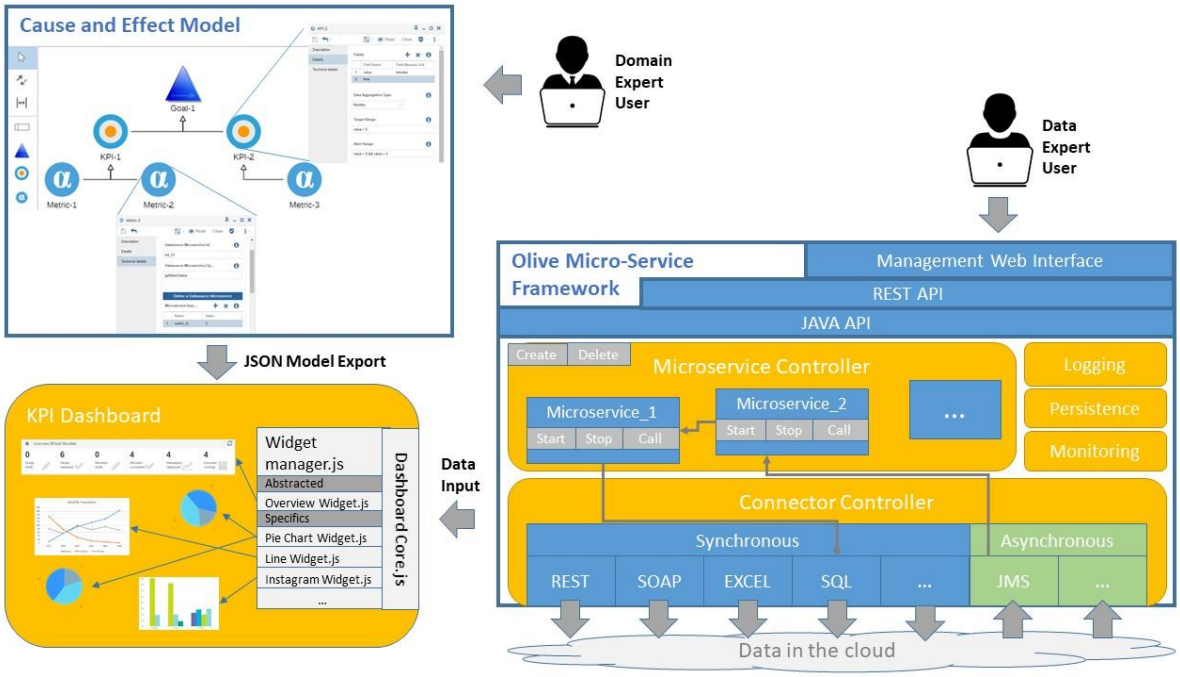

Figure 22: DISRUPT Knowledge Layer Architecture

In the following, a detailed description for every component is provided.

5.1.1.1 Meta-model

The Cause and Effect Meta-model contains all the relevant information necessary to configure the dashboard, instructing it on how to evaluate goals, how to calculate KPIs based on metrics or Sub-KPIs and where to retrieve metric measures. In particular, the most relevant classes in this meta-model are the following:

- Goal: This class is used to represent a business or an operational goal. This class is able to have dependencies from other Goals or KPIs and is able to collect the following attributes:
  - Aggregation Type: Specify when this goal is evaluated over time through the aggregation of sub-Goals or Sub-KPIs. This value usually has to reflect the aggregation type of its sub-objects. Possible values are: Not Aggregated, Yearly, Half-Yearly, Quarterly, and Monthly.
  - Goal Definition Algorithm: Define which algorithm to use in order to combine sub-goals and sub-KPIs to return the current Goal status. Possible values are:
    - All dependencies succeed: The goal succeeds only if all the sub-goals and Sub-KPIs succeed.
    - At least one dependency succeeds: The goal succeeds if at least one of the sub-goals or sub-KPIs succeed.
    - Custom: The goal status is evaluated using the custom algorithm provided
  - Custom JS Algorithm: If the Goal Definition Algorithm is Custom this must contain a JavaScript code that return the Goal status as JSON in the following format:

```
{
  status: -1, // -1=Goal failure, 0=Goal unevaluable, 1=Goal success
  moreInfo: {
    time: '…',
    …
  }
}
```

  - More Info: A list of key value pair representing additional custom information on the Goal.

- KPI: This class is used to represent a Key Performance Indicator and can have dependencies on other KPIs or metrics. The relevant attributes for this class are:
  - Fields: This attribute describe the data format of the KPI and contain a list of Field Name and Field Measure Unit pairs indicating the kind of measure available in this KPI. Common Fields names are "value" with relative measure unit (es: minutes, or days or cars depending on the measure) indicating the main measure and "time" representing the time instant relative to the value. These fields have a key role and must be filled in relation to the available measures.
  - Data Aggregation Type: Specify when these KPIs are evaluated over time through the aggregation of sub-KPIs or sub-metrics. This value usually has to reflect the aggregation type of its sub-objects. Possible values are: Not Aggregated, Yearly, Half-Yearly, Quarterly, and Monthly.

- Target Range: An expression representing the range in which the measures of this KPI are good. If the measure is out of the range the KPI is considered failing. The expression will use the field names defined in the fields attribute in order to create a range. A typical example of Target range is "`value < 10`" where "`value`" is a defined field name. In this example every time the value is lower than 10 then the KPI is considered good, else it is considered as failing. The expression can be any JS expression so complex range definitions are possible like: "value>0 && value <=5" or "`value==10 || value==15`" or "`value < 10 && new Date(time) < new Date().getDate()-1`" that combine together a value range with information on time.
- Alerts Range: A list of ranges indicating different levels of alerts. Every element in the list contains a range like in the Target Range attribute and a colour indicating the severity of the Alert.
- More Info: A list of key value pair representing additional custom information on the KPI.

- Metric: This class is used to represent a metric as a raw measure. A metric does not have information on target and alert Ranges like KPIs but contain information about where to retrieve a measure (in terms of which micro-service is able to return the required measure) and how to combine sub-metrics together in order to calculate its measure. It can have dependencies only on other metrics. The relevant attributes for this class are:
  - Data-source Micro-service Id: the ID of a micro-service available in the Olive Micro-service Framework component. More details on this will be provided in the next section.
  - Data-source Micro-service Operation Id: The ID of a specific operation for the specified micro-service. The operation can be seen as a function that returns the measure. The measure returned must be in the following JSON format in order to be recognized as valid:

```
{
  columns: ['value', 'instant', ...],
  data: [{
      value: '10',
      instant: '2019-01-30T10:40:50'
    }, ...
  ]
}
```

    In case the format returned is not aligned, it is possible to provide a custom JS algorithm that will process the result and return the right measure. More details on this will be provided in the next section.
  - Micro-service Inputs: A List of Input names and values that the micro-service operation require in order to work and can be considered as the parameter to provide the operation in order to be performed

- Custom JS Algorithm: A custom JavaScript algorithm can be provided in order to fix the data format returned by a micro-service or combine the measures of sub-metrics. The measure returned must be a JSON in the following format:

```
{
  columns: ['value', 'instant', ...],
  data: [{
      value: '10',
      instant: '2019-01-30T10:40:50'
    }, ...
  ]
}
```

The cause-and-effect meta-model has been implemented in the ADOxx Platform along with the mechanism to export the model in the JSON format accepted by the KPI Dashboard.

5.1.1.2 Olive Micro-Service Framework

The Olive micro-service framework allows to create micro-service via configuration. This macro-component is used to configure micro-services that connect to a generic data-source and return measure information. The framework can be used locally via a JAVA API or remotely via a REST API. A Web UI is also available allowing to create and manage micro-services from the browser.

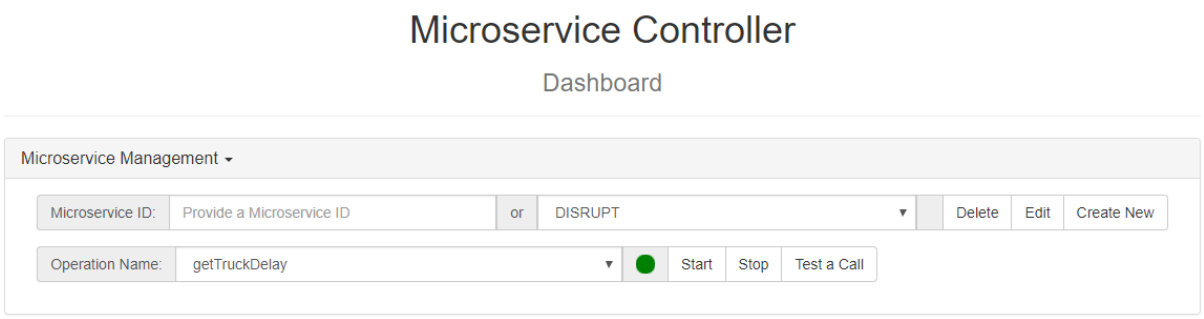

Figure 23: Olive Micro-service Framework Web UI

A micro-service in Olive is uniquely identified by an ID, generated automatically in the creation phase. A micro-service is composed of operations. An operation is identified by is name and can be considered as the function that perform all the work. A single micro-service operation in the future will be considered as a micro-service itself in order to simplify the explanation. In that way a micro-service is uniquely identified by an ID and its operation name.

The key part of a micro-service is the definition of its connector. The connectors are managed by a controller and represent the way in which the micro-service communicates to the external world and, more in general, performs its operation. The connectors are plugin based and can be extended based on specific needs. In particular, the connectors can be categorized into two groups: Synchronous and Asynchronous. A synchronous connector is able to perform a direct call to an endpoint or perform an operation and return immediately the result, while an asynchronous connector is used in all the cases where we must wait for the data. The asynchronous connector, due to its nature, requires a handler that in our context means another micro-service managing the data obtained.

A synchronous connector is defined by a Java-interface that requires a starting method with operations that initialize the connector, a stopping method with operations that finalizes the connector, and a calling method that will perform a call and return data. In an asynchronous connector this last method is substituted with a handler method that is called when the data are available. These methods of the controller will be reflected in the creation of a micro-service, meaning that the parameters required by the starting and calling methods of the connector have to be provided in the micro-service configuration. Additionally, the specific inputs required for the micro-service can be configured and mapped to fill specific parts of the parameter for the connector's method call. Once a micro-service has been created, all of its operations can be started, called, stopped and checked for the status separately.

5.1.1.3 KPI Dashboard

The KPI Dashboard is a web component able to merge the goals, KPIs and metrics information provided by the cause-and-effect Model, with the measures returned by the micro-service framework and shows the results in a widget-based web interface.

The KPI dashboard so will offer mainly visualization and rendering functionalities provided by generic widgets. The widgets did not implement logics so they can be adapted in every context due to their configurability. For example, when a new line chart widget is added to the dashboard, the data to be used in the X- and Y-axis will be asked. The dashboard will allow specification of this in the fields of values from the KPIs and metrics so that the widget will use them without knowing anything more than its visualization logic.

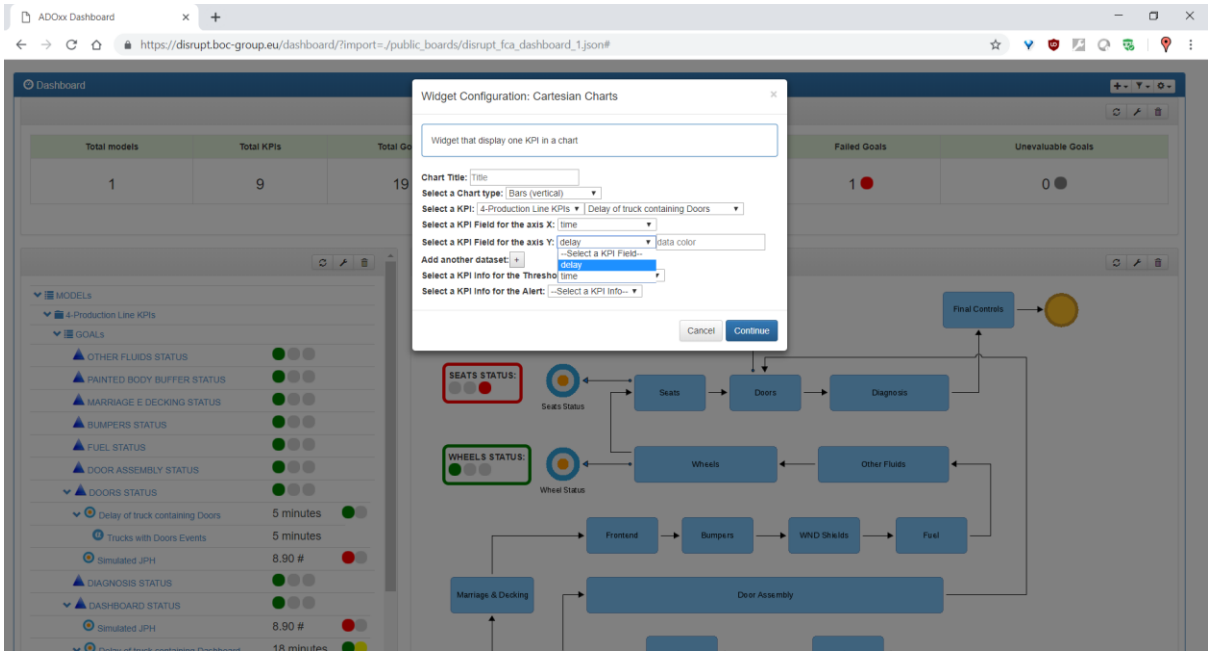

Figure 24: KPI Dashboard – New Widget creation

Once a cause-and-effect model has been imported in the dashboard through its configuration button in the menu, the user can start to add the desired widgets. Currently the available widgets are:

- Table Chart: Visualize the data in a simple tabular format

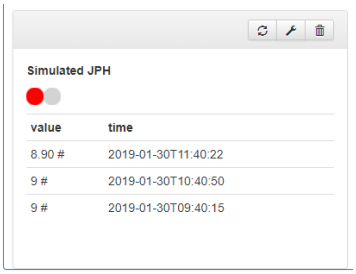

Figure 25: Table Chart Widget

- Table Overview: Visualize an overview of all the defined KPIs and goals with their status

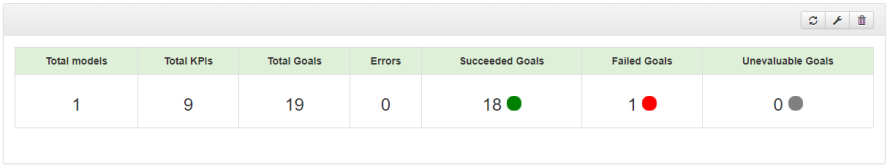

Figure 26: Table Overview Widget

- Table Overview Detailed: Visualize a more detailed overview of all the defined KPIs and goals in a three based view.

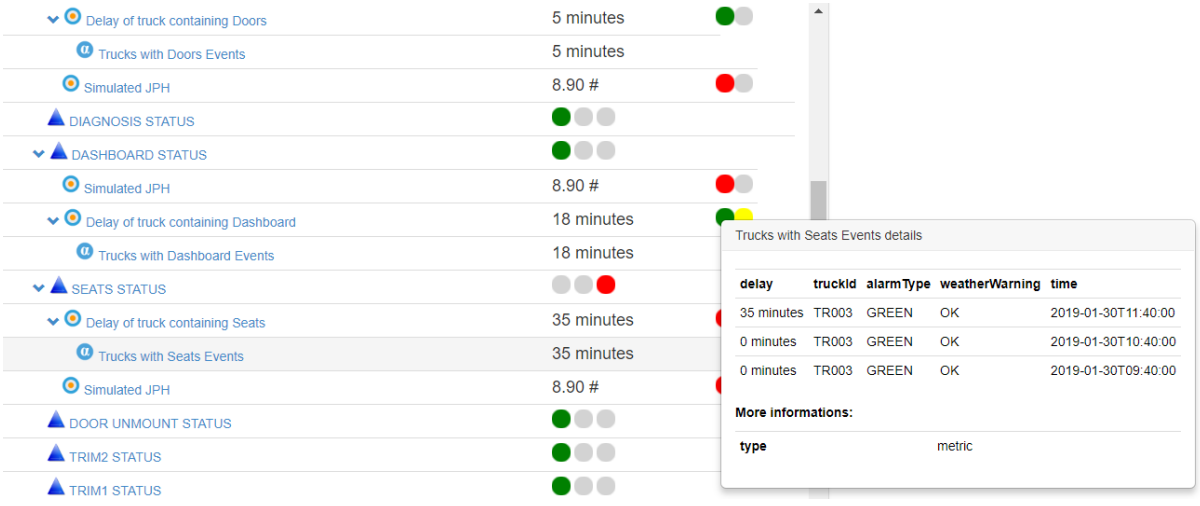

Figure 27: Table Overview Detailed Widget

- Image Map: Visualize custom KPIs and goals over a configurable image

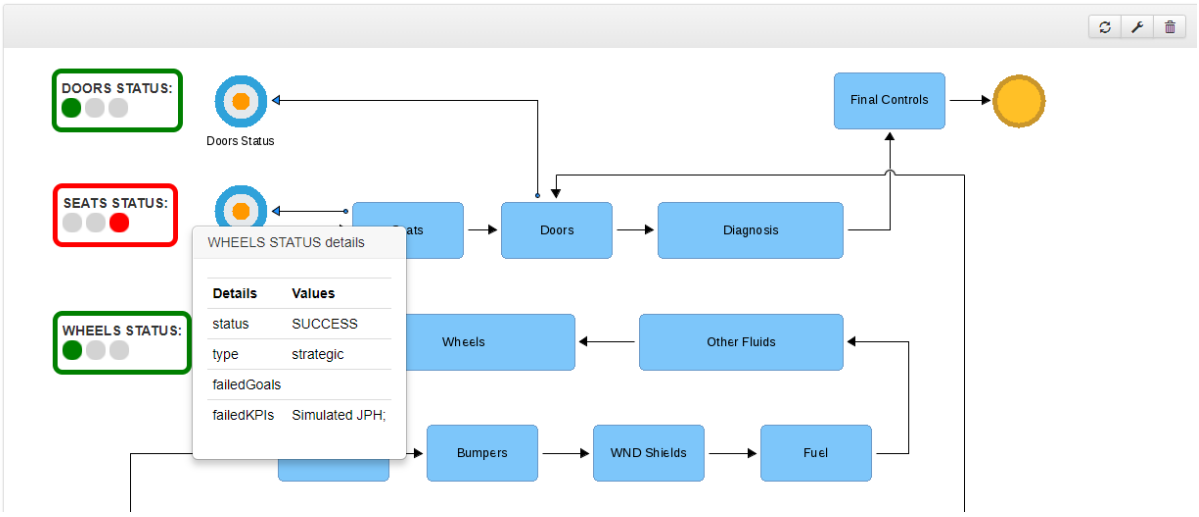

Figure 28: Image Map Widget

- Cartesian Charts: Visualize one KPI or metric value over different Cartesian charts

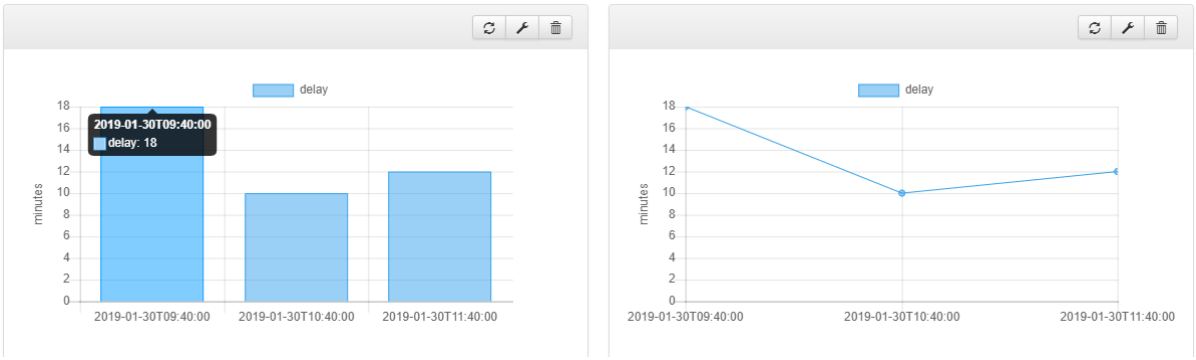

Figure 29: Cartesian Chart Widget

Every widget can be moved by means of Drag-&-Drop, reconfigured and deleted. Every dashboard can be locally saved and exported, and parameters like auto-update interval time can be configured. The Dashboard also contains a filtering engine that allows to visualise only the widgets with the KPIs or goals of interest, hiding the rest. This can be done via the menu filtering through a specific KPI or goal, or through a specific value. Every widget can decide to manage the filtering in its favourite way so a widget may react on filtering events generated by another widget allowing customized behaviours.

5.1.2 Usage Manual

In the following chapters all the materials about usage, installation requirements and instructions for every specific component will be collected and provided.

5.1.2.1 Usage of Cause and Effect Model

All the information about system requirements, installation and usage instructions for the cause-and-effect-model in the ADOxx platform are available here: https://www.adoxx.org/live/dashboard-version-2. Please refer to this page for the last updated code and installation manual.

5.1.2.2 Usage of Olive Micro-service Framework

All the information about system requirements, Installation and usage instructions for the Olive micro-service framework are available here: https://www.adoxx.org/live/olive. Please refer to this page for the last updated code and installation manual.

5.1.2.3 Usage of KPI Dashboard

All the information about system requirements, Installation and usage instructions for the KPI dashboard are available here: https://www.adoxx.org/live/dashboard-version-2. Please refer to this page for the last updated code and installation manual.

5.1.3 Extend

In the following chapters all the materials and instructions about extensions for every specific component will be collected and provided.

5.1.3.1 Extend the Cause and Effect Meta-model

All the Documentation including example and tutorial relative on the extension of the cause-and-effect model can be found here: https://www.adoxx.org/live/dashboard-extend

Instructions on how to start editing a meta-model in ADOxx are also extensively provided by the ADOxx community in form of

- FAQs: https://www.adoxx.org/live/faq/-/message_boards/category/16830
- And video tutorial: https://www.youtube.com/user/adoxxorg/playlists?shelf_id=8&view=50&sort=dd

5.1.3.2 Extend the Olive micro-service framework

All information about customization of the Olive Micro-service framework is available here: https://www.adoxx.org/live/olive.

Please refer to this page for the last updated code and instructions.

5.1.3.3 Extend of the KPI Dashboard

All information about customizations for the KPI Dashboard is available here: https://www.adoxx.org/live/dashboard-extend.

Please refer to this page for the last updated code and instructions.

5.2 Approach

BOC's semantic lifting of data pertains to the framework of a balanced scorecard in order to extract individual business objectives from different scenarios in the DISRUPT project.

5.2.1 Principle of the Balanced Scorecard

The Scorecard is a principle of a balanced controlling system that distinguishes different perspectives, different goals and measurable KPIs with the aim to achieve a balanced overall success. This principle became popular when it was applied to optimize the financial success via the so-called Balanced Scorecard. Since then, different variations from the underlying flexible principle in using a multi-view based monitoring system has been created. We are using a configuration on performance monitoring, which has been developed in the security domain, were success factors other than the financial one, where important. We apply this principle now for the industry in order to elaborate, if such an approach is feasible for the aim of Factory of the Future.

Every Scorecard needs a User to whom the Scorecard is reporting to.

When constructing a scorecard it is necessary to define:

- Who, needs
- What and
- Why
- When

In the following Scorecard (Figure 30) we interpreted the need that:

- The Plant Engineer, needs
- To check if decisions that have to be done in order to react on disruption
- Are in line with the overall strategy of the company, hence support the achieve goals,
- Whereas the Scorecard is needed at the time the decision has to be taken, or even before by comparing alternatives and their estimated impact on the goals.

5.2.2 Success Factors

First, the four perspectives are defined. Although the scorecard approach allows to add perspectives – which is interpreted as a synonym for viewpoint – we prefer to stick to the number four, which has proven to work fine.

	Cooperation	Production	Innovation
Product - Impact	I.3) Coopeation and Networked Enterprises	I.1.a) Impact on Quantitiy I.1.b) Impact on Costs I.1.c) Impact on Quality	I.2.a) Innovation on Productivity I.2.b) Innovation on Costs I.2.c) Innovaation on Quality
Effectiveness	II.3) Supply chain related issues	II.1.a) Processes on Quantitiy II.1.b) Processes on Costs II.1.c) Processes on Quality	II.2.a) Process Improvement on Productivity II.2.b) Process Improvement on Costs II.2.c) Process Improvement on Quality
Capabilities and Competence	III.3) Partner related issues	III.1.a) Capabilities III.1.b) Competence	III.2.a) Capabilities improvements III.2.b) Competence improvements
Ressources	IV) Financial Resources, Infrastructure, Mashines, Material, Know-How and Information related Issues		

Figure 30: Perspectives on Scorecard

The above figure introduces the proposed perspectives for our scorecard for industry.

First, the "Product Impact" viewpoint is introduced that measures the intended main goal. In our case, this is the quantity, the quality, and the cost of a produced unit.

Second is the "Effectiveness" viewpoint that is also understood as the "process perspective." This perspective observes the efficiency of the processes.

Third is the "Capabilities and Competence" viewpoint. This is an adaptation which we introduce, as typically it is the resource allocation within an organisation. In our case, we see man and machine capacity and capabilities as the perspective that describes the best potential of a factory.

Forth, is the "Resource" viewpoint, that introduces an outside view on the system concerning budget, legal framework, material, infrastructure, information, know how, etc.

Those four perspectives, which are based on the traditional scorecard approaches and adapted where appropriate to fit the needs of factories, is complemented with three pillars.

The major and most important pillar is the "production pillar" that is in the centre. This pillar is concerned with production relevant monitoring. The second pillar, the "cooperation pillar," is concerned with the interaction with other enterprises. In the production field, we consider cooperation, supply chains and networked enterprises as well as freelancers, start-ups and associated SMEs in this pillar. The third pillar is concerned with "innovation." It is shown at the right in the figure. Change goals, new market, continuous improvement, new skills and competences are monitored in this pillar. Setting up such a raster of 3*3 plus one – three perspectives * three pillars plus one resources that are seen as a bottleneck – enables focusing on a particular part monitoring.

5.2.3 Cause and Effect Model

The cause and effect model translates the critical success factor model into a specification map of measurable indicators and the consequences to the previously defined goals.

It consists of:

1. Perspectives that are copied from the Critical Success Factors
2. Critical Success Factors become goals (the blue pyramids)
3. Influence factors become Key Performance Indicators KPI (the target sign)
4. High level KPIs may become sub goals (orange circle), or sub-critical success factors become sub goals (orange circle) to better design the scorecard.

5.2.4 Criteria Model

The criteria model corresponds to the aforementioned cause and effect model and specifies the technical access to the data. The complexity in calculating the data can either be delegated to the CEP, which results in a quick operation, but which loses the transparency and abstraction. On the other hand, the complexity of calculating the indicators can be modelled in the scorecard, which results in a slower calculation, but also enables abstraction, documentation and transparency on how the indicators are calculated. It needs to be decided per indicator where to perform the calculation, and therefore where to deal with the complexity.

The criteria model distinguishes between:

- Atomic sensors:
    - These are sensors that should not be divided and are hence "atomic." Similar to the alphabet, they provide the basis of calculation. Typically, a scorecard starts with nearly all sensors as atomic, and while the scorecard evolves, and while more sophisticated KPIs are elaborate, the atomic sensors are exchanges with complex sensors. Each atomic sensor describes the access to the data stream, the URI of the Web-Service, the query or event registration that is necessary to retrieve the data stream. Those sensors will be invoked by the cockpit to load the data stream.
- Complex Sensors:
    - Complex sensors create a data stream out of already existing atomic sensors. For transparency reasons, such complex sensors can be expressed here and enable a certain abstraction and documentation how atomic sensors are used.

5.2.5 Management Dashboards

The management dashboard is the specific part of the knowledge management solution for the responsible decision makers to monitor and analyse whether the goals outlined in the models are being met through a visualisation webpage. This webpage is updated in real-time, displaying the level at which KPIs are being currently met, along with other relevant information for the decision makers such as graphs or maps, depending on what is being visualised.

5.3 Use Case Organization - CRF

5.3.1 Focus Areas

Based on the information in D2.4 and D5.3, we grey out the parts that we do not focus on and leave the production as well as the cooperation on process level – the logistics in focus.

	Cooperation	Production	Innovation
Product - Impact	I.3) Coopeation and Networked Enterprises	I.1.a) Impact on Quantitiy I.1.b) Impact on Costs I.1.c) Impact on Quality	I.2.a) Innovation on Productivity I.2.b) Innovation on Costs I.2.c) Innovaation on Quality
Effectiveness	II.3) Supply chain related issues	II.1.a) Processes on Quantitiy II.1.b) Processes on Costs II.1.c) Processes on Quality	II.2.a) Process Improvement on Productivity II.2.b) Process Improvement on Costs II.2.c) Process Improvement on Quality
Capabilities and Competence	III.3) Partner related issues	III.1.a) Capabilities III.1.b) Competence	III.2.a) Capabilities improvements III.2.b) Competence improvements
Ressources	IV) Financial Resources, Infrastructure, Mashines, Material, Know-How and Information related Issues		

Figure 31: Perspectives for CRF with focus

5.3.2 Success Factors

The next step is to fill the focus area with goals.

This is an interpretative step and hence needs to be performed in workshop sessions with the domain experts. The current proposal is based on an internal workshop without domain experts for demonstration purposes. Hence changes in the interpretation are expected.

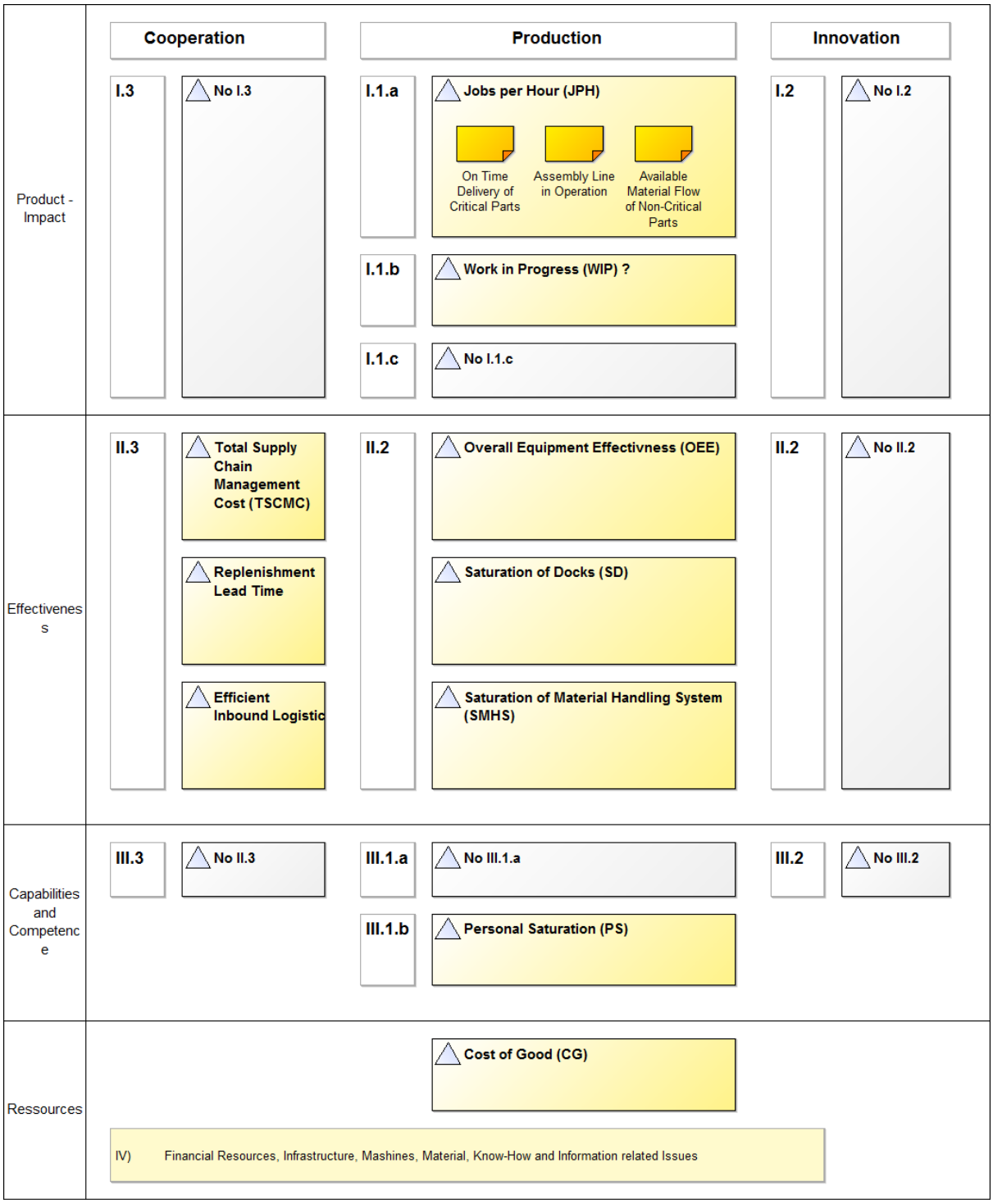

Figure 32: Initial CRF Critical Success Factor Model – without domain expert interpretation

Figure 32 introduces the initial Critical Success Factor model without domain expert interpretation. Typically, the identification of critical susses factors and their corresponding influence factors are performed in several iterations.

For explanation and demonstration purposes, we explain "**Jobs per Hours**":

- As it is a key impact indicator, it is placed in the highest layer

- JpH needs to be renamed following the SMART principle
  - Specific defined
  - Measurable
  - Ambiguous
  - Realistic
  - Time dependent

  Possible Name: "Keeping JpH constant at defined Speed" with should value 90 pph

- We assume three influencing factors on JpH
  - On time delivery of critical parts
  - Assembly line in operation (man, machine, material)
  - Available Material Flow for also the non-critical parts (but with more possible compensation actions)

In a similar way in order to identify influencing factors for critical success factors, the whole cockpit has to be filled.

5.3.3 Cause and Effect Model

In order to explain the idea, a detail on JpH is depicted, before introducing the whole view. Figure 33 shows a detail of a possible construction of JpH measurement.

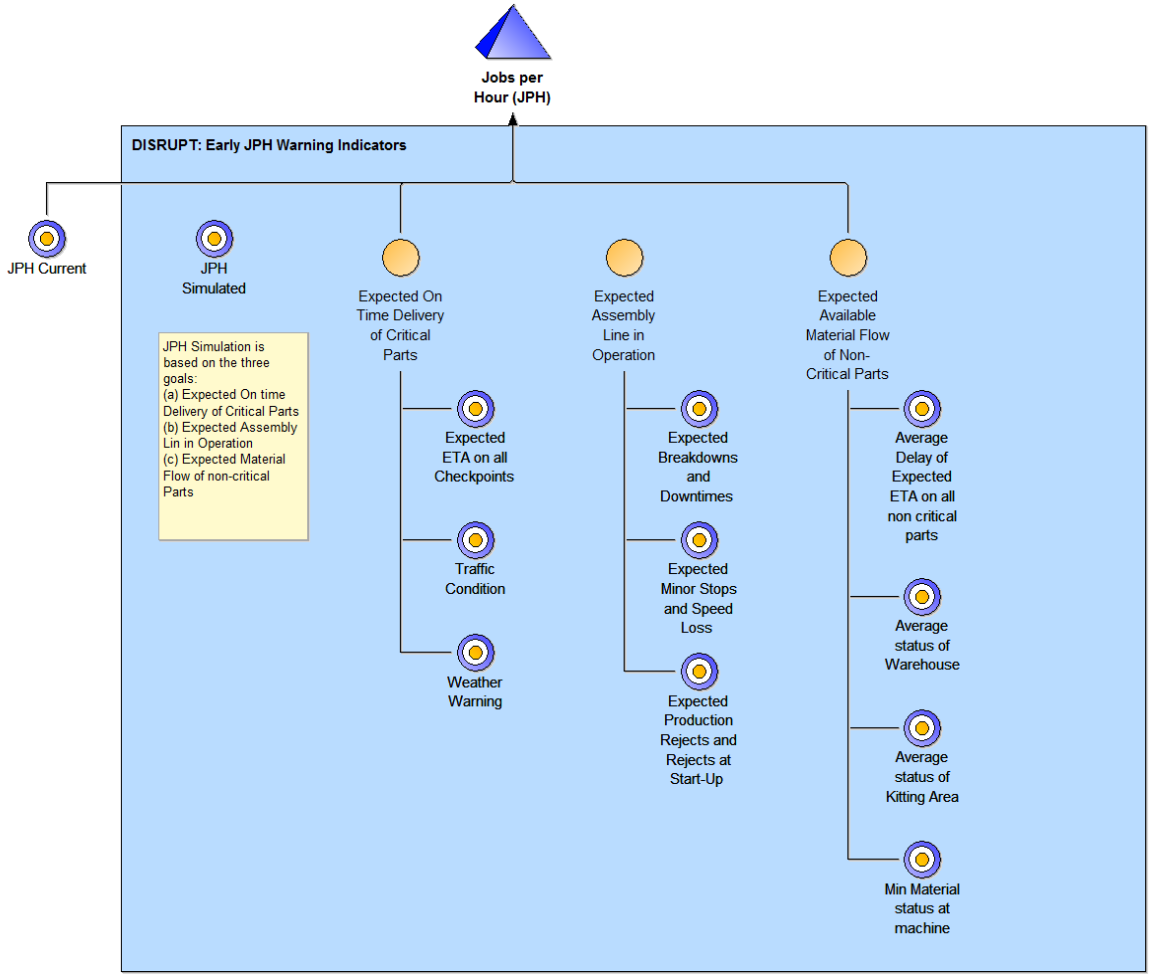

Figure 33: Initial Cause and Effect Model – CRF – Detail of JpH

The "JpH Current" value is collected in form of an integer and is modelled in the top left of the figure. There is "JpH Current":

- a "should" value (90),
- a current is value (to be delivered from the event bus),
- there is a measure (integer),
- a threshold from the bottom (less is not allowed, between green, yellow and red tolerance (to be defined with domain experts)

In addition to this – which is marked with the blue box that indicates the added value of DISRUPT – there is another JpH – Simulated that shares the same should value, measure, threshold and tolerance but the current value is delivered from the Simulation engine.

In order to provide data for the simulation, we designed possible forecast scenarios.

- Expected On Time Delivery of Critical Parts
- Expected Assembly Line in Operation
- Expected Available Material Flow of Non-Critical Parts

The current scorecard does not include "JpH Simulated" and "Expected Line Operation," therefore there is now a connection between the goal, the corresponding KPI "JpH Simulated," and the sub-goal – "Expected Line Operation." Hence, in the interpretation of the model while generating the cockpit, those elements are currently not considered. In case they will be connected, they become activated.

In the following, we analyse the operative goal "**Expected On Time Delivery of Critical Parts":**

- Expected ETA (Expected time of Arrival) at checkpoints
  - Each critical part can be identified per truck
  - Each truck can be identified per ETA per checkpoints
  - Each checkpoint is a defined node during the route that provides an insight on the expected arrival at the plant

  This indicator is calculated by CEP, as getting all the traffic logs from the truck and by semantically lifting that truck information with the critical parts. Then, the critical parts can be tracked and delays at the checkpoints raise an alert.

  The indicator turns yellow or red, in case to identify which critical part is meant, the user needs to drill down the cloud board or the CEP cockpit.
- Traffic conditions
  - Traffic warnings can be accessed, like:
    - https://www.asfinag.at/traffic/travel-time/
    - https://routenplaner.asfinag.at/trafficmap.html
    - https://routenplaner.asfinag.at/reisezeit.html
    - https://www.asfinag.at/traffic/webcams/?tabActive=favorites

  Based on such open data, there is an indication if there will be an additional delay based on the current information form the tracked checkpoints. This can be processed in the CEP.
- Weather conditions
  - Weather warnings can be accessed, like

- https://at.wetter.com/wetterwarnungen/oesterreich/
- https://at.wetter.com/wetterkarten/gewitterrisiko/
- https://at.wetter.com/oesterreich/EUAT.html
- http://netzwerk.wetter.com/?_ga=2.88780063.582774134.1542288420-765379783.1542288420

Based on such open data, there is an indication if there will be an additional delay based on weather conditions.

In addition, we can analyse the non-critical parts and observe (simulate) the material status at the machine, the kitting area, the warehouse or at the docks. This part of non-critical parts needs to be further elaborated, especially regarding what is needed for proper decision making.

The idea is to not only calculate the aforementioned estimations on influence factors of on time delivery of critical parts, but ideally, to introduce those figures to the simulation – in order to run a simulation based on those expectations - and estimate the impact on JpH.

The whole blue area is introduced as a possible improvement of DISRUPT with its decision support system.

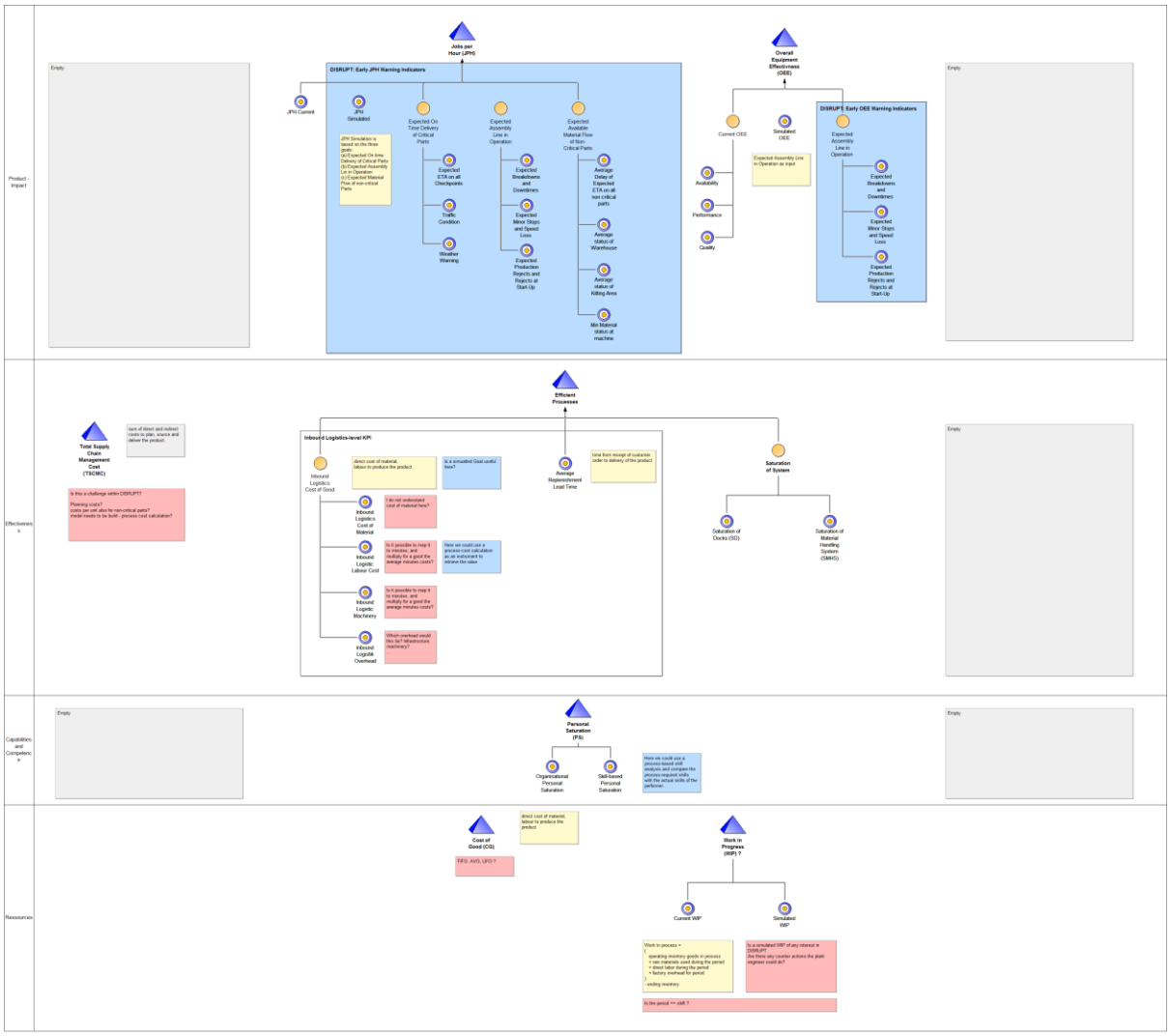

Figure 34: Initial Cause and Effect Model – CRF – Overview

Figure 34 indicates how the whole cause-and-effect model for a scorecard can look like. Although there is the aim to provide a complete overview, the benefit of using DISRUPT technologies are highlighted with the blue boxes to demonstrate how decision making can be supported. Some of the goals need clarification on whether they are relevant for decision support. Some of the fields will not be considered for the scorecard.

5.3.4 Criteria Model

Figure 35 shows the initial criteria model for the aforementioned scorecard.

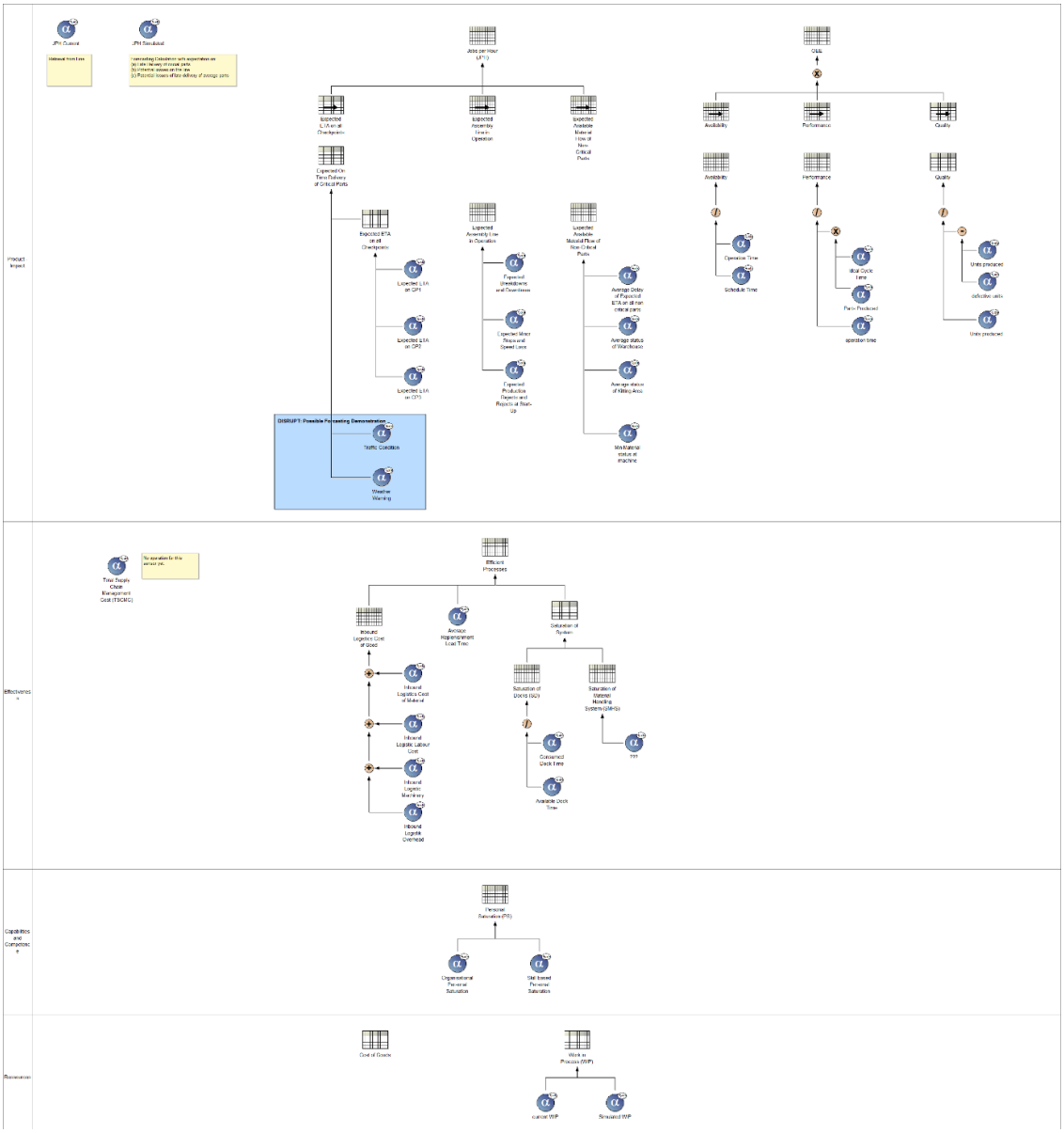

Figure 35: Initial Criteria Model – CRF

Each KPI from the cause-and-effect model is related with either an atomic sensor or a complex sensor. The atomic sensors are the "alpha indicators," which store the technical details to access the data stream. The complex sensors combine atomic sensors and perform a pre-processing before forwarding the data stream to the scorecard.

A sample for our expected critical parts is that:

- The checkpoints are aggregated from three atomic checkpoints.
- There is a design decision to be taken, if all critical parts are abstracted, or for each critical part an own indicator will be built.
- Some of the complex sensors provide the calculation to pre-process the data.

This model is a consequence of the previous cause and effect model and beside some design decisions on how to best collect the data is a technical exercise.

5.3.5 Management Dashboards - CRF

In the CRF use-case, dashboards are important in areas where monitoring the developing status of inbound logistics of vehicle parts, and the production line of vehicles produced is required.

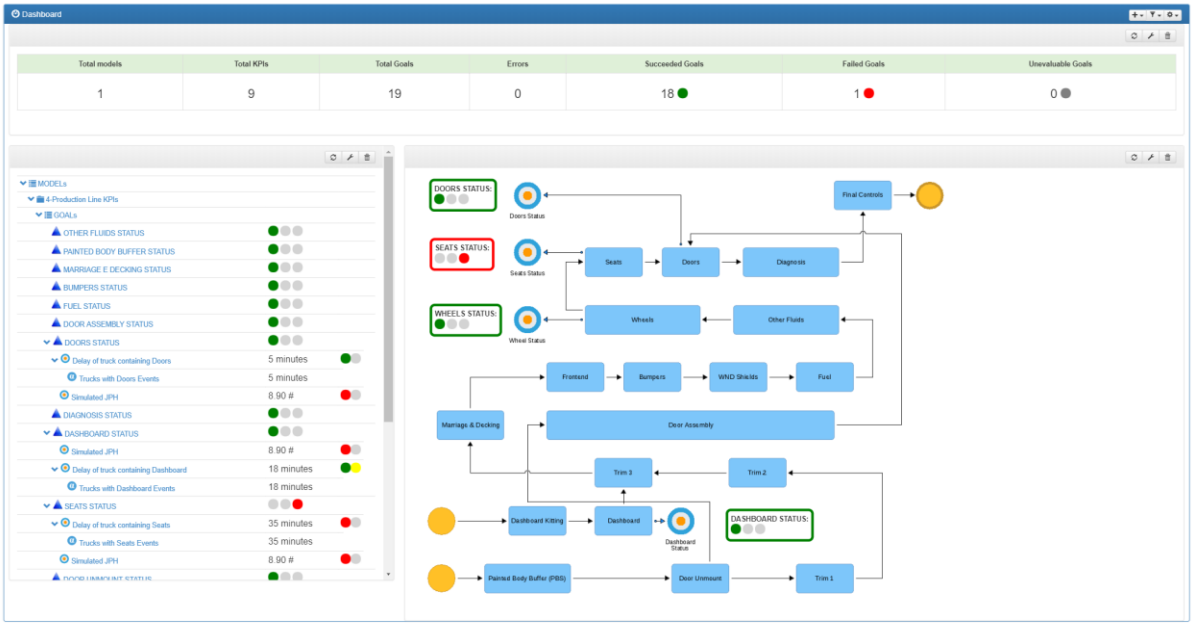

Figure 36: Dashboard of Production - CRF

In Figure 36, the end-user has an insight into the current situation of production at his organisation. In the top banner of the dashboard, the end-user can get a general overview of efficiency in production. Below, the process layout for the production line is modelled, and linked KPIs are shown as the "Targets," with either a green, yellow, or red window next to them indicating the level of data that are meeting the KPI standards. Such a visualisation brings concise descriptions to big data analytics, and pilots the user to further analysis.

5.4 Use-Case Organization - ARCELIK

5.4.1 Focus Areas

Based on the information in D2.4, we grey out the parts that we do not focus on and leave the production on process and product leave in focus.

	Cooperation	Production	Innovation
Product - Impact	I.3) Coopeation and Networked Enterprises	I.1.a) Impact on Quantitiy I.1.b) Impact on Costs I.1.c) Impact on Quality	I.2.a) Innovation on Productivity I.2.b) Innovation on Costs I.2.c) Innovaation on Quality
Effectiveness	II.3) Supply chain related issues	II.1.a) Processes on Quantitiy II.1.b) Processes on Costs II.1.c) Processes on Quality	II.2.a) Process Improvement on Productivity II.2.b) Process Improvement on Costs II.2.c) Process Improvement on Quality
Capabilities and Competence	III.3) Partner related issues	III.1.a) Capabilities III.1.b) Competence	III.2.a) Capabilities improvements III.2.b) Competence improvements
Ressources	IV) Financial Resources, Infrastructure, Mashines, Material, Know-How and Information related Issues		

Figure 37: Perspectives for ARCELIK with focus

It has to be stated, that this interpretation has been derived via an "argumentative approach" from the Performance Scorecard and hence is new in the Production area.

5.4.2 Success Factors

Also in the ARCELIK case, the success factors influencing a goal are in interpretation, which is based on internal workshops at BOC, literature research and the analysis of the D2.4. This must be completed by workshop sessions with the domain experts. Therefore, changes in the interpretation are expected.

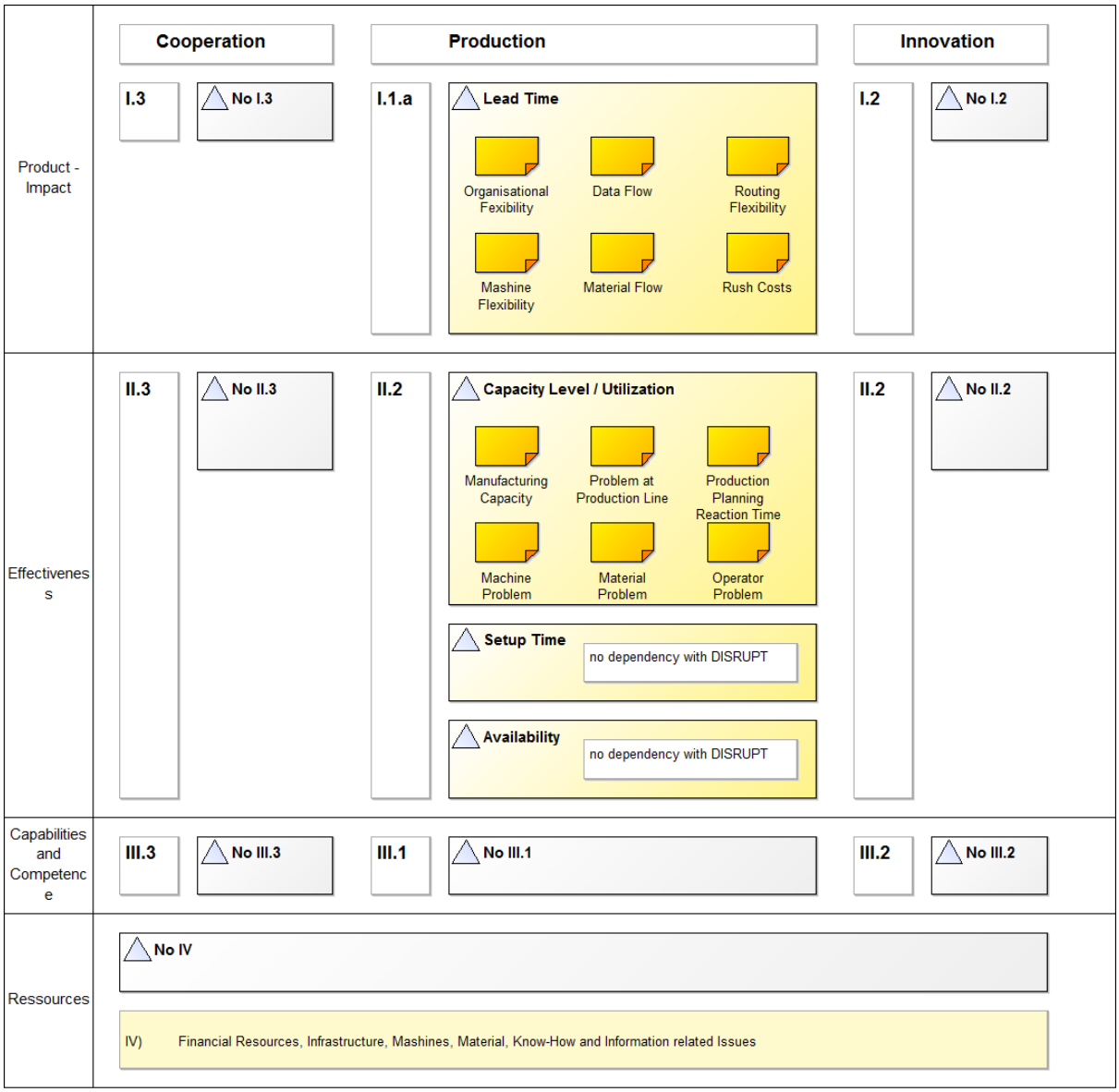

Figure 38: Initial ARCELIK Critical Success Factor Model – without domain expert interpretation

The above figure indicates the initial critical success factor model from ARCELIK, identifying two goals that may be further elaborated.

- Similar to CRF, the **SMART** principle on naming goals must be applied:
  - Specific defined
  - Measurable
  - Ambiguous
  - Realistic
  - Time dependent

The major KPIs "Lead Time" and "Capacity Level / Utilisation" has been selected that can be translated into a goal, meaning that the plant engineer has to have this goal in mind when performing decisions.

The "Setup Time" and "Availability" has currently not been selected, as we currently do not see the possibility of the plant engineer to influence those KPIs when performing his every day decisions. Hence the DISRUPT platform, with its Decision Support System, the Data layer and Cloudboard as well as with its improvement in CPS, will not change the "Setup Time" neither the "Availability" of the workforce.

A possible incorrect interpretation from the modelling workshop may be corrected with the experts from ARCELIK.

The "Lead Time" is understood as the lead time within the plant – excluding the supply chain, or time after the production. We interpreted to measure the FLEXIBILITY with the goal to improve flexibility. This can be achieved with DISRUPT technology, especially to improve the Data Flow, which is one mentioned aspect.

The "Capacity Level / Utilisation" is interpreted to target EFFICIENCY in the plant. Hence, efficiency enhancing tools and techniques from DISRUPT are suggested to be used by the plant engineer, to perform daily operations, but still have these two goals in mind.

5.4.3 Cause and Effect Model

Figure 39 shows the two goals that have been identified for the cause-and-effect model at the ARCELIK use case.

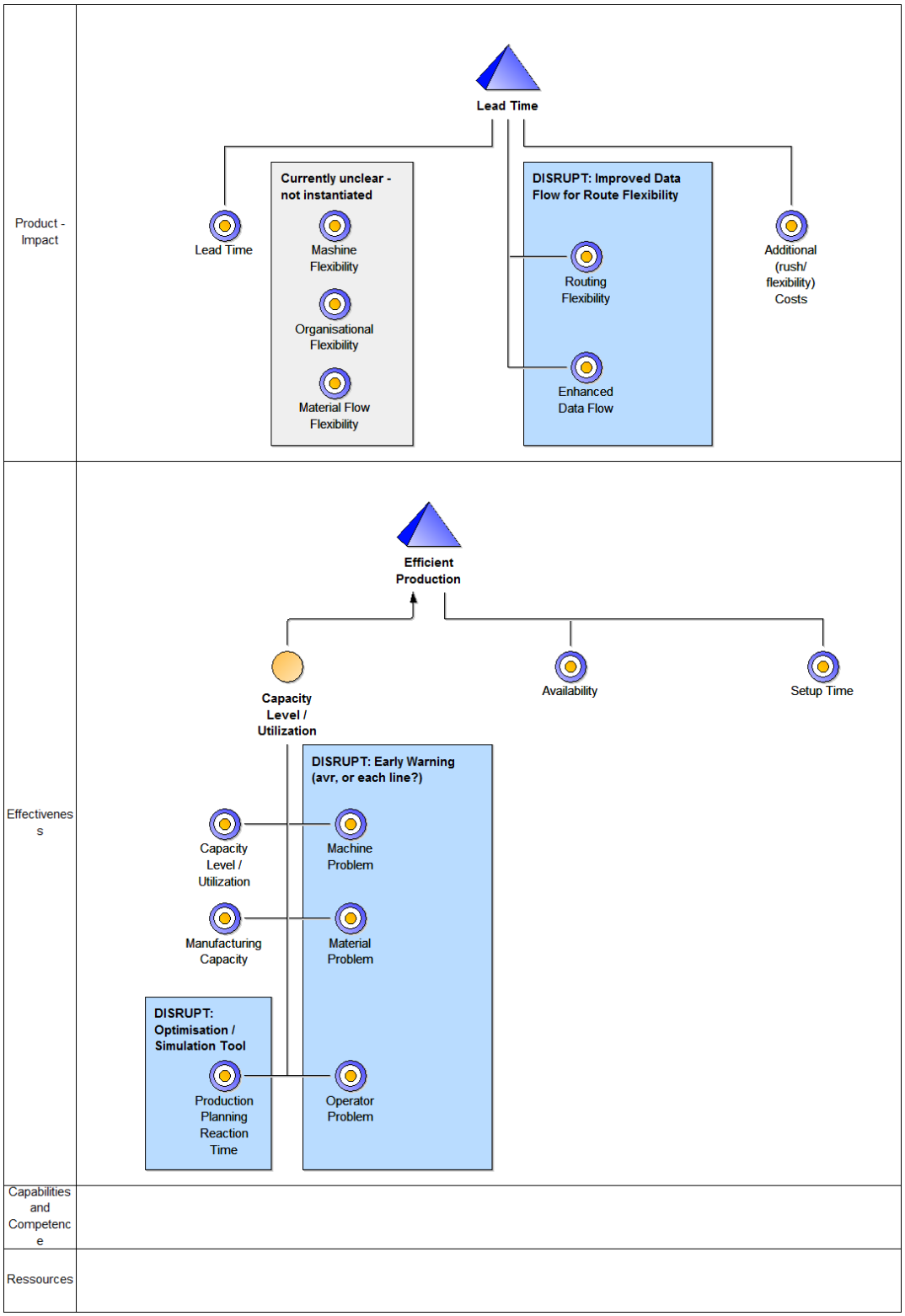

Figure 39: Initial Cause and Effect Model – ARCELIK – Overview

First, Lead time is defined in D2.4 and D5.3 as the response time to customer request. The goal is to become more flexible. There are several criteria that measure the flexibility, where the following has been used:

- Lead Time, this criterion displays the actual lead time
- Based on the research of flexible manufacturing there are the following influence factors:
    - Machine flexibility: This cannot be covered by DISRUPT, as this depends on the architecture of the plant
    - Organisational flexibility: This has strong dependencies on the used machinery, and the required skills. One aspect can be the flexibility of working time, to allow a broader set of simulation alternatives. Currently we do not see coverage in DISRUPT.
    - Material flow flexibility: This corresponds to warehousing and supply chain logistics. Currently this is not considered in DISRUPT as a use case.
- Candidates for DISRUPT relevant KPIs are:
    - Routing Flexibility: Enhanced methods in combining different production plans to influence the flexibility. The simulation and optimisation possibilities may influence the routing flexibility.
    - Enhanced Data Flow: The data flow is essential for gaining flexibility, hence the DISRUPT data flow capabilities will enable a higher flexibility. However, a concrete measure needs to be worked out that defines the improvement of the dataflow.
- Additional (rush/flexibility) costs: The costs that occur, when rushing material delivery, use overtime or other additional costs that are caused by flexibility demand are collected. Further logic on relating them to items or customer satisfaction may be useful.

Second, the Capacity Level / Utilisation has been identified as an important indicator. For methodological reasons, we introduced the goal Efficient Production. The two other KPIs form D2.4 and D5.3 "Availability" and "Setup Time" are both not in scope of DISRUPT, as the setup time depends on the machinery, and the availability depends on the downtime. Neither KPI can be influenced by the DISRUPT platform.

The Capacity Level / Utilisation have some traditional elements:

- Actual Capacity Level / Utilisation value
    - This figure calculated
- Manufacturing Capacity value, the baseline indicator
    - This figure gets reported after the checking the plant status
- Interesting KPIs are
    - Machine Problem
        - Real time alert, in case of a machine stop
    - Material Problem
        - Real time alert, in case of a material stop
    - Operator Problem
        - Real time alert, in case of an operator stop

As all these indicators speed up the notification of the project planning from real time line data. It has to be clarified, if the average of the line problems or for each individual line is preferable. Current proposal is for each line separately. A finer granular identification of issues – including early sights and warnings – needs to be elaborated with the experts.

- Production Planning Reaction Time: This indicator has been derived out of the description, as the goal is to quickly change the production planning according to known issues. Hence, the early warning of the aforementioned line problems, together with simulation and optimisation to quickly produce a new production plan, is an interesting KPI that may improve thanks to the DISRUPT platform.

5.4.4 Criteria Model

Figure 40 shows an initial version of the criteria model of ARCELIK.

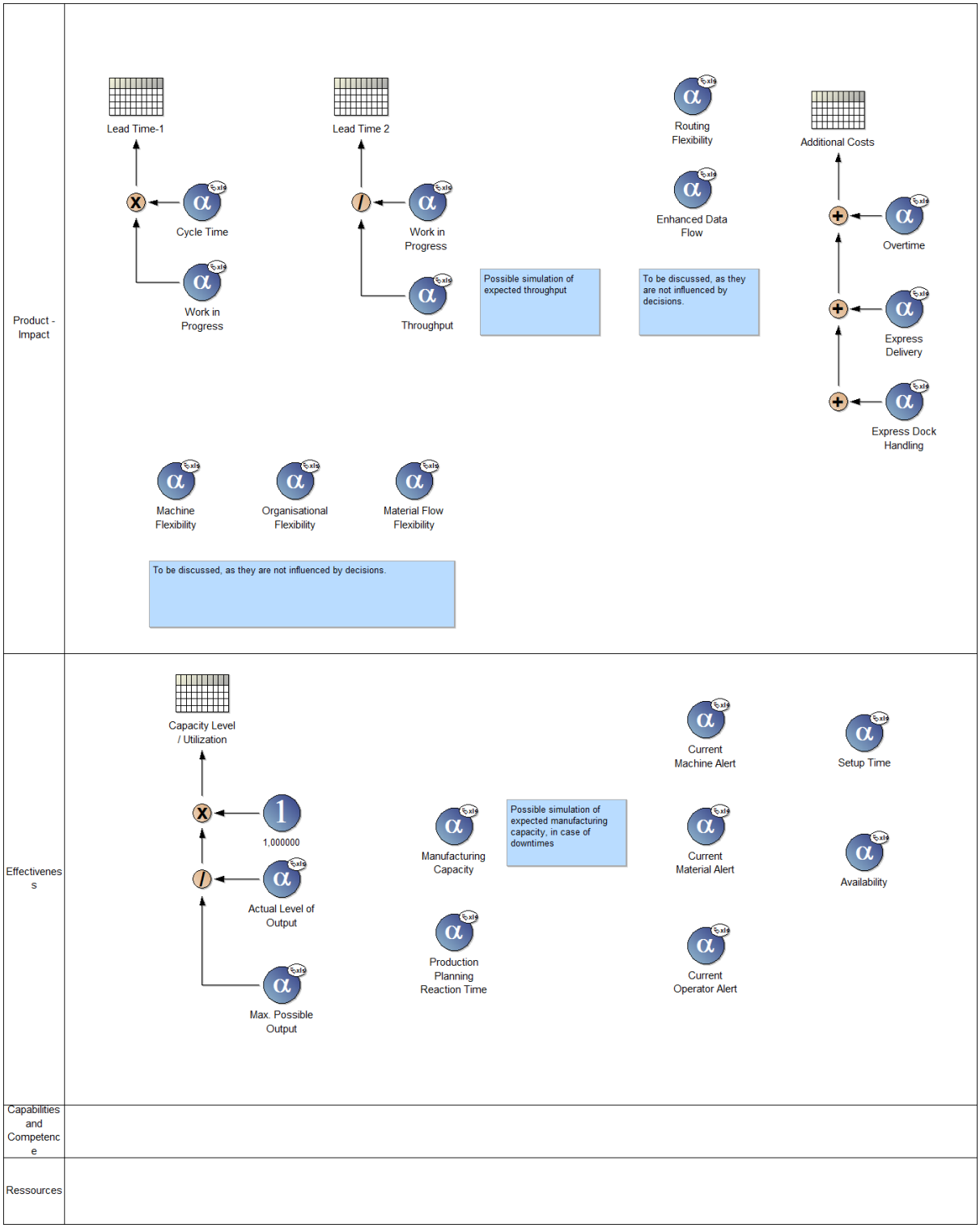

Figure 40: Initial Criteria Model – ARCLEIK

It has to be worked out, which of the criteria are calculated on the data layer, which of the criteria are calculated in the cockpit, and which are simulated after process runs. Some of the criteria may be simulated.

5.4.5 Management Dashboards - ARCELIK

In the ARCELIK use-case, dashboards are important in areas such as to monitor the developing status of inbound logistics of parts to produce their home appliances, and to analyse the level at which their production objectives are being met.

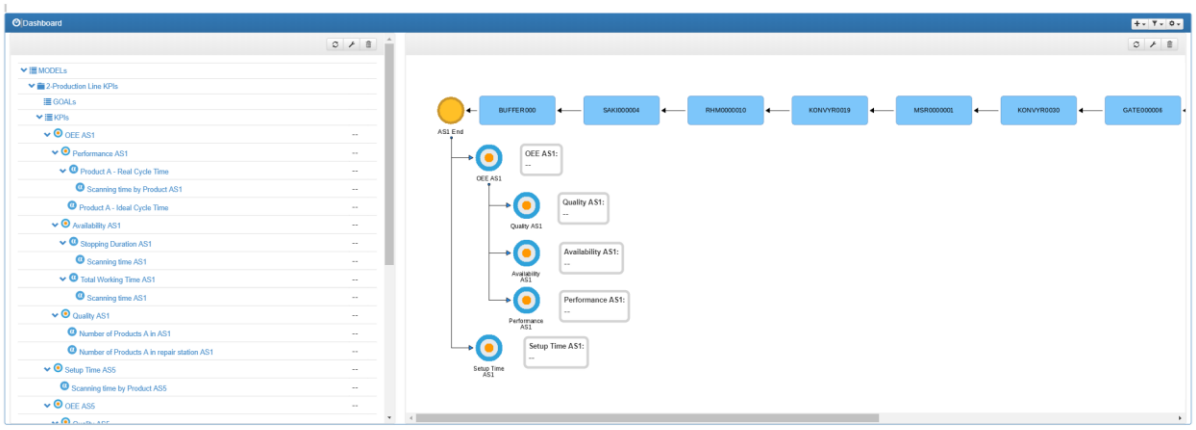

Figure 41: Management Dashboard - ARCLEIK

In figure 41, the end-user has an insight into the current situation of production at ARCELIK. Here, the end-user can get a general overview of efficiency in production. The process layout for the production line is modelled, and linked KPIs are shown as the "Targets." Once data is received from the use-case, it will appear linked in a window in green, yellow, or red by them, indicating the level of data that is meeting the KPI standards. Such a visualisation brings concise descriptions to big data analytics, and pilots the user to further analysis.

6 Conclusion

The offered framework is specifically proposed for the improvement of operations in the areas of production and logistics management by means of the concrete visualization of information being aggregated by the IoT-devices, and operators, to the responsible decision makers in an appropriate contextual form. In this approach, production objectives and critical-success-factors can be traced from the top, strategic level, all the way down to the bottom, sensor level, therefore being able to reply swiftly to unforeseen circumstances.

The approach first integrates available tools, introduces semantics, and lastly introduces semantic algorithms in order to implement smart instruments for support. Importantly, a decentralised integration between all stakeholders in production is critical to the implementation of this approach. Such an approach allows companies to gain competitive advantage.

Annex A: References

[1] Löffler C., Westkämper E. and Unger, K., Change drivers and adaptation of automotive manufacturing, International Conference on Manufacturing Systems (ICMS), p. 6 (2011)

[2] Westkämper E., Zahn E., Balve P. and Tilebein M., Ansätze zur Wandlungsfähigkeit von Produktionsunternehmen, WT. Werkstattstechnik 90, p.22-26 (2000)

[3] P. Eirinakis, J. Buenabad-Chavez, R. Fornasiero, H. Gokmen, J. Mascolo, I. Mourtos, S. Spieckermann, V. Tountopoulos, F. Werner and R. Woitsch, A proposal of decentralised architecture for optimised operations in manufacturing ecosystem collaboration, Working Conference on Virtual Enterprises PRO-VE (2017)

[4] N. Marz, J. Warren, Big Data: Principles and best practices of scalable real-time data systems, Manning (2015)

[5] Company White Paper: The APAMA Platform, Under-the-covers: an in-depth view of Apama, Software AG (2016)

[6] Product Fact Sheet: Universal Messaging, Software AG (2016) https://resources.softwareag.com/products-analytics-decisions/universal-messaging-fact-sheet, last accessed 13.02.2018

[7] Michael R. Berthold, Nicolas Cebron, Fabian Dill, Thomas R. Gabriel, Tobias Kötter, Thorsten Meinl, Peter Ohl, Christoph Sieb, Kilian Thiel and Bernd Wiswedel, KNIME: The Konstanz Information Miner, Studies in Classification, Data Analysis, and Knowledge Organisation, Springer (2007)

[8] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann and Ian H. Witten, The WEKA data mining software: an update, SIGKDD Explorations, p. 10-18 (2009)

[9] R Core Team (2013), R: A language and environment for statistical, computing, R Foundation for Statistical Computing (2013)

[10] J. Krumeich, M. Zapp, D. Mayer, D. Werth and P. Loos, Modeling Complex Event Patterns in EPC-Models and Transforming them into an Executable Event Pattern Language, Multikonferenz Wirtschaftsinformatik (MKWI), p. 81-92 (2016)

[11] J. Krumeich, N. Mehdiyev, D. Werth and P. Loos, Towards an Extended Metamodel of Event-driven Process Chains to Model Complex Event Patterns, 2nd International Workshop on Event Modeling and Processing in Business Process Management, Cham, Springer Switzerland (2015)

[12] Company White Paper: Why you need Zementis, Predictive Analytics, Software AG (2017) https://resources.softwareag.com/products-analytics-decisions/why-zementis-whitepaper, Accessed 13.02.2018

[13] Frost & Sullivan, Automotive Industry IT Spending, CIO Focus, Trends, and Highest Growth Areas, Report (2016)

[14] Woitsch R. and Hrgovcic V., Modelling Knowledge: An Open Model Approach, Proceedings of the 11th International Conference on Knowledge Management and Knowledge Technologies (2011)

[15] Guschlbauer E., Lichka C., Umsetzung des Prozesscontrollings, Prozessmanagement für Experten, Impulse für aktuelle und wiederkehrende Themen, Springer-Gabler Verlag Berlin Heidelberg (2013)

[16] Woitsch R., Process-Oriented Knowledge Management: A Service-based Approach, PhD Thesis, Vienna (2004)

[17] Woitsch R., Utz W., Hrgovcic V., Integration von Prozess- und Wissensmanagement, Prozessmanagement für Experten, Impulse für aktuelle und wiederkehrende Themen, Springer-Gabler Verlag (2013)

[18] Lichka C., Der modellbasierte Business Scorecarding-Ansatz zur Strategieoperationalisierung, University of Vienna, PhD Thesis (2006)

[19] Karagiannis D., Woitsch R., Knowledge Engineering in Business Process Management, Business Process Management 2, Strategic Alignment, Governance, People and Culture, Springer (2010)

[20] Roussopoulos N., Utz W., Design Semantics on Accessibility in Unstructured Data Environment, Domain Specific Conceptual Modelling, Concepts, Methods and Tools, Springer (2016)

[21] Utz W., Woitsch R., A Model-Based Environment for Data Services: Energy-Aware Behavioral Triggering Using ADOxx. Collaboration in a Data-Rich World. PRO-VE 2017, vol 506, Springer (2017)

[22] Karagiannis D. , Mayr H. , Mylopoulos J., Domain Specific Conceptual Modelling, Concepts, Methods and Tools, Springer Switzerland (2016)

[23] M. Wooldridge, An Introduction to Multi-Agent Systems, John Wiley & Sons, (2002)

[24] P. Leitão, Agent-based Distributed Manufacturing Control: A State-of-the-art Survey, Engineering Applications of Artificial Intelligence, vol. 22, pp. 979-991 (2009)

[25] S. Middelhoek, A.C. Hoogerwerf, Smart sensors: when and where ?, Sensors and Actuators, Vol. 8, Issue 1, pp. 39-48 (1985)

[26] M. A. Montironi, P. Castellini, L. Stroppa, N. Paone, Adaptive autonomous positioning of a robot vision system: application to quality control on production lines, Robotics and Computer Integrated Manufacturing, vol.30, pp. 489-498 (2014)

[27] A. Davis, J- Parikh, W.E. Weihl, Edge Computing, Extending Enterprise Applications to the Edge of the Internet, ACM New York (2004)

[28] M. Satyanarayanan, P. Simoens, Y. Xiao, P. Pillai, Z. Chen, K. Ha, W. Hu and B. Amos, Edge analytics in the Internet of Things, IEEE Pervasive Computing, vol. 14, pp. 24-31 (2015)

[29] E. A. Lee, J. Rabaey, B. Hartmann, J. Kubiatowicz, K. Pister, A. Sangiovanni-Vincentelli, S. A. Seshia, J. Wawrzynek, D. Wessel, R. Jafari, D. Jones, V. Kumar, R. Mangharam, G. J. Pappas, and T. S. Rosing, The Swarm at the Edge of the Cloud, IEEE Design & Test, vol. 31, no. 3, pp. 8-20 (2014)

[30] M. Kabáč, C. Consel and N. Volanschi, Designing parallel data processing for enabling large-scale sensor applications, Personal and Ubiquitous Computing (2017)

[31] Rossiter J., Model-Based Predictive Control: A Practical Approach, CRC Press (2003)

[32] Bemporad, A., Model Predictive Control Design: New Trends and Tools, Proceedings of 45th IEEE Conference on Decision and Control (2006)

[33] The Kouvaritakis, B. and M. Cannon. Non-linear Predictive Control: Theory and Practice, ISBN 978-0852969847, The Institution of Engineering and Technology, IEE Publishing, 2001

[34] Park, K., Zheng, R. and X. Liu, Cyber-physical systems: Milestones and research challenges, The International Journal for the Computer and Telecommunications Industry, n.36 (2012)

[35] Nenad Stojanovic, Marko Dinic, Ljiljana Stojanovic, Big data process analytics for continuous process improvement in manufacturing, pp. 1398-1407, Big Data, IEEE Publishing, Santa Clara, CA, USA (2015)

[36] Heemels, W., De Schutter, B. and A. Bemporad, Equivalence of hybrid dynamical models, Automatica, vol. 37, pp. 1085–1091 (2001) 37. Juloski, A., Wieland, S. and W.P.M.H. Heemels, A Bayesian approach to identification of hybrid systems. IEEE Transactions on Automatic Control, 50(10), pp. 1520–1533 (2005)

[37] Ferrari-Trecate, G., Muselli, M., Liberati, D. and M. Morari. A clustering technique for the identification of piecewise affine systems, Automatica, pp. 205–217, (2003)

[38] Woitsch Robert, Hrgovcic Vedran and Buchmann Robert, Knowledge Product Modelling for Industry: The PROMOTE Approach, 14th IFAC Symposium on Information Control Problems in Manufacturing, International Federation of Automatic Control (2012)

[39] Baldoni, R.; M. Contenti, and A. Virgillito. "The Evolution of Publish/Subscribe Communication Systems." Future Directions of Distributed Computing", Springer Verlag LNCS Vol. 2584, 2003],

[40] Website, last visited 14.01.2019 https://developer.here.com/documentation/weather/topics/request-constructing.html

[41] Website, last visited 14.01.2019 https://developer.here.com/products/routing-and-navigation

[42] Otto Boris et. al, IDS Reference Architecture Model, Industrial Data Spaces Association / Fraunhofer Society (Version 2.0, 2018)

[43] Werner, F., & Woitsch, R., Data Processing in Industrie 4.0 - Data Analytics and Knowledge Management in Industrie 4.0. Datenbank-Sprektrum, 1-11. (2018)

[44] Universal Messaging Concepts, Software AG (Version 10.3, 2018) https://documentation.softwareag.com/onlinehelp/Rohan/num10-3/10-3_Universal_Messaging_Concepts_Guide.pdf

[45] Universal Messaging Administration Guide, Software AG (Version 10.3, 2018) https://documentation.softwareag.com/onlinehelp/Rohan/num10-3/10-3_Universal_Messaging_Administration_Guide.pdf

[46] Gualtieri, M. (2016). 15 "True" Streaming Analytics Platforms For Real-Time Everything. Forrester Research. Retrieved from https://go.forrester.com/blogs/16-04-16-15_true_streaming_analytics_platforms_for_real_time_everything/

Annex B: List of Acronyms

API	Application Programming Interface
BPM	Business Process Management
BPMN	Business Process Model Notation
CEP	Complex Event Processing
CPS	Cyber Physical Systems
CRF	Centro Ricerche Fiat
EDA	Event Driven Architecture
EPL	Event Processing Language
ETA	Estimated Time of Arrival
FoF	Factory of the Future
IBPM	Industrial Business Process Management
ICT	Information & Communication Technology
IIoT	Industrial Internet of Things
IT	Information Technology
IoT	Internet of Things
JS	JavaScript
JSON	JavaScript Object Notation
JMS	Java Message Service
JpH	Jobs Per Hour
KPI	Key Performance Indicator
OEM	Original Equipment Manufacturer
PMML	Predictive Model Markup Language
RTA	Required Time of Arrival
SEOR	Systematic Energy Operational Rating
SMART	Specific Defined, Measurable, Ambiguous, Relaistic, and Time Dependent
TTL	Time To Live