White Paper

Manufacturing
Smart Manufacturing
Digital Manufacturing

Guideline for operating resilient and flexible production facilities using runtime risk management

Proposal of a new safety paradigm: operational safety intelligence - optimizing operations by runtime risk management and predictive modeling.

Authors and Contents:

Parkin Borsum,
TÜV SÜD Industrie Service GmbH

Christian Donitzky,
Intel Deutschland GmbH

Alex Klimovitski,
Intel Deutschland GmbH

Alexander Kurdas,
TÜV SÜD Industrie Service GmbH

Michael Pfeifer,
TÜV SÜD Industrie Service GmbH

Dr. Detlev Richter,
TÜV SÜD Product Service GmbH

Michael Schmidt,
Intel Deutschland GmbH

Dr. Marcel Wagner
Intel Deutschland GmbH

1. Abstract

The topics of risk management and safety in production are becoming an increasing challenge for companies in the interconnected and digitalized Industry 4.0 age. The reasons for this include changing market requirements, turbulence in supply chains and stricter energy efficiency requirements due to sustainability. These aspects lead to the use of resilient and flexible production systems, which contradicts the use of traditional and static safety concepts. Another reason is the demographic change, with the difficulty of finding skilled workers, for example for conducting manual risk assessments.

These dynamic boundary conditions no longer permit static and manual risk management with a smaller operating team. Therefore, in this paper a new safety paradigm is proposed: a dynamic, adaptable safety that reacts to the current context and environmental situation. This **operational safety intelligence** brings together flexible production and safety, handles data from large distributed systems in the appropriate context, understands risks from dynamic processes and optimizes operations by runtime risk management as well as predictive modeling.

The core building block of this novel operational safety intelligence is a cutting-edge information management system that can handle contextualized information and ensure data integrity. This is achieved by use of the Semantic Web concept, based on knowledge graphs with unified semantics and enhanced with deduction and constraints checking abilities. By bringing together these systems, and appropriate use of artificial intelligence (AI), operational safety intelligence is becoming an essential methodology for guaranteeing safety in Industry 4.0.

2. Glossary

ACID Atomicity, Consistency, Isolation and Durability (Constraints)
Database transactions shall guarantee this set of properties even in case of failures or errors.

AGV Automated Guided Vehicle
AGVs are driverless vehicles for intralogistics, mainly following predefined routes; in this paper they are also seen under the term **AMR**.

AI Artificial Intelligence
It is difficult to arrive at a precise definition of AI, according to the VDE-AR-E 2842-61-1 standard[1], but in the context of this paper it can be understood, broadly, as **advanced software**.

AMR Autonomous Mobile Robot
AMRs are driverless vehicles for intralogistics, whereas these allow mostly more dynamic routes than AGVs; in this paper, a broad range of types is considered under this term.

AVX Advanced Vector Extensions
Advanced instruction set for CPUs, including SIMD operations.

CAP Consistency-Availability-Partition Tolerance (Theorem)
The **CAP theorem** (also **Brewer's theorem**), states that in distributed data storage systems only two capabilities out of consistency, availability and partition tolerance can be guaranteed.

CPU Central Processing Unit
The processor or core calculating unit of a computer.

DTC Digital Twin Consortium
The **DTC** drives the awareness, adoption, interoperability and development of digital twin technology.

ECC Error Correction Code (Memory)
Special type of computer memory that detects and corrects data corruption occurring in memory chips.

EMQX Software platform for MQTT.

EN European Norm.

HTTP Hypertext Transfer Protocol
High-level protocol for transferring data via networks.

IEC International Electrotechnical Commission
The **IEC** is an international standardization organization focused on electrical- and electronic-related topics.

IFF IndustryFusion Foundation
The **IFF** is a consortium for digitalization and networking of European industry, ensuring its continuous further development.

IIoT Industrial Internet of Things
The **IIoT** describes the network of sensors, instruments and other devices including computers and controllers in industrial applications.

ISO International Organization for Standardization
The **ISO** sets and provides norms for many fields on a global scale.

IT Information Technology
IT in this context describes the enterprise network, computers and software that are not related to production systems.

JSON JavaScript Object Notation
A human-readable data format for storing data objects.

JSON-LD JavaScript Object Notation for Linked Data
A human-readable data format for storing graphs like RDF.

LLM Large-Language Model
LLMs are a special type of AI that are trained on huge amounts of data, which enables them to do things like generate text.

MPX Memory Protection Extensions
MPXs offer security features for memory in computers.

MQTT **MQTT** is a network protocol for machine-to-machine communication, for example, in IIoT.

NGSI-LD Next Generation Service Interface-Linked Data
An extension to the human-readable data format JSON-LD for storing verified data and graphs with extended context and metadata.

OT Operational Technology
OT describes the communication network of the production system in contrast to IT.

OWL Web Ontology Language
OWL is a programming language that provides a structure for describing human knowledge in a software, whereas such a description is also called an **ontology**.

PLC Programmable Logic Controller
Industrial computer specialized in the control of machinery and plants.

RDF Resource Description Framework
A standard for description and management of graph structures.

SGX Software Guard Extensions
Technology for protecting data at the CPU level.

SHACL Shapes Constraint Language
A programming language to express and check constraintson data stored in graph structures.

SIL Safety Integrity Level
The **SIL** describes the level of risk reduction of safety equipment according to the IEC 61508 standard[2].

SIMD Single Instruction Multiple Data
Instruction for CPUs that allows the CPU to perform one operation on multiple data in parallel.

SME Small and Medium-sized Enterprises
All enterprises that employ fewer than 250 persons and whose annual turnover does not exceed Euro 50 million are regarded as SMEs.

SPARQL SPARQL Protocol and RDF Query Language
A programming language for input and output of data stored in RDF graphs.

SQL Structured Query Language
A programming language to manage structured data, for example, in databases.

TXT Trusted Execution Technology
TXT is a hardware-based security feature for booting computers.

UEFI Unified Extensible Firmware Interface
UEFI is a specification for the booting of computers and the interface between hardware and operating system.

URI Uniform Resource Identifier
A **URI** is a unique name for an arbitrary physical or digital resource.

W3C World Wide Web Consortium
International standardization organization concerning the World Wide Web and related programming languages.

YAML YAML Ain't Markup Language
A human-readable data format for storing data objects.

3. Motivation

The industry today is challenged by an increasing product variety, customization of mass production and highly specialized single-unit production. Additional challenges result from even shorter product life cycles and turbulence in supply chains. Moreover, the demand for sustainable products has created requirements across all facets of the business. This not only concerns, for example, the selection of eco-friendly materials and the qualification of suppliers,

but also topics such as energy efficiency in production. The necessary improvements are often not easy to achieve and partially contradicting.

For managing all these challenges, the industry has reached considerable results in machine digitization, which is called **Industry** 4.0. However, the safety implications of this transformation have been neglected so far. The flexibility required to cope with the aforementioned challenges means that currently used manual safety assessments are too slow to keep up in this fast-paced digital world. Furthermore, the lack of experienced and trained personnel, reinforced by the demographic changes in Western countries, calls for further automation. Even safety concepts need to be adapted, and advanced tools for support of decisions for the remaining human experts are needed. Therefore, smart manufacturing and Industry 4.0 demand a smarter and more automatic safety framework.

To accommodate all the requirements mentioned above, the promise of Industry 4.0 becomes more and more concrete: seamlessly interconnected machines converge to optimize efficiency, flexibility and innovation across the entire value chain by means of data analytics and artificial intelligence (AI). Industry 4.0 platforms developed advanced technical solutions, including the Industrial Internet of Things (IIoT), interconnections, exchange of data across systems, autonomous machinery and collaborative machines. The promise of these technological advances is higher productivity and more flexibility to achieve a smart and resilient production system.

However, current safety concepts and requirements disrupt and even limit this new world. Safety aspects today are considered in isolated scenarios including static assumptions about processes and worst-case scenarios.

In isolation, safety assessment is well understood and even certifiable as **functional safety**, see for example the IEC 61508 standard[2]. Challenges arise from the interconnections and interactions of components, which make the overall system, or an even more complex system of systems, almost unmanageable with traditional methods. The static safety approach comes to its limits when applied to interconnected, distributed, dynamically changing systems. Especially autonomous systems pose unsolvable challenges to static predefined safety concepts, as autonomy means to appropriately react to situations, that are unforeseen at the design phase. Additionally, even the regulatory framework is meanwhile advancing. For example, the new European machinery regulation requires that autonomous machinery "shall alert the supervisor of the occurrence of unforeseen or dangerous situations present or impending"[3]. This requires the machine to identify dangerous situations that are not considered at the design phase. This is impossible to achieve with traditional static safety, which does not program (or teach) hazard identification to the machine, but just sets protective measures.

A striking example: The current safety regulations prevent broader adoption of automated guided vehicles (AGVs), autonomous mobile robots (AMRs) and collaborative robots by enforcing their slow motion to ensure safe operation. The current safety paradigm prescribes that the machine must always be able to reach a safe state immediately, mostly to stop, if it can potentially harm a person. For example, a person could potentially step into the movement area — in which case, the machine must stop immediately. However, the slow-motion operation mode reduces productivity — an unacceptable price given intensifying international competition and growing cost pressure.

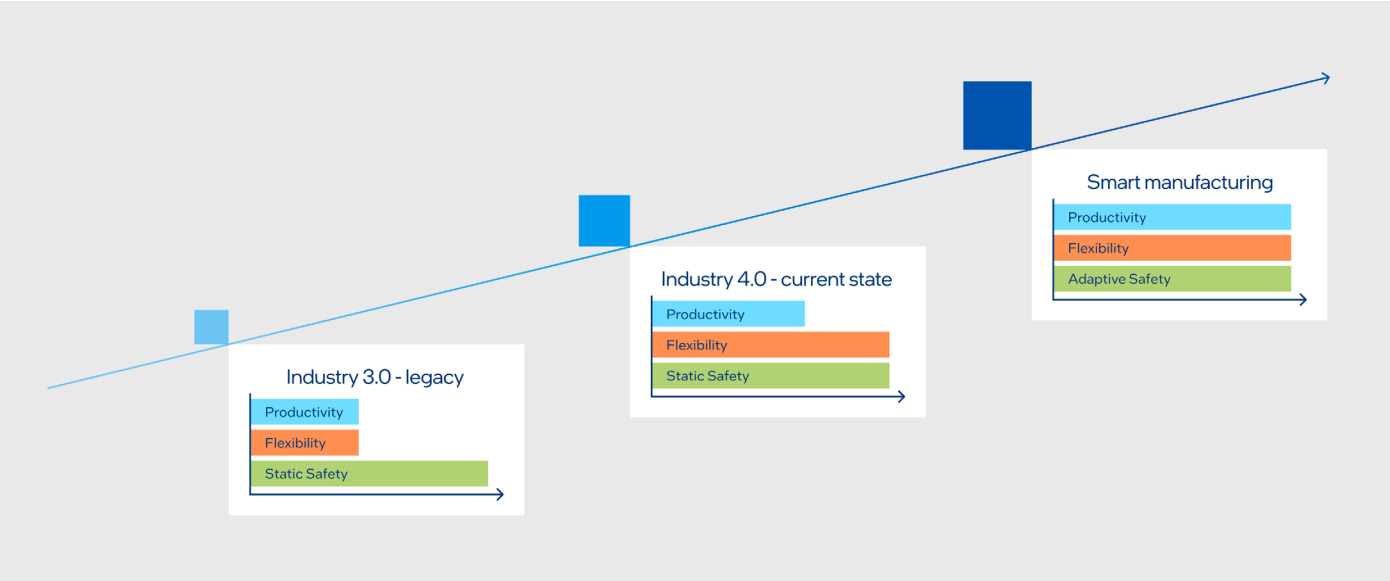

Figure 1. Innovation of Smart Manufacturing.

To summarize, the requirements for modern production systems, high productivity, flexibility and safety are in conflict with each other under the current safety paradigm (see also Figure 1). To resolve the contradiction, a new safety paradigm is proposed in this paper: a dynamic, adaptable safety that reacts to the current context and environmental situation. This is **operational safety intelligence**:

- Joins productivity, flexibility and safety.
- Handles data from large distributed systems in the appropriate context.
- Understands risks from dynamic processes.
- Optimizes operations by runtime risk management as well as predictive modeling.

4. Trust in autonomous and flexible production systems

Currently, when introducing changes in production, a distinction must be made between changes impacting the hazard and risk situation and those changes that only switch the parameterization at runtime. For both cases, there are already well-established treatment processes available. The first case forces a new (manual) risk assessment, while, in the latter one, for example, the brief muting of a light curtain at runtime is state of the art in functional safety. But new challenges arise from developments like the intended use of AMRs in narrow production areas where the range and width of view of safety sensors must be restricted to allow movement in the first place. However, at a cross-section the sight of the AMR will be limited, hence it needs to slow down. At this point relating only to today's paradigms, such as intrinsic safety, reduces productivity. To overcome this issue while still being safe, the AMR's information set needs to be expanded by external sensors or other machines, forming a production network. The AMR can react to the information from the environmental sensors, which enables "looking around the corner."

As seen in this example, to enhance the flexibility of safety in Industry 4.0 production, it is necessary to make use of contextual information. This leads to a change of certain paradigms in safety concepts, resulting in operational safety intelligence as a novel methodology in safety. This paper will detail the requirements of the implementation of such a novel safety system — the **operational safety intelligence software** — and outline an approach for implementing the information management and hardware structure. Core features of this information management are the handling of data from large distributed systems, managing contextualized information and, based on this, ensuring data integrity. By using information fulfilling these crucial aspects, it is possible to establish operational safety intelligence as a new higher-level approach to manage risks at runtime. Thereby, the implementation of the operational safety intelligence software is not meant to replace the underlying functional safety of today. Rather, it helps resolve the arising contradiction between productivity, flexibility and safety, as functional safety is still a crucial element in the methodology of operational safety intelligence.

Some steps toward operational safety intelligence have already been taken, as detailed in the article "Safe and efficient production through agent systems"[4]. Therein, an agent system for production is proposed, including safety agents, which are part of the operational safety intelligence software. The relevant parts for operational safety intelligence are mainly located at the tactical level according to this article, which is also depicted in Figure 2. Installed at this level in the middle, the operational safety intelligence software has the possibility to gather data from various sources and have a kind of overview of the operational level. By processing the received data, it can provide enhanced information and knowledge to the functions at the operational level and hence trigger improved reactions of the machines. Apart from this, there are also some parts of the operational safety intelligence software located at the strategic level, which are responsible for providing predictive safety information based on simulations.

An important aspect of the operational safety intelligence software is to gather and process information of many sources from the machine, but also contextual information. A so-called **knowledge graph**[4,5] is a suitable concept for data management and processing of this type of information. A knowledge graph is able to link diverse and huge amounts of data and information, while still offering fast and easy access to the content. This linkage of information constitutes the knowledge in the graph, which is the key aspect and elevates this concept beyond the traditional methods. Especially using the knowledge graph concept to evaluate and process safety information offers enormous potential. The connection of pieces of information that may lead to a hazardous situation is called **hazard rule**[5], whereas **safety rules** describe how safety measures mitigate the risks. With these elements, it is possible for the operational safety intelligence software to identify hazards and match mitigating safety measures. The so-far-proposed, high-level concept[4,5] will be further elaborated in terms of level of detail and software architecture in the following chapters.

Furthermore, a knowledge graph can be seen as a **digital twin** of a system, or as the information management part of a digital twin. Throughout this paper, the definition of the digital twin of the Digital Twin Consortium (DTC) will be used: "A digital twin is a virtual representation of real-world entities and processes, synchronized at a specified frequency and fidelity"[6]. Of special importance in this

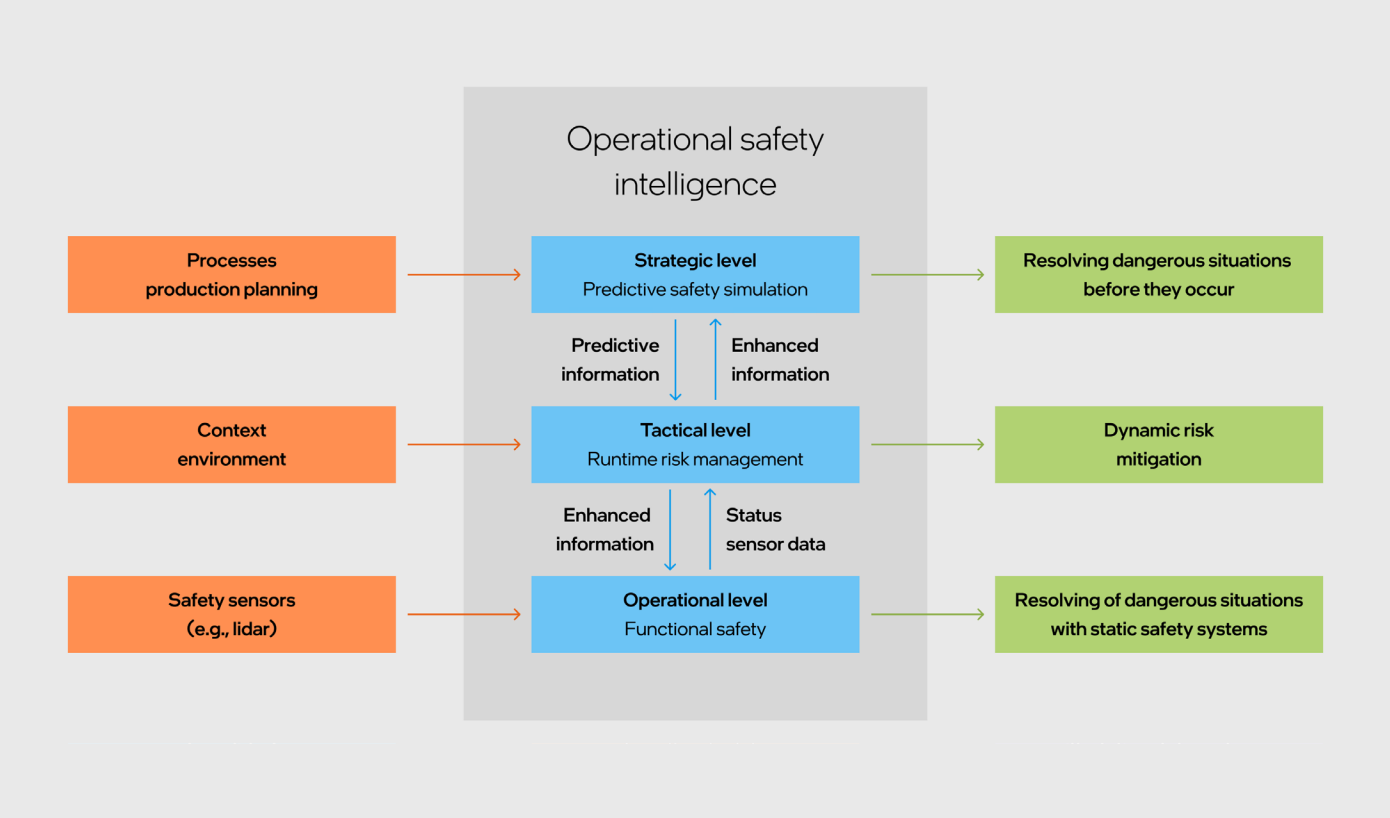

Figure 2. Inputs and outputs at different levels of operational safety intelligence.

definition is that **synchronized** is meant in both directions, so the digital twin gathers data from the real-world system but also influences the behavior of the real system. If there is no synchronization with runtime data at all it is called a **digital model** and if there is only data gathered but no feedback loop to the real system is established, it is called a **digital shadow**, according to DTC. In the herein-proposed concept, this feedback loop is established by an agent system[4]. Furthermore, a digital twin is composed of diverse aspects, which are represented as dimensions in Figure 3, showing a conceptual depiction of a digital twin. Important to note is that time is one of the aspects, as the content might change over time. Additionally, the digital twin is seen as a **modular twin**; as for providing the requested flexibility, it is necessary to easily change parts of the system.

Another dimension of the digital twin, as represented in Figure 3, but also of safety nowadays, is **trustworthiness**. Trustworthiness contains multiple aspects but herein mainly the aspects safety, security, privacy, reliability and resilience are summed up in this term. So, trustworthiness describes the overall goal to reach, but also shows that it is only achievable if all aspects are fulfilled. For example, functional safety without proper security could be hacked and misused to stop the machines and reduce productivity of competitors. Another important aspect is that according to safety standards it is not required to consider intended manipulation during the design of safety functions. However, such blind spots can be exploited by cyberattacks. Hence, cybersecurity must be considered throughout the lifecycle, although information technology (IT) cybersecurity management systems claim to isolate the whole operational technology (OT). For establishing fast and trustworthy communication on the machinery level, the concept of the trust vector[6] is introduced. However, for a holistic trustworthiness it is crucial to also ensure that data transfer and data processing on each end are performed in a trustworthy manner. This means data integrity must be maintained throughout all systems. Ultimately, these aspects require consideration of trustworthiness for all processing of the operational safety intelligence software and the digital twin; further requirements and an approach for hardware are proposed in this paper.

The basis on which the decisions of the operational safety intelligence software are made must be comprehensible and verifiable at all times. Apart from the fact that the operational safety intelligence software has the ability to recognize the context and propose configuration changes, an independent validator software is needed to assess and approve these proposals. From a technological point of view, this is achievable today. However, the question of liability arises, potentially driving up costs. Up to now, there has been no specific standard for such a system. Hence, one would risk the reversal of the burden of proof in the regulatory framework, which is an obstacle for companies that want to use such a system. Therefore, the legislative

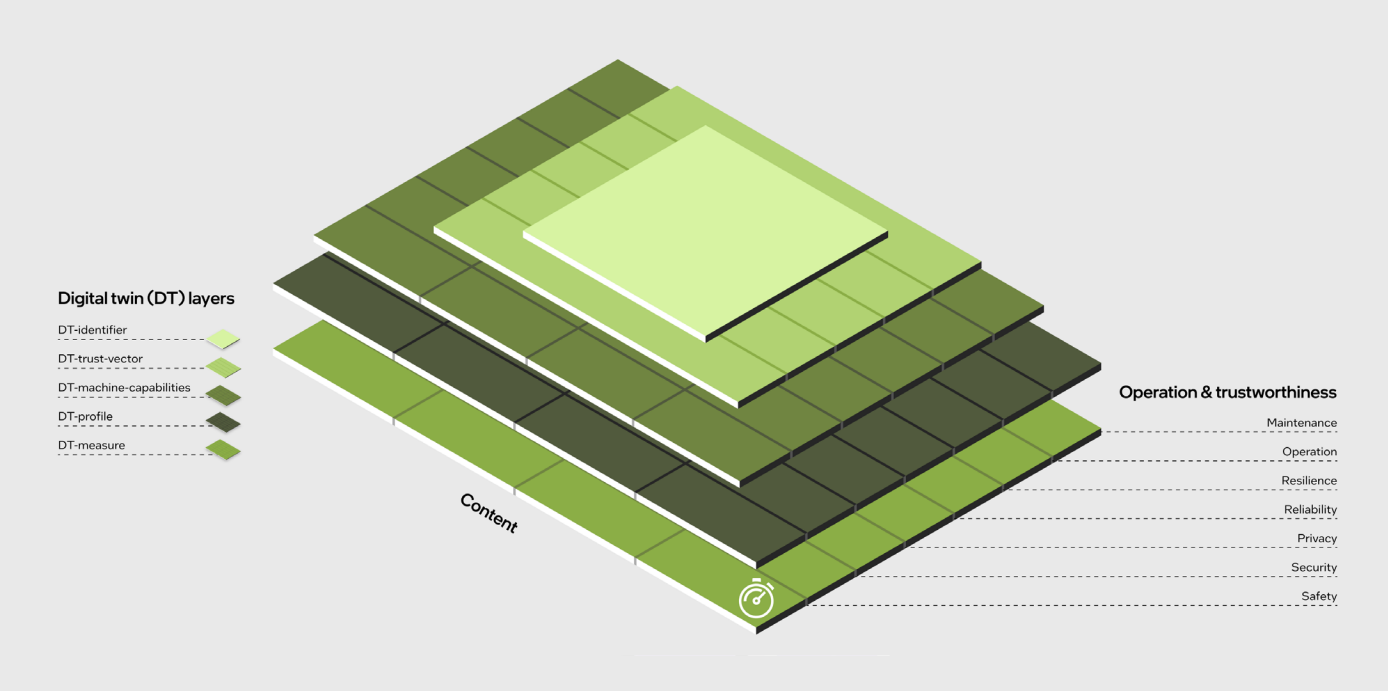

Figure 3. 4D representation of a digital twin.

authorities have a duty to set boundary conditions that allow the use of technologies in order to maintain competitiveness in the international market. Despite these challenges, the proposed system is already applicable in situations where the timing allows for human approval of the system's decisions, for example in commissioning, decision support in maintenance or for strategic simulations for changes.

4.1 Runtime risk management

In traditional manual safety concepts, there are several tasks required to fulfill the regulatory requirements, which are basically set by the ISO 12100 standard[7]. The relevant tasks include hazard identification, risk estimation (and evaluation) and selecting appropriate protective measures. Traditionally, these tasks are done during the design and construction of the machinery, or, at the latest, at commissioning. However, the crucial aspect is that these tasks are done once and manually, and the selected protective measures stay the same for the machinery's whole lifetime. A fundamental assumption in this process is that the machinery and its context are static or, if parts are not expected to be static, worst-case scenarios are considered. However, the future flexibility of production with its infinite and unforeseeable combination of possibilities of hardware, software and operational processes will lead to overly restrictive protective measures, due to the necessary excessive worst-case scenarios. Furthermore, the rising complexity of the production and its context will make it impossible to consider all safety-relevant possibilities during the design and construction phases. To overcome these challenges, we propose to dynamically react to the current context and adaptively activate or release protective measures. With this approach to operational safety intelligence, overly restrictive measures are avoided and productivity is kept high and competitive, along with being flexible to react to changes.

To guarantee overall safety, the operational safety intelligence software continuously supervises the selection of appropriate protective measures, so it manages the risks at **runtime**. This runtime risk management requires runtime risk evaluation and runtime hazard identification. It should be noted that the term runtime depends on the context, so given a modular process plant, the changes might only occur every few weeks, whereas an AMR might enter a new context every few minutes or even seconds. This implies at least partly automated solutions. Higher degrees of automation enhance the productivity even further and offer other advantages. Automation of the aforementioned three tasks for safety is provided by the operational safety intelligence software that is established to identify hazards, estimate the risks and select protective measures. If direct intervention of the automatic functionality is not possible or requested, the system can at least support in these tasks by making suggestions and help in safety-related decisions. The operational safety intelligence software can finally be considered to have kind of a "feeling" for hazards and risks. Throughout the following chapters, the requirements and a proposed architecture of such a system are derived.

4.1.1 Hazard identification

Our approach is to investigate how human experts fulfill the tasks and extract the requirements for the software from this manual process. The first task is hazard identification, which is focused on in this paper. Today's hazard identification by experts is mainly based on their experience and may be supported by specialized standards. The task for the experts in the future will be to teach their knowledge and experience to the software system. Hence, it is essential that the experts can easily and efficiently program their decision-making into the software. To reach this goal, it is firstly crucial that the experience and the knowledge of the experts are formalized and structured. This formalization can be regarded as a unified language for safety with precise definitions of terms — the **safety semantic**. The digitalized safety knowledge of experts will then be part of the operational safety intelligence software, see also the overview of the data and information flow in Figure 4.

Given the semantically structured language, the digital representation of the machine can be built up — the digital twin. This collection of all information about the machine, enhanced with contextual information and common knowledge, needs to be understandable and evaluable for the operational safety intelligence software working on it. The digital twin includes offline static data, for example, data sheets, as well as online dynamic data, for example, sensor measurement values. Of major importance is that it must be very easy to search pieces of information in the digital twin with its huge collection of information. This means it must be possible for the software to find elements with specific properties, but also with specific attributes and context.

As an example, two temperature values shall be considered in the following. One temperature value is a measured value of a sensor, the other is a fixed parameter, for example, the maximum measurable temperature of the sensor. For the values in the digital twin, not only the numerical value itself must be stored, but also contextual information to distinguish these temperature values. This contextual information includes, for example, the physical unit of the values, measurement accuracies or similar. However, this information is not yet sufficient for automatic hazard identification. For example, the maximum measurement temperature could be detected as a burn hazard, since an automatic search for temperature values greater than a burn temperature threshold would return both measurement and parameter values. Thus, it must be stored in the contextual information of the values whether it is a real measured value or a parameter. The meanings of **measured value and parameter** are defined in a standardized semantics, so that the automatic hazard identification can distinguish between them. The assignment of the semantic terms must be entered correctly once when creating the digital twin of the sensor. Then, an automatic search function can correctly interpret the values, not only the numeric value itself (value greater than a burn temperature threshold), but also its meaning (value is a real, current temperature measurement).

The search function indicated above is another necessary innovation, which is called **hazard** rule in this concept. A logic structure for the hazard rules that is similar to a search function makes it possible not to require fixed links between the information source and the processing logic. The hazard rules are based on semantic terms, which are then used to search the information in the digital twin.

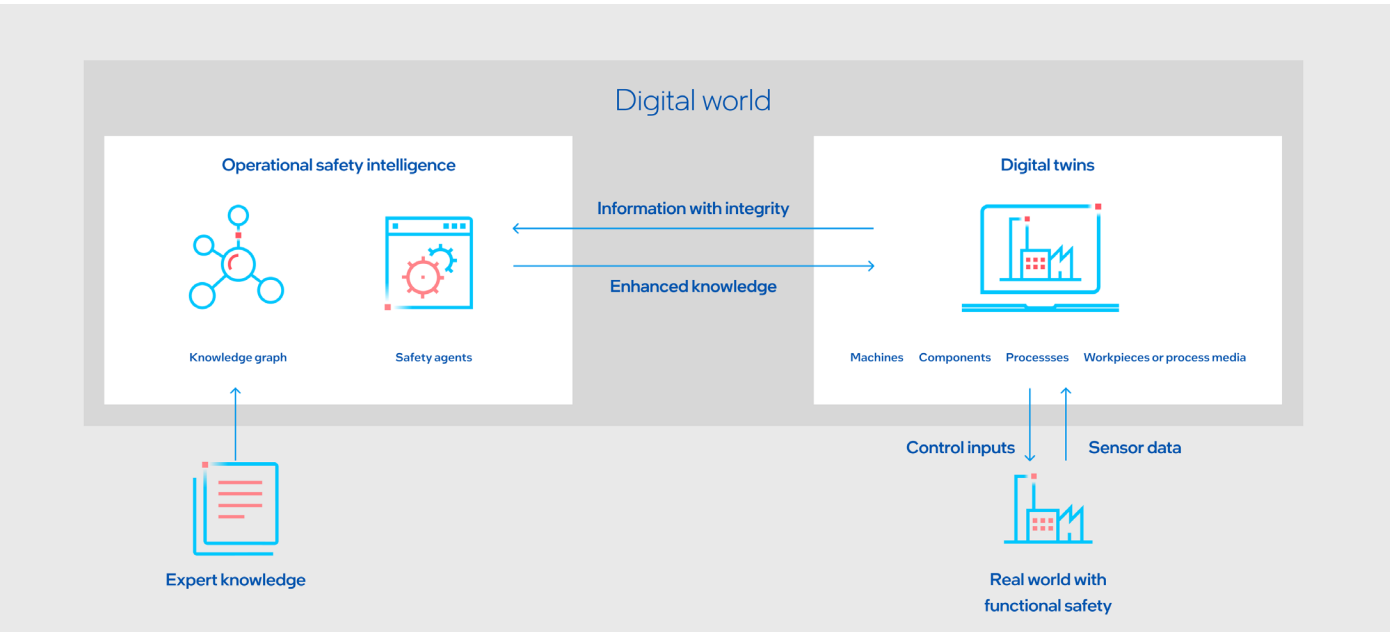

Figure 4. Flow of information.

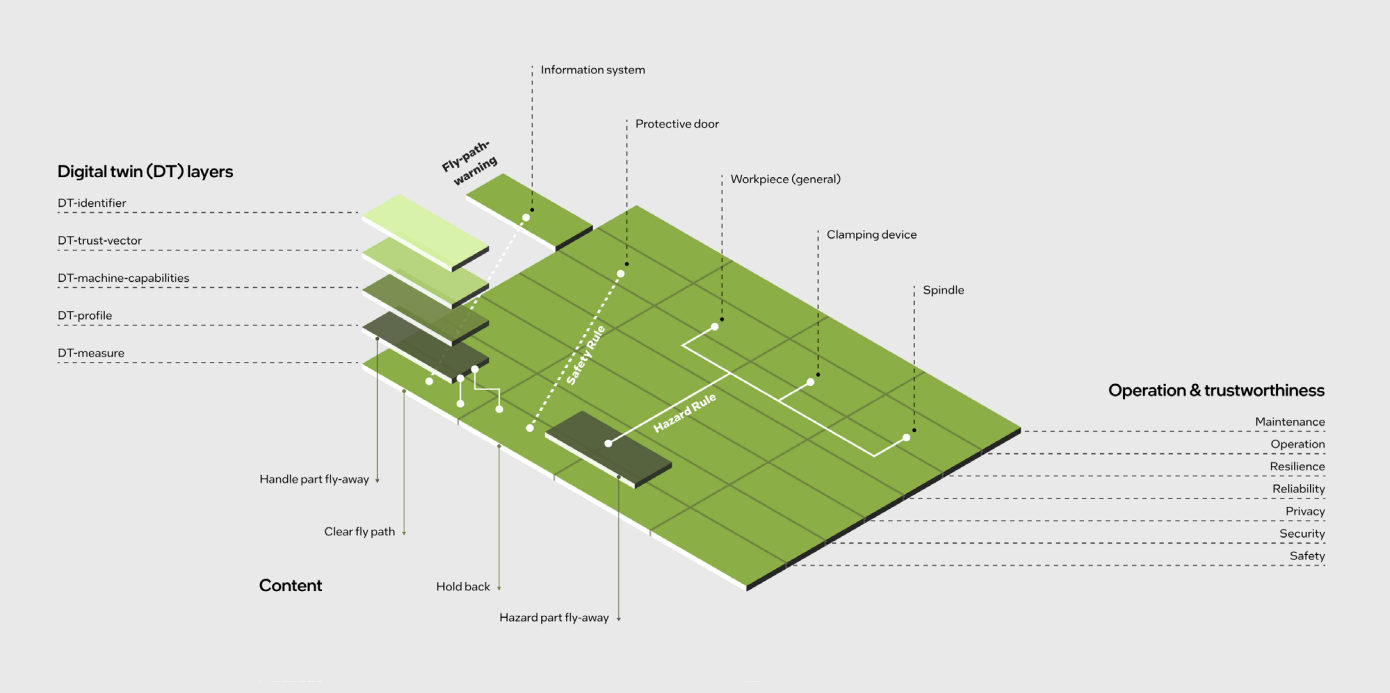

Figure 5. Exemplary depiction of the hazard rule part fly-away.

To give an example, this means that a port of the programmable logic controller (PLC) does not have to be statically linked to the burn temperature threshold in an if-then function to indicate the hazard, as was previously the case. The hazard rule would be a search for current temperature measurements that are above the burn temperature threshold. This means that not only one predefined sensor is evaluated, but other dynamically added sensors and computed parameters could also be monitored. The result is a part of the operational safety intelligence software that monitors all temperature values that fulfill certain semantic and contextual information. If one of these temperatures is above the threshold value, a burn hazard is detected at the position of the sensor (position would be another piece of contextual information), or if none of the temperatures is above the threshold, the burn hazard is (currently!) not given. This can then be used to react accordingly, either by displaying a warning or by giving the green light for maintenance, for example.

The hazard rules describe when a certain hazard occurs, that is, which conditions must be fulfilled for a dangerous situation to be possible from a technical and physical point of view. However, this does not mean that an accident will immediately occur if the hazard rule is fulfilled. In the context of the example of temperatures, the hazard rule would be active if a sensor measured a temperature above the burn temperature threshold. However, as long as no one is near the hot object, no one can burn themselves. Conversely, if all temperatures are below the threshold, the hazard (rule) is inactive, and the objects can be touched safely. This example also shows that the hazard rules link many different sources of information in the digital twin, as illustrated in Figure 5.

Additionally, it must be possible to manually enter, check or correct the hazard rules. A problem might be that the automatic functions may not recognize some specific hazards from the available information, maybe due to the complexity of the hazards or incomplete information. It must therefore be possible to initialize further hazards manually during a check by experts so that completeness is achieved. Furthermore, (safety) data sheets, standards or regulations may also contain information on special hazards, which must then be considered independently of automatic detection functions. However, if the operational safety intelligence software is not taught why these hazards exist, meaning based on which information these hazards can be evaluated, the system will fall back to the recent static systems, as it cannot decide about these hazards.

4.1.2 Risk calculation

The term **risk calculation** is introduced in this context as a bracket around risk estimation and risk evaluation according to the ISO 12100 standard[7]. But it is not limited to the manufacturer's perspective — the operator side will also make use of it. Furthermore, risk calculation mainly describes a mathematical and programmable way of treating risk, as in the operational safety intelligence software, risk calculation will be performed often.

The mechanism for risk calculation is similar to the descriptions in the previous sections. The current level of risk is calculated based on the collection of all information in the digital twin. So, roughly speaking, if, for example, few hazard rules are active, there is low risk; or, if nobody is near the machine, there is also low risk. All in all, the risk will be variable and dynamically calculated, depending on a lot of input information. Finally, this current value is compared to the (static) risk acceptance level, given by standards, society or common sense. In case the current risk is too high, more or other protective measures are necessary to mitigate the risk. Or, if this is not possible, the machine needs to be shut down. Conversely, if certain hazard rules are inacive, so the current risk level is low, protective measures can be reduced, which avoids overly restrictive measures.

It is worth noting here that the input information might originate from other devices outside of a single machine, which may influence the risk calculation and finally the behavior. An example would be an AMR that can monitor objects or humans in a certain area around itself based on its own sensors. But at cross-sections at the end of a narrow corridor it needs to slow down, as it cannot see around the corners. This slowdown is an additional protective measure, as the calculated risk rises at the corner, due to blind spots. However, if there would be a (trustworthy) connection to a fixed camera system at the cross-section or other AMRs, so that the AMR can exclude humans from being around the corner, it may drive faster in the cross-section. So based on the available information and trustworthiness of this information, the behavior of the system may change.

4.1.3 Protective measures

As mentioned already in the example in the previous section, the final step is to select appropriate protective measures. This step first requires knowing the available protective measures. In this case, protective measures cannot be automatically recognized as such from a digital twin, unlike hazards, as protective measures usually must meet specific normative requirements. So, the available measures need to be introduced manually in the system at the design phase. Nevertheless, it is possible afterward to automatically compare the protective measures with the identified hazards and the calculated risk by the operational safety intelligence software, by use of the safety rules. If the protective measures mitigate the calculated risks below the risk acceptance threshold for all hazards, the machine is considered to be safe. In case the risk is still too high, other protective measures are needed and may need to be set up manually. In this case, the operational safety intelligence software may support the operator's decisions by making suggestions about possible measures, prioritizing according to the principles given in standards and calculating how the possible measures affect the risk level.

As technology develops further, the protective measures that need to be applied to fulfill the state-of-the-art regulations evolve, too. Keeping track of these changes, hence keeping the production plant always up to date, can be easily realized with the system described here. This can be achieved by updates to the knowledge base of the system and the evaluated standards that the system uses. This functionality is an important feature, especially for operators of production plants, to exclude liability situations arising from old-fashioned or insufficient protective measures.

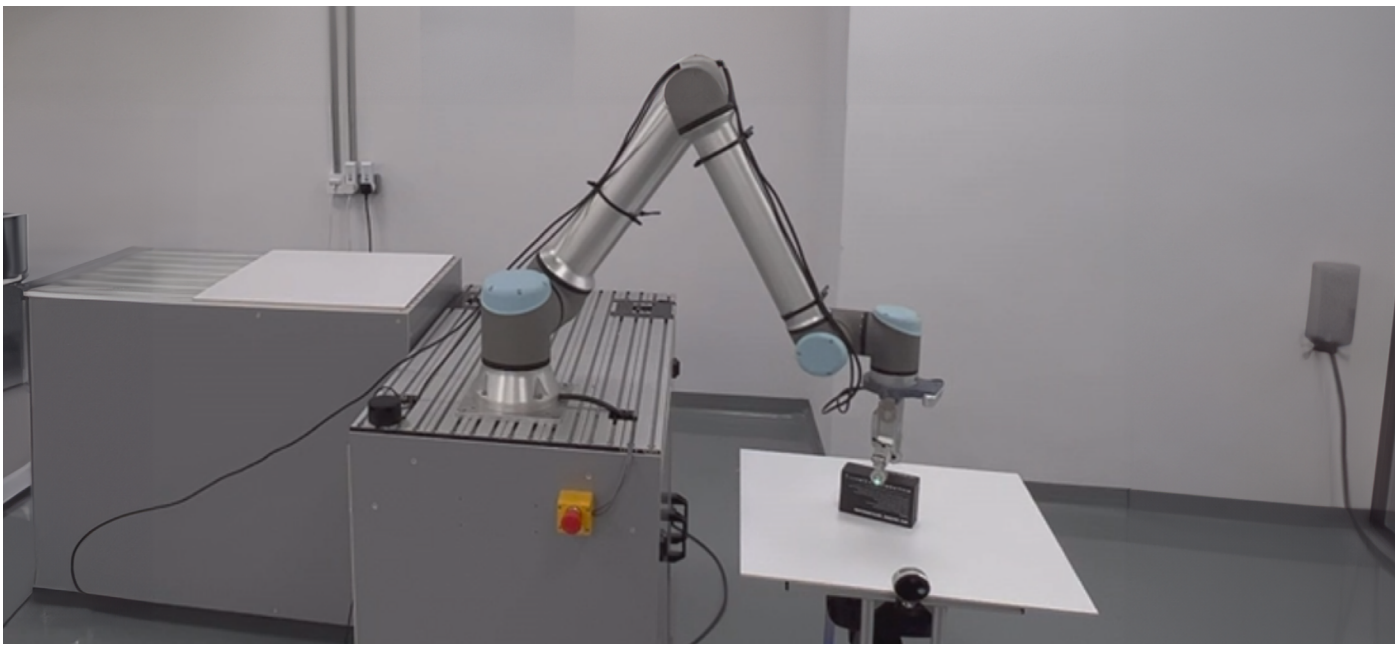

Figure 6. Depiction of the use case of operational safety intelligence.

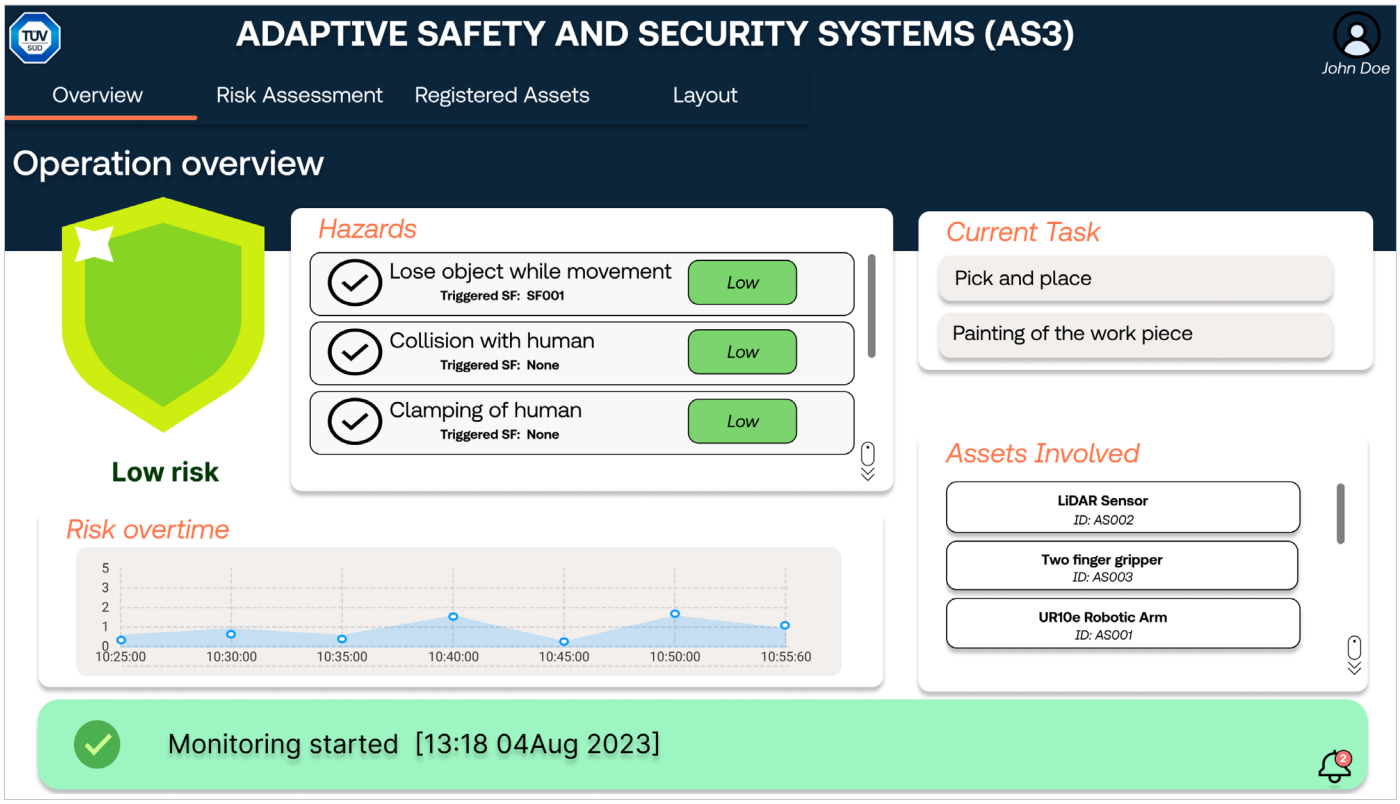

Figure 7. Dashboard of the Adaptive Safety and Security Systems (AS3) of TÜV SÜD, which is a prototype of the operational safety intelligence software.

4.2 Use case runtime risk management

To show the working principle of operational safety intelligence, a simple demonstrator is set up, as shown in Figure 6. The setup consists of a UR10e robot of Universal Robots, which is equipped with a two-finger gripper and mounted on a platform. As it is a collaborative robot, there are no fences around the setup and humans could interact with the robot. Nevertheless, there is a lidar sensor at the setup to track whether humans are in the vicinity of the robot. The robot should carry out a simple pick-and-place task. It grabs carton boxes on one table and puts them on another table. Furthermore, the operational safety intelligence software is monitoring the setup and evaluating whether the state of the setup is safe. The regarded data and the reported state are shown on a dashboard, which can be seen in Figure 7.

In the first scenario, the hazard to be considered is that the robot might drop the object on its way from one table to the other. This could happen if the object is too heavy for the two-finger gripper and the robot turns too quickly. Therefore, this case is represented as a hazard rule in the operational safety intelligence software. This hazard rule evaluates the weight of the object and the speed limit of the robot. In the case that the speed limit is too high for the specific weight of the object, the hazard rule reports an unsafe state before the task is really started. This prediction skill based on the preset speed limit means that the hazardous situation is recognized before the actual speed of the robot is at the limit. The data of the workpiece, but also the current speed limit of the robot, are gathered from the respective digital twins of the workpiece and the robot. However, if the speed limit of the robot is reduced, it will not lose the object, and hence the state is considered safe again. In the scenario the speed reduction determines the safety rule, which mitigates the considered hazard rule. By using a dynamic speed limit dependent on the weight of the object, the productivity of the setup is higher for lightweight boxes, in comparison to a fixed and low speed limit, which considers the heaviest box as a worst case to determine the speed limit of the robot.

In the second scenario, the interaction of the lidar sensor and the robot is regarded. The underlying hazard in this case is that a human might enter the workspace of the robot and a collision might occur. To mitigate this risk, virtual walls are activated for the robot as soon as the lidar sensor recognizes a human. This behavior is represented by a safety rule in the operational safety intelligence software. It is worth noting that in this scenario there is no direct information connection between the lidar sensor and the robot. The information transfer is managed by the operational safety intelligence software, which gathers all safety-relevant data of the whole system, but also processes this data and provides enhanced information to the specific components of the setup. For example, if one replaces the lidar sensor with a different sensor type or adds another

sensor, the connection to the sensor or the sensor fusion capabilities are managed in the operational safety intelligence software. This adds flexibility to the system, as all safety-relevant data is processed and provided by the operational safety intelligence software and the single components do not need to care about data management and integrity.

4.3 Safety and artificial intelligence (AI)

Operational safety **intelligence** as proposed contains a certain intelligence for its specific tasks, and it is **artificial** as it is created by humans, however calling it an **AI** is critical. Firstly, according to the VDE-AR-E 2842-61-1 standard[1], a precise definition of AI is difficult, and the understanding of AI is very different among individuals and departments, leading to misunderstandings. Secondly, the use of AI for safety systems might be seen skeptically by many people, especially in the functional safety community. Hence it needs to be emphasized that the use of AI in this system is restricted to noncritical parts and several requirements must be strictly fulfilled. Furthermore, in case there are decisions made by AI a human is, up to this point, closely involved and supervises or approves the decisions.

The opacity of AI inferencing causes a growing skepticism toward AI. Making AI decisions transparent will allow human experts or automated routines to prove the correctness of the AI, which is called **explainable AI**. But even clearly comprehensible AI decisions leave room for the question of liability. Jurisprudence, or society at large, has not yet reached a consensus on how AI can be made liable. That means the use of AI in this system is thus far restricted to cases in which no liability issue may arise. Possible examples are production planning or only simulated environments, but in real production, only humans are allowed to take action, perhaps supported by AI. Hence, at this point in time, AI can possibly provide advanced levels of support to humans but cannot be solely in charge of decision-making.

Another recent development involves large-language models (LLMs) that can understand human language and generate texts. This type of AI is of special interest in the presented context as it can operate on semantic data structures. This means that specially trained LLMs can possibly support humans to operate on the large-scale collected data of machinery and its contextual information. But the results of the LLMs are still checked and verified by humans, so that the responsibility is still in human hands. A possible use case would be the support for the generation of the hazard rules. It might be easier for the safety experts while generating these rules to use rather natural but semantically structured language than directly "programming" them in a computer language. Later on, the generated code or rules respectively can be checked and verified by the (human) experts.

It is important to emphasize that the knowledge that describes how hazards arise originates directly from experts. The hazard identification is not learned by AI in the sense of machine learning algorithms, as it is in the case of image recognition, for example. Machine learning algorithms would require a great deal of data on hazardous and dangerous situations that should actually be avoided. Also, simulations to generate this data without endangering real people are not suitable either, as these would also have to be programmed first. The specialist knowledge of experts in hazard identification is therefore directly transferred into software and not indirectly acquired using machine learning algorithms. Nevertheless, the experts are supported by AI for setting up the system.

5. Semantic data framework

5.1 Overview

In this section, we introduce the **semantic data framework**, which is capable of fulfilling the requirements of operational safety intelligence. Data and information modeling is crucial as it provides a structured and clear understanding of data, which is essential for accurate analysis and decision-making. Furthermore, it enhances the efficiency and effectiveness of operations by enabling seamless data integration, ensuring data consistency and facilitating communication between different systems, thereby supporting operational safety intelligence. From the discussion so far, we derive the following questions on the data model side:

1. How can data be contextualized and shared between parties without losing the context?

2. How can formal data constraints and inferencing rules be described?

3. How can data constraints and inferencing rules be enforced at runtime?

In a globally connected world, where data sharing and collaboration occur across diverse domains and disciplines, the concept of shareable data with cross-domain semantics becomes crucial. To implement a shareable data concept, several elements are necessary:

- Standardized ontologies:

In the context of information technology, an **ontology** is a structured representation of knowledge within a specific domain. It systematically defines and categorizes concepts and outlines the relationships between them. The call for standardized ontologies stems from the need for a

universally recognized system that can simplify the understanding and usage of complex information within a domain. This standardization promotes uniformity in data interpretation, thereby enhancing communication and comprehension across various platforms.

▪ Semantic interoperability

Semantic interoperability is a critical aspect of information technology that ensures the precise exchange and understanding of data across diverse systems. It guarantees that when data is transferred from one system to another, the meaning of the data remains consistent and unambiguous. This is crucial in maintaining the integrity of the data and ensuring that it is accurately interpreted, regardless of the system it is used in. Achieving semantic interoperability is essential for enabling seamless communication and effective data utilization among different technological ecosystems.

▪ Linked data principles

Linked data principles are a set of best practices for sharing structured data on the web, making it interrelated and easily accessible. These principles advocate using standard hypertext transfer protocol (HTTP) uniform resource identifiers (URIs) to identify data objects, allowing them to be accessed and manipulated using standard web protocols. Furthermore, they encourage linking data objects to other related data objects, thereby creating a web of interconnected data. The implementation of these principles fosters a more open and interconnected web environment, enhancing the accessibility and utility of data across various platforms.

These elements can be implemented with concepts of the **Semantic Web**[8,9]. The Semantic Web aims to enhance the way information is processed and understood by both humans and machines on the internet. It achieves this by adding a layer of structured data and meaning to content, allowing for more effective data integration, sharing and interoperability. Originally, the Semantic Web was conceived to enhance the World Wide Web. However, over time, the concept of Semantic Web has evolved beyond its original scope of just the web and has been recognized as a broader approach to achieving semantic integration and interoperability across various domains and applications.

5.2 The Semantic Web

Recently, advances of the Semantic Web have been driven by:

1. Interoperability demands: With the growth of digital information and the proliferation of disparate data sources, the need for interoperability and data integration across diverse systems became increasingly important. The Semantic Web principles provided a solution for achieving meaningful communication and integration between data sources.

2. Semantic interlinking: The idea of linking and interconnecting data using semantic relationships gained traction as a powerful means of deriving new insights and knowledge. The interlinking concept extended beyond the boundaries of the web and was embraced in other contexts, such as enterprise data integration, research collaboration and data analysis

3. Machine learning and AI: The rise of machine learning and artificial intelligence further underscored the importance of structured and semantically enriched data. These technologies rely on clear semantics to understand and reason about data, making the Semantic Web principles relevant beyond traditional web applications.

4. Cross-domain benefits: The benefits of shareable and serializable semantics such as improved data discovery, integration and reasoning proved to be valuable across different domains and industries. This led to exploration and adoption of semantic technologies in areas that were not originally part of the web-centric vision.

5.2.1 Resource description framework

The foundation of Semantic Web is the **resource description framework** (RDF)[10]. RDF allows the serialization of a graph dataset by so-called **triples**: Subject, Object, and Predicate (see Figure 8). Formally a triple can also be described as three terms (in Turtle[11] serialization syntax):

```
ex:subject ex:predicate ex:object .
```

Here `ex:` declares an example namespace to clarify where the terms come from. As can be seen from Figure 9, these triples are building blocks for arbitrarily complex graphs.

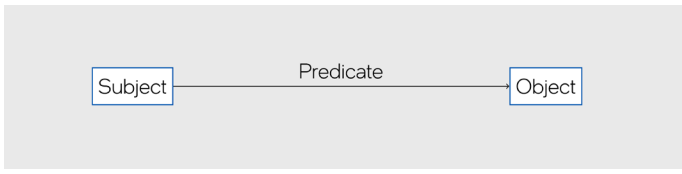

Figure 8. RDF triple.

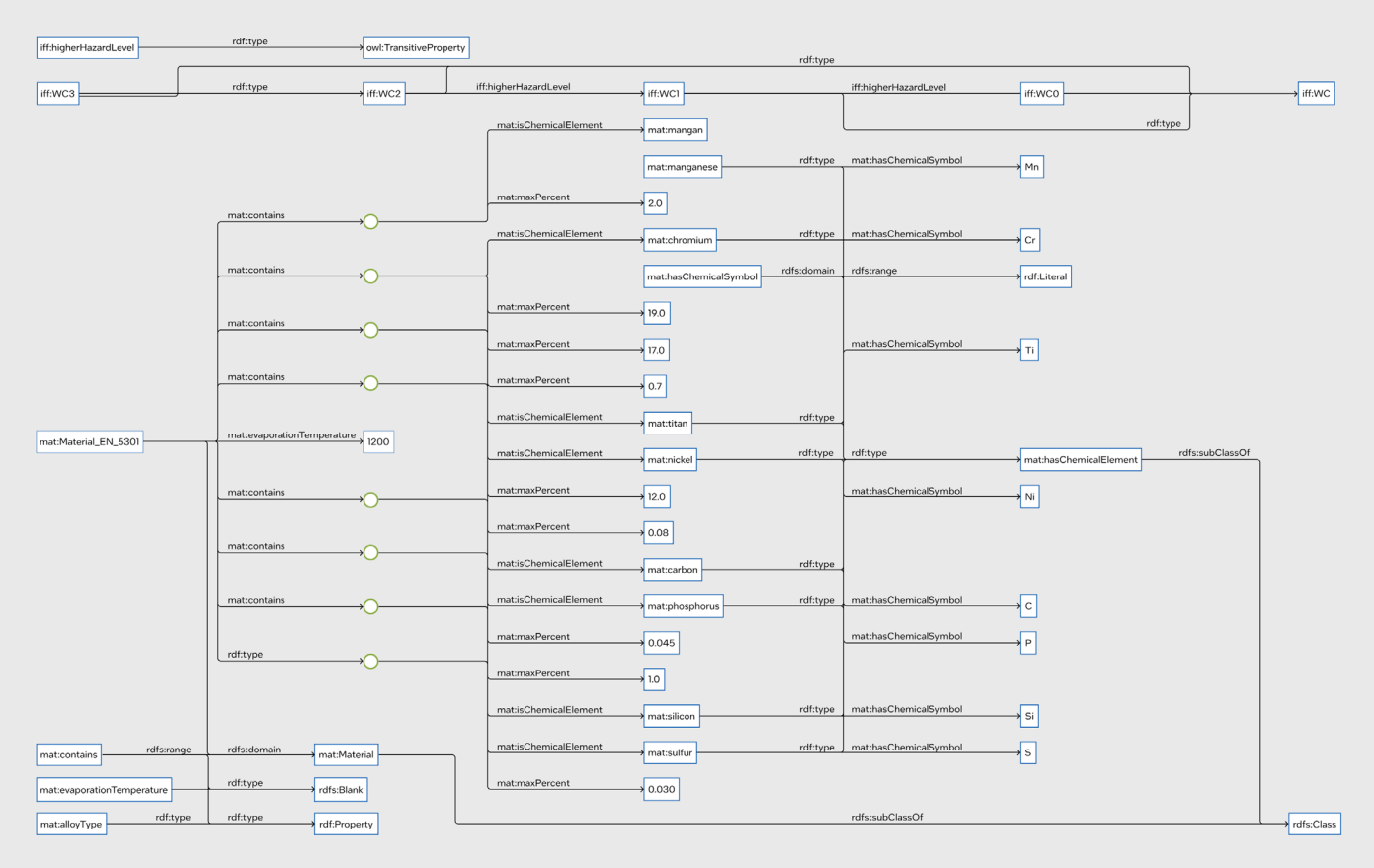

Figure 9. Graph built from RDF triples.

Typical relationships, such as classes, subclasses, instances and functional properties such as transitivity are described in RDF schema (RDFS) and Web Ontology Language (OWL)[12].

One of the long-term challenges in RDF is its expressivity. OWL can create undecidable rules and thus was restricted by profiles such as OWL2 RL[13].

5.2.2 Semantic data objects and JSON-LD

In the realm of modern software systems, object serialization is predominantly handled using JSON or YAML, making these formats familiar territory for developers. However, the concept of RDF triples and the graph view of data often remain foreign and complex. To bridge this gap, JSON-LD[14] was developed, serving as a conduit between the intricate world of RDF triple data and the more digestible JSON. Essentially, JSON-LD is a serialization of RDF data that seamlessly fits into a standard JSON object. This compatibility allows developers to continue leveraging their existing JSON-based infrastructure while simultaneously unlocking the potential of the semantic data world. Thus, JSON-LD not only simplifies the integration of structured metadata into web services but also fosters a more widespread adoption of semantic web technologies. The relationship between JSON-LD and RDF is shown in Figure 10.

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix iff: <https://industry-fusion.com/types/v0.9/> .
@prefix schema: <http://schema.org> .

<urn:plasmacutter:1> rdf:type iff:plasmacutter .
<urn:plasmacutter:1> schema:serialNumber "12345" .
<urn:plasmacutter:1> iff:hasFilter <urn:filter:1> .
<urn:plasmacutter:1> iff:hasWorkpiece <urn:workpiece:1> .
<urn:plasmacutter:1> iff:state> "OFF" .
```

```
"@context": [ … ],
„@id": "urn:plasmacutter:1",
„@type": "iff:plasmacutter",
"iff:state": "OFF",
"schema:serialNumber": "12345",
"iff:hasWorkpiece": {
    "@id": "urn:workpiece:1"
},
"iff:hasFilter": {
    "@id": "urn:filter:1"
}
```

Figure 10. RDF triple serialization with Turtle (left) and with JSON-LD (right).

Figure 11. Plain JSON-LD (left) and the extension NGSI-LD (right).

5.2.3 NGSI-LD: Data reification, properties and relationships

Reification is a process in semantic web technologies that allows users to make statements about other statements, essentially providing a way to express complex ideas or relationships. However, JSON-LD, while being a powerful tool for bridging the gap between RDF triples and JSON, falls short in its ability to handle reification effectively. This limitation can lead to complexities and inefficiencies in data representation and processing.

To address this shortcoming, NGSI-LD[15] was developed. **Next Generation Service Interface-Linked Data**, or NGSI-LD, extends the capabilities of JSON-LD by introducing a standardized way to express reified statements. It enhances the JSON-LD data model by incorporating the concepts of **Property-of-Property and Relationship-of-Relationship**, thereby enabling more complex and nuanced data relationships to be expressed in a structured and standardized way.

Moreover, NGSI-LD introduces semantic concepts for time such as "observedAt," "createdAt" and "modifiedAt." These temporal attributes add another layer of information to the data, enabling more precise and meaningful interpretations. By incorporating these time-related semantic concepts, NGSI-LD not only enhances the capabilities of JSON-LD but also makes it more suitable for applications requiring detailed temporal data and sophisticated time-based queries. This not only improves the expressiveness of JSON-LD but also makes it more suitable for a wider range of applications, particularly those requiring complex data relationships and semantic interoperability. An example for a JSON-LD object and its NGSI-LD extension is given in Figure 11. It shows an object of type `iff:plasmacutter` that has the identification `urn:plasmacutter:1` containing for instance properties describing state value, serial numbers and relationships to other entities (`iff:state`, `iff:serialNumber`, `iff:hasWorkpiece`, `iff:hasFilter`). The NGSI-LD example in this figure shows in addition to the former JSON-LD–based information that all attributes are Properties or Relationships to other entities. Moreover, the `iff:state` Property contains additional metadata such as time when it was observed or safety-relevant information on how reliable the value is. Since NGSI-LD can explicitly differentiate between links to entities and links to knowledge, NGSI-LD objects build a graph of entities as shown in Figure 12.

5.2.4 Data constraints and validation

Data constraints and inferencing have been around in the Semantic Web for more than 20 years. Even though it is possible to express constraints in OWL, it is essential to clarify that OWL's main purpose is deduction of facts and not checking constraints of data structures. For instance, assume that the predicate (or more specific RDF Property) `ex:hasBiologicalFather` describes that a person has a biological father. One logical constraint would then be `owl:maxCardinality 1`, i.e., every person can have at most one biological father. How would an OWL system react to the following?

```
ex:Bob ex:hasBiologicalFather ex:Chris .

ex:Bob ex:hasBiologicalFather ex:Dave .
```

In this case, one would expect that the cardinality constraint of 1 for the RDF Property `ex:hasBiologicalFather` would trigger a constraint violation. Actually, what OWL deduces from this is

```
ex:Chris owl:sameAs ex:Dave .
```

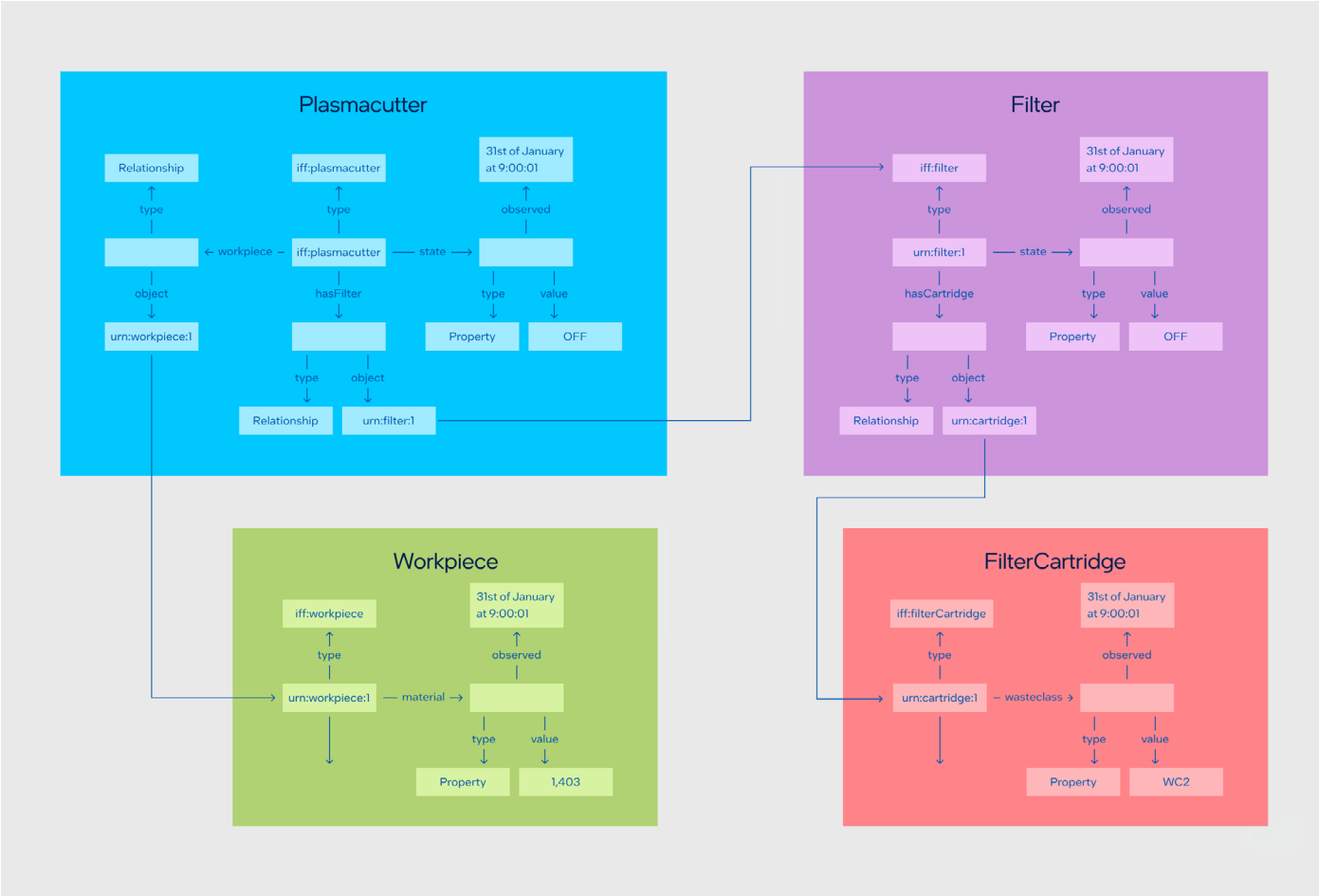

Figure 12. Visualization of a graph of entities described by NGSI-LD.

which is a logical conclusion from the constraints. This is interesting to know, however, it is not a meaningful answer to the question about data consistency.

The World Wide Web Consortium (W3C) realized that problem and closed the gap by creating Shapes Constraint Language (SHACL) [16]. SHACL can be used to express constraints on data. For instance, the constraint above can be formulated (in Turtle syntax[11]) as:

```
ex:Person

rdf:type sh:NodeShape ;

sh:property [

        sh:path ex:hasFather ;

        sh:maxCount 1 ;

] .
```

In the case above, the data constraint violation of `ex:Bob` would be detected if `ex:Bob rdf:type ex:Person` is fulfilled, i.e., `ex:Bob` is an instance of `ex:Person`.

SHACL can express complex relationships within a graph by using **SPARQL**, the W3C standardized query language for graphs. Since SPARQL can be mapped to SQL, it is possible to formulate SHACL constraints in SQL and, thus, SHACL validation can be achieved with modern SQL query engines. OWL is still relevant in areas where complex ontologies are defined and deduction is needed. A typical setup is that OWL is used to describe ontologies of an expert domain and is then used to derive rules and domain knowledge that is relevant for the real-time SHACL evaluation. Since SHACL can be applied to an arbitrary RDF graph, it can be applied to a graph of NGSI-LD entities as well. With that, it is possible to describe runtime constraints for entity graphs as shown in Figure 13.

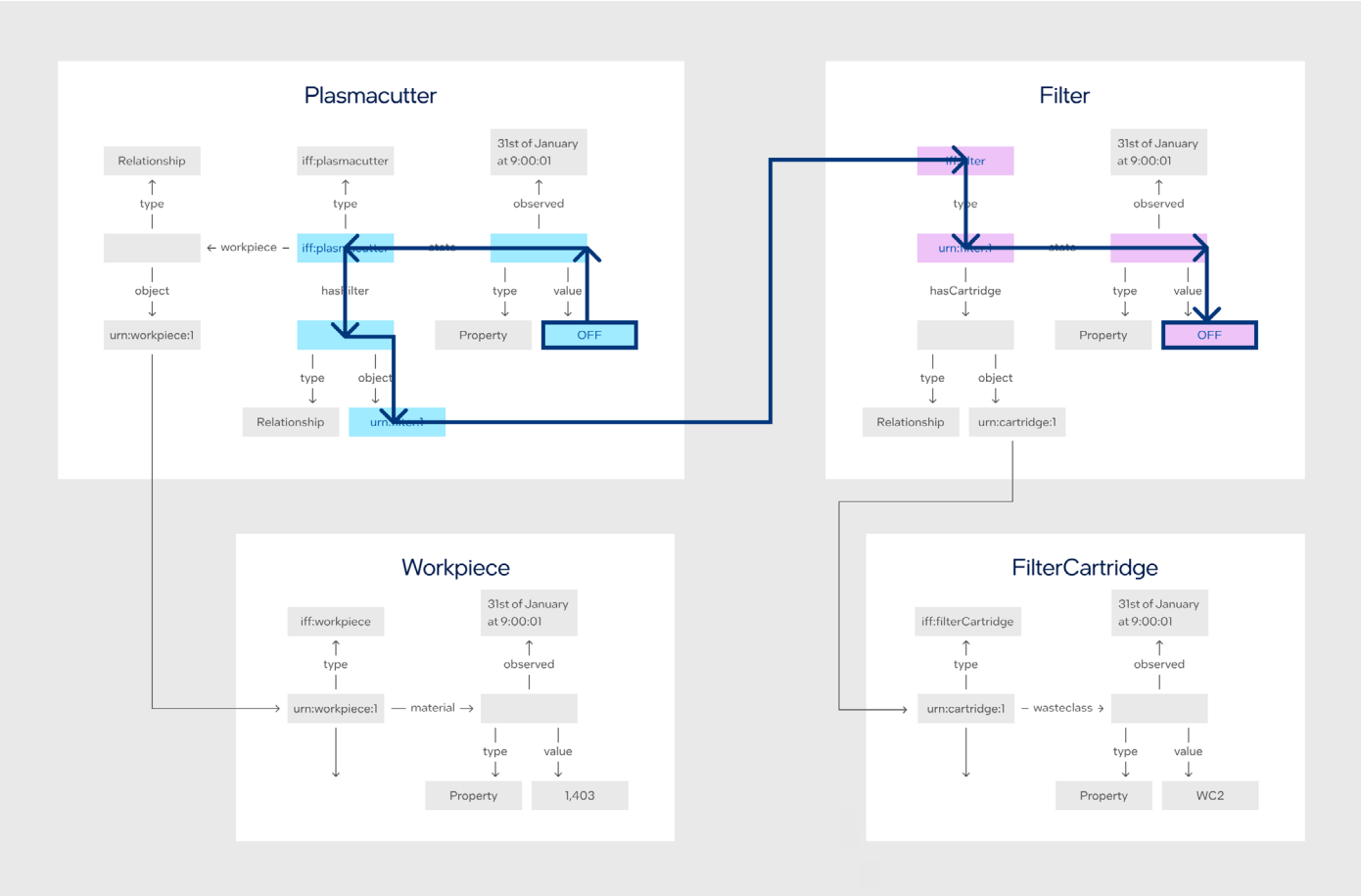

Figure 13. Runtime constraints in an entity graph defined by SHACL.

5.2.5 Summary

In this section, the semantic data framework was introduced, designed to fulfill the requirements of operational safety intelligence. The importance of data and information modeling was emphasized, highlighting its role in providing a structured and clear understanding of data for accurate analysis and decision-making. The framework was shown to enhance the efficiency and effectiveness of operations by enabling seamless data integration, ensuring data consistency and facilitating communication between different systems.

Several questions about data modeling were derived from the discussion, including how data can be contextualized, i.e., shared without losing context, how formal data constraints and inferencing rules can be described and how these rules can be enforced at runtime.

The necessity of shareable data with cross-domain semantics in a globally connected world was discussed. To implement this concept, several elements were identified as necessary, including standardized ontologies, semantic interoperability and the application of linked data principles.

The Semantic Web was presented as a tool capable of implementing these elements, enhancing the way information is processed and understood by both humans and machines on the internet. The evolution of the Semantic Web beyond its original scope of just the web was also discussed, recognizing it as a broader approach to achieving semantic integration and interoperability across various domains and applications.

The **resource description framework** (RDF), the foundation of the Semantic Web, was introduced, which allows for the serialization of a graph dataset. The long-term challenge of RDF's expressivity was acknowledged. To bridge the gap between RDF triples and the more digestible JSON, the benefits of JSON-LD were shown.

NGSI-LD was described to address the shortcomings of JSON-LD in handling reification (such as changes in time) effectively. It was shown to enhance the JSON-LD data model by incorporating the concepts of Property-of-Property and Relationship-of-Relationship.

The importance of data constraints and validation was discussed, and SHACL, the Shapes Constraint Language, was introduced as a tool to express constraints and

inferencing rules on data. The application of SHACL to an arbitrary RDF graph was demonstrated, making it possible to describe runtime constraints of combinations of NGSI-LD entity graphs with knowledge graphs.

6. Derived software and hardware architecture

6.1 Software architecture

In this section, the requirements from the Dynamic Safety concept are translated to a software architecture.

1. We conclude that a Dynamic Safety concept must be able to work on a huge amount of data. This creates a need to analyze data in a scalable way.

   Requirement: The system must be scalable and hence distributed.

   Solution: Cloud-native large-scale analytics architecture as described in section 6.1.2.

2. The data must be analyzed at runtime with low latency to guarantee a deterministic reaction to detected safety hazards.

   Requirement: Determinism and low latency.

   Solution: Due to CAP theorem (see section 6.1.1 and below), there cannot be a single system that fulfills the requirement. Therefore, such a system must consist of several autonomous systems. The low-latency data analytics and the PLC layer must be independent of each other, linked only by a common data model.

3. The data that is analyzed must be shareable (or serializable), queryable, semantically defined and must be capable of expressing knowledge.

   Requirement: Formal knowledge and declarative search/querying.

   Solution: Semantic Web, RDF, JSON-LD, OWL.

4. It must be possible to define constraints and rules on the data in a standardized way, shareable for specific expert domains.

   Requirement: Declarative data language.

   Solution: Semantic Web, SHACL and SPARQL.

5. The data from the digital twin must be synchronized with the real world in low latency.

   Requirement: No bottom-up or unidirectional data processing.

   Solution: Hybrid edge architecture (as described in section 6.1.4).

6.1.1 Cloud-native architecture, the CAP theorem and scaling

Cloud-native architecture, with its emphasis on scalability and reliability, is fundamentally enabled by horizontal scalability[17]. By distributing workloads across multiple instances, cloud-native applications can dynamically adapt to changing demands, ensure high availability and optimize resource usage. Therefore, whenever a system with large-scale data needs is designed, the cloud-native architecture principles of designing distributed systems are applied.

The **CAP theorem**, also known as **Brewer's theorem**, is a fundamental principle in distributed systems that has significant implications for cloud-native architectures. The theorem states that in a disturbed system, it is impossible to achieve all three of the following properties: consistency, availability and partition tolerance[18].

A famous conclusion of the CAP theorem is that SQL databases with their Atomicity, Consistency, Isolation and Durability (ACID) constraints cannot scale well and therefore, the No-SQL databases have been invented, trading availability for consistency and having a so-called Eventually Consistent model.

Therefore, one can conclude that every distributed system design must be clear about the trade-offs made.

6.1.2 Large-scale analytics

As industries generate vast volumes of data from their operations, the application of data analytics opens doors to uncover valuable insights, patterns and trends. Through sophisticated algorithms and computational techniques, these analytics platforms disclose hidden information within the data, enabling industries to make informed decisions and optimize or even innovate their processes. Leading platforms like Apache Kafka, Apache Flink and Apache Spark play a pivotal role in this landscape, offering powerful tools for processing, analyzing and visualizing data. Apache Spark, renowned for its versatility, excels in processing both batch and real-time data, rendering it ideal for diverse applications ranging from data warehousing to machine learning tasks. Meanwhile, Apache Flink

specializes in real-time stream processing, enabling seamless analysis of data as it flows, thus facilitating immediate insights and responses. Complementing these platforms is Apache Kafka, an efficient data streaming solution that excels in handling vast data streams with fault tolerance and scalability. The synergistic interplay of Spark, Flink and Kafka marks a significant step forward in data utilization, enabling businesses to harness their information for strategic insights, informed decision-making and improved operational efficiency. According to CAP theorem, a trade-off between consistency and availability has to be made. In the case of combining Kafka and Flink, the emphasis is on consistency, i.e., it can happen that the system turns unavailable to enforce consistency.

6.1.3 PLC model

The **programmable logic controller** (PLC) model is a cornerstone of industrial automation, providing a realistic and deterministic framework for controlling and monitoring complex processes. PLCs are specialized digital computers designed to withstand harsh industrial environments and ensure consistent operation. They excel in real-time control applications where timing precision is crucial, such as manufacturing, power plants or process industry productions. Using a combination of inputs, outputs and logical programming, PLCs execute tasks according to predefined sequences, enabling machines to function safely, efficiently and predictably. While offering unparalleled determinism and reliability, the PLC model is not easy to scale to larger data. According to CAP theorem, this would require a trade-off of consistency versus availability. But both are needed to reach determinism.

6.1.4 Balance between determinism, safety and scalability

In the realm of industrial systems and applications a dynamic interplay between determinism and scalability has emerged as a pivotal challenge. Industries are confronted with the compelling need to keep stringent real-time requirements while simultaneously accommodating the escalating influx of data. The criticality of executing operations within prescribed deadlines persists, as it underpins safety, efficiency and reliability. Yet, as these systems expand in scope with larger data volumes, the feasibility of guaranteeing absolute determinism becomes progressively intricate. This prompts a profound exploration of the trade-off between maintaining deterministic behavior and accommodating the ever-growing demands for data scalability. Industries have to actively navigate this challenge to create systems that address the evolving needs.

The convergence of PLCs with the dynamic duo of Kafka and Flink presents a compelling proposition for industrial automation. While PLCs ensure precise control and real-time execution of operations, Kafka provides a resilient and scalable platform for streaming data. Flink, in turn, offers advanced stream processing capabilities, allowing the analysis of data as it flows from PLCs through Kafka. This synergy facilitates immediate insights and responses to changing conditions on the factory floor or in industrial processes. The integration empowers industries to not only maintain deterministic control, but also leverage real-time data streams for predictive maintenance, anomaly detection and optimization.

The earlier mentioned CAP theorem describes the dilemma: In a sufficiently unreliable system, one has to decide between availability and consistency. For safety purposes, we can derive that an important requirement is consistency. With inconsistent data, no safe decision is possible. The consequences are severe: It means that if one always expects to get a consistent answer, such a system cannot always guarantee to provide an answer in time if the system is sufficiently complex. On the other hand, there is a clear requirement for a system to react in real time to safety hazards. One way to marry the large-scale system with a real-time system is to allow the real-time system to locally deviate from the overall system state, based on the local knowledge. For instance, if the distributed analytics system does not give an answer about potential hazards in the movement line of an AMR, the local safety unit of the AMR must be able to conclude on its own with its locally available sensor data how long the AMR can be moved with a certain speed until the existing oversight is lost. For example, the AMR knows from the large-scale system that for at least 10 seconds no human being can cross the path, but after losing the connection (or the trust), it can keep up the operation for 10 seconds and then go into a safe, slow-motion state if it did not get a decision from the large-scale system until then. When within the 10 seconds the AMR gets trusted advice from the large-scale system, it can continue to move within the agreed parameters without interruption.

This hybrid edge architecture is shown in Figure 14. A local system that has to execute the safety functions must be able to make decisions based on a consistent view. When it trusts the global system and the global system indicates health, it can use the larger system model to make decisions. In the case that either the trust or the availability is not given (right side), the local system will only rely on local information to execute the safety function.

An example for such a local system is an AMR that relies on local sensors and runs with a speed that ensures that it can stop at any time. A global system could be comprised of

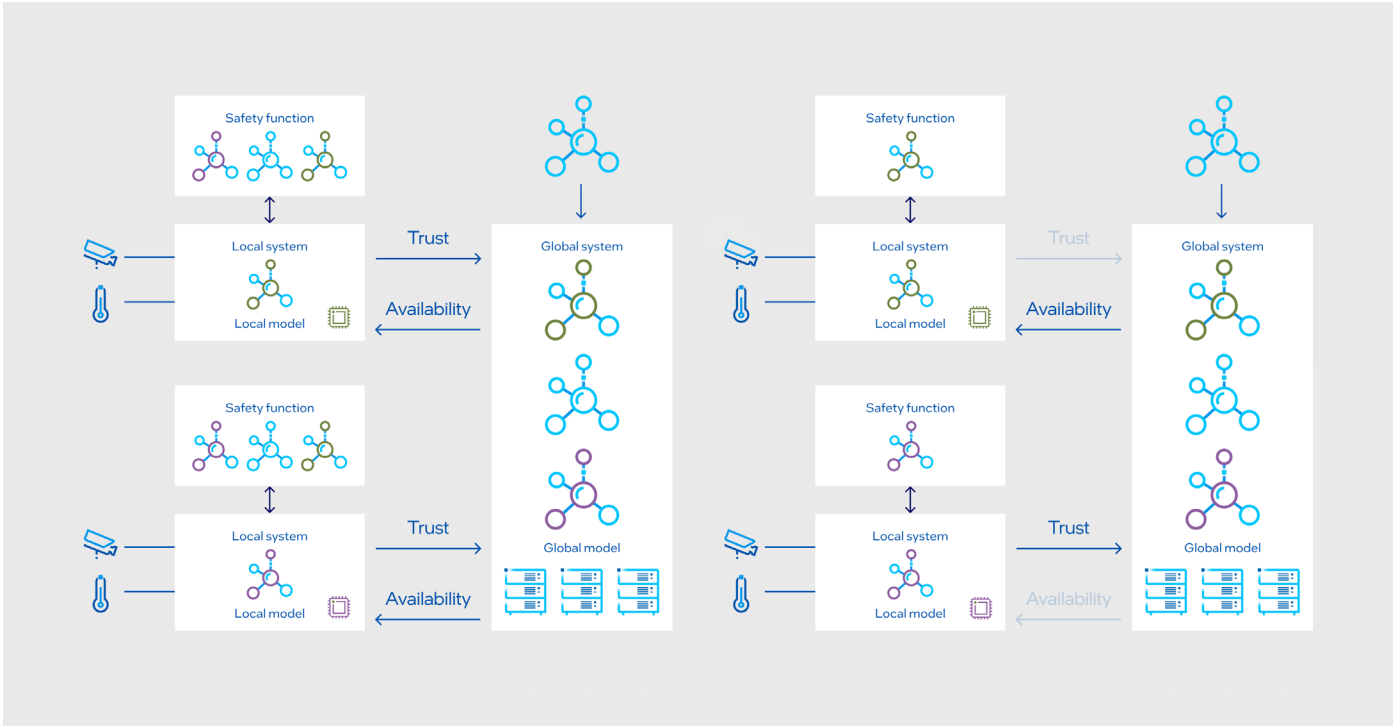

Figure 14. Hybrid edge architecture: Connection of local and global systems with trust and availability.

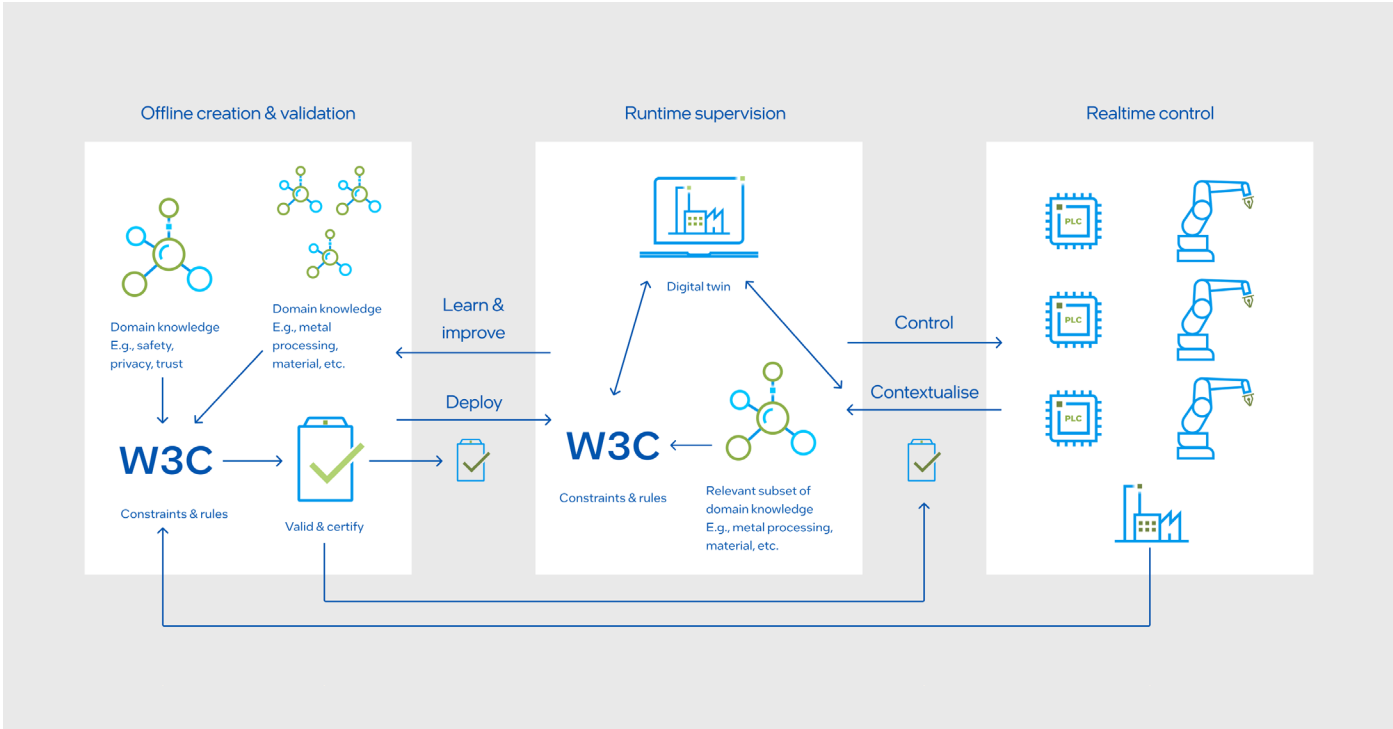

Figure 15. Data model is created and validated offline and is deployed on the digital twin. In parallel, the data contextualization from the machines must be defined for the model as well to achieve the needed trust.

cameras that are distributed over the whole area that can confirm that no person is close to the AMR. With this additional information, the AMR can act differently, for example, run faster for some time. When the information of the global system cannot be trusted, the AMR switches back to the local safety implementation.

It is important to have a consistent data model when synchronizing the data between the different systems. A digital twin provides a consistent data model of the physical environment that can be used for the constraints and rule analytics but also by the real-time system to synchronize the data. This model must not only be consistent, but also be able to inform whether the consistency of data can be assured. When the consistency of the data model is not given, the local systems will have to fall back to their local data models and logic to make their decisions.

6.1.5 Software architecture and implementation

The data model describes the use cases, knowledge and respective rules and constraints. However, it is important to emphasize that a data model is only useful with a defined underlying data contextualization. For instance, when the data model is expecting an internal temperature value of the machine but the temperature sensor is outside of the machine, there is an obvious mismatch between the context and the model. In order to cope with this fact, the contextualization cannot be done independently of the data model. This is shown in Figure 15.

In this section, we are focusing on the runtime supervision part in Figure 15. Putting all the ingredients together, and adding the requirement to use open-source frameworks, we propose the following architecture for the analytics system:

Data is collected from machines and other sources through a streaming data interface. All is connected to a low-latency message bus. Around the message bus, the semantic data analytics, object broker and alerts management are grouped. The semantic data analytics is configured and controlled by the knowledge graph and constraints management system.

This architecture contains the minimal ingredients for the low-latency analytics system. Of course, there are more details to consider for such a system, for example, authorization and authentication and time series data management, etc. Also, it is important to emphasize that this platform is only focusing on the **data core**, i.e., data integrity, validity, plausibility and execution of rules on data. Dashboarding and process integration are kept separate.

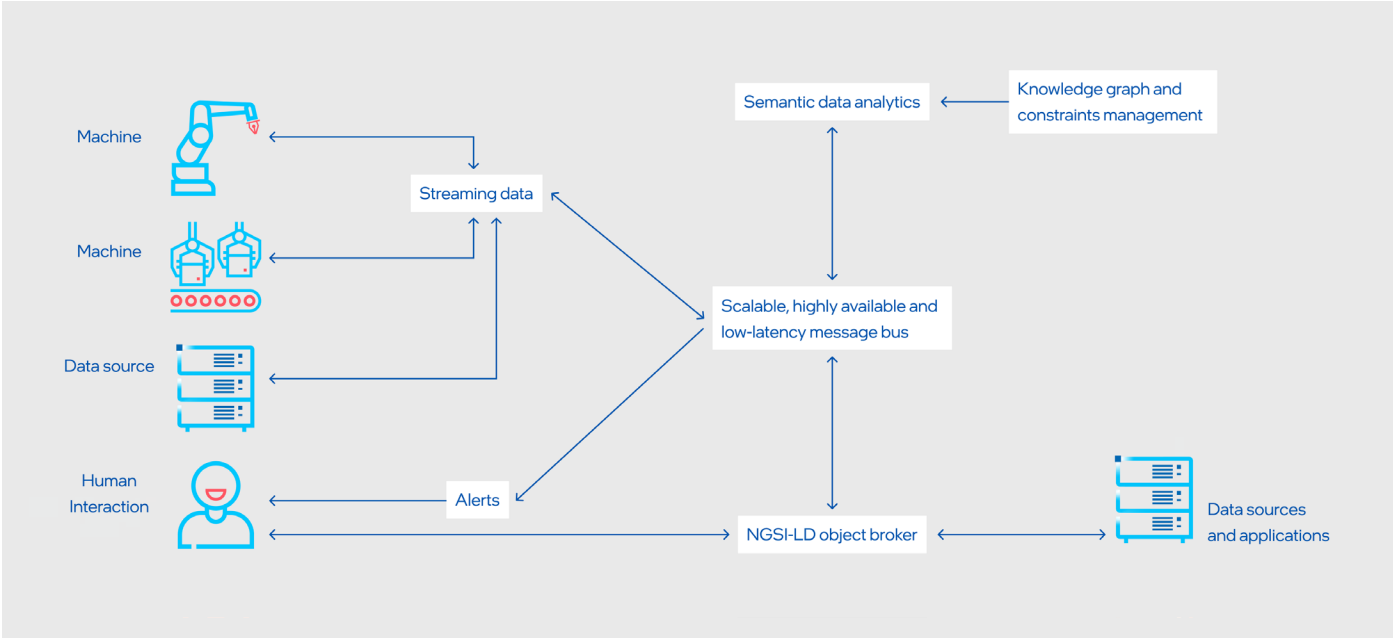

Figure 16. Minimal ingredients for the low-latency analytics part.

6.2 Hardware: Functional safety and integrity/cascading for large-scale safety analytics

For the previously described concept, some strict requirements must be fulfilled not only by the software, which is running on such a system, but also the underlying hardware and the peripherals that are involved need to provide capabilities accordingly. In the realm of data exchange and processing, two primary challenges exist. Firstly, ensuring the reliability and integrity of data during transfer, and secondly, ensuring reliable data processing while minimizing potential failures.

A common approach to secure data transfer is the use of a **black channel** as defined in the IEC 61508 standard[2], which includes error-detecting measures like a consecutive telegram counter, checksums and timer data. However, reliable data transfer is only beneficial if the data can be processed reliably on the corresponding computing unit before and afterward.

Powerful CPUs, designed for use in industrial computers or servers, can provide the necessary features to achieve the required **safety integrity level** (SIL) or an adequate safety level, combined with strong computing power and reliable data processing.

To increase system reliability, it is crucial to decrease the residual failure rate of undetected faults, provide efficient detection against transition and permanent faults and decrease the common cause failure rate. Additionally, the system should provide diagnostic capabilities for latent faults and fault accumulation.

Redundancy is a widely used technique in industrial systems for safety-critical data processing. This technique is also applicable on a higher level than that at which the machines are located. It can be achieved through parallel or serial redundancy, hardware-based or software-based redundancy or a combination of these.

Parallel redundancy involves identical components within a system performing the same task simultaneously, while **serial redundancy** involves components performing the same task sequentially. **Hardware-based redundancy** involves multiple instances of the same or different hardware components capable of performing the same task, while **software-based redundancy** involves implementing an algorithm in two different ways or with two different programming languages. In general, this leads to the five characteristics that come along with trustworthiness, which are security, privacy, safety, reliability and resilience.

Each of those characteristics has very specific requirements to an underlying hardware. Some of those characteristics are assigned to individual features that are provided only by some little particular components within the system. Others again must be considered by the entire system across all involved components.

Security is defined as the property a system must provide to protect itself from unintended or unauthorized access, change or destruction[19]. To fulfill this aspect, systems often provide mechanisms to make sure not to execute any code that has not been previously verified and checked to make sure it was not modified. This unbroken chain is called **root of trust** chain and must be guaranteed end-to-end from the first instruction that gets executed during boot of the system up to the applications that need to be launched by an operating system in the middle. Furthermore, this also needs to be fulfilled during the entire runtime of the system and not only at boot time.

Using, for example, UEFI secure boot and hardware technologies such as software guard extensions (SGXs), memory protection extensions (MPXs) and trusted execution technology (TXT) enhances operational security for a system in an industrial context.

Privacy is the right of an individual or group to control or influence what information related to them may be collected, processed and stored, and by whom, and to whom that information may be disclosed[20].

This very generally phrased request also applies in today's digital industrial landscape, where safeguarding operational integrity is paramount. As also mentioned in the security paragraph above, there is a suite of features in modern CPUs to bolster this integrity. Below it is explored how these features can enhance operational security in an industrial context.

Software guard extensions (SGXs)[21]

SGX is a technology designed to protect selected code and data areas from disclosure or modification. It creates protected areas, known as **enclaves**, which are safeguarded by the processor itself. These enclaves can be used to store sensitive operational data, such as system passwords or encryption keys, ensuring their security even if the rest of the system is compromised.

Memory protection extensions (MPXs)[22]

MPX offers a range of features to enhance software security by checking memory references at runtime. It helps prevent

buffer overflow and underflow, which can lead to security breaches. By preventing such attacks, MPX helps protect sensitive operational data from unauthorized access.

Trusted execution technology (TXT)[23]

TXT provides a hardware-based security solution designed to protect the integrity of the boot process. It helps prevent attacks that attempt to manipulate the boot process. By ensuring that only trusted software is loaded, TXT contributes to protecting operational integrity by preventing the installation of malware or spyware on the system.

Safety refers to a system's ability to function without posing an unacceptable risk of physical harm or health hazards to individuals, either directly or indirectly, due to property damage or environmental impact.

In the context of this document, the focus is not on traditional functional safety, which is often applied at the machine level. Instead, a holistic safety approach that encompasses the entire system is considered. However, the platform requirements are typically similar, if not largely identical.

In other words, it is crucial to ensure that the hardware hosting the software described above can handle transient and permanent faults, resist random and systematic failures and remain robust against common cross-failures.

Hardware-provided features, also known as **integrity features**[24], can contribute to achieving a high level of safety. Examples of such integrity features include CPU-core hardware lock step to achieve dual-channel opcode execution, power-on self-tests, on-demand diagnostic coverage, memory protection (such as error correction code [ECC] memory) and monitoring of clock, thermal and voltage conditions.

Reliability refers to a system or component's ability to perform its required functions under stated conditions for a specified period of time. In the context of this document, the hardware hosting the setup should be designed and built to operate reliably over its expected lifespan. This includes the ability to handle system workload in the expected amount but also within the expected environmental conditions. To ensure reliability, the hardware should be designed to withstand a variety of such conditions and loads, effectively addressing potential points of failure, preferably ahead of when potential issues happen. This is what is generally understood under the term **predictive maintenance capabilities**. Predictive maintenance utilizes data analysis, machine learning and predictive modeling to determine a system's likelihood of failure. This allows for timely maintenance and updates, which can help extend its operational life and maintain its performance levels.

Resilience is an emergent property of a system that allows it to avoid, absorb and manage dynamic adversarial conditions while completing assigned missions, and reconstitute operational capabilities after incidences. In the context of this document, the hardware should be designed with resilience in mind. This means it should have the ability to quickly recover from any disruptions or failures, ensuring minimal impact on the user's operations. This could be achieved through features such as redundant systems, fault tolerance mechanisms and robust error detection and especially correction capabilities. Furthermore, the system should be able to adapt to changing conditions and workloads, ensuring it can maintain its performance and functionality in a variety of scenarios.

Two hardware features modern CPUs offer are:

Rapid start technology[25]: This technology allows systems to quickly resume from deep sleep or cold boot, providing fast access to stored data and applications. This quick machine recovery time enhances system resilience.

Advanced vector extensions (AVXs)[26]: An **AVX** is a set of instructions for doing single instruction multiple data (SIMD) operations. SIMD allows one operation to be performed on multiple data points simultaneously, improving performance and resilience in the face of high computational demands.

7. Implementation case study

The IndustryFusion Foundation (IFF) has been founded by small and medium-sized enterprises (SMEs) in the metal processing and manufacturing industry. Their aim is to implement innovative use cases such as shared production and Equipment as a Service. The precondition for such use cases is to create a common shareable data model. Therefore, IFF decided to implement a semantic data model, as described in the sections above. A typical use case is to operate and optimize jointly cutting systems with attached air filtering and related workpieces and filter cartridges. One use case is to optimize the filter configuration based on the workpiece size and material. For this, the use case must first be modeled with JSON-LD and then the constraints and rules can be defined with SHACL. The material information can be provided by an ontology derived from the EN 10027-2 standard[27] and this builds up an RDF/OWL-based knowledge graph. In addition, state information and other details of the manufacturing process are defined in RDF/OWL. A simplified model is already used in the previous sections and shown in Figure 12. First of all, it must be ensured that the links between the entities are correct, for example, that a cutter is connected to a filter. This is a constraint that can be checked with SHACL as displayed in Figure 17.

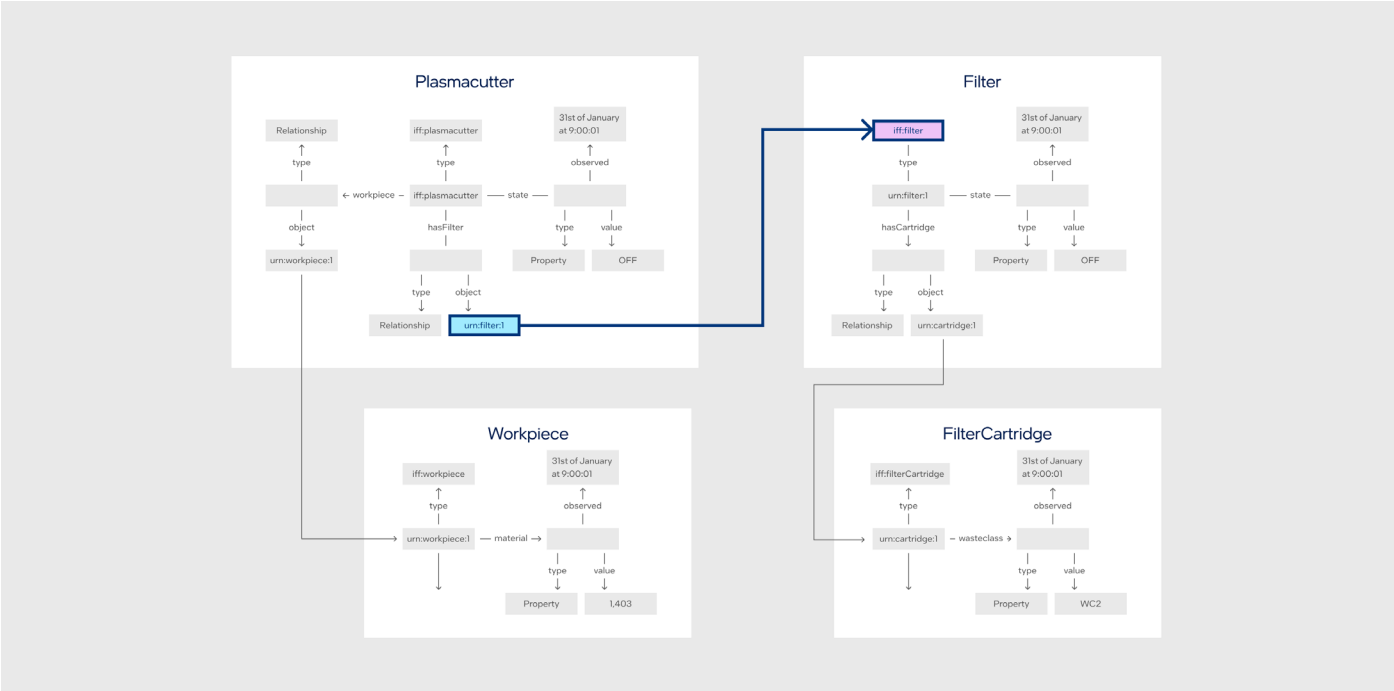

Figure 17. SHACL constraint to make sure that an object of type plasmacutter is linked with an object of type filter.

A typical constraint is to continuously compare the states of connected cutters and filters. For instance, when the cutter is in state iff:state_PROCESSING it must be ensured that the connected air filter is running with state iff:state_ON. This constraint can be checked with SHACL and is already shown in Figure 13. Finally, the entities and the knowledge graph need to be combined to understand how a material type is changing the waste class of the linked filter cartridge. This is shown in Figure 18.

This system, described in Figure 16, has been implemented as an open-source solution by the IFF[28]. The following components have been used:

- EMQX[29] as MQTT broker for streaming data.
- ScorpioBroker[30] as NGSI-LD object broker.
- Apache Flink[31] as semantic data analytics.
- Apache Kafka[32] as low latency message bus.
- Alerta[33] as alerts management system.
- Streaming SQL[34] for knowledge graph and constraint management running on Apache Flink.

In addition, authorization and authentication are managed by Keycloak[35] and the synchronization of the data object broker and the streaming system is implemented with Debezium[36]. Time series data is managed by TimescaleDB[37].

8. Conclusion and outlook

Discussions with companies from a wide range of industrial sectors show that the degree of automation will continue to increase. This means that autonomous, flexible systems and collaborative production will be used to cope with the recent challenges in industry, like changing market requirements, turbulence in supply chains, sustainability requirements and lack of skilled workers. The newly introduced flexibility in high-productivity automation systems also requires a change of fundamental paradigms for safety technology. In this paper, a novel paradigm for safety of machinery is shown that is able to react to the current context and environmental situation of a machine: operational safety intelligence. This safety concept requires advanced data and information management underneath, providing, for example, high data integrity. The paper shows how such information management can be established based on a Semantic Web concept and knowledge graphs with unified semantics that can handle data in large, distributed systems. By taking into account the data with its context, operational safety intelligence can understand risks from dynamic processes and can optimize operations using runtime risk management. Hence, it unites productivity, flexibility and safety and, therefore, tackles the challenges of autonomous and flexible systems regarding safety. The authors are convinced that it is only a matter of time until runtime risk management with operational safety intelligence becomes widespread in the industry.

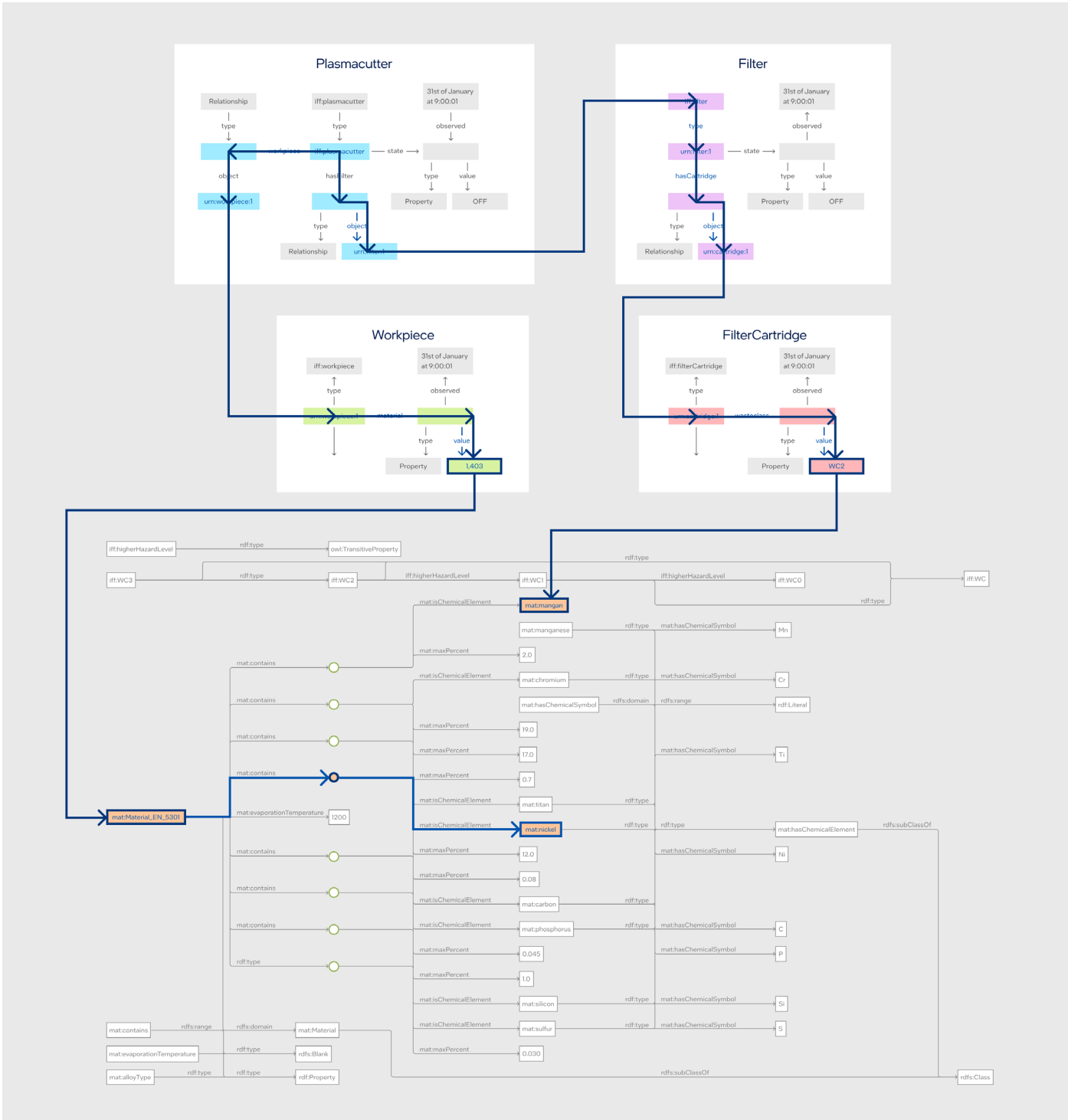

Figure 18. Combined entities and knowledge graph to determine the waste class change of the filter cartridge. This figure shows the complete overview; the detailed content can also be seen in Figure 12 and Figure 9.

9. References

[1] VDE-AR-E 2842-61-1 – "Development and trustworthiness of autonomous/cognitive systems – Part 61-1: Terms and concepts," 2021.

[2] IEC 61508 – "Functional safety of electrical/electronic/programmable electronic safety-related systems," 2010.

[3] Regulation (EU) 2023/1230 of the European parliament and of the council of 14 June 2023 on machinery, 2023.

[4] Michael Pfeifer, Detlev Richter, Dimitri Harder, William Motsch, Nigora Gafur, Henning Gösling, Alexander David, Manuel Heid, Oliver Thomas, Martin Ruskowski, Achim Wagner. "Safe and efficient production through agent systems," ATP Article, 2022.

[5] Michael Pfeifer, Dimitri Harder, Detlev Richter, Klaus Reichenberger, Bernd Neuschwander, Matthias Schweiker, Alexander David, Pascal Rübel, Manuel Heid, Achim Wagner, William Motsch, Martin Ruskowski. Smart Safety—Das Konzept Knowledge Graph zur Umsetzung von Safety in Digitalen Zwillingen, Smart Factory KL Whitepaper SF-3.4, 2022.

[6] Anto Budiardjo, Jon Geater, Frederick Hirsch, Michael Pfeifer, Detlev Richter. Assuring Trustworthiness in Dynamic Systems Using Digital Twins and Trust Vectors, Digital Twin Consortium Foundational Paper, 2022.

[7] ISO 12100 – "Safety of machinery – General principles for design – Risk assessment and risk reduction," 2010.

[8] Tim Berners-Lee, James Hendler and Ora Lassila. "The Semantic Web: A New Form of Web Content that is Meaningful to Computers will Unleash a Revolution of New Possibilities," Linking the World's Information: Essays on Tim Berners-Lee's Invention of the World Wide Web (1st ed.), Association for Computing Machinery, 2023, https://doi.org/10.1145/3591366.3591376.

[9] W3C – Semantic Web – https://www.w3.org/2001/sw/.

[10] W3C – RDF 1.1 Concepts and Abstract Syntax – https://www.w3.org/TR/rdf11-concepts/.

[11] Eric Prud'hommeaux, Gavin Carothers. RDF 1.1 Turtle, W3C Recommendation, 2014, https://www.w3.org/TR/turtle/.

[12] Boris Motik, Peter F. Patel-Schneider, Bijan Parsia. OWL 2 Web Ontology Language: Structural Specification and Functional-Style Syntax (Second Edition), W3C Recommendation, 2012, http://www.w3.org/TR/2012/REC-owl2-syntax-20121211/. Latest version available at http://www.w3.org/TR/owl2-syntax/.

[13] Boris Motik, Bernardo Cuenca Grau, Ian Horrocks, Zhe Wu, Achille Fokoue, Carsten Lutz. OWL 2 Web Ontology Language: Profiles (Second Edition), W3C Recommendation, 2012, http://www.w3.org/TR/2012/REC-owl2-profiles-20121211/. Latest version available at http://www.w3.org/TR/owl2-profiles/.

[14] Gregg Kellogg, Pierre-Antoine Champin, Dave Longley. JSON-LD 1.1, W3C Working Draft, 2019, https://www.w3.org/TR/json-ld11/.

[15] ETSI GS CIM 009 V1.7.1 (2023-06) – Context Information Management (CIM), NGSI-LD API.

[16] Holger Knublauch, Dimitris Kontokostas. Shapes Constraint Language (SHACL), W3C Candidate Recommendation, 2017, https://www.w3.org/TR/shacl/.

[17] Oyeniran et al. "A comprehensive review of leveraging cloud-native technologies for scalability and resilience in software development," International Journal of Science and Research Archive, 2024.

[18] Eric Brewer. Towards robust distributed systems, PODC, 2000.

[19] Committee on National Security Systems (CNSS) Glossary – information security.

[20] IAPP – https://iapp.org/about/what-is-privacy/.

[21] Intel – Intel Software Guard Extensions (Intel SGX) – https://www.intel.com/content/www/us/en/products/docs/accelerator-engines/software-guard-extensions.html.

[22] Intel – Support for Intel Memory Protection Extensions (Intel MPX) Technology – https://www.intel.com/content/www/us/en/support/articles/000059823/processors.html.

[23] Intel – Intel Trusted Execution Technology (TXT) for Client Platforms – https://www.intel.com/content/www/us/en/developer/articles/tool/intel-trusted-execution-technology.html.

[24] Riccardo Locatelli, Elisa Spanò, Laura Spinella. Intel Silicon Integrity Technology Tech Brief, Intel whitepaper, 2024, https://www.intel.la/content/www/us/en/content-details/827568/intel-silicon-integrity-technology-tech-brief.html.

[25] Intel – General Q&A for Intel Rapid Start Technology – https://www.intel.com/content/www/us/en/support/articles/000024078/technologies/intel-rapid-storage-technology-intel-rst.html.

[26] Intel – Intel Advanced Vector Extensions 512 (Intel AVX-512) Overview – https://www.intel.com/content/www/us/en/architecture-and-technology/avx-512-overview.html.

[27] EN 10027-2 – "Designation systems for steels – Part 2: Numerical system," 1992.

[28] IndustryFusion Foundation – Process Data Twin – https://github.com/IndustryFusion/DigitalTwin.

[29] EMQX – MQTT Broker – https://www.emqx.io/.

[30] ScorpioBroker – NGSI-LD Compatible Context Broker – https://github.com/ScorpioBroker/ScorpioBroker.

[31] Apache Flink – Stateful Computations over Data Streams – https://flink.apache.org/.

[32] Apache Kafka – Distributed Even Streaming Platform – https://kafka.apache.org/.

[33] Alerta – Alerta Management Platform – https://alerta.io/.

[34] E. Begoli, T. Akidau, F. Hueske, J. Hyde, K. Knight, K. Knowles. "One SQL to rule them all — an efficient and syntactically idiomatic approach to management of streams and tables," Proceedings of the 2019 International Conference on Management of Data, 2019.

[35] Keycloak – https://www.keycloak.org/.

[36] Debezium – https://debezium.io/.

[37] TimescaleDB – https://www.timescale.com/.

intel + TÜV SÜD

1124/AK/CAT/PDF ♻Please Recycle 362330-001EN