Journal of Intelligent Manufacturing
https://doi.org/10.1007/s10845-023-02319-6

Towards a knowledge graph framework for ad hoc analysis in manufacturing

Bart Meyers[1] · Hans Vangheluwe[2] · Pieter Lietaert[1] · Geert Vanderhulst[1] · Johan Van Noten[1] · Michel Schaffers[1] · Davy Maes[1] · Klaas Gadeyne[1]

Received: 15 May 2023 / Accepted: 22 December 2023

Abstract
The development of artificial intelligence models for data driven decision making has a lot of potential for the manufacturing sector. Nevertheless, applications in industry are currently limited to the actionable insights one can discover from the available data and knowledge of a manufacturing system. We call the process to obtain such insights "ad hoc analysis". Ad hoc analysis at system level is very complex in an industrial setting due to the inherent heterogeneity of data and existence of data silos, the lack of information and knowledge formalization, and the inability to meaningfully and efficiently reason about the data, information and knowledge. In this paper, we provide and outline a framework for ad hoc analysis in manufacturing based on knowledge graphs and influenced by the metamodelling paradigm. We derive its requirements and key elements from an analysis of several industry application cases. We show how manufacturing data, information and knowledge can be combined and made actionable using this framework. The framework supports workflows and tools for the data consumer (i.e., data scientist), and for the knowledge engineer. Furthermore, we show how the framework is integrated with existing data sources. Then, we discuss how we applied the framework to several application cases. We discuss how the framework contributes when applied, and what challenges still remain.

Keywords Knowledge graphs · Data analysis · Data management · Knowledge management

Introduction

In Industry 4.0 data play a crucial role in companies. The promise is that actionable insights in the data influence the manufacturing system, offering previously untapped business value, often through but not limited to the application of artificial intelligence (AI) techniques. AI applications have potential in cost reduction, self monitoring, self configuration and self learning systems, insights and forecasting, decision support, supporting new business models such as servitization.

Nevertheless, the success of generating business value out of these data is typically limited to specific applications such as predicting tool wear or quality control using image recognition. These applications are limited in scope, concentrating on a single or limited machine, product type or manufacturing operation. To fully realize the goals of Industry 4.0, manu-

✉ Bart Meyers
bart.meyers@flandersmake.be

Hans Vangheluwe
hans.vangheluwe@uantwerpen.be

Pieter Lietaert
pieter.lietaert@flandersmake.be

Geert Vanderhulst
geert.vanderhulst@flandersmake.be

Johan Van Noten
johan.vannoten@flandersmake.be

Michel Schaffers
michel.schaffers@flandersmake.be

Davy Maes
davy.maes@flandersmake.be

Klaas Gadeyne
klaas.gadeyne@flandersmake.be

[1] CodesignS, Flanders Make, Oude Diestersebaan 133, Lommel 3920, Belgium

[2] Ansymo-CosysLab, Flanders Make at University of Antwerp, Middelheimlaan 1, Antwerpen 2020, Belgium

Published online: 29 February 2024

facturing companies need to be able to create system-level business intelligence applications that take into consideration the complete manufacturing system Wang et al. (2018); Zheng et al. (2022).

While a manufacturing company's amount of data grows, the ability to create more complex business intelligence applications (by means of a process we call "ad hoc analysis" in this paper) is heavily impeded by the inability to find the necessary data insights in a manufacturing system CrowdFlower (2016). The key barriers for ad hoc analysis that are identified in the state-of-the-art are heterogeneity of data Nagorny et al. (2017); Wang et al. (2018), lack of information and knowledge formalization Shilov (2020); Zheng et al. (2022), and lack of inductive reasoning capabilities Wuest et al. (2016); Wang et al. (2018); Zheng et al. (2022). No framework exists that addresses all of these challenges, and that provides concrete workflows and software tools that can be applied in the manufacturing domain.

This paper presents 24 system level requirements for a framework for ad hoc analysis as a result of analysing several ad hoc analysis application cases. We introduce a reference architecture of a knowledge graph based framework for ad hoc analysis, with concrete workflows and tools. We show how the framework provides a solution to the three key barriers and highlight key contributions on top of the state-of-the-art. We validated the framework on three application cases.

The paper starts with related work in Section "Related work". Section "Ad hoc analysis" analyses application cases and results in requirements and a reference architecture of a framework for ad hoc analysis. We define a knowledge graph based framework that adheres to these requirements in Section "Ontology-based knowledge graph framework". We define this framework's components and workflows. We show how we applied the framework to three application cases in Section "Validation on application cases". We discuss its value and open challenges in Section "Discussion" and conclude with an outline for future work in Section "Conclusion and future work".

Related work

Knowledge graphs for manufacturing receive increasing interest. The majority of the work can be classified a knowledge fusion Buchgeher et al. (2021). Knowledge graphs are used in specific application domains such as assembly Kalaycı et al. (2020), quality monitoring Kwon et al. (2020) for specific applications such as welding Svetashova et al. (2020) or drilling Weckx et al. (2022), root cause analysis Martinez-Gil et al. (2022) or in the context of Industry 4.0 Grangel-González (2019); Grangel-González et al. (2016). Other work focuses on education Kumar et al. (2022). In most of these cases, knowledge graphs are used for data integration and access, often connecting different data silos. One popular technique to accomplish this by providing an abstraction layer on the data, is ontology based data access (OBDA) Calvanese et al. (2016) or virtual knowledge graphs Xiao et al. (2019). This has already been applied in a number of practical use cases Kalaycı et al. (2020); Kharlamov et al. (2017). Another important reason for using knowledge graphs in manufacturing is to capture and reuse manufacturing knowledge, often obtained from unstructured data sources Kumar et al. (2022).

A lot of work is being done representing manufacturing knowledge on a more general level, through the development of domain ontologies. This has been the case for robotic path planning Gayathri and Uma (2018), welding Saha et al. (2019), quality control Leitão et al. (2012), anomaly detection Steenwinckel et al. (2018), system design Arista et al. (2023) and generally in the context of Industry 4.0 Sampath Kumar et al. (2019), such as for decision making Kourtis et al. (2019), context modelling Giustozzi et al. (2018), or interoperability Ameri et al. (2022). Formal, high-level ontologies can then be used to connect these domain ontologies, thereby easing the process of connecting different data silos together. For example, in Hildebrandt et al. (2020), ontologies based on standards are used for interoperability between systems in manufacturing. Furthermore, there are ongoing efforts creating formal ontologies for industry, such as IOF Kulvatunyou et al. (2018) or the Ontocommons[1] project.

Using an ontology-based knowledge graph in manufacturing, automatic deductive and inductive reasoning can also be applied to extract additional knowledge from the knowledge graph Guimarães et al. (2022); Munch et al. (2019). Knowledge graphs can be deployed to handle (semi-)live or streaming data Kharlamov et al. (2016), so that they can support the construction of a data-based applications such as digital twin models for manufacturing data for the alignment of streaming data Ringsquandl et al. (2017), condition-based monitoring Singh et al. (2021), and knowledge graph generation Banerjee et al. (2017). Furthermore, if the knowledge graph is amenable to reasoning, it can also give rise to what is referred to as a cognitive digital twin Zheng et al. (2022); Lu et al. (2021) that offers intelligent decision support, for example for systems engineering Jinzhi et al. (2022); Arista et al. (2023). Grevenitis et al. (2019) uses a triple store as landing zone for data, enriched with knowledge, and focuses on data ingestion. It is applied in the context of zero-defect manufacturing. For zero-defect manufacturing, an ontology is proposed in Psarommatis et al. (2023) to semantically integrate multiple data sources.

In order to use knowledge graphs in practice, tooling is important. In particular, visualizing and designing ontologies

[1] https://ontocommons.eu/. Last visited: 15/05/2023.

Dudáš et al. (2018); Lohmann et al. (2016) is an important part of creating and exploring the knowledge graph. Research has been done on graph based representation of ontologies, with tools such as VOWL Lohmann et al. (2016), ViSMod García-Peñalvo et al. (2014) KC-Viz Motta et al. (2011), or GLOW Hop et al. (2012), or in an UML-type diagram as a UML profile Djuric et al. (2005), by defining a mapping to classes and objects Bartalos and Bielikova (2007), or vice versa Chávez-Feria et al. (2022).

In conclusion, while several applications, models and tools exist for ad hoc analysis, the state-of the art presents no framework to fully address the challenge of ad hoc analysis, describing its requirements and concept, its implementation with necessary software tools, and its validation in the manufacturing industry.

Ad hoc analysis

In this section we describe ad hoc analysis and its role in creating business intelligence applications, its challenges from industrial use cases, and the requirements for an ad hoc analysis framework that results from these challenges.

The ad hoc analysis challenge in manufacturing

The generic high level workflow to create business intelligence applications is shown in Fig. 1, with the operational technology (factory) on the left, followed by data acquisition, i.e., ingestion, cleaning, and curation by a data engineer who manages the data lake. This data lake is used by data scientists and subject matter experts to find interesting insights in data with the goal of creating business value, an activity we call "ad hoc analysis". Once these insights have been found, confirmed, modelled, etc., a specific business application can be created by an application developer. We purposely define this process to have an iterative life cycle, compatible with the emerging vision on life cycle management of business applications, such as ModelOps Hummer et al. (2019).

The main challenge for creating business applications is "ad hoc analysis" (highlighted in Fig. 1), namely finding the actionable insights in a manufacturing system that lead to purposeful business applications. In ad hoc analysis, there is a gap between the existing data and the consumer of this data (the data scientist). Due to this gap, data scientists spend up to 80

Requirements for a framework for ad hoc analysis

Figure 2 shows the overall research methodology of the paper. We analyzed the ad hoc analysis needs of a number of industry-level application cases with different problem domains as shown in Table 1. Section "Validation on application cases" explains the first three in more detail.

Table 1 Industry-level application cases that were analysed for requirements

Sector	Ad hoc analysis goal
Manufacturing of air compressor core elements (see Meyers et al. (2022))	Predictive and prescriptive analytics for zero defect manufacturing
Manufacturing of textile machines	Flexible dashboard creation for the monitoring of costs such as material consumption
Material bonding and debonding	Predictive and prescriptive analytics with uncertainties for an experimentation lab
Additive manufacturing	Development of real-time corrective algorithms at run time
Assembly of mechatronic systems	Flexible root cause analysis with uncertainties for pFMEA (process failure modes and effects analysis)
Manufacturer of communication products	Test coverage analysis in embedded software development
Drivetrain design and testing	Experimental design, and reuse of experiments
Horticulture robotics	Anomaly detection and diagnostics of products

For each of these cases, we analyzed the common needs of these application cases together with the state-of-the-art and concluded that all cases share three main challenges:

- **Main challenge 1: Heterogeneity of data** Nagorny et al. (2017); Wang et al. (2018); Adolphs et al. (2015). A typical manufacturing system has a plethora of different data sources (silos, databases) and formats (tabular, time series, blob, etc.) for various aspects of the manufacturing system (product design, manufacturing execution, quality assurance, sensor readings, etc.).
  Acquiring the right data is often a highly iterative process with data engineers, especially for manufacturing system-wide analysis, e.g., linking product quality to production circumstances.
- **Main challenge 2: Lack of information and knowledge formalization** Shilov (2020); Zheng et al. (2022). The cognitive gap needs to be bridged, firstly between data scientists and domain experts, and secondly between the problem domain and technical solution domain. Therefore, there is a need to formalize knowledge.
  In an enterprise, information and knowledge is typically spread across different knowledge sources (subject matter experts, data scientists, machine manuals,

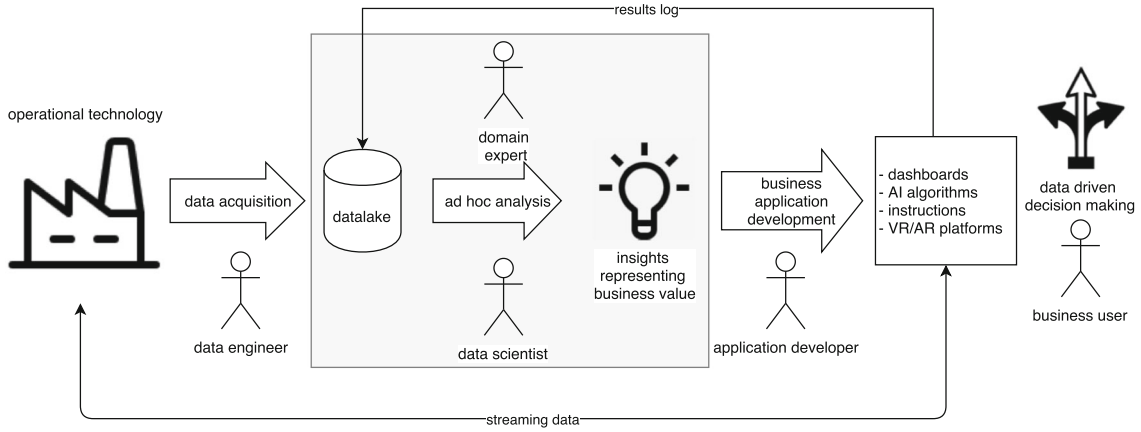

Fig. 1 High level workflow for creating business intelligence applications

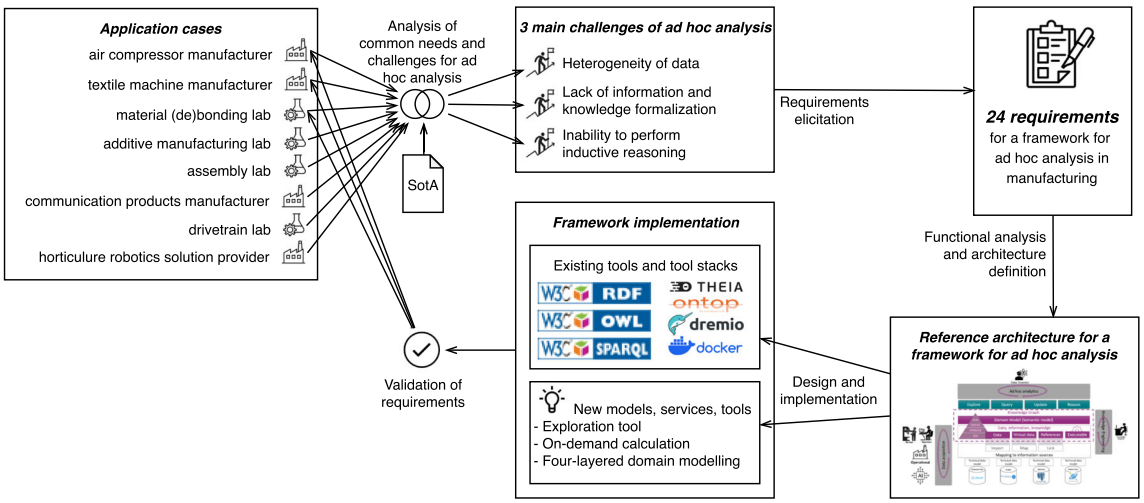

Fig. 2 Research methodology of this paper

research papers, AI/simulation models, etc.). Typically, this knowledge remains implicit or tacit, or not readily available to the data scientist. This leads to lack of understanding, data analysis errors (wrong assumptions or conclusions), or inefficiencies (long iterations and repetition of interpretation of the same data) in ad hoc analysis. The challenge is making such knowledge explicit and formal, and relating it to the data.

- **Main challenge 3: Inability to meaningfully and efficiently perform inductive reasoning** Wuest et al. (2016); Wang et al. (2018); Zheng et al. (2022). Ad hoc analysis requires flexible experimentation with data and knowledge. There is a need for efficiently carrying out statistics, logical inferencing, and AI workflows, including correlation analysis, feature modelling, training, and validation. Moreover, discovered insights need to be findable and accessible in an enterprise for future use, and explicitly linked to the concepts they are related to. Furthermore, following the ModelOps paradigm Hummer et al. (2019), there is a clear need for monitoring deployed AI models, that need to be optimized or retrained when necessary.

Based on analysis and user interviews of each of the application cases, we refined the three challenges and elicited 24 system level requirements in 5 categories for a framework for ad hoc analysis in Table 2. In the remainder of the paper, we refer to these requirements and explain how our framework fulfills them.

Ontology-based knowledge graph framework

This section proposes a framework and shows how it implements the requirements of Table 2. While we are able to build on existing tools and tool stacks, technical contributions are

Table 2 System level requirements for a framework for ad hoc analysis

Metadata	
REQ1	**Abstraction.** The heterogeneous knowledge about the problem domain as well as interpretation and semantics of data (answering "how" and "why" questions) must be explicit and linked to information and data
REQ2	**Formalization.** Data, information and knowledge must be formalized, well-formed and consistent
REQ3	**Explainability.** The information for explainability of data (categorization and typing, answering "who", "what", "where", "how many" and "when" questions) must be explicit and linked to data
REQ4	**Traceability.** Data must be traceable at system level, so that the desired analysis can be performed. This can be product, operator, resource traceability, depending on the problem domain that needs to be tackled
REQ5	**Visualization.** It must be possible to visualise the data structure in a graphical way
REQ6	**Schema discovery.** Extracting the structure of the data must be automated
Data management	
REQ7	**Single point of access.** A single point of access must be available to all data, information and knowledge at enterprise level
REQ8	**Data flow.** The framework must support a workflow for ingesting new or existing data, information and knowledge
REQ9	**Heterogeneous data.** Heterogeneous data (i.e., raw observations) from different data silos must be explicitly linked
REQ10	**Live data.** The framework must provide support for perpetually changing data, information and knowledge
REQ11	**Data operator.** A user must be able to create new data from existing data
Data exploitation	
REQ12	**Exploration.** A user must be able to incrementally explore and understand what data, information and knowledge exist
REQ13	**Querying.** A user must be able to formulate queries to retrieve data, information and knowledge
Workflow support	
REQ14	**Incremental workflow.** The user must be able to interact according to an incremental and flexible workflow
REQ15	**Reasoning.** The framework must be able to automatically reason about data, information and knowledge
REQ16	**Automation.** Repetitive work must be automated. The workflows must be managed and executed workflows must be preserved
Framework and tooling	
REQ17	**Multi user.** The framework must support interaction of multiple users with the same enterprise level data, information and knowledge
REQ18	**Performance.** The performance overhead for querying data, information and knowledge must be acceptable compared to state-of-the-practice querying
REQ19	**Openness.** The framework must support the definition of user-specific tools for the interaction with data, information and knowledge
REQ20	**Interoperability.** The querying service must be well integrated with existing software packages for data science
REQ21	**Extensibility.** The framework must support an extension mechanism for organization-independent problem-specific knowledge
REQ22	**Adaptability.** The framework must support gradual adoption, by ensuring that its value (i.e., the development of an AI application) can be achieved without having to setup a complete organization wide framework first
REQ23	**Flexible deployment.** The framework must support a flexible deployment allowing extension with new tools, services and data sources tailored to an enterprise
REQ24	**Cloud support.** The tool must be usable from any place using a browser

necessary in our framework. The main technical contributions presented in this paper to achieve this are:

- a framework with a (to our knowledge) functionally complete set of data, information, knowledge access methods (**REQ1**, **REQ7**, **REQ9**, **REQ10**, **REQ11**), and interaction methods (**REQ5**, **REQ12**, **REQ13**, including workflows (**REQ8**, **REQ14**) and deployment architecture (**REQ23**);
- a knowledge graph structure for modelling data, information and knowledge that suits the needs of the manufacturing domain that heavily relies on domain models (**REQ1**, **REQ2**, **REQ3**, **REQ4**);
- an exploration tool to support efficient finding and understanding of data, information and knowledge (**REQ5**, **REQ6**).
- a method for on demand calculation to improve reasoning capabilities of the framework (**REQ11**, **REQ15**, **REQ16**).

The remainder of this section explains these contributions in more detail and positions them in the overall framework.

Combining the ontology paradigm and the metamodelling paradigm

Our framework for ad hoc analysis needs to have a foundational approach to model data, information and knowledge, and their relationships (**REQ2**). Our work is founded on a

combination of the ontology paradigm and the metamodelling paradigm. In databases, this need is acknowledged and partially addressed Razniewski and Nutt (2014). In data modelling, the metamodelling paradigm (e.g., database schemas) is used, supporting explicit typing (needed for **REQ6**). In order to model knowledge and classification[2] (**REQ3**) and link multiple types of data (**REQ1**, **REQ4**), and give specific support for reasoning (**REQ15**), the use of ontologies is the preferred paradigm Shilov (2020); Zheng et al. (2022).

For the ontology- and metamodel-based knowledge graph implementation in our framework, we have chosen to use the semantic web tool stack, which follows the ontology paradigm, on which we impose explicit restrictions so that we draw from the benefits of the metamodelling paradigm.

Outline of the architecture

The architecture for the knowledge graph based framework is shown in Fig. 3. It positions a knowledge graph (center) as the single point of access to all data, information and knowledge of the manufacturing system (**REQ7**) to the data scientist, for ad hoc analysis (top). Ad hoc analysis requires four types of interaction with the knowledge graph, namely explore, query, update and reason (see Sect. 4.4).

Based on the knowledge pyramid Rowley (2007) (center of Fig. 3), a knowledge graph contains:

- **data**: the observed data, created by software, sensors, operators. In our framework, this can be raw or curated data;
- **information**: the structure and context of the data, such as metadata;
- **knowledge**: the know-how of the manufacturing system. This can be based on data (e.g., the definition of a predictive machine learning algorithm) or not involve data at all (e.g., the description of a suspected correlation between two unmeasurable parameters in the manufacturing system).

In this paper we will use the term *knowledge graph content* as umbrella term for all data, information, knowledge (and wisdom) in a knowledge graph, and a *knowledge graph element* as a single record of data, information or knowledge. In our architecture, knowledge graph elements are typed by domain models (see Section "Domain model").

Manufacturing data is typically heterogeneous, so our framework supports several ways to access data through the knowledge graph (bottom of Fig. 3, see Sect. 4.3.2 for more details) (**REQ9**). As illustration, some examples of information sources are shown. Data in the knowledge graph can be materialized data, virtual data, references to data or on demand calculation of data (**REQ16**). The framework defines respective types of links to the manufacturing system's data: importing into, mapping onto or linking data to the knowledge graph. Next to accessing existing data, the knowledge graph typically contains standalone data, information and knowledge.

The data acquisition workflow (left of Fig. 3) represents the existing workflow to ingest data, which may be performed by workers, operators, operational technology or deployed digital twins. This includes the conversion of streaming data to data lakes. This workflow is executed perpetually, independent of the knowledge graph framework. This paper does not contribute to the activity of data acquisition.

The knowledge engineering workflow (right of Fig. 3) defines a workflow for building and extending a knowledge graph and its relation to information sources (see Sect. 4.3) (**REQ8**). This workflow is carried out by the knowledge engineer role, who takes ownership of the knowledge graph. Knowledge engineering requires a system level view on the manufacturing system, modelling skills, domain knowledge and data knowledge.

Knowledge engineering

Figure 4 shows the high level workflow and interaction of the knowledge engineer with the knowledge graph. In order to provide value as early as possible (**REQ22**), the knowledge engineering workflow starts with a business case or viewpoint that is flagged to provide business value. In a traditional data engineering context, this case directly leads to data engineering actions as shown at the bottom of Fig. 4. So, knowledge engineering introduces an additional cost. An organisation benefits once the same knowledge graph content is re-used many times. Moreover, the use of a knowledge graph avoids making similar mistakes multiple times, thus improving quality in the ad hoc analysis process. Also note that part of data curation and data modelling is now transferred from the data scientist to the knowledge engineer.

The high level workflow is highly collaborative and iterative, and consists of the following activities:

1. Understand the business value together with the business user, in order to understand the goal and the value of the case.
2. Understand the problem domain with the domain expert. This involves exploring several knowledge sources, such as process engineers, machining tool providers, quality engineers, product designers, etc.
3. Model the domain concepts in the problem domain by creating or extending a domain model with ontologi-

[2] In metamodelling, the difference between typing and classifying is defined by a linguistic and ontological metamodel Kühne (2006). While in ontologies, the linguistic metamodel is left unrestricted, in metamodelling, the ontological metamodel is traditionally overlooked.

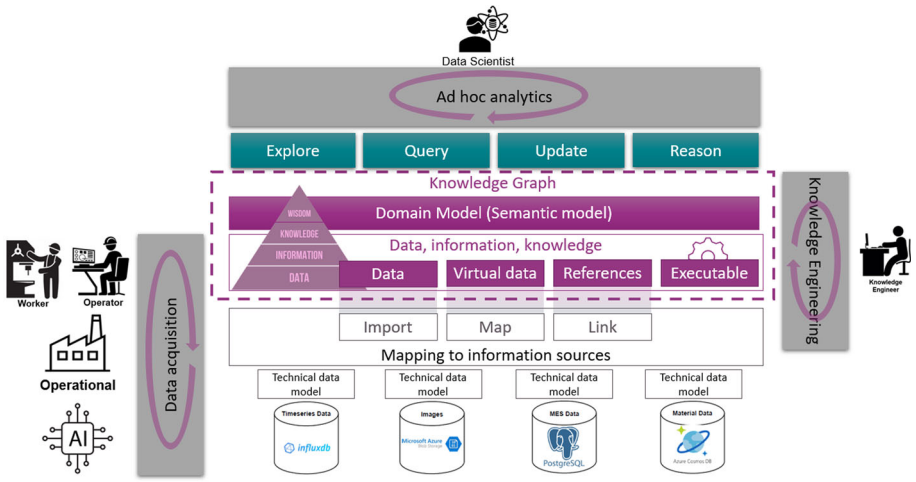

Fig. 3 Knowledge graph reference architecture

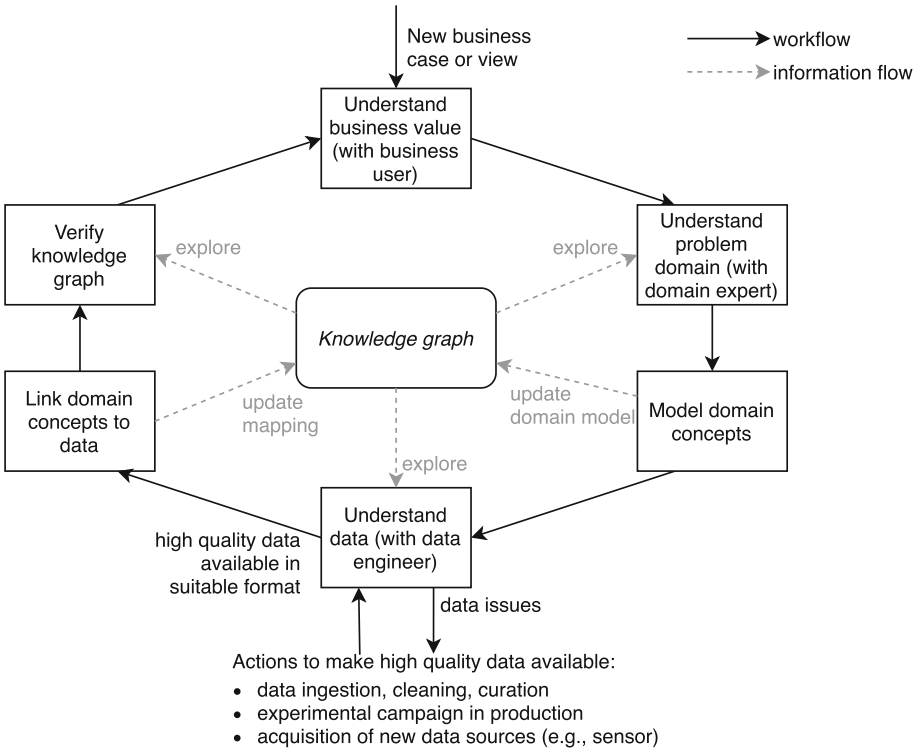

Fig. 4 Knowledge engineering workflow and interaction with the knowledge graph

cal concepts in the knowledge graph, explained more in detail in Section "Domain model".

4. Understand what data exists that is relevant to the case, and understand this data, together with the data engineer. There may be several issues with the data leading to data engineering actions, such as ingesting, cleaning or curating new data, carrying out an experimental campaign to acquire new data or acquisition of new sensors.
5. Map domain concepts to data, once high quality data is available and well understood by the knowledge engineer (see Sect. 4.3.2).
6. Verify the knowledge graph is verified, if necessary together with other stakeholders, to verify whether its goal is met, i.e., whether the data, information and knowledge for the business case is available in the knowledge graph.

Throughout the workflow, the knowledge engineer interacts with the knowledge graph to explore its concepts and data (**REQ6**, **REQ12**) (more detailed in Section 4.4), create new concepts in a domain model (**REQ8**) (see Section "Domain model") and map concepts to data (**REQ2**, **REQ3**, **REQ4**) (see Sect. 4.3.2).

Domain model

The framework uses domain models to improve support for defining system level semantics and accessibility of the knowledge graph content (**REQ1**), formalization, consistency and well-formedness of data, information and knowledge (**REQ2**), explainability (**REQ3**), and traceability (**REQ4**).

The domain model provides structure to all data, information and knowledge as a single point of access (**REQ7**) in the knowledge graph in terms of the *problem domain*, whereas available information typically is defined in a technical *solution domain*. Figure 5 shows by example how we position domain models in a four layered architecture. Domain models are implemented as ontologies as defined in the semantic web tool stack, but we provide several extensions that originate from the metamodelling paradigm that improve the use of ontologies in manufacturing:

A generic, minimal **upper ontology** based on the Resource Description Framework (RDF) The W3C and RDF Working Group (2014), RDF Schema (RDFS) The W3C and RDF Working Group (2014) and the Web Ontology Language (OWL2) The W3C and OWL Working Group (2012) that defines a linguistic structure for domain models (first layer in Fig. 5). The use of open standards ensures openness (**REQ19**). In order to adequately use the metamodelling paradigm, we defined our own minimal Metamodelling (MM) upper ontology. We use the following idioms in the upper ontology (some of which are shown in Fig. 5):

- Classification of elements (rdf:type, grey dashed arrows) using rdfs:Class. The framework highly encourages the use of class disjointness (owl:disjointWith) in order to model explicity typing.
- The use of hierarchy to classify types (rdfs:subClassOf, rdfs:subPropertyOf) where appropriate.
- Explicit modelling domain (rdfs:domain) and range (rdfs:range) specifications for relations between elements (rdf:Property) where appropriate.
- The use of mm:ownedProperty (an instance of owl: ObjectProperty) to model a class' exclusive ownership a property, i.e., the property cannot exist without the class.

Restricting a domain model in terms of these idioms makes nonsensical information explicitly invalid, e.g., a jml:Sample cannot be an instance of jml:AdhesiveApplication. On the other hand, the framework leaves the possibility open for classification: a jml:Sample can be classified as a *good* sample or a *bad* sample (not shown in figure). Next to validity checking, a second benefit is that tools in the framework are able to leverage this additional information, for example to provide more dedicated support for exploring the knowledge graph compared to existing visualisation tools (see Section 4.4).

Reusable **domain ontologies** define ontologies for different types of problem domains, such as digital twins, or knowledge on influence factors, etc. (second layer in Fig. 5). Figure 5 shows an excerpt of the *dt* ontology to model manufacturing systems, with sequences of manufacturing steps (dt:ProcessStepExecution), that have a start time (dt:startTime), and that consume, produce, transform or involve products and machines (dt:Asset). The elements in *dt* are typed by the upper ontology. Furthermore an excerpt of the *tacit* ontology is shown, that models explicitly dependencies (tacit:influence) between properties, which are useful for correlation analysis and root cause analysis. The framework can be extended with different domain ontologies (**REQ21**).

The **domain model** represents the company-specific manufacturing system in the problem domain (third layer in Fig. 5). The domain model in this layer is created during the knowledge engineering workflow of Fig. 4. It instantiates and extends the upper ontology and domain ontologies of interest. The example model glued (jml:AdhesiveApplication) samples for which the tensile strength (jml:maxBreakStress) is important.

The **individuals** are the data, information and knowledge of the knowledge graph (fourth layer in Fig. 5). In our framework, the goal is to provide explicit types for all individuals by domain model, in accordance to the metamodelling paradigm that assumes that all objects are typed. The principle of strong typing improves human and computer interpretability of knowledge graph content. Individuals are modelled in terms of the problem domain (which may be of different structure than the underlying data sources), making them easier to understand by domain experts. Next to representing data and information about their context, additional manufacturing knowledge is explicitly defined as an influence relationship (jml:humidityCausality) that states that humidity during adhesive application influences the sample's maximum break stress.

The framework provides a base for ad hoc analysis that can be extended to improve support for specific types of ad hoc analysis (**REQ21**). The main extension method is to define a new domain ontology, compliant to the upper ontology (see Section "Domain model" and Fig. 5). A domain ontology may have dependencies with existing domain ontologies. The application cases (see Section "Validation on application cases") made use of these extension mechanisms.

Mapping domain model to data sources

While the individuals layer in Fig. 5 represents a logical structure, there are several strategies to access the data without the need for importing them in the knowledge graph triple store.

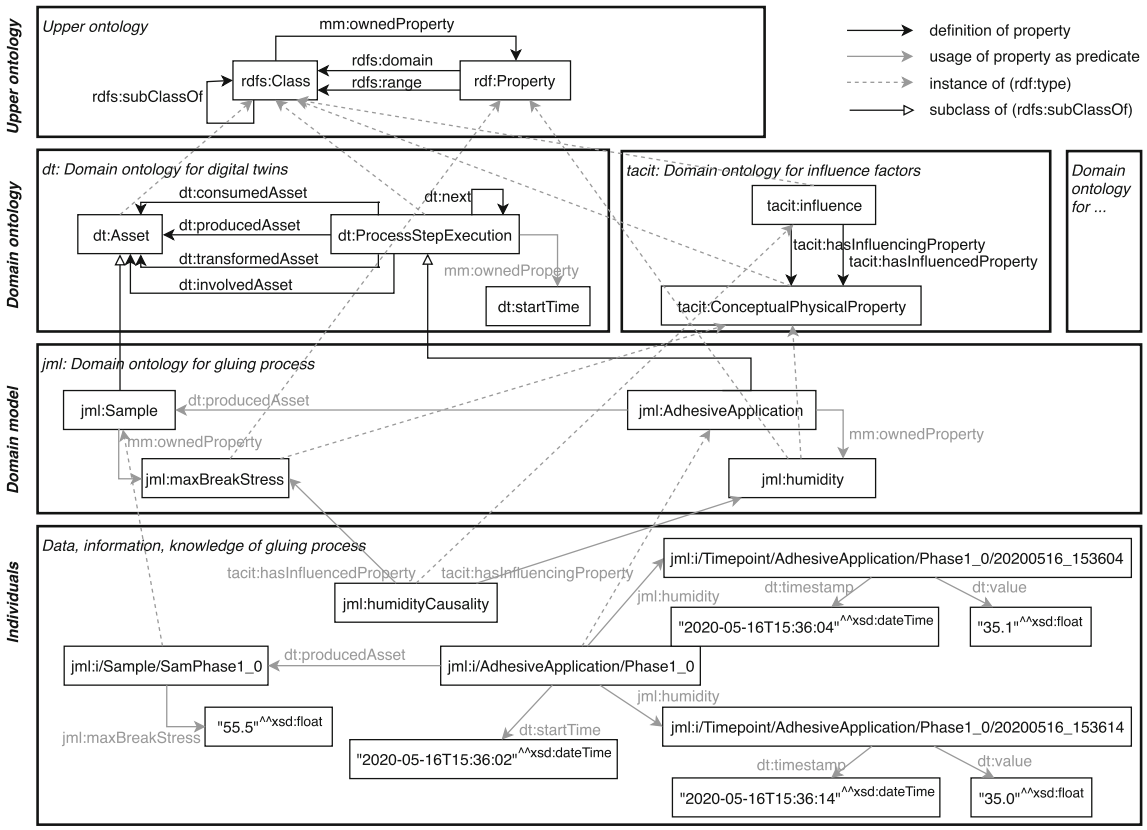

Fig. 5 A simplified excerpt of the structure of a knowledge graph of a gluing process in terms of the problem domain

This section describes how data sources are mapped to the domain model:

- By offering four different data access methods that can be used and combined for all necessary types of data sources (**REQ9**) to ensure performance (**REQ18**), allowing gradual adoption (**REQ22**), supporting evolving data (**REQ10**);
- By supporting a conversion from the problem domain (domain model) to the solution domain (data structure) (**REQ1**).

The four access methods are explained in the following, and guidelines on when to apply them are given in Table 3.

Materialization: creating RDF triples by using existing API libraries, for example in Python (rdflib[3] and kglab[4]) or in Java (Jena API[5]). This is the knowledge graph equivalent of Extract-Transform-Load (ETL) scripts. For knowledge that is *tacit* (i.e., only exists in the minds of people or is implicit from documents), triples can be the format to materialize this knowledge. As shown in Table 3, materialization is suitable for implicit or tacit information and data or knowledge that represent relations between concepts. Materialization is not suitable for data or knowledge that is represents calculation, or for time series data;

Virtualization, also known as ontology-based data access (OBDA) Kogalovsky (2012); Xiao et al. (2019): data is not copied into the knowledge graph but is accessed on demand via rules that map triples to databases. This has become a popular data access method (e.g., Ontop Calvanese et al. (2017), Mastro Calvanese et al. (2011), Stardog[6]). OBDA removes the need of copying and keeping consistent of data and managing ETL pipelines, while providing reasonable query execution performance. The current state-of-the-practice focuses on highly efficient virtualization of relational databases using mappings to SQL. As shown in Table 3, virtualization is suitable for databases that can be queried with SQL, as well as semantic data types that can be made accessible with SQL by virtualization tools like Dremio[7]) or Denodo[8]). There is interesting research to expand OBDA outside of SQL sources Kalayci et al. (2019); Botoeva et al. (2019); Bereta et al. (2019), but these approaches are not yet well supported by tools.

[3] https://rdflib.readthedocs.io/en/stable/, accessed 24/04/2023.

[4] https://derwen.ai/docs/kgl/, accessed 24/04/2023.

[5] https://jena.apache.org/documentation/ontology/, accessed 24/04/2023.

[6] http://www.stardog.com, accessed 24/04/2023.

[7] https://www.dremio.com/, accessed 24/04/2023.

[8] https://denodo.com, accessed 24/04/2023.

Table 3 Suitability for access methods of the knowledge graph to data, information, knowledge sources

Semantic type of data, information, knowledge	Example data persistence formats	access method			
		Materialization	Virtualization	Referencing	On demand calculation
Relational data	SQL	+	+++	− −	− −
Columnar data	Excel, CSV, Parquet	+	++	− −	− −
Hierarchical data	json, XML	++	+	−	-
Geometric data	CAD	−	− − −	+++	+
Event based data	log	+	++	− −	±
Time series data	InfluxDB, TimescaleDB	− −	− − −	+	++
NoSQL data	MongoDB	++	−	+++	+
Graph data	Neo4J	+++	− − −	++	+
Blob data	png, mpeg	− − −	− −	+++	±
Simulation	MATLAB scripts	− − −	− − −	+	+++
Cloud service	REST API	− − −	− − −	+	+++
Metadata	*often columnar data*	++	+++	− −	− − −
Domain concepts	UML class diagram	+++	− − −	− − −	− − −
Acquired data insights	*correlation index, feature value*	+++	−	− −	+
Relational knowledge	*influence factors*	+++	−	− − −	− − −
Modelled knowledge	SysML	+++	−	− − −	− − −
Executable knowledge	Python code	−	− − −	+	+++

Referencing: data can be accessed through the knowledge graph by storing a reference to the data in knowledge graph (access method, location, credentials). This can include a URI, database client information, and low level queries. Ontologies can be defined to define explicitly different reference methods depending on the data source. The actual retrieval of data is in this case done in a separate step. This decoupling of linking and retrieving data offers a benefit in case of performance, when the actual retrieval is inherently long in case of very large data sizes, as the data scientist can delay the actual retrieval of data and the execution of their analysis (e.g., in the case of training an AI model, which may take several hours). Furthermore, as shown in Table 3, referencing is suitable for blob data (images, videos, binary data) and geometric data, and is potentially useful to provide queries to specialized databases, services or models (time series, REST, simulation, SysML) in order to maximally exploit the features of these underlying databases.

On demand calculation: access by executing predefined and parameterizable code. Our approach for on demand calculationVanderhulst et al. (2023) allows users to code customized access to underlying data, information and knowledge sources. Our method allows registering "virtual" predicates that will calculate triples on demand by custom logic. For the user, they are semantically equivalent to regular triples.

While the idea of on demand calculation is not new Regalia et al. (2016); Taelman et al. (2018) and is supported by tools in the state-of-the-practice as magic predicates,[9] our approach completely shields the on demand calculation module from the user, and provides optimization in terms of filtering. Furthermore, our framework supports registering on demand calculation as a REST service in a programming language of choice. This allows optimal flexibility and separates the calculation module from the knowledge graph, thus lowering the barrier for creating new on demand calculation modules (**REQ11**). The REST service can implement any kind of stateless parameterizable logic. As shown in Table 3, on demand calculation is suitable for time series data, executing simulations, calling cloud services and for executable knowledge that was previously created by a data scientist (e.g., a correlation analysis, an AI based prediction module). On demand calculation is not suitable for directly accessing relational or columnar data, metadata or domain concepts, and relational or modelled knowledge.

All access methods can be combined for a single knowledge graph by our deployment architecture (see Sect. 4.5).

[9] Ontotext GraphDB https://graphdb.ontotext.com/documentation/10.2/pdf/GraphDB.pdf, accessed 02/05/2023.

Ad hoc analysis

Once a knowledge graph is constructed, it can be used by data scientists for ad hoc analysis. The goal is to generate insights that represent business value. Using a knowledge graph improves the ad hoc analysis process by improving data access (**REQ7**), making knowledge available (**REQ1**), and improving reasoning capabilities on data and knowledge (**REQ15**) in an automated way (**REQ16**), and integrating well with the core activity of data scientists, namely data analytics (**REQ20**). We position our knowledge graph based framework in the complete data analytics life cycle, from the creation of a model up to the business outcome. Due to its iterative and flexible nature, this life cycle complies with the ModelOps paradigm Hummer et al. (2019). In this paradigm our framework provides most value. In this section, we present a generic incremental ad hoc analysis workflow (**REQ14**) that interacts with the knowledge graph in four interaction patterns: explore (**REQ12**), query (**REQ13**), update (**REQ11**) and reason (**REQ15**). We then explain what methods and tools our framework for ad hoc analysis provides to support these interaction patterns for multiple users and workflows at the same time (**REQ17**), using cloud-based tools (**REQ24**).

Figure 6 shows an iterative ad hoc analysis workflow.

1. Understand and agree on the **goal of the business user**, and determine what insight they have in mind. For example, the end product quality is insufficient and the cause needs to be found, and quality needs to be improved by changing settings in the manufacturing process. This needs to be understood well: what is meant by quality? In what range of products is quality insufficient? What is considered to be an improvement? What are boundary conditions?
2. Translate the business goal into one or more **hypotheses in the problem domain**, together with the domain expert(s). For example, how is quality measured in the manufacturing process? Is quality measured for every product? What are the production steps that are performed before the quality measurement? What influences quality? What experiments have been done before related to quality? The knowledge graph is used in this phase as a replacement of a whiteboard, and as a way to improve cross domain understanding by providing access to nomenclature in the domain model, previous insights, etc. Existing knowledge, domain concepts and insights are the knowledge graph elements that are explored to support determining hypotheses.
3. **Look for the data, information and knowledge** that is needed to confirm or invalidate a hypothesis. In order to efficiently do this, the knowledge graph is explored to determine what data, information and knowledge exists,

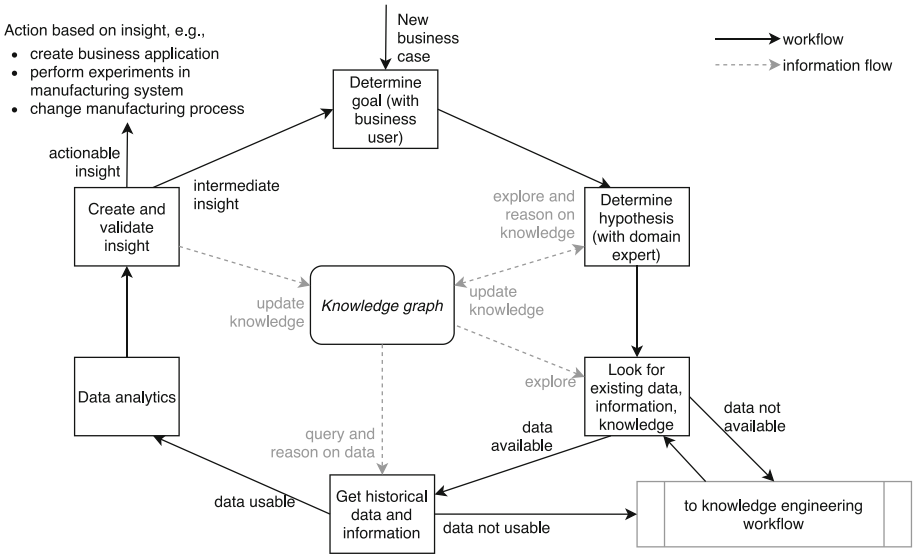

Fig. 6 Generic data scientist workflow for ad hoc analysis and interaction with the knowledge graph

by exploring directly the domain concepts of the problem domain (rather than the technical solution domain). In case the necessary data, information or knowledge is not available to analyse the hypothesis, a knowledge engineering workflow is started.

4. **Retrieve data by querying** (possibly including reasoning on) the knowledge graph in terms of problem domain concepts. In this step, a single query may combine different data sources and access methods, shielding the underlying technical structure and access of the data, information and knowledge. In the example, a data scientist may query the maximum break stress values of tested samples, together with the process settings and measurements during the gluing process that influence the break stress (using the modelled knowledge in the form of tacit:influence relationships as shown in Fig. 5).
5. With the necessary data, information and knowledge, the **core data analysis activity** is performed. In our framework this is considered to be the core expertise of the data scientist and covers a wide range of activities statistical analysis of data, feature modelling, training of an AI model, executing simulations, etc. This activity is maximally supported through the other steps in the ad hoc analysis workflow.
6. As a result of the data analytics step, an **insight is explicitly formulated, and new actions are defined**. The insight may trigger a new iteration of the workflow, or may trigger other workflows like creating an end user application. The insight may be stored in the knowledge graph, by using the domain model. For instance, analysis of the data may lead to a regression model of the causation between plasma settings and maximum break stress. This insight is saved as a tacit:ModelledInfluence instance in the knowledge graph, and may lead to the development of a dashboard application that predicts maximum break stress. Based on this insight, a next iteration of the ad hoc analysis workflow may be triggered to determine under which conditions an additional quality assurance test is needed.

As shown in Fig. 6, the data scientist uses four types of interaction with the knowledge graph.

Query. The data scientist can **query** the knowledge graph to retrieve data, information and knowledge in a computer readable format (**REQ13**).

In the semantic web tool stack, SPARQL The W3C SPARQL Working Group (2013) is the suggested language to query an RDF triple store via a SPARQL service. A SPARQL service is an HTTP service endpoint that can process SPARQL queries. In our framework, we can use the SPARQL language without any modification, since the knowledge graph is fully compatible with the semantic web tool stack. SPARQL can be used in typical data science environments as it is compatible with e.g., Python and Pandas **REQ20**).

In SPARQL, queries are graph patterns. As the knowledge graph relies on explicit domain models, patterns over these domain models are very intuitive.

Additionally, RDF allows SPARQL queries across all four layers of the knowledge graph, this allowing to query for data, information, knowledge, and also domain concepts. For a concrete example of such SPARQL queries, we refer to our previous work Meyers et al. (2022).

Explore.

We developed a visual exploration tool that allows users to interactively browse the knowledge graph content at the level of the domain model (**REQ5**, **REQ12**).

This tool is designed according to the following principles:

- It shields the user from the different data sources of the knowledge graph.
- It visualizes only knowledge graph concepts of interest, at the level of the problem domain. The tool avoids visualizing technical concepts like rdfs:domain, owl:DatatypeProperty or owl:Thing, or technical data structures of the solution domain.
- It provides an interactive interface that allows gradually exploring domain elements. This allows the user to limit themselves to a specific subdomain, without having to find domain elements in a complete knowledge graph visualization which can become unwieldy.
- Its content is kept synchronous with the knowledge graph as it is based on SPARQL queries to retrieve the elements that need to be visualised.
- It provides a combination of the ontology paradigm and metamodelling paradigm, as discussed in Section "Domain model". This is achieved by supporting on the one hand **domain-based exploration**. This follows the metamodelling paradigm where data is explicitly typed according to an explicit domain model. On the other hand, evidence-based exploration by querying instances is also supported. This follows the ontology pattern, where instances can be of multiple classes and an open world is assumed where elements can be of no class.
- It is generic and works for any knowledge graph defined according to the four-layer domain modelling structure of Section "Domain model". Nevertheless, it is view based and extensible, allowing dedicated views for each domain ontology.
- It is browser based, implemented as a Theia IDE.[10] A screenshot is shown in Fig. 7 showing graph bases exploration, data exploration based on plots, and an integrated SPARQL editor and executor.

Update. The knowledge graph can be updated by the data scientist to add new insights so that they can be explored, queried and reasoned on (**REQ11**). This can be done in two ways: by directly by adding calculated instance values or knowledge as triples to the knowledge graph by means of a SPARQL INSERT Query, or indirectly by adding the calculation code itself and make this a virtual predicate by means of registering a REST-based on demand calculation module.

Reason. The knowledge graph can automatically reason on data, information and knowledge to generate new knowledge by semantic reasoning, execution of algorithms, triggered by queries (**REQ15**, **REQ16**). For semantic reasoning on the one hand, the semantic web tool stack supports various means for logical inference, with wide spread reasoners that implement this logic, such as the OWL reasoner,[11] which we use for example in combination with the tacit domain ontology for influence relationships to infer a property's transitively influential properties. Execution of algorithms on the other hand is implemented by on demand calculation. The algorithms can be products of data analytics by the data scientist, making them available and reusable through the knowledge graph, or can be defined by the knowledge engineer. They can be of arbitrary complexity, and can include 3rd party tools for simulation, AI inference, etc. For both semantic reasoning and execution of algorithms the user is shielded from the underlying mechanisms, as they can use regular queries to trigger such computation.

Deployment

Our framework needs a versatile cloud based deployment method (**REQ24**), that can be used depending on the needs and data sources of each organization (**REQ23**). For this reason, we deploy the tools as a containerized application, which can be easily configured to deploy the containers that an organization needs. The framework includes a template based installation process that allows the quick configuration of new knowledge graph deployments. An example deployment is shown in Fig. 8. The data sources are shown at the bottom. All data sources are shown at the bottom, where local storage represents csv, Parquet, json and Excel files. In this configuration, we use Dremio[12] to provide an SQL interface for all data formats except InfluxDB. We use Ontop to provide virtual access to these data. For InfluxDB, access is via on demand calculation through a Python REST server implemented using our framework. These on demand calculation modules and the Ontop endpoint are accessible to the user through the on demand calculation (ODC) proxy endpoint. The ODC endpoint allows users (1) to register new on demand calculation modules and (2) automatically route, process and forward SPARQL queries. In this example, the Exploration tool, ODC endpoint and Python REST server are based on our own contributions of our framework, while Ontop and Dremio are existing tools that we use in our framework.

Validation on application cases

In this section we discuss how we validated our framework on three application cases from the eight we used as input for the requirements analysis in Sect. 3. For each, we briefly explain

[10] https://theia-ide.org/

[11] A list of OWL reasoners is maintained on the W3C website https://www.w3.org/2001/sw/wiki/OWL/Implementations. Visited on 12/05/2023.

[12] https://www.dremio.com/. Visited 12/05/2023.

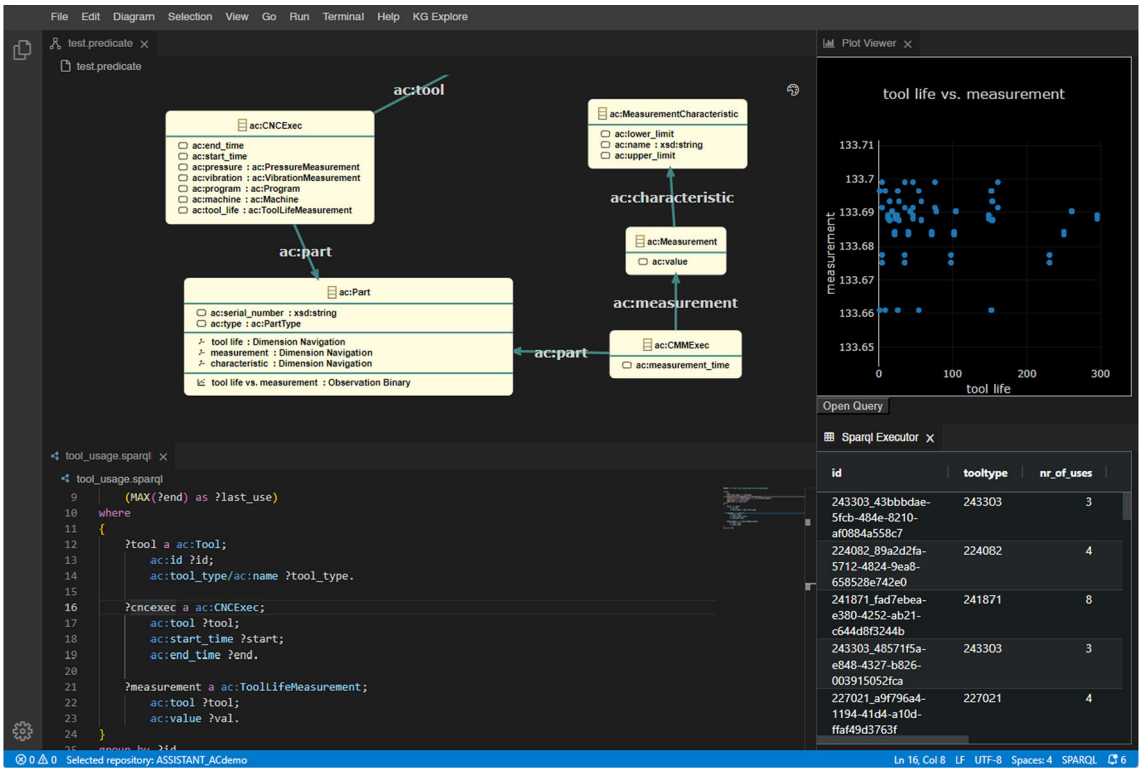

Fig. 7 Exploration tool for manufacturing knowledge graphs

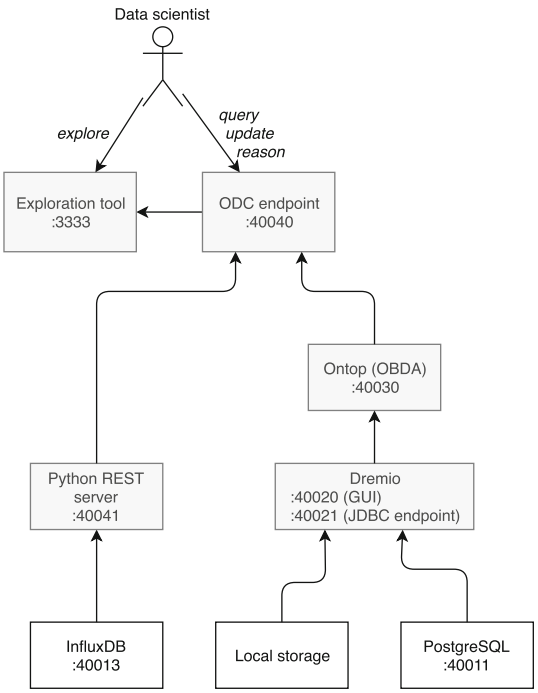

Fig. 8 Technical deployment architecture overview for the gluing example

the context, the ad hoc analysis goal, how the three main challenges of ad hoc analysis are present for the application case, and how the framework was applied to overcome these challenges.

Knowledge graph for zero defect manufacturing

Context. This application case investigates methods for zero defect manufacturing of an Atlas Copco compressor air end, the core element of a compressor. The case focuses on the detection and prediction strategies of zero defect manufacturing Psarommatis et al. (2019). The role of data for these strategies has been marked important for quality improvement Psarommatis et al. (2022), and has been identified as a key enabling technology in Powell et al. (2022). One of the main manufacturing steps is a machining step using a CNC (computer numerical control) machine to mill, drill, grind, etc. the housings of these compressor core elements. Manual (CAQ and coordinate-measuring machine (CMM) tests are performed to measure the geometric quality of these housings after they have been machined. Geometric quality is essential to ensure high performance of the compressor.

Ad hoc analysis goal. The goal is to reduce cost by reducing the number of quality tests, while maintaining confidence in quality. For this, Atlas Copco is researching an adaptive measurement strategy, that uses data based methods (i.e., machine learning) to predict quality based on machine and environmental data Meyers et al. (2022). This involves a com-

plete AI life cycle: gather data, find and analyze insights like correlations within the manufacturing system, machine learning, deployment, execution and validation of an AI model, and, whenever deemed necessary, retraining.

Challenges. The three main challenges were identified as follows:

- **Main challenge 1: Heterogeneity of data.** To analyze what influences quality, data must be combined from sensors (time series), the MES system (SQL), a CAQ database (a different SQL database), CMM (pdf).
- **Main challenge 2: Lack of information and knowledge formalization.** Traceability of products throughout the production system needs to be explicitly formalized, in order to link data at the level of individual products.
- **Main challenge 3: Inability to meaningfully and efficiently perform inductive reasoning.** Without our framework, this is a process that involve long iterations for ad hoc analysis and (re)deployment of the AI model. This is because data and knowledge unavailability extends data retrieval times and undermines the quality of insights. Due to the impractical retrieval process, it may for example take weeks to discover that there are not a sufficient number of data points to evaluate a given hypothesis.

Solution. In this application case, we showed how our knowledge graph based framework improves the AI life cycle, as shown in Fig. 9. The AI life cycle is shown on the right side. It is a specific variant of the data scientist workflow shown in Fig. 1.

The goal of this workflow is finding useful correlations in the data. During this workflow, the knowledge graph is gradually built up, when new data sets are needed for analysis. The ML model that results from this workflow can be automatically deployed in an automated consumer zone together with a dashboard (see Fig. 10), where the model can be executed for inference, in this case quality prediction and assessment whether a test is needed. We deployed our architecture in an experimental environment that is detached from Atlas Copco production. This environment is designed so that it mimicks the manufacturing system by providing raw and streaming data in the same format as the factory floor. At the time of writing, Atlas Copco is still incrementally including new hypotheses and data sources into this workflow, in order to improve results.

Knowledge graph for flexible dashboard creation

Context. In this application case, we applied our knowledge graph framework to an undisclosed machine builder. The machines they build are equipped with a large number of sensors, that measure its operation, material consumption, etc. Machines are virtually unique, as are highly configured with custom sensors, actuators and software, depending on customer need.

Ad hoc analysis goal. There is a customer need to obtain insight in material consumption to improve performance. For this reason, the machine builder is looking into expanding their portfolio to offer data analytics dashboards for their customers that are customized to their needs and their machines.

Challenges. The three main challenges were identified as follows:

- **Main challenge 1: Heterogeneity of data.** As the machines are real-time cyber-physical systems, a lot of the data is stored as time series data. The major data sources are InfluxDB, Parquet and Microsoft SQL server.
- **Main challenge 2: Lack of information and knowledge formalization.** Reuse of information and knowledge is hampered by a lack of formalization. Typical features of the machine need to be explicitly formalized, such as material consumption per machine subprogram.
- **Main challenge 3: Inability to meaningfully and efficiently perform inductive reasoning.** For the creation of custom dashboards, the ad hoc analysis is the major bottleneck to find and validate the insights that need to be converted into the dashboard applications. As dashboards are custom, the demand for ad hoc analysis rises together with the customer demand for new dashboards. The machine builder has extensively investigated GraphQL to address this challenge, but found that it still lacked flexibility and extensibility.

Solution. We used the knowledge graph framework to provide a single model at the functional (rather than technical) level to relate the data silos (relational vs. time series), and to support the reuse of previous insights to build a data analytics knowledge base. These insights take the form of executable scripts for data analysis or AI and can be reused for other customers.

The overview of the approach is shown in Fig. 11 where a data scientist (top right) needs to acquire business insights to create multiple customer dashboards from a technically curated zone that includes SQL, InfluxDB and Parquet data. We introduced a knowledge graph with a domain model that takes the role of functional model and that enables the data scientist to query for data scattered across the different data sources. Insights in the form of data features, aggregations, correlations or other calculations are stored as on demand calculation modules and made available in the knowledge graph. Data can be directly retrieved for ad hoc analysis by means of the virtualization or on demand calculation access methods. Data can also be retrieved indirectly via the referencing access method. The latter is used for creating the dashboard applications that can be deployed at the customer.

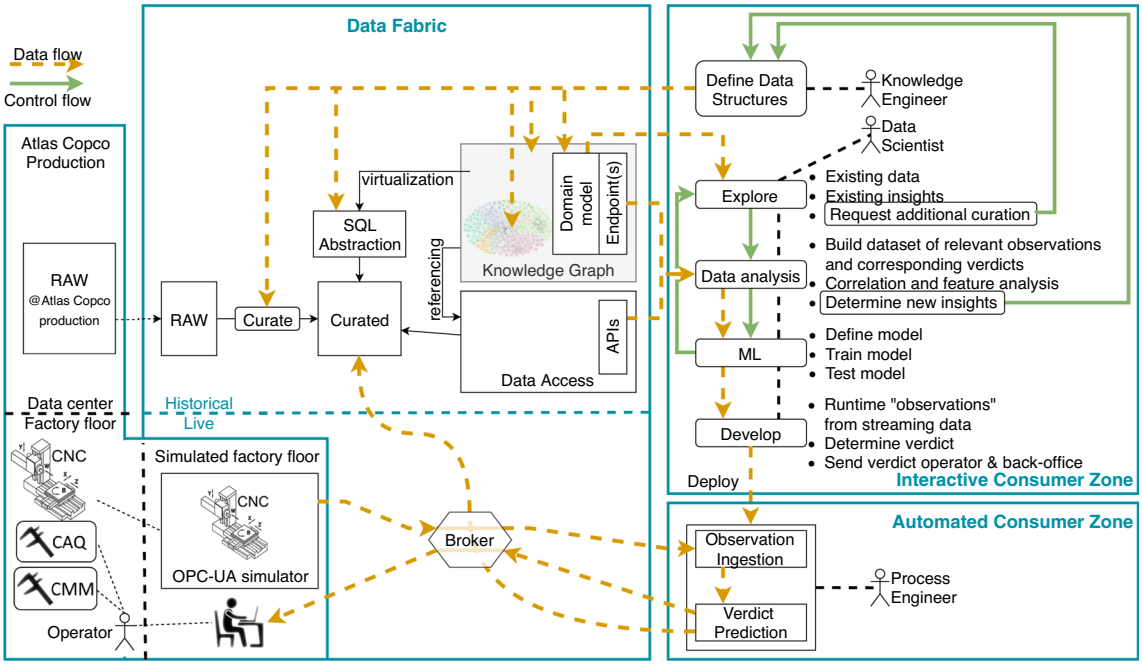

Fig. 9 Overview of the Atlas Copco case for zero defect manufacturing

Fig. 10 Dashboard of the deployed AI model for adaptive measurement (synthetic data)

The dashboards can operate on the technically curated customer data at the customer (which is of the same format as the technically curated zone), removing the need for a deployed knowledge graph in the customer zone.

Knowledge graph for data analytics under uncertainty

Context. In this application case bonding (gluing) and debonding experiments for several materials are carried out in an experimentation lab. In this manufacturing process, two substrates are glued to form a sample. The goal is to obtain a strong adhesion, which can be tested using a break stress

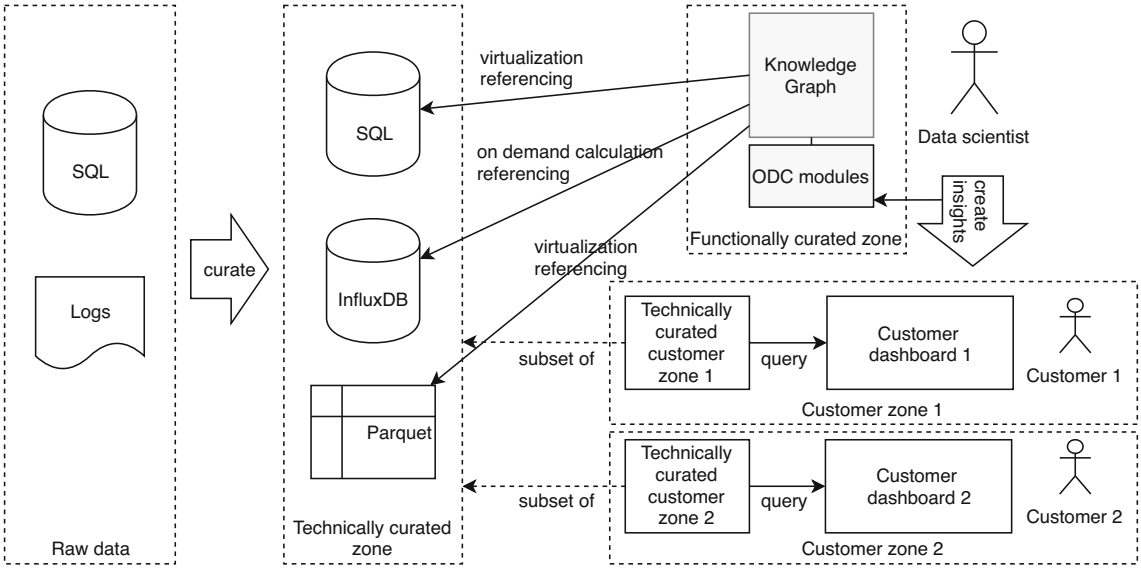

Fig. 11 Custom dashboard creation using the knowledge graph framework

test, i.e., a tensile strength test that determines the maximum break stress, while maintaining visual quality of a sample (i.e., the sample is not burnt).

Ad hoc analysis goal. The goal is to understand how process parameters influence properties of interest, and to perform root cause analysis in case of quality problems. Subsequently, the goal is to train a machine learning model that is able to find optimal settings for new glues or materials based on a minimal number of experiments. When analyzing dependencies in a manufacturing system, it is often desirable to take uncertainties into account, to obtain an answer to questions like *"what is the probability that maximum break stress will be within tolerance and the visual quality will be good, given the plasma settings and contact angle measurements"*.

Challenges. The three main challenges were identified as follows:

- **Main challenge 1: Heterogeneity of data.** Data are semi-formalized an partly automatically ingested (sensors as Parquet, MES system as json), and partly manually created (quality measurements as Excel file).
- **Main challenge 2: Lack of information and knowledge formalization.** There is a need for explicitly reasoning with uncertainties. The probabilistic variables need to be explicitly formalized.
- **Main challenge 3: Inability to meaningfully and efficiently perform inductive reasoning.** Given the data in the process, the root cause analysis is a cumbersome process that is not flexible for testing different hypothesis in an efficient way. Data needs to be manually linked and converted in a statistical model, in order to analyse how parameters influence each other. For each analysis, the process needs to be carried out from the start. This process is inefficient, inflexible, and error prone.

Solution. We introduced a domain ontology for probabilistic reasoning using Bayesian inference into the knowledge graph. In this application case, we used the knowledge graph to add all information that is needed to create a trained Bayesian network in three steps:

- determine influence relationships between properties of the manufacturing system (arrows in Fig. 12). The knowledge graph allowed us to explicitly model these influence relationships using the "tacit" domain ontology (see Fig. 5). Such influence may be found in the data, of which their retrieval is supported by knowledge graph. We also used different knowledge sources, i.e., the subject matter expert, scientific publications, and machine manuals. All knowledge is homogeneously modelled in the knowledge graph;
- determine a semantically useful categorization for each value (shown as annotations on properties in Fig. 12). Starting from the requirement that the maximum break stress must be at least 8 MPa and that the visual quality should be acceptable (i.e., the sample cannot be burnt), categories for each property are determined by analyzing the data. Multiple strategies are possible: by simple discretization of the value range, by clustering, or by regression analysis and intersection (as shown in the graph on the bottom left of Fig. 12). The resulting categories are stored using the PR-OWL domain ontology Carvalho et al. (2017);
- determine the conditional probabilities in each table cell. The probabilities need to be quantified for each cell in the conditional probability tables. If sufficient data is available in the knowledge graph, this can be done using simple statistical methods. In other cases, an estimate

can be inserted based on literature, know-how, etc. The probabilities are stored according to PR-OWL.

Automated reasoning under uncertainty is implemented in this application case in the form of Bayesian inference, by extracting the Bayesian network elements from the knowledge graph and generating a pgmpy[13] Bayesian network. With this network, pgmpy can infer given variables with given evidence. We implemented an on demand calculation module that defines a property of a sample that returns the probability that the maximum break stress will be in tolerance. The module automatically extracts all existing data for that sample from the knowledge graph to formulate the evidence for the Bayesian inference. For the end user, this results in the possibility to simply query for the probability that the maximum break stress will be in tolerance for any sample in an intuitive way, as if it were implemented as a data field of sample. Since all data and all elements of the Bayesian network are stored in the knowledge graph and are connected, the barrier for determining model drift and retraining the Bayesian network is lowered.

Discussion

The framework description and application cases show that the framework fulfills the requirements defined in Sect. 3. After analysis of our application cases we conclude that the framework is functionally complete in terms of:

- access methods. We are able to adequately treat all different data, information and knowledge methods using the four access methods of Sect. 4.3.2. While some semantic types are best addressed using materialization or on demand calculation (see Table 3), a fall-back to materialization may be required for performance issues, or, in the worst case, referencing. In any case, the framework still improves the capability to find and understand data;
- interaction methods. The full ad hoc analysis workflow makes use of query, explore, update, reason interaction with the knowledge graph as shown in Fig. 6. These interactions and their supporting tools turn out to be efficient to carry out the application cases.

Value proposition of the knowledge-graph based framework

We discuss the value proposition of our framework in three categories as determined in previous research on knowledge graph adoption Atkin et al. (2022):

- *Capability enhancement*. Our framework enhances the level of complexity that can be dealt with, in three dimensions: the number of concepts (data, information and knowledge), the number of domains (with each different people, goals for ad hoc analysis, domain ontologies and views), and the number of reusable insights (actionable insights, knowledge, AI models and life cycles). In our framework, these dimensions are connected by **explicitly modelling data, information and knowledge** in a knowledge graph. By providing this **abstraction layer**, ad hoc analysis is expanded to span different data and knowledge silos and multiple problem domains, and obtained insights are introduced in the body of knowledge of an organisation. For example, Sect. 5.1 shows how the knowledge graph is used to find correlations in a manufacturing system with multiple data and knowledge sources. Section 5.2 improves the reasoning capabilities on data by introducing obtained insights in the knowledge graph for reuse. Section 5.3 shows how complex analysis can be done while taking a complete manufacturing system into account.
- *Control environment*. Our framework allows to mitigate risks better by improving **control over business processes**. Furthermore, the knowledge graph provides **better understanding and explainability** of relationships among data, information, knowledge. While the adoption of the framework itself comes with an additional cost, the value driven, **incremental adoption** we propose by starting from business goals, mitigates the risk of high initial cost. For example, Sect. 5.1 shows how the technical risks can be mitigated to support the complete AI life cycle from the first iteration of an AI model.
- *Cost containment*. Our framework provides means to **support ad hoc analysis workflows** to reduce the time it takes to find insights by introducing a single point of access for data, information and knowledge. Data understanding and retrieval is improved by allowing **exploring, querying and reasoning** at the right level of abstraction. This is especially shown in Sect. 5.1 where correlation analysis is made more efficient and Sect. 5.2 where the cost to create flexible dashboards is reduced.

Open challenges

During our research, we identified the following open challenges.

Tools for the knowledge engineering workflow. Industrial adoption is still hampered by the current tools, that are still far from the maturity that is required. A lot of responsibility is on the knowledge engineer to avoid mistakes when creating a domain model, and there is no single tool to use all data access methods. Additionally, a number of important aspects are not addressed in this paper, such as security

[13] https://pgmpy.org/. Visited 14/05/2023.

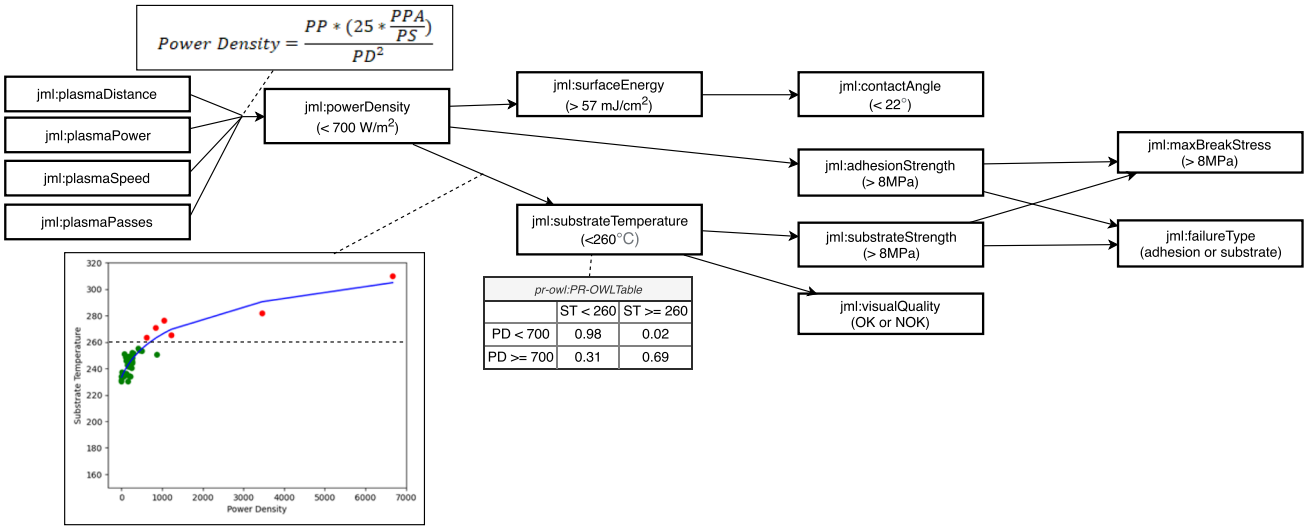

Fig. 12 Excerpt of the Bayesian network structure in the knowledge graph based on tacit:influence relations

and data governance, and validation and verification of the knowledge graph.

Knowledge engineer role. Knowledge engineering requires a system level view on the manufacturing system, modelling skills, domain knowledge and data knowledge. This is typically not concentrated in one profile in an organisation. Hence, good collaboration is required to fulfill the knowledge engineering workflow, and all involved people must be aligned in order to be successful.

Tools for the ad hoc analysis workflow. The query, explore, update and reason tools must be well integrated with the data scientist tool stack, e.g., notebooks in Databricks. This can be done in the form of direct support for SPARQL (similar to support for SQL), including input completion and validation based on the domain knowledge content. Our exploration tool can be expanded with different views to also visualize data and knowledge, next to domain models.

Query execution performance. Query execution can be unexpectedly bad, due to certain misuses of data access constructs. Currently, improving performance is an ad hoc process. There is a need for a better understanding and support for the data access methods we provide in terms of query execution performance.

Systematic modelling of knowledge. Formalized knowledge is almost always available in a manufacturing system. It should be used optimally to build a knowledge graph's domain model. Existing standards for manufacturing such as SOSA, SysML or ISA-95 can be used as a starting point for creating an organization's domain model. The evolution of the knowledge graph must be made explicit, to allow querying its status at a certain moment in time, and to support what-if analysis.

Conclusion and future work

We provide a framework for system level ad hoc analysis in manufacturing based on knowledge graphs. The novelty is:

- we identified **framework requirements** that are driven by industrial cases and the three challenges of ad hoc analysis, namely data heterogeneity, knowledge formalization and reasoning capabilities;
- we generalized a **framework structure and specific workflows** for knowledge engineers and data scientists, so that the reader can understand the interaction with the ad hoc analysis framework;
- we presented a **technical implementation** of our framework combining existing tools from the semantic web tool stack and our own tools that:
  - combine the **ontological and metamodelling paradigms** and introduce a four layered domain modelling approach that structures data, information and knowledge;
  - introduces novel **data, information and knowledge exploration** approach that exploits this domain modelling approach, and that is implemented as a software tool;
  - combine all identified ways of **data access** depending on the data source and data user's requirements (rather than limiting to one or the other);
- we **validated the framework** on different application cases from industry, indicating that the framework is versatile and extensible. This is different from the state-of-the-art, where framework propositions for system-level analysis are at a conceptual level, or are tailored to a specific domain or case;

- we provided an analysis of the value proposition and open challenges of our framework.

In future work, we will focus on extending our framework from the manufacturing domain to the product life cycle, spanning design, manufacturing, operation and reuse. The goal is to allow decision making that spans the full life cycle, i.e., improving the ability to feed back insights from one life cycle phase to another. We are looking into research on digital passports and the asset administration shell (AAS) Neidig et al. (2022).

Acknowledgements This research was supported by Flanders Make, the strategic research center for the manufacturing industry. This paper is partially funded by the European Commission under the framework program Horizon 2020, grant agreement number 101000165, project name ASSISTANT Beldiceanu et al. (2021). This research was supported by Flanders Make, the strategic research center for the manufacturing industry. This paper was partially funded by the DTDesign ICON (Flanders Innovation and Entrepreneurship FM/ICON:: HBC.2019.0079) project.

References

Adolphs, P., Bedenbender, H., Dirzus, D., Ehlich, M., Epple, U., Hankel, M., Heidel, R., Hoffmeister, M., Huhle, H., Kärcher, B., Koziolek, H., Pichler, R., Pollmeier, S., Schewe, F., Walter, A., Waser, B., & Wollschlaeger, M. (2014). Status Report: Reference Architecture Model Industrie 4.0 (RAMI 4.0). https://www.zvei.org/fileadmin/user_upload/Presse_und_Medien/Publikationen/2016/januar/GMA_Status_Report__Reference_Archtitecture_Model_Industrie_4.0__RAMI_4.0_/GMA-Status-Report-RAMI-40-July-2015.pdf. Accessed: April 13, 2023x

Ameri, F., Sormaz, D., Psarommatis, F., & Kiritsis, D. (2022). Industrial ontologies for interoperability in agile and resilient manufacturing. *International Journal of Production Research, 60*(2), 420–441. https://doi.org/10.1080/00207543.2021.1987553

Arista, R., Zheng, X., Lu, J., & Mas, F. (2023). An ontology-based engineering system to support aircraft manufacturing system design. *Journal of Manufacturing Systems, 68*, 270–288. https://doi.org/10.1016/j.jmsy.2023.02.012

Arista, R., Zheng, X., Lu, J., & Mas, F. (2023). An ontology-based engineering system to support aircraft manufacturing system design. *Journal of Manufacturing Systems, 68*, 270–288. https://doi.org/10.1016/j.jmsy.2023.02.012

Atkin, M., Wisnosky, D., Scharffe, F., & Deely, T. (2022) Knowledge Graph Industry Survey Report. Accessed: 2023-04-24. https://www.knowledgegraph.tech/knowledge-graph-industry-survey-2022/

Banerjee, A., Dalal, R., Mittal, S., & Joshi, K.P. (2017). Generating digital twin models using knowledge graphs for industrial production lines. https://doi.org/10.1145/3091478.3162383

Bartalos, P., & Bielikova, M. (2007). An approach to object-ontology mapping. In: IIT. SRC-Student Research Conference, pp. 9–16

Beldiceanu, N., Dolgui, A., Gonnermann, C., Gonzalez-Castañé, G., Kousi, N., Meyers, B., Prud'homme, J., Thevenin, S., Vyhmeister, E., & Östberg, P.-O. (2021). ASSISTANT: Learning and robust decision support system for agile manufacturing environments. *IFAC-PapersOnLine, 54*(1), 641–646. https://doi.org/10.1016/j.ifacol.2021.08.074

Bereta, K., Xiao, G., & Koubarakis, M. (2019). Ontop-spatial: Ontop of geospatial databases. *Journal of Web Semantics*. https://doi.org/10.1016/j.websem.2019.100514

Botoeva, E., Calvanese, D., Cogrel, B., Corman, J., & Xiao, G. (2019). Ontology-based data access - beyond relational sources. *Intelligenza Artificiale, 13*(1), 21–36. https://doi.org/10.3233/IA-190023

Buchgeher, G., Gabauer, D., Martinez-Gil, J., & Ehrlinger, L. (2021). Knowledge graphs in manufacturing and production: A systematic literature review. *IEEE Access, 9*, 55537–55554. https://doi.org/10.1109/ACCESS.2021.3070395

Calvanese, D., Cogrel, B., Komla-Ebri, S., Kontchakov, R., Lanti, D., Rezk, M., Rodriguez-Muro, M., & Xiao, G. (2016). Ontop: Answering SPARQL queries over relational databases. *Semantic Web, 8*(3), 471–487. https://doi.org/10.3233/SW-160217

Calvanese, D., Cogrel, B., Komla-Ebri, S., Kontchakov, R., Lanti, D., Rezk, M., Rodriguez-Muro, M., & Xiao, G. (2017). Ontop: Answering SPARQL queries over relational databases. *Semantic Web, 8*(3), 471–487. https://doi.org/10.3233/SW-160217

Calvanese, D., Giacomo, G. D., Lembo, D., Lenzerini, M., Poggi, A., Rodriguez-Muro, M., Rosati, R., Ruzzi, M., & Savo, D. F. (2011). The MASTRO system for ontology-based data access. *Semantic Web, 2*(1), 43–53. https://doi.org/10.3233/SW-2011-0029

Carvalho, R. N., Laskey, K. B., & Costa, P. C. G. (2017). PR-OWL - a language for defining probabilistic ontologies. *International Journal of Approximate Reasoning, 91*, 56–79. https://doi.org/10.1016/j.ijar.2017.08.011

Chávez-Feria, S., García-Castro, R., & Poveda-Villalón, M. (2022). Chowlk: from UML-based ontology conceptualizations to OWL. In: Groth, P., Vidal, M., Suchanek, F.M., Szekely, P.A., Kapanipathi, P., Pesquita, C., Skaf-Molli, H., Tamper, M. (eds.) The Semantic Web - 19th International Conference, ESWC 2022, Hersonissos, Crete, Greece, May 29 - June 2, 2022, Proceedings. Lecture Notes in Computer Science, vol. 13261, pp. 338–352. https://doi.org/10.1007/978-3-031-06981-9_20

CrowdFlower: 2016 Data Science report. Accessed: 2023-04-21 (2016). https://visit.figure-eight.com/rs/416-ZBE-142/images/CrowdFlower_DataScienceReport_2016.pdf

Djuric, D., Gasevic, D., Devedzic, V., & Damjanovic-Behrendt, V. (2005). A UML profile for OWL ontologies. In: Aßmann, U., Aksit, M., Rensink, A. (eds.) Model Driven Architecture, pp. 204–219. Springer, Berlin, Heidelberg. https://doi.org/10.1007/11538097_14

Dudás, M., Lohmann, S., Svátek, V., & Pavlov, D. (2018). Ontology visualization methods and tools: a survey of the state of the art. *The Knowledge Engineering Review*. https://doi.org/10.1017/S0269888918000073

García-Peñalvo, F. J., de Pablos, P. O., García, J., & Therón, R. (2014). Using OWL-VisMod through a decision-making process for reusing OWL ontologies. *Behaviour & Information Technology, 33*(5), 426–442. https://doi.org/10.1080/0144929X.2012.709538

Gayathri, R., & Uma, V. (2018). Ontology based knowledge representation technique, domain modeling languages and planners for robotic path planning: A survey. *ICT Express, 4*(2), 69–74. https://doi.org/10.1016/j.icte.2018.04.008

Giustozzi, F., Saunier, J., & Zanni-Merk, C. (2018). Context modeling for Industry 4.0: an ontology-based proposal. *Procedia Computer Science, 126*, 675–684. https://doi.org/10.1016/j.procs.2018.08.001

Grangel-González, I. (2019). A knowledge graph based integration approach for Industry 4.0. PhD thesis, Universität Bonn.

Grangel-González, I., Halilaj, L., Coskun, G., Auer, S., Collarana, D., & Hofmeister, M. (2016). Towards a semantic administrative shell for Industry 4.0 components. https://doi.org/10.1109/ICSC.2016.58

Grevenitis, K., Psarommatis, F., Reina, A., Xu, W., Tourkogiorgis, I., Milenkovic, J., Cassina, J., & Kiritsis, D. (2019). A hybrid framework for industrial data storage and exploitation. In: 52nd CIRP Conference on Manufacturing Systems (CMS), Ljubljana, Slovenia, June 12-14. Procedia CIRP, vol. 81, pp. 892–897 (2019). https://doi.org/10.1016/j.procir.2019.03.221

Guimarães, R., & Ozaki, A. (2022). Reasoning in knowledge graphs. In: Bourgaux, C., Ozaki, A., Peñaloza, R. (eds.) International Research School in Artificial Intelligence in Bergen (AIB 2022). Open Access Series in Informatics (OASIcs), vol. 99, pp. 1–31. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl, Germany. https://doi.org/10.4230/OASICS.AIB.2022.2

Hildebrandt, C., Köcher, A., Küstner, C., López-Enríquez, C.-M., Müller, A. W., Caesar, B., Gundlach, C. S., & Fay, A. (2020). Ontology building for cyber-physical systems: Application in the manufacturing domain. *IEEE Transactions on Automation Science and Engineering, 17*(3), 1266–1282. https://doi.org/10.1109/TASE.2020.2991777

Hop, W., de Ridder, S., Frasincar, F., & Hogenboom, F. (2012). Using hierarchical edge bundles to visualize complex ontologies in glow. In: Proceedings of the 27th Annual ACM Symposium on Applied Computing. SAC '12, pp. 304–311. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/2245276.2245338

Hummer, W., Muthusamy, V., Rausch, T., Dube, P., Maghraoui, K.E., Murthi, A., & Oum, P. (2019). ModelOps: Cloud-based lifecycle management for reliable and trusted AI. In: IEEE International Conference on Cloud Engineering, IC2E 2019, Prague, Czech Republic, June 24–27, (2019), pp. 113–120. https://doi.org/10.1109/IC2E.2019.00025

Jinzhi, L., Zhaorui, Y., Zheng, X., Jian, W., & Dimitris, K. (2022). Exploring the concept of cognitive digital twin from model-based systems engineering perspective. *The International Journal of Advanced Manufacturing Technology, 121*, 5835–5854. https://doi.org/10.1007/s00170-022-09610-5

Kalaycı, G., Gonzalez, I.G., Lösch, F., Xiao, G., ul-Mehdi, A., Kharlamov, E., & Calvanese, D. (2020). Semantic integration of Bosch manufacturing data using virtual knowledge graphs. In: Proceedings of the 19th International Semantic Web Conference (ISWC 2020). Lecture Notes in Computer Science, vol. 12507, pp. 464–481. https://doi.org/10.1007/978-3-030-62466-8_29

Kalayci, E. G., Brandt, S., Calvanese, D., Ryzhikov, V., Xiao, G., & Zakharyaschev, M. (2019). Ontology-based access to temporal data with Ontop: A framework proposal. *International Journal of Applied Mathematics and Computer Science, 29*(1), 17–30. https://doi.org/10.2478/amcs-2019-0002

Kharlamov, E., Hovland, D., Skjæveland, M., Bilidas, D., Jiménez-Ruiz, E., Xiao, G., Soylu, A., Lanti, D., Rezk, M., Zheleznyakov, D., Giese, M., Lie, H., Ioannidis, Y., Kotidis, Y., Koubarakis, M., & Waaler, A. (2017). Ontology based data access in Statoil. *Web Semantics: Science, Services and Agents on the World Wide Web, 44*, 3–36. https://doi.org/10.1016/j.websem.2017.05.005

Kharlamov, E., Kotidis, Y., Mailis, T., Neuenstadt, C., Nikolaou, C., Özçep, Ö., Svingos, C., Zheleznyakov, D., Brandt, S., Horrocks, I., Ioannidis, Y., Lamparter, S., & Möller, R. (2016). *Towards Analytics Aware Ontology Based Access to Static and Streaming Data*. Cham: Springer. https://doi.org/10.1162/dint_a_00011

Kogalovsky, M. R. (2012). Ontology-based data access systems. *Programming and Computer Software, 38*(4), 167–182. https://doi.org/10.1134/S0361768812040032

Kourtis, G., Kavakli, E., & Sakellariou, R. (2019). A rule-based approach founded on description logics for Industry 4.0 smart factories. *IEEE Transactions on Industrial Informatics, 15*(9), 4888–4899. https://doi.org/10.1109/TII.2019.2916622

Kühne, T. (2006). Matters of (meta-)modeling. *Software & Systems Modeling, 5*(4), 369–385. https://doi.org/10.1007/s10270-006-0017-9

Kulvatunyou, B.S., Wallace, E., Kiritsis, D., Smith, B., & Will, C. (2018). The industrial ontologies foundry proof-of-concept project. In: Moon, I., Lee, G.M., Park, J., Kiritsis, D., von Cieminski, G. (eds.) Advances in Production Management Systems. Smart Manufacturing for Industry 4.0, pp. 402–409. Springer, Cham. https://doi.org/10.1007/978-3-319-99707-0_50

Kumar, A., Bharadwaj, A., Starly, B., & Lynch, C. (2022). FabKG: A knowledge graph of manufacturing science domain utilizing structured and unconventional unstructured knowledge source. In: Proceedings of the Workshop on Structured and Unstructured Knowledge Integration (SUKI), pp. 1–8. Association for Computational Linguistics, Seattle, USA. https://doi.org/10.18653/v1/2022.suki-1.1

Kwon, S., Monnier, L. V., Barbau, R., & Bernstein, W. Z. (2020). Enriching standards-based digital thread by fusing as-designed and as-inspected data using knowledge graphs. *Advanced Engineering Informatics, 46*, 101102. https://doi.org/10.1016/j.aei.2020.101102

Leitão, P., Rodrigues, N., Turrin, C., Pagani, A., & Petrali, P. (2012). Grace ontology integrating process and quality control. In: IECON 2012 - 38th Annual Conference on IEEE Industrial Electronics Society, pp. 4348–4353. https://doi.org/10.1109/IECON.2012.6389189

Lohmann, S., Negru, S., Haag, F., & Ertl, T. (2016). Visualizing ontologies with VOWL. *Semantic Web, 7*, 399–419. https://doi.org/10.3233/SW-150200

Lu, J., Zheng, X., Schweiger, L., Kiritsis, D. (2021). A cognitive approach to manage the complexity of digital twin systems. In: West, S., Meierhofer, J., Ganz, C. (eds.) Smart Services Summit, pp. 105–115. Springer, Cham. https://doi.org/10.1007/978-3-030-72090-2_10

Martinez-Gil, J., Buchgeher, G., Gabauer, D., Freudenthaler, B., Filipiak, D., & Fensel, A. (2022). Root cause analysis in the industrial domain using knowledge graphs: A case study on power transformers. *Procedia Computer Science, 200*, 944–953. https://doi.org/10.1016/j.procs.2022.01.292

Meyers, B., Van Noten, J., Lietaert, P., Tielemans, B., Hristov, H., Maes, D., & Gadeyne, K. (2022). Knowledge graphs in digital twins for manufacturing - lessons learned from an industrial case at Atlas Copco Airpower. *IFAC-PapersOnLine, 55*(10), 13–18. https://doi.org/10.1016/j.ifacol.2022.09.361

Motta, E., Mulholland, P., Peroni, S., d'Aquin, M., Gomez-Perez, J.M., Mendez, V., & Zablith, F. (2011). A novel approach to visualizing and navigating ontologies. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) International Semantic Web Conference ISWC 2011, pp. 470–486. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-25073-6_30

Munch, M., Dibie-Barthelemy, J., Wuillemin, P.-H., & Manfredotti, C. (2019). Interactive causal discovery in knowledge graphs. In:

International Semantic Web Conference ISWC 2019, vol. 2465, pp. 78–93. CEUR-WS.org

Nagorny, K., Monteiro, P., Barata, J., & Colombo, A. (2017). Big data analysis in smart manufacturing: A review. *International Journal of Communications, Network and System Sciences, 10*, 31–58. https://doi.org/10.4236/ijcns.2017.103003

Neidig, J., Orzelski, A., & Pollmeier, S. (2022) Asset Administration Shell Reading Guide. Accessed: 2023-04-24. https://www.plattform-i40.de/IP/Redaktion/EN/Downloads/Publikation/AAS-ReadingGuide202201.html

Powell, D., Magnanini, M. C., Colledani, M., & Myklebust, O. (2022). Advancing zero defect manufacturing: A state-of-the-art perspective and future research directions. *Computers in Industry, 136*, 103596. https://doi.org/10.1016/j.compind.2021.103596

Psarommatis, F., Fraile, F., & Ameri, F. (2023). Zero defect manufacturing ontology: A preliminary version based on standardized terms. *Computers in Industry, 145*, 103832. https://doi.org/10.1016/j.compind.2022.103832

Psarommatis, F., May, G., Dreyfus, P.-A., & Kiritsis, D. (2019). Zero defect manufacturing: state-of-the-art review, shortcomings and future directions in research. *International Journal of Production Research, 58*(1), 1–17. https://doi.org/10.1080/00207543.2019.1605228

Psarommatis, F., Sousa, J., Mendonça, J. P., & Kiritsis, D. (2022). Zero-defect manufacturing the approach for higher manufacturing sustainability in the era of Industry 4.0: a position paper. *International Journal of Production Research, 60*(1), 73–91. https://doi.org/10.1080/00207543.2021.1987551

Razniewski, S., Nutt, W.: Databases under the partial closed-world assumption: A survey. In: Klan, F., Specht, G., & Gamper, H. (eds.) Proceedings of the 26th GI-Workshop Grundlagen Von Datenbanken, Bozen-Bolzano, Italy, October 21st to 24th, 2014. CEUR Workshop Proceedings, vol. 1313, pp. 59–64 (2014)

Regalia, B., Janowicz, K., & Gao, S. (2016). VOLT: A provenance-producing, transparent SPARQL proxy for the on-demand computation of linked data and its application to spatiotemporally dependent data. In: Sack, H., Blomqvist, E., d'Aquin, M., Ghidini, C., Ponzetto, S.P., Lange, C. (eds.) The Semantic Web. Latest Advances and New Domains - 13th International Conference, ESWC 2016, Heraklion, Crete, Greece, May 29 - June 2, 2016, Proceedings. Lecture Notes in Computer Science, vol. 9678, pp. 523–538. https://doi.org/10.1007/978-3-319-34129-3_32

Ringsquandl, M., Lamparter, S., Lepratti, R., & Kröger, P. (2017). Knowledge fusion of manufacturing operations data using representation learning. In: Lödding, H., Riedel, R., Thoben, K.-D., von Cieminski, G., Kiritsis, D. (eds.) Advances in Production Management Systems. The Path to Intelligent, Collaborative and Sustainable Manufacturing, pp. 302–310. Springer, Cham. https://doi.org/10.1007/978-3-319-66926-7_35

Rowley, J. (2007). The wisdom hierarchy: representations of the DIKW hierarchy. *Journal of Information Science, 33*(2), 163–180. https://doi.org/10.1177/0165551506070706

Saha, S., Usman, Z., Li, W. D., Jones, S., & Shah, N. (2019). Core domain ontology for joining processes to consolidate welding standards. *Robotics and Computer-Integrated Manufacturing, 59*, 417–430. https://doi.org/10.1016/j.rcim.2019.05.010

Sampath Kumar, V. R., Khamis, A., Fiorini, S., Carbonera, J. L., Olivares Alarcos, A., Habib, M., Goncalves, P., Li, H., & Olszewska, J. I. (2019). Ontologies for Industry 4.0. *The Knowledge Engineering Review*. https://doi.org/10.1017/S0269888919000109

Shilov, N., Smirnov, A.V., & Ansari, F. (2020). Ontologies in smart manufacturing: Approaches and research framework. In: 26th Conference of Open Innovations Association, FRUCT 2020, Yaroslavl, Russia, April 20-24, 2020, pp. 408–414. https://doi.org/10.23919/FRUCT48808.2020.9087396

Singh, S., Shehab, E., Higgins, N., Fowler, K., Reynolds, D., Erkoyuncu, J. A., & Gadd, P. (2021). Data management for developing digital twin ontology model. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture, 235*(14), 2323–2337. https://doi.org/10.1177/0954405420978117

Steenwinckel, B., Heyvaert, P., De Paepe, D., Janssens, O., Vanden Hautte, S., Dimou, A., De Turck, F., Van Hoecke, S., & Ongenae, F. (2018). Towards adaptive anomaly detection and root cause analysis by automated extraction of knowledge from risk analyses. In: Proceedings of the 9th International Semantic Sensor Networks Workshop, Co-located with 17th International Semantic Web Conference (ISWC 2018), vol. 2213, pp. 17–31

Svetashova, Y., Zhou, B., Pychynski, T., Schmidt, S., Sure-Vetter, Y., Mikut, R., Kharlamov, E.: Ontology-enhanced machine learning: A Bosch use case of welding quality monitoring. In: Pan, J.Z., Tamma, V., d'Amato, C., Janowicz, K., Fu, B., Polleres, A., Seneviratne, O., Kagal, L. (eds.) The Semantic Web – ISWC 2020, pp. 531–550. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-62466-8_33

Taelman, R., Herwegen, J.V., Sande, M.V., & Verborgh, R. (2018). Comunica: A modular SPARQL query engine for the web. In: Vrandecic, D., Bontcheva, K., Suárez-Figueroa, M.C., Presutti, V., Celino, I., Sabou, M., Kaffee, L., Simperl, E. (eds.) The Semantic Web - ISWC 2018 - 17th International Semantic Web Conference, Monterey, CA, USA, October 8-12, 2018, Proceedings, Part II. Lecture Notes in Computer Science, vol. 11137, pp. 239–255. https://doi.org/10.1007/978-3-030-00668-6_15

The W3C and OWL Working Group: OWL 2 Web Ontology Language Document Overview (Second Edition). Accessed: 2023-04-24 (2012). https://www.w3.org/TR/owl2-overview/

The W3C and RDF Working Group (2014). RDF 1.1 Concepts and Abstract Syntax. Accessed: 2023-04-24. https://www.w3.org/TR/rdf11-concepts/

The W3C and RDF Working Group: RDF Schema. Accessed: 2023-04-24 (2014). https://www.w3.org/TR/rdf-schema/

The W3C SPARQL Working Group: SPARQL 1.1 Query Language. Accessed: 2023-05-12 (2013). https://www.w3.org/TR/sparql11-overview/

Vanderhulst, G., Van Noten, J., & Maes, D. (2023). SPARQLe up your knowledge graphs with on-the-fly computed triples. In Proceedings of The 22nd International Semantic Web Conference (ISWC 2023). CEUR-WS.org

Wang, J., Zhang, W., Shi, Y., Duan, S., Liu, J.: Industrial big data analytics: Challenges, methodologies, and applications. CoRR:1807.01016 (2018)

Weckx, S., Meyers, B., Jordens, J., Robyns, S., Baake, J., Lietaert, P., De Geest, R., & Maes, D. (2022). Development and deployment of a digital twin for monitoring of an adaptive clamping mechanism, used for high performance composite machining. *IET Collaborative Intelligent Manufacturing, 4*, 112–122. https://doi.org/10.1049/cim2.12052

Wuest, T., Weimer, D., Irgens, C., & Thoben, K.-D. (2016). Machine learning in manufacturing: Advantages, challenges, and applications. *Production & Manufacturing Research, 4*, 23–45. https://doi.org/10.1080/21693277.2016.1192517

Xiao, G., Ding, L., Cogrel, B., & Calvanese, D. (2019). Virtual knowledge graphs: An overview of systems and use cases. *Data Intelligence, 1*(3), 201–223. https://doi.org/10.1162/dint_a_00011

Zheng, X., Lu, J., & Kiritsis, D. (2022). The emergence of cognitive digital twin: vision, challenges and opportunities. *International Journal of Production Research, 60*(24), 7610–7632. https://doi.org/10.1080/00207543.2021.2014591