3rd Conference on Production Systems and Logistics

Knowledge Graphs For Data And Knowledge Management In Cyber-Physical Production Systems

Beatriz Bretones Cassoli[1], Nicolas Jourdan[1], Joachim Metternich[1]
[1]TU Darmstadt, Institute of Production Management, Technology and Machine Tools (PTW), Darmstadt, Germany

Abstract

Cyber-physical production systems are constituted of various sub-systems in a production environment, from machines to logistics networks, that are connected and exchange data in real-time. Every sub-system consumes and generates data. This data has the potential to support decision making and optimization of production processes. To extract valuable information from this data, however, different data sources must be consolidated and analyzed. A Knowledge Graph (KG), also known as a semantic network, represents a net of real-world entities, i.e., machines, sensors, processes, or concepts, and illustrates their relationship. KG allows us to encode the knowledge and data context into a human interpretable form and is amenable to automated analysis and inference. This paper presents the potential of KG in manufacturing and proposes a framework for its implementation. The proposed framework should assist practitioners in integrating raw data from multiple data sources in production, developing a suitable data model, creating the knowledge graph, and using it in a graph application. Although the framework is applicable for different purposes, this work illustrates its use for supporting the quality assessment of products in a discrete manufacturing production line.

Keywords

Knowledge Graph; Cyber-Physical Production Systems; Data and Knowledge Management.

1. Introduction

The concept of Cyber-Physical Production Systems (CPPS) applies to autonomous and cooperative sub-systems (e.g. machines, sensors, actuators) that are connected and exchange data across all levels of production [1]. The data generated by CPPS, particularly from Internet of Things (IoT) devices, can be used to monitor manufacturing processes online and support decision-making. A challenge inherent to CPPS is managing this data and extracting useful information which humans can use to improve productivity and prognosis [2]. Especially for applications that use Artificial Intelligence (AI) technologies, the data delivery itself and how this data is provided (in which quality, quantity, and context) to the data analytics applications are decisive for its use [3]. Contextual information in particular allows AI applications to deal with ambiguity, thus improving predictions and their capacity of aiding the decision-making process [4]. Semantic technologies, such as Knowledge Graphs (KG), which capture context and domain knowledge information, can contribute to the explainability and acceptance of AI for realizing flexible manufacturing systems [5]. Early adopters in the manufacturing industry have been using (industrial) KG to support the integration of various data sources and to enable inference and machine processing by providing a formal semantic representation of manufacturing domain knowledge [6]. However, the use of KG in the manufacturing industry is not as disseminated as in other sectors such as finance and biomedical, having mostly prototypical implementations [7]. Guidelines and frameworks for its implementation in the manufacturing industry were not found in the current literature. This paper thus contextualizes KG, provides a novel framework and a practical example of developing an industrial KG for a CPPS. This publication aims at providing guidelines for practitioners and researchers who are not yet familiar with the usage and capabilities of KG. The goal is to break the initial barrier to understanding how to develop a KG, especially for manufacturing data and facilitating the implementation of this semantic technology.

This paper is structured as follows; section 2 presents the theoretical background of KG, graph databases and ontologies. The goal is to situate KG in the broader spectrum of AI technologies. Section 3 introduces the methodology and the proposed framework. Finally, section 4 concretizes the framework by providing a step-by-step development of a KG application having as an example an openly accessible dataset provided by Bosch of four of its assembly lines. The section concludes with an outlook of how the developed KG can be used as part of a Machine Learning (ML) pipeline to predict part quality. The fifth and last section provides future work directions.

2. Theoretical Background

A Knowledge Graph (KG) is a collection of nodes (for each entity), and edges, representing the relationships which connect and relate these entities to the world [8]. Figure 1 provides an example of a basic graph. In this graph, three entities are shown (a person, a station, and a machine), together with the directed relationships that connect them (a person works in a station, and a station has a machine as resource).

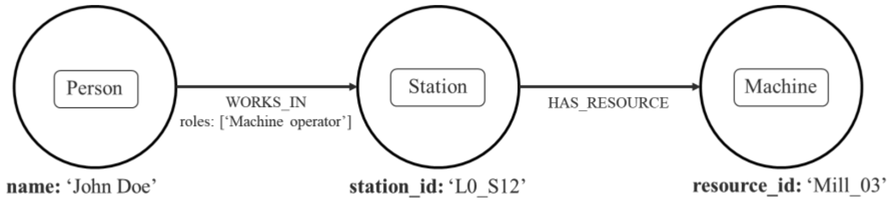


Figure 1: Nodes and relationships

KG are particularly useful because they add a layer of metadata to the data, providing context to it and defining rules for its structure and interpretation [9]. KG can, therefore, represent complex relationships in a domain in both machine friendly and human-readable forms [9]. Graph databases are databases developed specifically to store graph-shaped data and are designed to take advantage of the data connectivity to extract information and generate insights [10]. The uses of KG and of graph databases can typically be divided into three areas: graph statistics (statistic measures about the graph), graph analytics (analysis of the graph data to answer specific questions through queries or graph algorithms), and graph-enhanced ML and AI (application of graph data and analytics results to create or select features for ML models) [11]. In the context of graph-enhanced ML, KG embeddings are techniques used to insert entities and relationships from a KG into continuous vector spaces to allow further mathematical manipulation (e.g. use as input data for ML models) while still preserving the KG structure [12]. KG embeddings are being used in ML pipelines to improve the learning process through the integration of semantic rich features [4].

An ontology represents a domain, its objects, and the formalized relationships between them in a declarative formalism [13]. It is a human-readable text that describes entities in a domain and contains formal axioms (propositions) constraining the interpretation and use of these terms [13]. An ontology is used to share a common understanding of the information structure between people and machines, formalize and thus allow the reuse and analysis/reasoning with domain knowledge, make domain assumptions explicit, and, finally, separate domain knowledge from the operational knowledge [14]. KG and ontologies are part of the broader field of Knowledge Representation and Reasoning, a subfield of AI. Ontologies can be applied to define the KG's underlying structure and guarantee interoperability with other systems and applications from the same domain [15].

Graph technologies are particularly suitable if the relationships within the data are central for analytics and reasoning purposes. Although relational systems (also called SQL databases) are well documented and easier for developers, retrieving relationships out of relational systems is considerably more complex and less efficient than graph systems [16].

Recent literature on graph databases deals with whether graph technology is appropriate for specific use cases. Gosnell et al. [16] propose three questions to decide whether a particular problem is suited for graph technologies:

1. Does the problem need graph data?
   - Understand the shape of the data the problem requires – when understanding the use case, is it natural to represent the data in a graph shape? What is the type of information to be extracted from the data?
2. Do relationships between entities in the data help understand and solve the problem?
   - Are the relationships between entities in the data relevant to explain and clarify the problem?
3. What is the purpose of exploring the relationships in the data?
   - Is the goal to generate a report (classical business intelligence)? Perform research on graph-structured data? Provide a service to an end-user (retrieval)? Use as part of an ML pipeline?

Nowadays, KG are being applied in various industries in application areas such as question answering, recommender systems, information retrieval, and feature engineering [17]. KG are also being adopted in the manufacturing industry, mostly in the form of prototypes. Zhou et al. use a KG-based approach for resource allocation in a discrete manufacturing workshop [18]. To achieve a self-configurable manufacturing process, Zheng et al. introduce a KG-based multi-agent reinforcement learning method [19]. Zhao et al. developed a service platform based on KG for resource allocation and scheduling of manufacturing orders [20]. Additional use cases include the use of KG for creating the digital twin of a building used in applications for building management and services, risk management in engineering projects, process monitoring, and machine service operations planning [6, 21].

3. Methodology and proposed framework

This section introduces a framework for developing and using KG. The framework is based on the literature presented in the theoretical background section, mostly focused on the use of graph databases. Mainly the work from Robinson et al. [8], Barrasa et al. [9], Gosnell et al. [16], Sequeda and Lassila [22], combined with practical implementation experience from the authors, are translated into a general guideline for the adoption of KG. Figure 2 presents the framework consisting of five phases. The three questions proposed by Gosnell et al. [16] and introduced in section 2 are to be answered before the first phase of the framework. Next, each phase will be explained from a theoretical point of view.

1. Problem understanding and identification of data sources. As a natural first step, the problem understanding aims at having a clear goal for the KG application. In this step, the purpose of exploring the relationships in the data should be differentiated between generating a business report (calculating key performance indicators), providing a service to an end-user (retrieval system), or being used as part of an ML pipeline. The identification of data sources and their relationships aids the data modelling. We suggest conducting a workshop with stakeholders for the problem understanding step.

2. Data modelling. The data model represents the domain's entities and relationships and establishes the graph structure for receiving data in the subsequent steps. This step is paramount and should be guided by the goal defined in the problem understanding. When querying a graph, we talk about "traversing", which means going from one point (or node) of the graph to another. So how the entities and their relationships are represented in the data model will affect the traversals and, therefore, the efficiency of information retrieval and how the domain is represented in the graph. One of the biggest differences and advantages from graph data storage is that the conceptual data model (how the entities relate to each other in the domain, for example the one portrayed in Figure 1) translates directly to how the data is physically stored in the graph database [16]. We identify two main ways of defining the data model: through an existing or newly developed ontology stored, e.g., in a Resource Description Framework (RDF) file (**2.1**) or through graph modelling frameworks, such as the Labeled Property Graph (LPG) [8] or the Graph Schema Language (GSL) [16] (**2.2**). Independently of the chosen approach, each entity or object from the domain is assigned a specific node label. Node labels are used for entities from the same class, which will share the same type of relationships and properties. Similarly, relationships' labels define their type and properties. This step can likewise be conducted in a form of a workshop with stakeholders and using visual tools, e.g., whiteboards, to draw the entities and their relationships and derive the data model.

3. Data preparation and integration. In most cases, the data sources come from relational databases or are stored in a relational data format, such as tables in CSV files or as unstructured data in NoSQL databases. The question in this task is how to prepare the data and integrate the various data sources to be loaded into the specified graph data model defined in step 2. The data preparation can be as simple as creating new CSV files that translate the relationship within the data that will be then loaded into the KG. For more complex scenarios, there are the so-called "mappings". A mapping is "a function representing the relationship from a source data model to a target data model" [22]. Mappings are usually formulated as rules (IF (condition) THEN (conclusion)) and indicate how a source (e.g., relational database) can be represented in the target (a KG). These functions or rules are applied to fill the KG with data originating from relational or NoSQL systems. At this stage, the graph database (such as Neo4j, DataStax, Amazon Neptune, and Cassandra) should be chosen, as it will influence the necessary changes in the original data for its integration.

4. Knowledge Graph creation. Once the data sources are identified and prepared for integration, the next step is to load the data into the selected graph database. As mentioned before, different procedures depending on the graph database exist for conducting this task. The basic idea is to load the entities and relationships for the data as defined in the graph data model in step two. At the end of this step, entities and relationships from the data should be represented and connected in the KG. The KG can then be used to develop the application in the fifth step.

5. Application development. The fifth and last step of the framework, application development, summarizes the efforts for creating an application that will ultimately use the KG. These uses can be assigned to three main categories: Generating business reports (**5.1**), querying, and retrieving information for an end-user (**5.2**), or as part of an ML pipeline for either producing graph features or facilitating the selection of features, e.g., for regression or classification tasks (**5.3**).

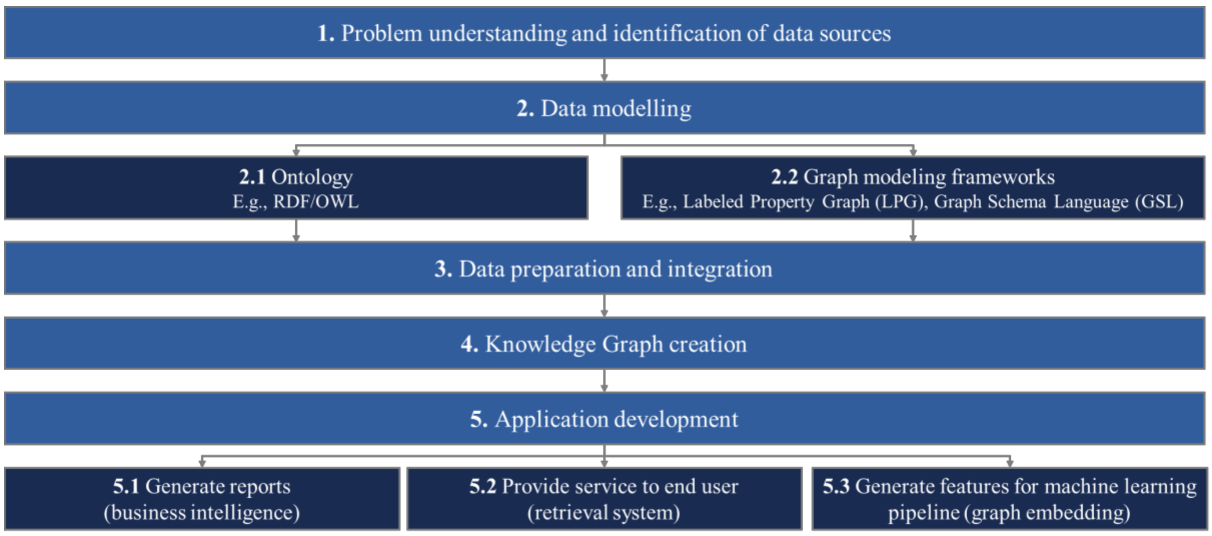

1. Problem understanding and identification of data sources
2. Data modelling
2.1 Ontology
E.g., RDF/OWL
2.2 Graph modeling frameworks
E.g., Labeled Property Graph (LPG), Graph Schema Language (GSL)
3. Data preparation and integration
4. Knowledge Graph creation
5. Application development
5.1 Generate reports (business intelligence)
5.2 Provide service to end user (retrieval system)
5.3 Generate features for machine learning pipeline (graph embedding)

Figure 2: Proposed framework

4. Framework Application: Case Study Product 360

A well-known use case for KG in enterprises is the so-called "Customer 360" applications. Customer 360 are considered data products that consolidate and connect various data sources about the customer, creating a common approach to expose this information for use [22]. The 360-degree view of customer's data allows using the available and most relevant information about each customer to improve and customize service offerings, e.g., products recommendations [23]. Similarly, we propose a 360-degree view of the product. The idea is to gather in a KG all available and relevant information about a manufactured product to support additional analysis, e.g., for creating graph features for an ML pipeline to predict which part will fail quality control (predictive quality). In this work, steps one to four (from problem understanding to KG creation) are presented to clarify how to use the framework. The fifth step (application development) gives an outline and future work directions for the predictive quality case. For this application development, the graph database Neo4j through the Neo4j Desktop application was used.

1. Problem understanding and identification of data sources. The data used to develop the product 360 KG is from the publicly available datasets "Bosch Production Line Performance" published in Kaggle in 2016[1]. The datasets contain measurements of parts as they move through Bosch's assembly lines and are divided into three types of features plus a response feature: numerical, categorical, date (timestamps) and binary labels indicating whether the part succeeded (Response = 0) or failed (Response = 1) quality control. The datasets are further divided into training and testing sets for each feature type. The training and testing sets have 1.184.687 and 1.183.748 samples, respectively. Further, the data contains 968 numerical features, 2.140 categorical features and 1.156 date features. This results in a combined dataset of 14,3 Gigabyte. Besides the challenge of working with large datasets, the ground truth is highly imbalanced (approximately one in every 125 products are defective), and several product variants are represented in the dataset (identified by the 7.148 unique flow paths) [24]. The identified problem is then predicting which parts will fail quality control. Therefore, the data sources are the training and testing datasets: train and test numerical, train and test date, and train and test categorical datasets.

2. Data modelling. The data model should aid the task of identifying which products pass or fail the quality control. The goal is to generate a KG with all information about that product. In this case, it means representing in the graph the numerical and categorical features related to each station and line, the sequence of stations and lines, and the data features in the form of timestamps. Figure 3 shows the data model developed using the LPG framework.

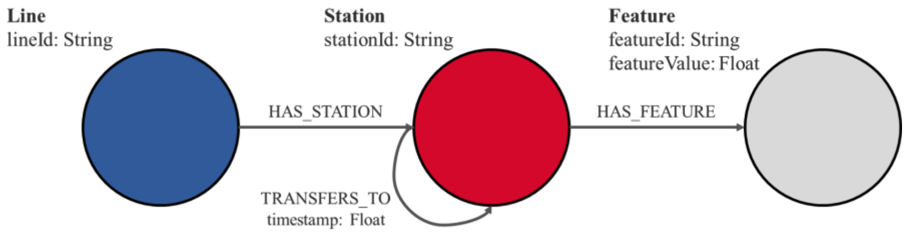


Figure 3: LPG data model for product 360 knowledge graph

In the figure, we identify three types of entities: Line, Station, and Feature. Lines are identified by the "lineId", which is saved as a String. Similarly, stations are identified by a "stationId" stored as a String. Features have two properties: the identifier "featureId" saved as String, and the "featureValue" stored as a Float and which holds the actual feature value (recordings from sensors or categorical data). The data model further defines three types of relationships: HAS_STATION, HAS_FEATURE, and TRANSFERS_TO. HAS_STATION is used to describe which stations the line comprises. HAS_FEATURE indicates in which

[1] Available at: https://www.kaggle.com/c/bosch-production-line-performance/data

station the feature was recorded. Finally, TRANSFERS_TO captures the sequence of stations and lines the product passed through during the assembly process. The "TRANSFERS_TO" relationship has a property called "timestamp", which records the timestamp of the estimated time the transfer between stations occurred. This data model serves as the basis for preparing the data and loading it in the Neo4j graph database. Alternatively, an ontology can be used as the data model and imported in Neo4j using the neosemantics (n10s) plugin. Figure 4 shows an exemplary ontology created using WebProtègè[2] and exported as a turtle (.ttl) file.

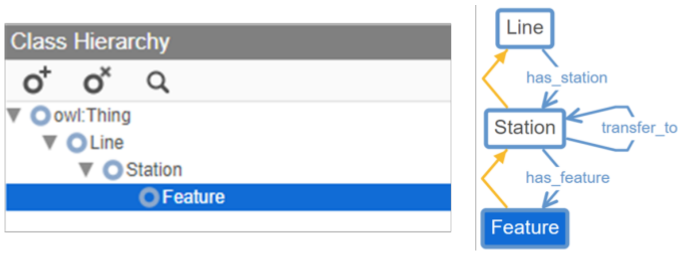


Figure 4: Ontology in WebProtègè

3. Data preparation and integration. The data preparation step consists of reshaping the Bosch production line dataset to be loaded into the graph database. Table 1 presents an excerpt of the first five rows and five columns of the raw numerical train dataset. This dataset was used to capture the entities for the Line, Station and Feature classes and the "featureValue" for the numerical features.

Table 1: Excerpt of Bosch's production line numerical train dataset

Index	Id	L0_S0_F0	L0_S0_F2	L0_S0_F4	L0_S0_F6
0	4	0.03	-0.034	-0.197	-0.179
1	6	nan	nan	nan	nan
2	7	0.088	0.086	0.003	-0.052
3	9	-0.036	-0.064	0.294	0.33
4	11	-0.055	-0.086	0.294	0.33

Table 2 depicts the result of shaping the data for the first product in the dataset, product Id 4. For manipulating the raw data, the Python library pandas was used. The pandas data frame was saved into a CSV file which will be used in the subsequent step to create the KG. This table allows identifying all entities, their respective Ids, and the relationships "HAS_STATION" and "HAS_FEATURE".

Table 2: Entities and feature values for product Id 4

Index	lineId	stationId	featureId	featureValue
0	L0	L0_S0	L0_S0_F0	0.03
1	L0	L0_S0	L0_S0_F2	-0.034
2	L0	L0_S0	L0_S0_F4	-0.197
3	L0	L0_S0	L0_S0_F6	-0.179
4	L0	L0_S0	L0_S0_F8	0.118

[2] Available at: https://webprotege.stanford.edu/

The relationship “TRANSFERS_TO” was obtained from the date train dataset. The timestamps in this dataset were used to determine the station sequence and the estimated timestamp of transfer between stations. Table 3 shows the resulting table with the station of origin (stationId_s, s for the subject) and the destination (stationId_o, o for the object), and the timestamp listed in the column “transfers_to”.

Table 3: TRASNFERS_TO relationship

Index	stationId_s	transfers_to	stationId_o
0	L0_S0	82.24	L0_S1
1	L0_S1	82.24	L0_S2
2	L0_S2	82.24	L0_S4
3	L0_S4	82.26	L0_S7
4	L0_S7	82.26	L0_S8

4. Knowledge Graph creation. The CSV files were used to create and save the data into the KG in Neo4j. Cypher commands to add the data to the graph database were written in the Neo4j Browser application. An excerpt of the commands used to create the nodes (entities) and relationships are listed in Table 4.

Table 4: Excerpt of Cypher commands for creating nodes and relationships

Command explanation	Cypher command
Create the nodes for lines, stations and features from product Id 4’s CSV file	LOAD CSV FROM 'file:///productid4.csv' AS row WITH row[1] AS lineId, row[2] AS stationId, row[3] AS featureId, toFloat(row[4]) AS featureValue MERGE (l:Line {lineId: lineId}) MERGE (s:Station {stationId: stationId}) MERGE (f:Feature {featureId: featureId}) SET f.featureValue = featureValue RETURN count(l);
Create relationships “HAS_FEATURE” from product Id 4’s CSV file	LOAD CSV WITH HEADERS FROM 'file:///productid4.csv' AS row WITH row.lineId AS lineId, row.stationId AS stationId, row.featureId AS featureId MATCH (l:Line {lineId: lineId}) MATCH (s:Station {stationId: stationId}) MATCH (f:Feature {featureId: featureId}) MERGE (s)-[rel:HAS_FEATURE]->(f) RETURN count(rel);
Visualize “TRANSFERS_TO” relationship and numerical features	MATCH (s:Station)-[rel:TRANSFERS_TO]->(s:Station) MATCH (s:Station)-[rel2:HAS_FEATURE]->(f:Feature) RETURN s, f, rel, rel2;

The result from the last query can be visualized in Figure 5. Figure 6 provides a detailed view of the "TRANSFER_TO" property, indicating the transfer's timestamp to the next station occurred.

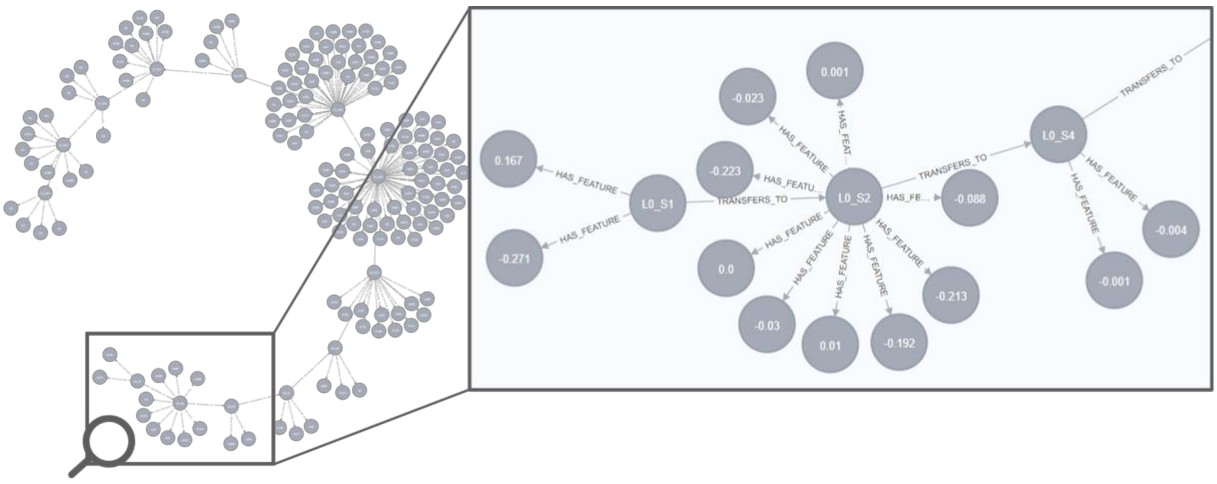
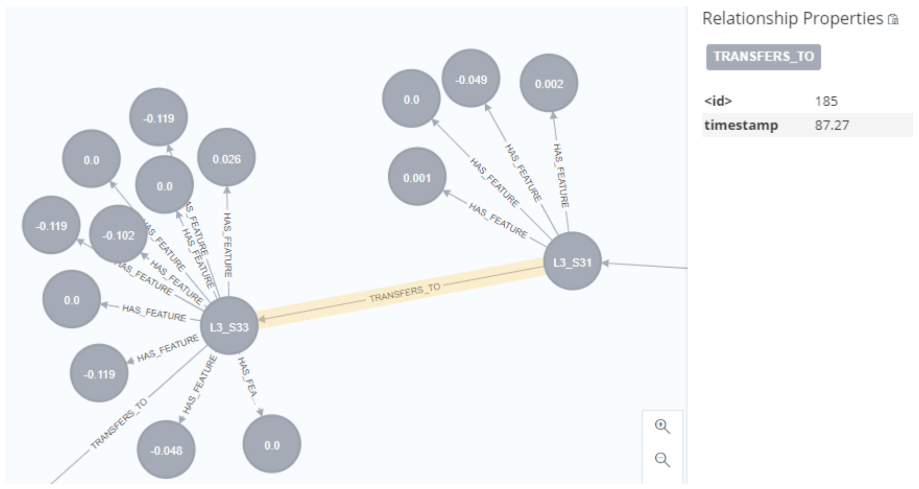


Figure 5: Knowledge Graph in Neo4j for product Id 4



Relationship Properties

TRANSFERS_TO

<id>	185
timestamp	87.27

Figure 6: Relationship property for product Id 4

The same procedure used to create the KG for product Id 4 is applied to the remaining products. The result is one KG per product.

Outlook: 5. Application development. The KG created in step four can now be used for various applications, from generating key performance indicators for each product and the assembly line and part of an ML pipeline for graph classification. Here we outline a predictive quality application currently being developed and seen as future work. The predictive quality application consists of conducting a graph classification task. First, it is necessary to generate low-dimensional vector representations of the KG created in step four through KG embeddings. For that, GraphSAGE[3] can be used for products from the same variant and, therefore, present the same stations and lines sequence. The graph embeddings are used for training a classifier which should then predict whether the assembled product has Response = 0 (succeeded the quality control) or Response = 1 (failed the quality control).

5. Conclusion

This paper presented step-by-step guidance to create a KG from relational data sources. It starts with framing the problem as a graph to develop a data model and load it into the graph database. The guideline should assist practitioners and researchers who want to use semantic technologies as a source of information. Either

[3] Available at: http://snap.stanford.edu/graphsage/

for calculating key performance indicators and generating business reports by ingesting data into the KG and performing graph analytics or as a retrieval system for providing a service to an end-user or as part of an ML pipeline. Future work includes providing more insights into the application development and exploring the potential to enrich feature expressiveness for ML pipelines.

Acknowledgments

The research leading to these results has received funding from the European Commission's H2020 program under grant agreement number 958357 (InterQ).

References

[1] Francalanza, E., Borg, J., Constantinescu, C., 2017. A knowledge-based tool for designing cyber physical production systems. Computers in Industry 84, 39–58.

[2] Saqlain, M., Piao, M., Shim, Y., Lee, J.Y., 2019. Framework of an IoT-based industrial data management for smart manufacturing. Journal of Sensor and Actuator Networks 8 (2), 25.

[3] Janssen, M., Brous, P., Estevez, E., Barbosa, L.S., Janowski, T., 2020. Data governance: Organizing data for trustworthy Artificial Intelligence. Government Information Quarterly 37 (3), 101493.

[4] Needham, M., Hodler, A.E., 2019. Graph Algorithms: Practical Examples in Apache Spark and Neo4j. O'Reilly Media, United States of America.

[5] Bretones Cassoli, B; Jourdan, N; Nguyen, P. H.; Sen, S.; Garcia-Ceja, E.; Metternich, J., 2021. Frameworks for data-driven quality management in cyber-physical systems for manufacturing: A systematic review, 15th CIRP Conference on Intelligent Computation in Manufacturing Engineering - CIRP ICME '21 Virtual Conference, 14-16 July 2021, Procedia CIRP, Elsevier, ISSN: 2212-8271, (in print).

[6] Hubauer, T., Lamparter, S., Haase, P., Herzig, D.M., 2018. Use Cases of the Industrial Knowledge Graph at Siemens, in: International Semantic Web Conference (P&D/Industry/BlueSky).

[7] Chen, X., Jia, S., Xiang, Y., 2020. A review: Knowledge reasoning over knowledge graph. Expert Systems with Applications 141, 112948.

[8] Robinson, I., Webber, J., Eifrem, E., 2015. Graph Databases: New Opportunities for Connected Data. O'Reilly Media, Inc., United States of America.

[9] Barrasa, J., Hodler, A.M., Webber, J., 2021. Knowledge Graphs: Data in Context for Responsive Businesses. O'Reilly Media, United States of America, 87 pp.

[10] Janev, V., Graux, D., Jabeen, H., Sallinger, E., 2020. Knowledge Graphs and Big Data Processing 12072.

[11] Hodler, A., Needham, M., 2021. Graph Data Science (GDS) For Dummies®, Neo4j Special Edition. John Wiley & Sons, Inc., United States of America.

[12] Wang, Q., Mao, Z., Wang, B., Guo, L., 2017. Knowledge Graph Embedding: A Survey of Approaches and Applications. IEEE Trans. Knowl. Data Eng. 29 (12), 2724–2743.

[13] Gruber, T.R., 1993. A translation approach to portable ontology specifications. Knowledge acquisition 5 (2), 199–220.

[14] Noy, N.F., McGuinness, D.L., 2001. Ontology development 101: A guide to creating your first ontology. Stanford knowledge systems laboratory technical report KSL-01-05.

[15] Ko, H., Witherell, P., Lu, Y., Kim, S., Rosen, D.W., 2021. Machine learning and knowledge graph based design rule construction for additive manufacturing. Additive Manufacturing 37, 101620.

[16] Gosnell, D., Broecheler, M., 2020. The Practitioner's Guide to Graph Data: Applying Graph Thinking and Graph Technologies to Solve Complex Problems. " O'Reilly Media, Inc.".

[17] Zou, X., 2020. A survey on application of knowledge graph, in: Journal of Physics: Conference Series. IOP Publishing, p. 12016.

[18] Zhou, B., Bao, J., Li, J., Lu, Y., Liu, T., Zhang, Q., 2021. A novel knowledge graph-based optimization approach for resource allocation in discrete manufacturing workshops. Robotics and Computer-Integrated Manufacturing 71, 102160.

[19] Zheng, P., Xia, L., Li, C., Li, X., Liu, B., 2021. Towards Self-X cognitive manufacturing network: An industrial knowledge graph-based multi-agent reinforcement learning approach. Journal of Manufacturing Systems 61, 16–26.

[20] Zhao, Y., Liu, Q., Xu, W., 2017. Open Industrial Knowledge Graph Development for Intelligent Manufacturing Service Matchmaking, in: 2017 International Conference on Industrial Informatics - Computing Technology, Intelligent Technology, Industrial Information Integration (ICIICII). 2017 International Conference on Industrial Informatics - Computing Technology, Intelligent Technology, Industrial Information Integration (ICIICII), Wuhan. 02.12.2017 - 03.12.2017. IEEE, pp. 194–198.

[21] Fensel, D., Simsek, U., Angele, K. (Hg.), 2020. Knowledge Graphs: Methodology, Tools and Selected Use Cases. SpringerLink (Ed.), Austria.

[22] Sequeda, J., Lassila, O., 2021. Designing and Building Enterprise Knowledge Graphs. Synthesis Lectures on Data, Semantics, and Knowledge 11 (1), 1–165.

[23] Hristoski, I., Dimovski, T. (Hg.), 2020. Graph Database Modeling of a 360-Degree E-Customer View in B2C E-Commerce. International May Conference on Strategic Management IMCSM20. Bor, Serbia, September 25 - 27.

[24] Mangal, A., Kumar, N. (Hg.), 2016. Using big data to enhance the Bosch production line performance: A Kaggle challenge. IEEE International Conference on Big Data.

Beatriz Bretones Cassoli, M. Sc. has been a researcher and PhD student at the Institute of Production Management, Technology and Machine Tools (PTW) at the Darmstadt University of Technology since 2019. Her research interests include using semantic technologies for manufacturing use cases, particularly Knowledge Graphs.

Nicolas Jourdan, M. Sc. Has been a researcher and PhD student at the Institute of Production Management, Technology and Machine Tools (PTW) at the Darmstadt University of Technology since 2020. His research interests include the robustness analysis of Machine Learning models for manufacturing applications.

Prof. Dr.-Ing. Joachim Metternich has been the head of the Institute for Production Management, Technology and Machine Tools (PTW) at the Darmstadt University of Technology since 2012. In addition, Prof. Metternich is the spokesman for the Center of Excellence for Medium-Sized Businesses in Darmstadt and president of the International Association of Learning Factories.