Semantic Integration of Bosch Manufacturing Data Using Virtual Knowledge Graphs

Elem Güzel Kalaycı[1,2], Irlan Grangel González[3], Felix Lösch[3], Guohui Xiao[1,4(✉)], Anees ul-Mehdi[3], Evgeny Kharlamov[5,6], and Diego Calvanese[1,4,7]

[1] Free University of Bozen-Bolzano, 39100, Bolzano, Italy *lastname*@inf.unibz.it
[2] Robert Bosch GmbH, Corporate Research, 70465, Stuttgart, Germany *name.lastname*@de.bosch.com
[3] Virtual Vehicle Research GmbH, 8010, Graz, Austria elem.guezelkalayci@v2c2.at
[4] Ontopic S.r.L., Bolzano, 39100, Italy *name.lastname*@ontopic.biz
[5] Robert Bosch GmbH, Bosch Center for Artificial Intelligence, 71272 Renningen, Germany evgeny.kharlamov@de.bosch.com
[6] University of Oslo, Blindern N-0316 Oslo, Norway evgeny.kharlamov@ifi.uio.no
[7] Umeå University, 90187 Umeå, Sweden diego.calvanese@umu.se

Abstract. Analyses of products during manufacturing are essential to guarantee their quality. In complex industrial settings, such analyses require to use data coming from many different and highly heterogeneous machines, and thus are affected by the data integration challenge. In this work, we show how this challenge can be addressed by relying on semantic data integration, following the Virtual Knowledge Graph approach. For this purpose, we propose the SIB Framework, in which we semantically integrate Bosch manufacturing data, and more specifically the data necessary for the analysis of the Surface Mounting Process (SMT) pipeline. In order to experiment with our framework, we have developed an ontology for SMT manufacturing data, and a set of mappings that connect the ontology to data coming from a Bosch plant. We have evaluated SIB using a catalog of product quality analysis tasks that we have encoded as SPARQL queries. The results we have obtained are promising, both with respect to expressivity (i.e., the ability to capture through queries relevant analysis tasks) and with respect to performance.

1 Introduction

The digitization trend in manufacturing industry, known as Industry 4.0, leads to a huge growth of volume and complexity of data generated by machines involved in manufacturing processes. These data become an asset of key relevance for enhancing the efficiency and efficacy of manufacturing. However, unlocking the potential of these data is a major challenge for many organizations. Indeed, often the data naturally reside in silos, which are not interconnected, but which contain semantically related data, possibly with redundant and inconsistent information. As a result, the effective use of data demands data integration, which includes cleaning, de-duplication, and semantic homogenization. As evaluated at Bosch, the integration effort needed for data integration is approximately 70-80%, in comparison to 20-30% required for data analysis [16], where the integration is mainly hampered by ad-hoc and manual approaches that are prone to

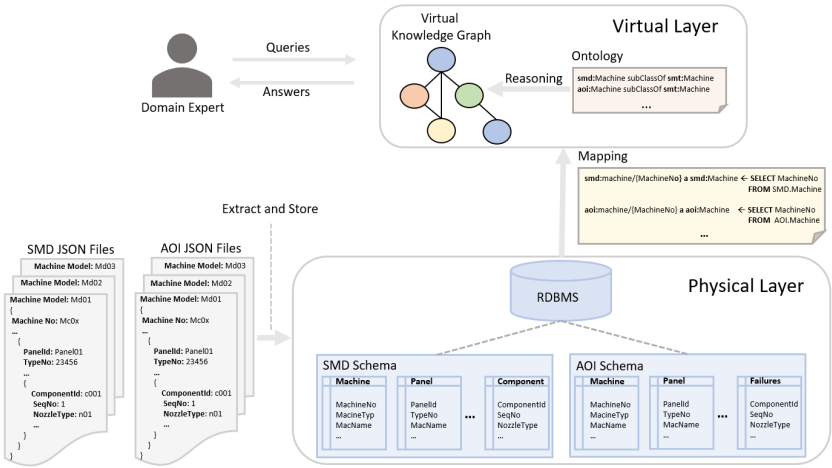

Fig. 1. Virtual Knowledge Graphs approach exemplified over Bosch SMT scenario.

data quality issues. In particular, this affects the reproducibility of analytical results as well as the consequent decision making [9,16,17].

A Bosch plant located in Salzgitter, Germany, that produces electronic control units, is not an exception in the digitization trend and the integration challenge [16]. Indeed, the product quality analysis that is performed at the plants requires integration of vast amounts of heterogeneous data. For instance, failure detection for Surface Mounting Process (SMT) fundamentally relies on the integration and analysis of data generated by the machines deployed in the different phases of the process. Such machines, e.g., for *placing electronic components* (SMD) and for *automated optical inspection* (AOI) of solder joints, usually come from different suppliers and they rely on distinct formats and schemata for managing the same data across the process. Hence, the raw, non-integrated data does not give a coherent view of the whole SMT process and hampers analysis of the manufactured products.

To address this problem, we propose to adopt an approach for semantic data integration and access based on virtual knowledge graphs (VKG)[8] [20,21,5], which we illustrate in Figure 1. In such an approach, we use an ontology that exposes, in terms of a VKG, a conceptual view of the concepts and properties of relevance for the SMT manufacturing process. The log data of the process are extracted directly from JSON files generated by the different machines, and are loaded into a PostgreSQL database, without further conversion from JSON into the relational format. From there the data is connected to the ontology by making use of semantic mappings [1].

The strength of this approach comes on the one hand from the domain knowledge encoded in the ontology. This knowledge is used to enrich answers to user queries describing product analysis tasks. Another key advantage is the use of semantically rich VKG mappings. These bridge the impedance mismatch between the data layer and the ontology layer, by relying on the template-based mechanism of R2RML [4] to construct

[8] Also known in the literature as Ontology-based Data Access/Integration (OBDA/I).

knowledge graph (KG) IRIs out of database values. Moreover, the mappings provide a solution to the integration challenge, since semantically homogeneous information coming from different log files, which use syntactically different representations, can be reconciled at the level of the KG. This makes it easy to query the overall data assets in an integrated way, by exploiting the semantics of the extracted information. For example, consider various types of machine failures, encoded in the different log files through different *magic numbers*, i.e., codes with a specific meaning that have to be known and interpreted by the Bosch engineers. Once the machine failures of the same type are reconciled in a single ontology class, Bosch engineers can effectively understand the meaning of failures, since the mappings guarantee that each failure data item (virtually) populates only the appropriate failure class. This is of crucial importance to allow engineers take the right decisions in the SMT process. In general, the approach enables to encode different product analysis tasks that are of importance in the Bosch SMT manufacturing process, by means of suitable SPARQL queries over the domain ontology. Notably, such queries make use of ontology terms to refer to the relevant information assets, and thus are very close to the natural language formulation of the analysis tasks, which in turn makes it easy for Bosch engineers to formulate them. We can then obtain the respective analysis data coming from the process logs, by simply executing such queries over the underlying database via a VKG engine.

The above solution has been implemented at Bosch in the VKG-based data integration framework called SIB (for Semantic Integration at Bosch) and deployed at Bosch. The purpose of this deployment has been to evaluate the feasibility of using semantic technologies and data integration based on them for supporting product quality analysis. More specifically, towards the development of SIB, we have overcome the technical challenges posed by semantic data integration, and we have provided the following contributions: *(i)* We developed the *SMT Ontology*, which is an OWL 2 QL ontology capturing the concepts and properties that are relevant for SMT manufacturing, together with important domain knowledge. This ontology is the basis of the VKG approach to integration for SMT manufacturing. *(ii)* We built a mapping layer that semantically connects the SMT ontology to a PostgreSQL database. Such a database in turn collects the relevant log data from JSON files produced by the machine components in the manufacturing pipeline. *(iii)* We encoded relevant product analysis tasks into a catalog of SPARQL queries formulated over the SMT Ontology. For processing such queries, taking into account both the SMT Ontology and mappings, we rely on the state-of-the-art VKG engine Ontop [2] .

Moreover, we carried out an experimental evaluation of the SIB approach, with a two-fold aim. First, we wanted to assess its effectiveness in addressing significant product analysis tasks by relying on the answers returned by the SPARQL queries encoding such tasks. Second, we were interested in understanding the performance in query execution, and whether such performance is compatible with the requirements coming from product analysis. As a baseline for the evaluation, we have used SANSA-DL (where "DL" stands for Data-Lake), which is an alternative implementation of semantic integration based on SANSA [14], a distributed framework for scalable RDF data processing in turn based on Apache Spark. Similarly to Ontop, SANSA-DL supports SPARQL queries over an ontology connected via mappings to a data source. However, the data source consists of *parquet* files, loaded from the JSON data via a Scala script.

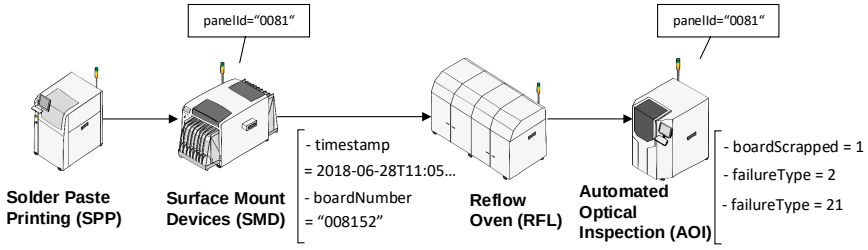

Fig. 2. Surface Mounting Process pipeline. The SMT process comprises four phases: SPP, SMD, RFL, and AOI. The machines, usually by different suppliers, rely on distinct formats and schemas for managing the same data across the process pipeline.

Our evaluation showed that, in this real-world use-case in industrial manufacturing, semantic data integration based on VKGs as realized in SIB is feasible, both from the point of view of semantics/expressiveness, and from the point of view of performance. Moreover, our results indicate that the VKG system Ontop adopted in SIB outperforms an alternative implementation based on Spark.

The rest of the paper is organized as follows. Section 2 describes the SMT manufacturing process and the challenges for integrating data generated by the different machines used for manufacturing. Section 3 describes the components of the VKG approach and its application to the SMT use case. In Section 4, we outline the evaluation results. In Section 5 we discuss the lessons learned, and we conclude the paper with an outlook and future work in Section 6.

2 Bosch Use Case: SMT Manufacturing Process

In this section we describe the SMT manufacturing process, the analytics it typically requires, and the challenges in implementing this analytics, which will be addressed further in the paper.

The SMT Process involves four main phases (cf. Figure 2): *(1) Solder Paste Printing*: This phase consists of pasting solder paste on print circuit boards (PCBs), and is conducted by a Solder Paste Printing Machine; *(2) Surface Mounting*: This is the phase where electronic components are actually mounted on PCBs in a Surface Mounting Device (SMD); *(3) Heating*: To solder the mounted components properly, in this phase the boards are heated inside a Reflow Oven; and *(4) Automated Optical Inspection.* In the final phase, the boards are inspected by an AOI machine to find out whether during the previous phases any failure occurred, such as component misplacement or bad soldering. When the whole process is completed, the system generates log files, which contain two types of data: the *placement logs* by the SMD machine, include information on which component is mounted on which board; the *failure logs* generated by the AOI machine, comprise information where at a board and with what component a failure is encountered.

Product Analysis Tasks. Several product analysis tasks are carried out by Bosch engineers in order to be able to assess and evaluate the quality of the manufactured PCBs, and such tasks require in an essential way to access in an integrated way the data produced during the SMT process. A typical analytical request for information in this domain may look as follows: "For the *panels* processed in a given *time frame*, retrieve the number of *failures* associated with *scrapped boards* and grouped by *failure types*." To retrieve this information, data that reside in the SMD and AOI machines need to be semantically integrated. Referring again to Figure 2, the SMD machine processes the panel, the board, and the timestamp information, i.e., the time at which a given panel was processed, while the AOI machine generates information to check whether a board was scrapped or not. The meaning of this information is encoded in numbers: a value of '1' when the board is scrapped and '0' when not. In addition, the AOI machine contains information about the different failure types, e.g., number 2 representing "false call" and 21 representing "misplaced component". The meaning of this information encoded in numbers is only available in internal documents and in the head of Bosch engineers. The panel provides the connection between the data generated by the machines. With this setup, the codes associated to various kinds of components need to be combined to create the ids for different types of objects, and this in turn complicates semantic interoperability between SMD and AOI data.

Technical Challenges. In this work, we focus on addressing the above challenges by means of semantic data integration. Specifically, we show how a VKG-based approach enables the semantic integration of the SMD and AOI data, and how the requests of domain experts can be answered on top of the semantically integrated data. Towards the objective of applying the VKG approach to the Bosch use case, we had to address the following technical challenges: **C1:** integration of syntactically heterogeneous machine log data involving semantic conflicts; **C2:** development of the SMT Domain Ontology capturing relevant domain knowledge; **C3:** transformation of product analysis questions into a catalog of SPARQL queries and evaluation of their effectiveness for product quality analysis; and **C4:** evaluation of the efficiency of the VKG approach in the execution of these queries, which are relevant to the Bosch use case. We discuss in this paper, how these challenges have been addressed and solved.

Related Work. The VKG-based integration approach has already been successfully applied in industrial settings [7]. These include, among many, the case of manufacturing [18], of assembling complex systems at Festo [6], of turbine diagnostics and other tasks at Siemens [8,11,17,10,12], and of exploration and analyses of geological data at Equinor [9,13]. In several of these, Ontop has been adopted as one of the key components. In this work we continue this line of applied research and bring semantic technologies into the Bosch corporate environment by adapting them to a concrete scenario.

3 Application of SIB at Bosch

In this section, we present the general VKG approach for semantically integrating data and its specific application at Bosch in SIB for the semantic integration of manufacturing

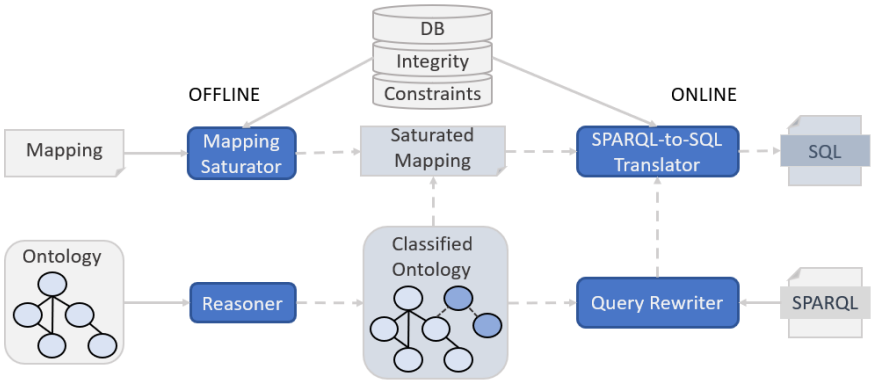

Fig. 3. The Ontop Workflow for query translation. The figure depicts the SPARQL-to-SQL query translation workflow of Ontop.

data of the SMT process. SIB comprises the following components: *1)* data sources of the SMT manufacturing process; *2)* the SMT Ontology, i.e, a semantic model of the SMT domain; *3)* mappings between the data sources and the SMT Ontology; *4)* queries for expressing the requirements of the domain; and *5)* an implementation of the VKG approach using Ontop to semantically answer the queries while integrating the SMD and AOI data sources.

3.1 Overview of the SIB Framework

We gave a general overview of the Virtual Knowledge Graph (VKG) paradigm in Figure 1 of Section 1. In this work we rely on the state-of-the-art VKG framework Ontop for developing SIB. Ontop computes answers end-user SPARQL queries by translating them into SQL queries, and delegating the execution of the translated SQL queries to the original data sources. We remark that with the VKG approach, there is no need to materialize into a KG all the facts entailed by the ontology. As seen in Figure 3, the workflow of Ontop can be divided into an off-line and an online stage. As the first step at the *off-line stage*, Ontop loads the OWL 2 QL ontology and classifies it via the built-in reasoner, resulting in a directed acyclic graph stored in memory that represents the complete hierarchy of concepts and that of properties. In the second step, Ontop constructs a so-called *saturated mapping*, by compiling the concept and property hierarchies into the original VKG mapping. This aspect is important also in SIB, since the domain knowledge encoded in the ontology allows for simplifying the design of the mapping layer. During the offline stage, Ontop also optimizes the saturated mapping by applying structural and semantic query optimization [19].

During the *online stage*, Ontop takes a SPARQL query and translates it into SQL by using the saturated mapping. To do so, it applies a series of transformations that we briefly summarize here [2,22]: *(i)* it rewrites the SPARQL query w.r.t. the ontology; *(ii)* it translates the rewritten SPARQL query into an algebraic tree represented in an internal format; *(iii)* it unfolds the algebraic tree w.r.t. the saturated mapping, by replacing the

Table 1. SMT Data. The main tables in the SMT schema are given on the left. The most frequently requested tables are given on the right, with a sample of data.

SMD Tables
smd_event
smd_location
smd_panel
smd_components

AOI Tables
aoi_event
aoi_location
aoi_panel
aoi_failures

smd_panel

panelId	boardNo	machineName	processedTS	location
p01	b01	SMD Machine 1	24-04-2020	mes01

smd_components

panelId	boardNo	headId	nozzleId	turnNo	pickSeqNo	placeSeqNo
p01	b01	h01	n01	2	1	3

aoi_failures

panelId	boardNo	refDesignator	windowNo	cPinNo	failureCode
p01	b01	rd01	w01	pn01	1

triple patterns with their optimized SQL definitions; and *(iv)* it applies structural and semantic techniques to optimize the unfolded query. One of the key points in the last step is the elimination of self-joins, which negatively affect performance in a significant way. To perform this elimination, Ontop utilizes in an essential way the key constraints defined in the data sources. In those cases where it is not possible to define these key constraints explicitly in the data sources, or to expose them as metadata of the data sources so that Ontop can use them, Ontop allows one to define them implicitly, as part of the mapping specification. The data we have been working with in the Bosch use case was mostly log data and stored as separate tables containing often highly denormalized and redundant data. Consequently, there were a significant amount of constraints in the tables that are not declared as primary or foreign keys, which brought significant challenges to the performance of query answering. To address these issues, we had to declare these constraints manually, and supply them as separate inputs to Ontop.

3.2 Data Sources

Despite the fact that the SMT process comprises multiple data sources, we have focused in SIB only on the SMD and AOI log data generated by the respective machines, as these are the main data sources in the SMT process and are required to answer the queries we derived for product quality analysis. We extract the SMD and AOI log data from nested JSON files and without any pre-processing, we store them in a PostgreSQL database, which contains the relational tables shown in Table 1. The `event` suffixed tables contain information about mounting and inspection processes, while the `location` suffixed tables describe locations where processed panels are inspected. The `panel` suffixed tables contain information about processed and inspected panels and boards. The SMD data set tracks the information about the pick-and-place sequences, which is stored in the `smd_components` table. It contains the sequence numbers in which the component is picked and placed, and the turn number in which the pick-and-place is performed. Finally, the `aoi_failures` table encompasses information of the failures that are detected during the SMT process by visually inspecting the solder joints. It contains information about panels, boards, components, and pins associated with failures.

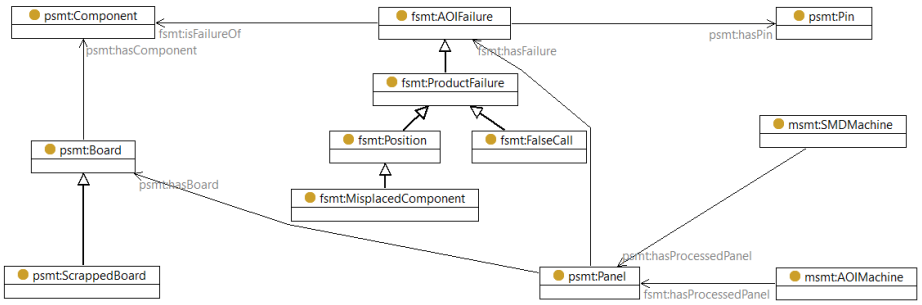

Fig. 4. The SMT Ontology. The SMT Ontology comprises the SMT Product, SMT Machine, and SMT Failure Ontologies. Above, we show a portion of the SMT Ontology.

3.3 SMT Ontology

In the VKG approach, the domain of interest is described in a ontology in terms of *classes*, their *data properties* (i.e., attributes), and the *object properties* (i.e., relationships) in which they are involved. The ontology serves as an abstraction over the data sources and is used by the domain experts to formulate queries over the data. To apply the VKG approach, we developed the SMT Ontology (cf. Figure 4), which is used as a domain model for semantic data integration and access in the domain, and is divided into three modules: *1) SMT Failure Ontology* (`fsmt`), modeling the SMT failures that can occur during the process; *2) SMT Product Ontology* (`psmt`), describing SMT products; and *3) SMT Machine Ontology* (`msmt`), modeling SMT machines. The SMT Ontology overall comprises 76 classes, 30 object properties, and 57 datatype properties.

The development of the SMT Ontology was the result of six workshops over a period of eight weeks, involving Bosch experts that include a line engineer, two line managers from the Bosch plant, an SMT process expert, an SMT data manager, a project manager, two Big Data managers, and four semantic experts. The ontology creation process took several iterations that were required to understand the SMT process, the concepts that are relevant to it, and the relationships between these concepts. We combined a top-down approach in which we modeled classes and properties based on expert knowledge provided by the process and machine experts, with a bottom-up approach in which we looked at the JSON data together with the data engineers and experts to identify additional attributes of classes. We also had to study numerous Wiki pages with technical documentation created by the process experts of the Bosch plant.

Let us briefly discuss the case of AOI failure types, since these play a crucial role in the product quality assessment, and understanding their meaning is of crucial importance to take decisions in the SMT process. The failure types were partially described in textual form in the Wiki and partially they were only in the head of the process experts. Typically, the way in which failures are coded differs between different plants inside the same organization. To address this issue, we semantically codified the failures in the SMT Failure Ontology. For instance, the failure type *misplaced component*, meaning that a component was misplaced on top of the board, is represented in the data with the *magic number* 21. We created a class `fsmt:MisplacedComponent` to represent this type

of failure. The class is a subclass of `fsmt:Position` as well as `fsmt:ProductFailure`, which semantically group all the failures that fall into these categories. Similarly, we created a class `psmt:ScrappedBoard`, for semantically representing all the boards that are scrapped, represented as subclass of `psmt:Board`.

3.4 Data-to-Ontology Mapping

A *mapping* is a set of assertions specifying how the classes and properties of the ontology are populated by the data from the sources. Each mapping assertion consists of an *identifier*, a *source part*, i.e., a SQL query over the data source schema, and a *target part*, i.e., an RDF triple template [3]. The standard language for representing a mapping is defined by the W3C R2RML specification [4].

The SMT VKG mapping contains 53 mapping assertions; they link the elements of the SMT data source to the basic ontological concepts and roles in the SMT Ontology. For the sake of readability and for lack of space, we present only a subset of the mappings that we developed. These mappings cover several important assertions relevant to the queries, and we formulate them using the Ontop mapping language [2], since we find it more compact and more end-user oriented than the R2RML mapping language.

The first mapping assertion we consider (with id `hasBoard`), instantiates the object property `psmt:hasBoard`, relating individuals of the classes `psmt:Panel` and `psmt:Board`, and also associates the individuals of `psmt:Panel` with their processing times.

```
mappingId hasBoard
target    psmt:Panel/{panelId} psmt:hasBoard psmt:Board/{panelId}/{boardNo} ;
                               psmt:pTStamp {smdpTStamp}.
source    SELECT panelId, boardNo, smdpTStamp  FROM smd_panel
```

The next mapping assertion places some of the individuals of the `psmt:Board` class (namely those selected by the `WHERE` condition in the source query) also in the `psmt:ScrappedBoard` class.

```
mappingId ScrappedBoard
target    psmt:Board/{panelId}/{boardNo} a psmt:ScrappedBoard .
source    SELECT panelId, boardNo FROM aoi_failures WHERE boardScrapped = 1;
```

As one can notice, this mapping assertion addresses the semantic interoperability problem, by carrying the information about scrapped boards that is encoded in numbers, from the data level to the conceptual level.

The following mapping assertion creates a bridge between SMD and AOI entities, by relating individuals of `fsmt:AOIFailures` with individuals of `psmt:Panel` via the `fsmt:hasFailure` object property.

```
mappingId hasFailure
target    psmt:Panel/{panelId} fsmt:hasFailure
              fsmt:AOIFailure/{panelId}/{boardNo}/{refDesignator}/{windowNo}/{cPinNo}.
source    SELECT  panelId, boardNo, refDesignator, windowNo, cPinNo
          FROM aoi_failures
```

Finally, the following mapping assertions label the individuals of failures with a type, which may be either `"FalseCall"` or `"MisplacedComponent"`, depending on the numeric value of the `failureCode` attribute of the `aoi_failures` table, which identifies the failure types.

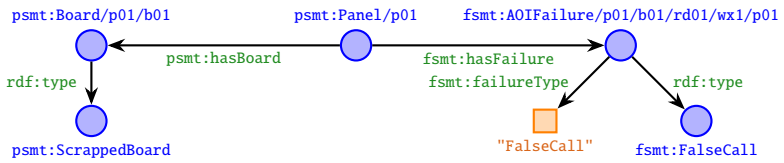

Fig. 5. A VKG. It is built from the mapping over the SMD and AOI data.

```
mappingId FalseCall
target    fsmt:AOIFailure/{panelId}/{boardNo}/{refDesignator}/{windowNo}/{cPinNo}
            rdf:type fsmt:FalseCall ;
            fsmt:failureType "FalseCall" .
source    SELECT panelId, boardNo, refDesignator, windowNo, cPinNo
          FROM aoi_failures WHERE failureCode = 2

mappingId MisplacedComponent
target    fsmt:AOIFailure/{panelId}/{boardNo}/{refDesignator}/{windowNo}/{cPinNo}
            rdf:type fsmt:MisplacedComponent ;
            fsmt:failureType "MisplacedComponent" .
source    SELECT panelId, boardNo, refDesignator, windowNo, cPinNo
          FROM aoi_failures WHERE failureCode = 21
```

Based on the above mapping assertions defined over the SMD and AOI data, a VKG (cf. Figure 5), is built. This graph contains instances of `psmt:Panel`, `psmt:Board`, and `fsmt:AOIFailure` as nodes, and also object properties `psmt:hasBoard` and `fsmt:hasFailure` as edges. Note that the special object property (i.e., edge) `rdf:type` is used to represent the membership of an object to a class.

4 Evaluation of SIB at Bosch

We have carried out an evaluation of SIB, both with respect to its effectiveness in supporting the formulation of typical product quality analysis tasks through SPARQL queries, and with respect to the efficiency with which such queries are executed by the underlying VKG system Ontop over distinct data sets with three different sizes.

4.1 Effectiveness of SIB

We start by describing our effectiveness evaluation. We measure the effectiveness be verifying whether typical product quality queries can be expressed over the ontologies that we developed. To this end, we developed a catalog of 13 queries that consolidate the requirements of Bosch experts. These queries were the result of a collaborative work and a careful selection during two visits to Bosch plants and meetings with Bosch line engineers and line managers. The queries offer a good balance among three dimensions: they are representative for product analyses, offer a good coverage of product analyses tasks, and they are complex enough to account for a reasonable number of domain terms. All these 13 queries were expressible over the ontologies of SIB.

Now we present three of these queries, formulating them both in natural language and in SPARQL. The queries are presented in increasing complexity, going from a simple

query performing joins and applying FILTERs, to a more complex query involving a nested sub-query, up to a query involving complex aggregation. The complete query catalog is provided as supplemental material.

Query q2: *"Return all panels that have been processed after a given panel P, and before 2018-06-28T11:05:42.000+02:00."*

This query fetches all the panels produced in a given time interval between the production time of panel P and the given time. Since timestamps recording processing time are associated to panels, we retrieve all panels processed after the given panel by comparing their timestamps. The query in SPARQL is formalized as follows:

```
SELECT DISTINCT ?pn2 ?ts2
WHERE {
  ?pn psmt:panelId ?panelId ;
      psmt:pTStamp ?ts .
  ?machine psmt:hasProcessedPanel ?pn2 .
  ?pn2 psmt:pTStamp ?ts2 .
  FILTER (?ts < ?ts2 && ?ts2 < '2018-06-28T11:05:42.000+02:00'^^xsd:dateTimeStamp)
  FILTER (?panelId = "085079990025218062222261041592") }
```

Query q3: *"Return all panels processed from a given time T up to the detection of a failure."*

This query is temporal in nature in the sense that we are interested in all panels that did not encounter failures in production *until* the first failure was encountered. The query can still be realized in SPARQL as shown below.

```
SELECT DISTINCT ?panel ?ts ?eventTime
WHERE {
  ?panel psmt:pTStamp ?ts . {
    SELECT ?eventTime
    WHERE {
      ?eventfailure fsmt:eTStamp ?eventTime .
      FILTER (?eventTime > '2018-06-01T00:06:00.000+02:00'^^xsd:dateTimeStamp)
    }
    ORDER BY (?eventTime) LIMIT 1
  }
FILTER (?ts > '2018-06-01T00:06:00.000+02:00'^^xsd:dateTimeStamp && ?ts < ?eventTime) }
```

Query q9: *"For the panels processed in a given time frame, retrieve the number of failures associated with scrapped boards and grouped by failure types."*

This query is formalized in SPARQL using an aggregation operator, as shown below.

```
SELECT (COUNT(?failure) as ?f) ?type
WHERE {
  ?pn psmt:pTStamp ?ts ;
      psmt:hasBoard ?board ;
      fsmt:hasFailure ?failure .
  ?failure fsmt:failureType ?type ;
           rdf:type fsmt:AOIFailure .
  ?board rdf:type psmt:ScrappedBoard .
  FILTER (?ts > '2018-06-01T02:17:54+02:00'^^xsd:dateTimeStamp)
  FILTER (?ts < '2018-06-02T07:04:14+02:00'^^xsd:dateTimeStamp)
} GROUP BY ?type
```

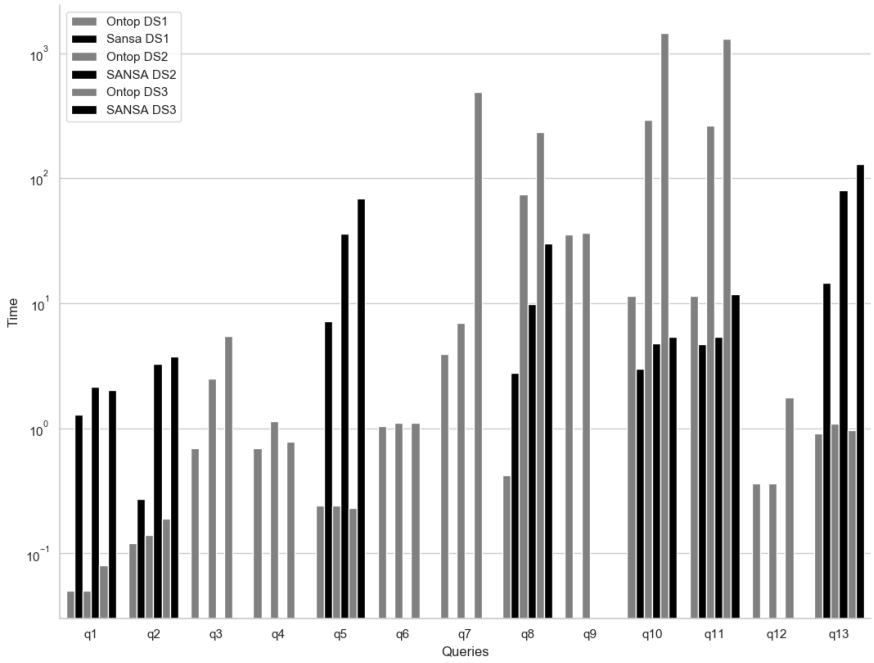

Fig. 6. Evaluation Results of executing the 13 queries using Ontop and SANSA on the three data sets used for the evaluation. The execution times are given in seconds. In general, Ontop outperforms SANSA. Ontop also supports more queries than SANSA.

4.2 Efficiency of SIB

In order to analyze the performance and scalability of the VKG approach for the SMT use case, we executed performance tests for the 13 queries in the query catalog. To evaluate the influence of the size of data on query performance, we executed all 13 queries on three SMD data sets of different sizes. These SMD data sets are combined with the AOI data set. Each line in the AOI data set describes when an error occurs. The three data sets in JSON, i.e., DS1, DS2, and DS3, have sizes of 3.15 GB, 31 GB, and 59 GB, respectively. Each line of the SMD data sets represents an event where a panel was produced. DS1 comprises 69995 lines, DS2 has 699962 lines, whereas DS3 has 1332161 lines. DS3 contains all the SMT data that was available for the experiment. The experiments were executed on a machine Intel(R) Xeon(R) Gold 6148 CPU @ 2.40GHz with 64 GB RAM and Red Hat Linux version 7 installed. As a database system we used PostgreSQL version 9.2.24. For the execution of the experiments, we used the Ontop v3.0 Plugin for Protégé, which provides an integrated framework for the definition of the ontology and the mappings to an underlying database, and gives also the possibility of executing SPARQL queries over such database.

We compared the results with SANSA [14], which is a distributed framework for scalable RDF data processing based on Apache Spark. SANSA allows for accessing both large RDF graphs, and heterogeneous large non-RDF data sources, thanks to its

sub-component SANSA-DataLake (SANSA-DL) [15]. SANSA-DL can query out-of-the-box various types of sources ranging from plain files and file formats stored in Hadoop distributed file system, e.g., CSV and Parquet, as well as various other sources via a JDBC connector. We included SANSA because it is an open source solution, and also uses a virtualized approach.

Since neither Ontop nor SANSA natively support the JSON format, the three datasets used for the evaluation have been ingested into a database and into Parquet files, respectively. For Ontop, we created a Python script to read the JSON data and insert it into a PostgreSQL database. For SANSA, we have developed a Scala script to convert the data into Parquet files. Figure 6 reports on the query execution times in seconds for executing the queries in Ontop and SANSA with the different data sets. We have used gray bars to depict the execution times for Ontop, and black bars for those of SANSA.

The experimental results show that Ontop performs well in general and outperforms SANSA. For most of the queries (namely q1–q5, q6, q7, q12, and q13) the execution times scale sub-linearly, and most of them finish in less than one second even over the largest dataset DS3. For most of the complex queries, i.e., q5, q7, q9, q10, and q11, the execution lasts in the order of 10s of seconds, which is considered a reasonable amount of time in the context of the SIB use case. We observe that SANSA does not support some queries, i.e., q3, q4, q6, q7, q9, and q12. This is because subqueries, queries containing variables at both object and subject position, i.e., joins on the object position, and GROUP BY clauses in SPARQL queries on variables appearing at subject position are currently not supported by SANSA. In two of the queries, i.e., q10 and q11 SANSA behaves better than Ontop. These queries comprise aggregation functions, i.e., *COUNT* in this case. The query processing in Ontop for aggregation functions require additional operations for query rewriting, while SANSA supports these functions as part of its SPARQL fragment. SANSA utilizes SPARK DataFrames to perform aggregation functions on top of the queried data. This seems to have an influence on the query performance as depicted in Figure 6. To summarize our evaluation, Ontop shows quite fast query answering times for most of the queries. Even complex queries involving sub-queries could be answered at reasonable time for our use case. In contrast to Ontop, SANSA could not answer some of the queries. Based on that SANSA was not the first choice for our case as important queries for product analysis tasks could not be executed.

5 Lessons Learned

Involvement of Domain Experts. Involving the Bosch domain experts in the process of deploying the VKG approach to the SMT use case early on was important for the project. Although these experts had no knowledge of semantic technologies, they quickly understood that the SMT Ontology we developed together with them will support their product quality analysis tasks. Before we formally modelled the SMT domain as an Ontology, they started depicting a model of the process on a whiteboard.

Integrated View of Data provided by SMT Ontology. The SMT Ontology and VKG approach enabled complex product analysis tasks that require an integrated view of manufacturing data over multiple processes. Without this integrated view, it would have

been very difficult for the Bosch domain experts to formulate their information needs as queries, as they were working on the level of incompatible raw data sources before.

Methodology and Tooling. The adoption of the VKG approach is a labor-intensive process. In particular, in the design phase of mappings three pieces of knowledge need to be combined: the domain knowledge of the SMT manufacturing process, the detailed database schemas, and how the VKG approach works. This makes it challenging to produce high-quality mappings. A proper methodology and tooling support will be important to improve the productivity. By applying the VKG approach we learned several guidelines that should be followed when designing the mappings, notably the following: *(i)* it is necessary to avoid joining multiple tables inside a mapping to reduce the complexity of query-rewriting; *(ii)* indexes in the database should be used for speeding up the processing of certain translated SQL queries; *(iii)* all necessary key constraints should ideally be defined and exported via JDBC, to enable the VKG system Ontop to perform extensive query optimization.

Handling Denormalized Data. The data we have been working with in this project is mostly log data. Each large log file was treated as a separate table. Such tables are often highly denormalized and contain a lot of redundancy. Consequently, there are a significant amount of constraints in the tables that are not declared as primary or foreign keys. These redundancies bring significant challenges to the performance of query answering. To address these issues, we have supplied the constraints as separate inputs to Ontop so as to avoid generating queries with redundancies. This optimization is critical and we have observed it turns many queries from timing out to almost finishing instantly. Currently, these constraints are declared manually, but in the future, we envision to automate this step as well.

Efficiency of VKG Approach. Bosch domain experts were skeptical in the beginning about the VKG approach, and they were confident that it could result in rather long query execution times for product quality analysis. After we presented our evaluation, they were quite enthusiastic about the results as their complex queries could be answered in a reasonable amount of time.

Impact of our approach to I4.0 at Bosch. The results of applying the VKG approach to the problem of integrating heterogeneous manufacturing data for the SMT manufacturing process are quite promising. We integrated the heterogeneous data sources to answer complex queries while at the same time achieving reasonable query execution times. We also showed that the SIB framework bridges the gap between the complex SMT domain and the underlying data sources, which is important for enabling the vision of I4.0. We see that the SIB framework can also be applied to other use cases at Bosch.

6 Conclusions and Outlook

Conclusions. We have presented the SIB Framework, in which we apply the VKG approach for the efficient integration of manufacturing data for product quality analysis tasks at Bosch. We introduced an ontology that models the domain knowledge and abstracts away the complexity of the source data for the *Surface Mounting Technology*

Manufacturing Process. We mapped the data sources into ontological concepts and formulated queries that encode important product quality analysis tasks of domain experts at Bosch. We evaluated SIB over three different SMT data sets of different scales. We have presented the evaluation results and have shown that our framework retrieves the requested information mostly at reasonable time.

Outlook. This work is a first and important step towards adapting the VKG technology for integration of manufacturing data at Bosch, but there is a number of further steps to do. So far, we heavily involved domain experts from the Bosch Salzgitter plant in the development of the query catalog and ontologies and in the interpretation of results. This effort has shown that it is important to extend SIB with GUIs that will ease the interaction with domain experts, and will allow them to use SIB for various analytical tasks. This would enable us to conduct a user study that is needed for a more in depth evaluation of SIB. Moreover, this will permit us to scale the evaluation of the system from 13 queries to a much more substantial catalog. Then, it is important to show the benefits of SIB with analytical tasks beyond product analysis and evaluate the ability of the SMT ontology to accommodate a wider range of use cases. Another important next step is to extend SIB with a number of dashboards to facilitate aggregation and visualisation of query results. Moreover, an additional core step is popularization and validation of the SIB technology and its benefits with Bosch colleagues from various departments, where we already did the first step by presenting our results in an internal Bosch "Open Day" for internal projects. All of this should allow us to make SIB accessible and attractive to Bosch domain experts from factories and move us closer towards integrating SIB in the tool-set of Bosch analytical software. Finally, we plan to extend our solution and directly connect ontologies to the native JSON data, avoiding the intermediate step of materializing JSON into a relational DB. We also plan to compare the VKG approach to native triple stores.

Acknowledgements. This research has been partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation, by the Italian Basic Research (PRIN) project HOPE, by the EU H2020 project INODE, grant agreement 863410, by the CHIST-ERA project PACMEL, by the Free Univ. of Bozen-Bolzano through the projects QUADRO, KGID, and GeoVKG, and by the project IDEE (FESR1133) through the European Regional Development Fund (ERDF) Investment for Growth and Jobs Programme 2014-2020.

References

1. Bienvenu, M., Rosati, R.: Query-based comparison of mappings in ontology-based data access. In: Proc. of the 15th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR), AAAI Press (2016) 197–206
2. Calvanese, D., Cogrel, B., Komla-Ebri, S., Kontchakov, R., Lanti, D., Rezk, M., Rodriguez-Muro, M., Xiao, G.: Ontop: Answering SPARQL queries over relational databases. Semantic Web J. **8**(3) (2017) 471–487
3. Cyganiak, R., Wood, D., Lanthaler, M.: RDF 1.1 concepts and abstract syntax. W3C Recommendation, World Wide Web Consortium (February 2014)
4. Das, S., Sundara, S., Cyganiak, R.: R2RML: RDB to RDF mapping language. W3C Recommendation, World Wide Web Consortium (September 2012)

5. De Giacomo, G., Lembo, D., Lenzerini, M., Poggi, A., Rosati, R.: Using ontologies for semantic data integration. In: A Comprehensive Guide Through the Italian Database Research Over the Last 25 Years. Volume 31 of Studies in Big Data. Springer (2018) 187–202
6. Elmer, S., Jrad, F., Liebig, T., ul Mehdi, A., Opitz, M., Stauß, T., Weidig, D.: Ontologies and reasoning to capture product complexity in automation industry. In: Proc. of the ISWC 2017 P&D and Industry Tracks. (2017)
7. Horrocks, I., Giese, M., Kharlamov, E., Waaler, A.: Using semantic technology to tame the data variety challenge. IEEE Internet Computing **20**(6) (2016) 62–66
8. Hubauer, T., Lamparter, S., Haase, P., Herzig, D.M.: Use cases of the industrial knowledge graph at siemens. In: Proc. of the ISWC 2018 P&D/Industry/BlueSky Tracks. (2018)
9. Kharlamov, E., Hovland, D., Skjæveland, M.G., 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.: Ontology based data access in Statoil. J. of Web Semantics **44** (2017) 3–36
10. Kharlamov, E., Kotidis, Y., Mailis, T., Neuenstadt, C., Nikolaou, C., Özçep, Ö.L., Svingos, C., Zheleznyakov, D., Ioannidis, Y.E., Lamparter, S., Möller, R., Waaler, A.: An ontology-mediated analytics-aware approach to support monitoring and diagnostics of static and streaming data. J. of Web Semantics **56** (2019) 30–55
11. Kharlamov, E., Mailis, T., Mehdi, G., Neuenstadt, C., Özçep, Ö.L., Roshchin, M., Solomakhina, N., Soylu, A., Svingos, C., Brandt, S., Giese, M., Ioannidis, Y.E., Lamparter, S., Möller, R., Kotidis, Y., Waaler, A.: Semantic access to streaming and static data at Siemens. J. of Web Semantics **44** (2017) 54–74
12. Kharlamov, E., Mehdi, G., Savkovic, O., Xiao, G., Kalayci, E.G., Roshchin, M.: Semantically-enhanced rule-based diagnostics for industrial Internet of Things: The SDRL language and case study for Siemens trains and turbines. J. of Web Semantics **56** (2019) 11–29
13. Kharlamov, E., Skjæveland, M.G., Hovland, D., Mailis, T., Jiménez-Ruiz, E., Xiao, G., Soylu, A., Horrocks, I., Waaler, A.: Finding data should be easier than finding oil. In: Proc. of the IEEE Int. Conf. on Big Data (Big Data), IEEE Computer Society (2018) 1747–1756
14. Lehmann, J., et al.: Distributed semantic analytics using the SANSA stack. In: Proc. of the 16th Int. Semantic Web Conf. (ISWC). (2017) 147–155
15. Mami, M.N., Graux, D., Scerri, S., Jabeen, H., Auer, S., Lehmann, J.: Squerall: Virtual ontology-based access to heterogeneous and large data sources. In: Proc. of the 18th Int. Semantic Web Conf. (ISWC). (2019) 229–245
16. Mehdi, A., Kharlamov, E., Stepanova, D., Loesch, F., Gonzalez, I.G.: Towards semantic integration of bosch manufacturing data. In: Proc. of ISWC. (2019) 303–304
17. Mehdi, G., Kharlamov, E., Savkovic, O., Xiao, G., Kalaycı, E.G., Brandt, S., Horrocks, I., Roshchin, M., Runkler, T.A.: Semantic rule-based equipment diagnostics. In: Proc. of the 16th Int. Semantic Web Conf. (ISWC). (2017) 314–333
18. Petersen, N., Halilaj, L., Grangel-González, I., Lohmann, S., Lange, C., Auer, S.: Realizing an RDF-based information model for a manufacturing company - A case study. In: Proc. of the 16th Int. Semantic Web Conf. (ISWC). (2017) 350–366
19. Rodriguez-Muro, M., Rezk, M.: Efficient SPARQL-to-SQL with R2RML mappings. J. of Web Semantics **33** (2015) 141–169
20. Xiao, G., Calvanese, D., Kontchakov, R., Lembo, D., Poggi, A., Rosati, R., Zakharyaschev, M.: Ontology-based data access: A survey. In: Proc. of the 27th Int. Joint Conf. on Artificial Intelligence (IJCAI), IJCAI Org. (2018) 5511–5519
21. Xiao, G., Ding, L., Cogrel, B., Calvanese, D.: Virtual knowledge graphs: An overview of systems and use cases. Data Intelligence **1**(3) (2019) 201–223
22. Xiao, G., Kontchakov, R., Cogrel, B., Calvanese, D., Botoeva, E.: Efficient handling of SPARQL optional for OBDA. In: Proc. of the 17th Int. Semantic Web Conf. (ISWC). Lecture Notes in Computer Science, Springer (2018) 354–373