Dynamic Knowledge Graph-based Measurement of Data Quality

Johannes Schrott
Rainer Meindl
Christian Lettner
Software Competence Center Hagenberg GmbH
Hagenberg, Austria
firstname.lastname@scch.at

Stefan Hammer
Magdalena Leitner
Robert Bosch AG
Linz, Austria

ABSTRACT

Graphs are a versatile concept for the representation of interconnected data, such as production processes. In combination with ontologies that provide context in the form of domain knowledge, graphs holding data can be seen as knowledge graphs. An interesting use case for knowledge graphs is data quality measurement, which is highly context-dependent. Consequently, ontologies for a knowledge graph-based definition of data quality metrics, such as the Data Quality Definition Ontology (DQD), have been developed. Unfortunately, context must be encoded as part of the data quality metric definitions, which limits the reusability of data quality metrics in other use cases.

Addressing the shortcomings of previous work, we develop a novel method for dynamic knowledge graph-based parameters for data quality metric definitions. We further provide an extended version of DQD that leverages our method to support dynamic parameters among other optimizations. Based on an example in the context of a manufacturing company, we highlight the practical applicability of our approach and discuss future uses of the concept.

VLDB Workshop Reference Format:
Johannes Schrott, Rainer Meindl, Christian Lettner, Stefan Hammer, and Magdalena Leitner. Dynamic Knowledge Graph-based Measurement of Data Quality. VLDB 2025 Workshop: 14th International Workshop on Quality in Databases (QDB'25).

VLDB Workshop Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://w3id.org/dqd/1.0.

1 INTRODUCTION

As an effect of the increasing digitalization of the world, the amount of data generated in almost all aspects of life is increasing. For example, in manufacturing companies the production steps of each manufactured item are recorded to create digital twins as part of the so-called "Industry 4.0" [22, 24]. Tracking of the production process enables detailed analyses, which can lead to insights that allow to optimize the production process. However, manufacturing data may have a complex structure, as shown in the following running example (cf. Figure 4): Products are sequentially assembled on a production line by various machines, and each machine provides a timestamp for each assembly it performs. Although the relational model [9] is a viable option for representing such data, a high number of joins leads to inefficient processing of this data. Therefore, the use of *graphs*, i.e., a collection of nodes connected by edges, is typically more efficient for highly interconnected data. However, the flexibility of graphs (e.g., dynamically evolving schemas) comes at the cost of data validation issues and hard-to-detect *data quality (DQ)* problems [27]. This is unfortunate, since high DQ is a prerequisite for tasks such as training accurate machine learning (ML) models [7] and performing meaningful data analyses [6].

A possibility to overcome these challenges of measuring DQ is to enrich graphs containing data with semantics. We consider such data graphs as directed edge-labeled graphs that are implemented using the Ressource Description Framework (RDF) [10] and that are provided with semantics through Web Ontology Language (OWL) [34] *ontologies* (cf. [13, 32] for definitions of ontologies). Following Hogan et al. [20], we refer to the combination of graphs that hold data and ontologies that provide domain context as *knowledge graphs (KGs)*. Consequently, KGs combine data with context, which makes them suitable for the context-dependent task of DQ measurements [35].

DQ is often viewed as a multidimensional concept, where each *dimension* (e.g., completeness) covers a particular aspect of DQ [6, 30, 35]. DQ *metrics* [6, 30] (e.g., the share of missing values) provide quantitative measurements for a particular dimension. When assessing the quality of a KG, users have to specify which parts of the KG are measured by which DQ metric. Various ontologies with different strengths and weaknesses [2, 11, 15, 16, 29] have been developed for this purpose, such as the recently published Data Quality Definition (DQD) Ontology[1] [29].

Although current ontologies allow the definition of reusable DQ metrics and enable a semantic-based assignment of DQ metrics, they offer only limited possibilities for metric configuration (e.g., for setting thresholds) in the form of static values. We refer to such metric configurations as *static metric parameterization*. However, for use cases based on complex data, such as the manufacturing data mentioned above, static parameterization is not always expressive enough. For example, a DQ metric that measures whether timestamps of performed assemblies are in order cannot be expressed using static parametrization. Thus, a mechanism that allows metrics to be parameterized based on the data to be measured is required. We refer to such a mechanism as *dynamic parameterization*. The need for such context-dependent DQ definitions is also mentioned

Proceedings of the VLDB Endowment. ISSN 2150-8097.

[1] https://w3id.org/dqd/0.5 (last visited in June 2025)

by Serra et al. [31], who emphasize that this increases the *reusability* across use cases.

In this paper, we address existing limitations and contribute as follows: (1) We present a novel method that enables the dynamic parameterization of DQ metrics. (2) To facilitate its usability, we implement our method in DQD, a state-of-the-art ontology for data quality definitions. Intuitively, values for the parameters of DQ metrics are dynamically retrieved from the KG based on the data to be measured. (3) Moreover, we showcase the applicability of our approach based on a simplified use case on manufacturing data.

The remainder of this paper is structured as follows: Section 2 provides related work in terms of ontologies that enable a KG-based definition of DQ measurements. Section 3 introduces the novel method for dynamic KG-based DQ metric parameterization. Section 4 describes the extension of DQD with support for the new method as well as further features. The practical application of the new method is shown in Section 5. A discussion of the effects of our new method and its impacts on other topics is provided in Section 6. Section 7 concludes this paper along with future research directions.

2 RELATED WORK

DQ measurements are typically performed by computing DQ metrics that quantitatively assess the fulfillment of various aspects, the so-called DQ dimensions [6, 30, 35]. Although there are common DQ dimensions, such as accuracy, completeness, and validity, standardized definitions are missing [30, 35]. Similarly, multiple DQ metrics for each DQ dimension have been proposed (see [12] for examples). Consequently, a common understanding of DQ metrics and DQ dimensions must be established within organizations to obtain a consistent view of DQ and its measurements.

Since this paper focuses on RDF-based KGs that follow OWL ontologies, this section investigates existing ontologies [29] enabling the specification of reusable DQ definitions. Even though some of those ontologies are called "vocabularies", in this paper they are referred to as ontologies, as they provide a "specification of a shared conceptualization" [32]. Given the objective of this paper (i.e., the dynamic parametrization of metrics for KG-based DQ measurement), we formulate five aspects by which we analyze related ontologies:

(1) **Data type.** For which type of data can DQ measurements be defined?
(2) **Structure.** Does the ontology use DQ dimensions and metrics to structure DQ information, as mentioned above?
(3) **Reusability.** Does the ontology provide support to reuse definitions of DQ metrics and dimensions?
(4) **Storing results.** Where and how can the DQ measurement results be stored?
(5) **Dynamic parameterization.** Does the ontology allow to dynamically parameterize DQ metrics based on data stored in the KG?

These aspects are analyzed on five ontologies in total, as shown in Table 1. In addition to the four ontologies mentioned in [29], an online search has been carried out to cover new developments, leading to the inclusion of one additional ontology [16].

Apart from the five ontologies investigated, the online search yielded further results, which are either (i) domain-specific (e.g., Spatial Data Quality Ontology (SDQO) [37]), (ii) rely on the Data Quality Vocabulary (DQV) for the representation of DQ measurement definitions (e.g., the Data Quality Assurance Ontology (DQAO) [25] and BIGOWL4DQ [5]), or (iii) poorly documented, which prohibits an analysis (e.g., the Image and Data Quality Assessment Ontology (IDQA) [33]).

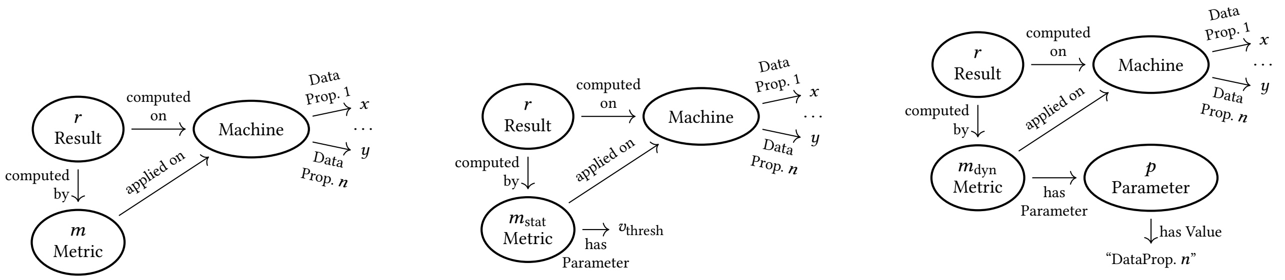
As Table 1 shows, all analyzed ontologies enable the reuse of DQ measurement definitions. However, regarding the other analysis aspects, the ontologies differ: the Data Quality Management (DQM) vocabulary [15] uses a slighly different terminolgy by defining so-called DQ rules that mainly return boolean results, the Dataset Quality Ontology (daQ) [11] and DQV [2] can only define DQ measurements on whole datasets, DQD [29] can be applied to data assets of any type, and DQStream [16] targets data streams. Overall, none of the discussed ontologies allow the definition of metrics that depend on dynamic parameters whose values are determined based on the measured KG.

For relational data, the situation is similar. Although there are frameworks for measuring DQ on relational data, such as *great expectations*[2] and *deequ*[3], frameworks like these typically only allow the definition of static parameters. Dynamic parameterization, e.g.,

[2]https://greatexpectations.io (last visited in June 2025)
[3]https://github.com/awslabs/deequ (last visited in June 2025)

Table 1: Comparison of ontologies that allow the definition of DQ measurements

Ontology	(1) Data type	(2) Structure	(3) Reusability	(4) Storing results	(5) Dynamic parameterization
Data Quality Management (DQM) Vocabulary [15]	RDF-based datasets	DQ rules	Possible	In the RDF-based dataset to be measured	Possible for certain types of rules
Dataset Quality Ontology (daQ) [11]	Whole datasets	DQ metrics are associated to DQ dimensions	Possible	In RDF graphs	Not possible
Data Quality Vocabulary (DQV) [2]	Whole datasets (represented using the Data Catalog Vocabulary (DCAT) [1])	DQ metrics are associated to DQ dimensions	Possible	In RDF graphs	Not possible
Data Quality Definition (DQD) Ontology [29]	Any type of data asset + parts of them (e.g., one attribute, one instance)	DQ metrics consist of statistics and are associated to DQ dimensions	Possible	In an external data store	Not possible
DQStream [16]	Data streams	DQ factors consisting of so-called DQ windows are associated with DQ metrics and DQ dimensions	Possible	As part of the processed data stream	Not possible



(a) No parameterization **(b) Static parameterization** **(c) Dynamic parameterization**

Figure 1: Visualization of KG-based DQ measurements on a small sample of the manufacturing data example

through values stored in certain attributes, is not possible. Even though there is also the need for dynamic parameterization in this context, developments for the relational model are considered out of scope, since the focus of this paper is on KGs-based DQ measurement.

Based on the shortcomings described above, in Section 3 a novel method enabling the dynamic KG-based parameterization of DQ metrics is introduced. Apart from its theoretical introduction, the method is also implemented as an addition to an ontology. In Section 4, we extend DQD to support the new parameterization method, as it is the only analyzed ontology that (i) follows the structure of DQ metrics and DQ dimensions and that (ii) allows to perform DQ measurements on parts of datasets such as instances of a particular class.

3 DYNAMIC KNOWLEDGE GRAPH-BASED PARAMETERIZATION

Following the investigation of ontologies, we observe that none of them allow the definition of dynamically parameterized DQ metrics. Furthermore, Serra et al. [31] mention the inclusion of context into DQ metrics as a topic for future research. Therefore, we extend the definition of DQ metrics [6, 21] to support dynamic KG-based DQ metric parameters.

From a mathematical perspective, a DQ metric m, can be defined as an unary function that maps data d of type D to a DQ result value r from a scale R (Equation (1)) [6, 14, 17, 21]. Adhering to that notation, a DQ measurement is the application of the metric function m onto data d returning a result value r (cf. Equation (2)).

$$m := D \to R \tag{1}$$

$$r = m(d) \tag{2}$$

Since the only argument of m is d, further parameters providing context to be considered in measurements must be directly encoded within m, as also described by [31]. For example, consider a DQ metric m that measures whether a value lies above a certain threshold. Hence, the threshold must be encoded as part of m, m's reusability is limited. Figure 1a shows how unparameterized DQ measurements can be implemented in the context of RDF- and OWL-based KGs. The structure of the defined DQ measurement is inspired by the ontologies introduced in Section 2, whereas the example is based on the manufacturing data mentioned in Section 1.

Static Parameterization. Based on this unparameterized definition of DQ metrics, we first extend the definition of DQ metrics to support parameters $v_1, \ldots, v_{n-1}$. As an effect of that change, a DQ metric m_{stat} is an n-ary function, as shown in Equation (3) and Equation (4). The first argument, data $d \in D$, remains unchanged compared to unparameterized metrics. The remaining $n-1$ arguments of m_{stat} are the parameters $v_1, \ldots, v_{n-1} \in V$.

$$m_{\text{stat}} := (D, V, \ldots, V) \to R \tag{3}$$

$$r = m_{\text{stat}}(d, v_1, \ldots, v_{n-1}) \tag{4}$$

The parameters are constant values $v_1, \ldots, v_{n-1}$ of type V that stay the same for all measurements performed by a DQ metric. Thus, these parameters are called static. Considering the example of the threshold metric, it is now possible to provide the threshold separately as a parameter v_{thresh}. Figure 1b visualizes such a metric in the context of KGs and the manufacturing use case.

Dynamic Parameterization. The static parameterization does still not consider the context provided by a KG. We address this limitation by defining a dynamically parameterized DQ metric m_{dyn} as an n-ary metric. Compared to m_{stat} the metric m_{dyn} takes functions $p_1, \ldots, p_{n-1}$ as arguments that determine a parameter's value based on the data $d \in D$ and a value $v \in V$, as the Equations (5), (6), and (7) show. Consequently, the dynamic computation of parameter values passed to a metric m_{dyn} is enabled.

$$m_{\text{dyn}} := \Big(D, \big((V, D) \to V\big), \ldots, \big((V, D) \to V\big)\Big) \to R \tag{5}$$

$$p_x := (V, D) \to V \tag{6}$$

$$r = m_{\text{dyn}}(d, p_1(v_1, d), \ldots, p_{n-1}(v_{n-1}, d)) \tag{7}$$

Figure 1c shows how a threshold in a DQ metric can be defined dynamically for a KG based on a certain ontology. This time, the metric is parameterized with a function that resolves the parameter value using a path relative to the node of the KG on which the metric is applied to. Consequently, the metric is tied to a specific class and its subclasses within that particular ontology. Subsequently, the metric can be applied on any KG following that ontology. As an effect, the value of the parameter will change dynamically. Compared to the other kinds of DQ metrics (cf. Figures 1a and 1b), this kind of DQ metric features an increased reuseability.

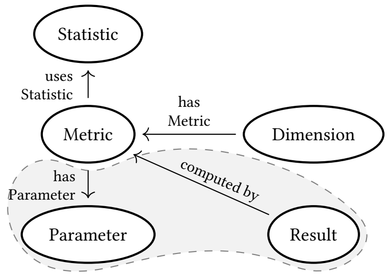


Figure 2: The `owl:Classes` of DQD and the `owl:ObjectProperties` between them. The additions of DQD version 1.0 are contained in the grey area.

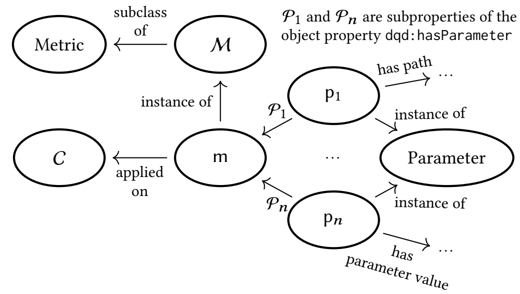


Figure 3: Visualization of dynamic and static parameterization of DQ metrics in DQD

4 EXTENDING THE DATA QUALITY DEFINITION ONTOLOGY

The original concept of the Data Quality Definition (DQD)[1] Ontology (cf. Section 2) was introduced in 2023 alongside an implementation of the ontology in version 0.5 by Schrott et al. [29].

In this section, we introduce version 1.0 of DQD[4], which enables the definition of (dynamic) parameters (cf. Section 4.1), provides an application profile (i.e., a guideline of how to use an ontology, cf. Section 4.2), and contains further refinements (cf. Section 4.3) that became apparent when using the original version in proof-of-concept use cases.

4.1 Implementation of (Dynamic) Parameterization

For the implementation of the (dynamic) KG-based parameterization of DQ metrics, DQD is extended with new classes and properties (cf. Figure 2).

The parameter functions mentioned in Section 3 are implemented in DQD in the form of the `dqd:Parameter` OWL class. The parameter d of the parameter function is inferred from the application of the DQ metric, and the values v are realized as OWL data type properties `dqd:hasPath` and `dqd:hasParameterValue`. The literal value of `dqd:hasPath`, a SPARQL Property Path [18], is used to resolve the actual value of the parameter relative to the class to which the metric is applied on. In contrast, `dqd:hasParameterValue`'s literal is treated as a static value. Thus, DQD version 1.0 supports not only dynamic but also static parameterization of DQ metrics. For the connection of parameters to DQ metrics, the OWL object property `dqd:hasParameter` has been added.

Following [29] and by incorporating DQD's new additions, the definition of a (dynamically) parameterized metric that follows DQD consists of two steps. First, a DQ metric $\mathcal{M}$ needs to be created by forming a subclass of `dqd:Metric`, as described in [29]. Second, to ensure that different parameters can be identified when computing a metric, for each parameter of $\mathcal{M}$, a subproperty $\mathcal{P}_x$ of `dqd:hasParameter` has to be defined. The domain of these subproperties must be $\mathcal{M}$, the range has to be `dqd:Parameter`.

To apply a metric $\mathcal{M}$ to the nodes of a class C in a KG, $\mathcal{M}$ must be instantiated to a metric instance m. By using the object properties $\mathcal{P}_x$, which denote the parameters' roles, this metric instance m is connected to instances p_x of `dqd:Parameter`. These parameter instances p_x reference literal values by using the `dqd:hasParameterValue` and `dqd:hasPath` data type properties.

A visualization of a DQ metric that is dynamically and statically parameterized and applied to a class C is shown in Figure 3. A larger example in the context of the manufacturing use case is provided in Section 5 and visualized in Figure 4.

4.2 Application Profile

Application profiles (APs) declare how terms defined in ontologies should be used in a certain use case [19]. As described in [29], an application profile (AP) is part of DQD's concept to ensure the correct usage of DQD. In contrast to DQD version 0.5, DQD's new release now provides an implementation of the AP in the form of Shapes Constraint Language (SHACL) [23] shapes provided in a separate file. SHACL was chosen over Shape Expressions (ShEx) [3], since SHACL provides better violation reporting and also supports reasoning on the RDF graphs, which is not the case for ShEx.

The shapes contained in the AP were partly created automatically [26] using Astrea [8] and were partly defined manually as well. In its current form, the AP (i) encourages users of DQD to provide a sufficient amount of metadata when defining DQ metrics, it (ii) ensures that ranges of DQ values are defined consistently over a hierarchy of DQ metrics, and it (iii) checks whether properties have correct values, e.g., each result holds exactly one result value. The AP is distributed separately from DQD and is available for download online[5].

4.3 Further Adaptions

Alongside the implementation of the dynamic parameterization (cf. Section 4.1), we introduce further adaptions to DQD that streamline the usage of parameters and address observed limitations of the original version.

To keep the ontology as consistent and concise as possible, all data properties that model metric parameters specific to certain DQ dimensions have been removed in favor of the `dqd:Parameter`

[4]https://w3id.org/dqd/1.0 (last visited in June 2025)

[5]https://w3id.org/dqd-ap/1.0 (last visited in June 2025)

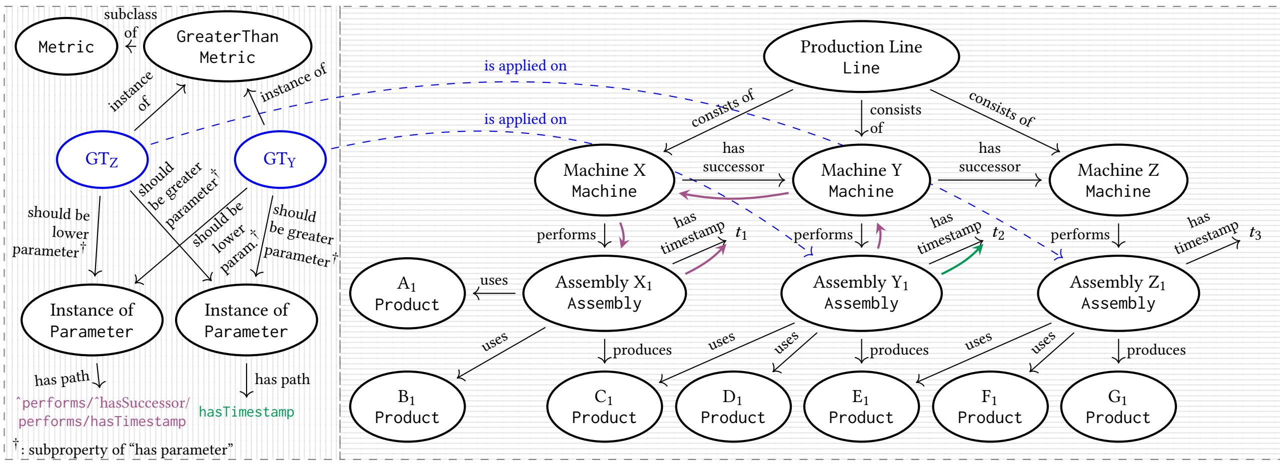


Figure 4: Exemplary visualization of a dynamically parameterized DQ metric and the KG it is applied to. Names written in *monospace* denote classes.

class (cf. Section 4.1). This concerns, e.g., the properties `dqd:hasValidityPattern` and `dqd:hasCustomNullValue` of DQD version 0.5. Even though this change breaks compatibility with some metrics defined in DQD version 0.5, it ensures a consistent definition of arbitrary parameters.

Another removal concerns the value-partition subclasses of the class `dqd:Dimension`, representing individual DQ dimensions. Since DQ dimensions are not standardized, as discussed in [28], pre-created dimensions are no longer part of DQD. Instead, users of DQD must create the instances of `dqd:Dimension` themselves, e.g., based on an organization-wide agreement. Similar to `dqd:Metrics`, the `dqd:Dimensions` can be reused via `dqd:Suites` (cf. [29] for an explanation of the suite concept). This ensures that only agreed DQ dimensions are available when using DQD, which was not supported so far.

Apart from the handling of DQ dimensions, we further improve the handling of DQ results in DQD version 1.0. While it is still possible to store results in an external data store, such as a time series database, and to use DQD as metadata, there is now a possibility to represent DQ measurement results using DQD. Instances of the `dqd:Result` class (i) reference a single literal value that denotes the value of the DQ measurement result (`dqd:hasDQValue`) and (ii) reference an instance of the metric used for the computation of this value (`dqd:computedBy`). This referenced metric instance transitively references the part of the KG for which the DQ result value was computed.

5 APPLICATION IN PRACTICE

We applied the new version of DQD with dynamic parameterization of DQ metrics in our manufacturing use case (cf. Section 1). The use case is kept generic and could be part of any manufacturing company as part of Industry 4.0 initiatives. It considers a production line consisting of three *Machines X*, *Y*, and *Z*. The machines sequentially perform *Assemblies* of *Products*. Products assembled by one machine may be used as a part in an assembly of the next machine. Each assembly contains a timestamp of when it was performed. The part with horizontal lines in the background in Figure 4 visualizes the KG, which can also be downloaded in the form of turtle files[6].

In order for the KG to be valid, the timestamps of the assemblies must have the same order as the machines performing the assemblies. For example, an assembly performed by *Machine X* must happen earlier than an assembly performed by *Machine Y*, which reuses the product assembled at *Machine X*.

Based on that description, we use DQD in version 1.0 to implement a dynamically parameterized DQ metric. The part of Figure 4 with vertical lines in the background shows the classes and instances that are created in order to implement the use case. It is visible that the "GreaterThanMetric" has two dynamic parameters: the path of one resolves to the timestamp of the assembly of the machine it is applied to (shown in green), whereas the path of the other one resolves to the timestamp of the assembly at the predecessing machine (shown in violet). In blue, individual instances GT_n of the "GreaterThanMetric" and their application to the machines X and Y are shown. Exemplarily, the path resolving is visualized for GT_Y. As an effect of the dynamic parameterization, this metric can be also applied to other KGs that follow the same ontology.

Although we demonstrated our novel method in only one use case that could be associated to the DQ dimension "validity", our method is general enough to be transferred to other domains and DQ dimensions as well. For example, a completeness metric considering custom use case-dependent `null` values, such as `NaN`, `None`, or `-1`, or regular expression-based metrics for dynamic syntax checking in the dimension "syntactic correctness" could be created without effort.

6 DISCUSSION

In this section, we discuss the effects of the proposed method for KG-based DQ metric parameterization and its impact on topics different than DQ.

[6]Please find the files for the example as part of DQD's documentation at https://w3id.org/dqd/1.0

Advantages. As highlighted in Section 5, our novel method and its implementation in DQD version 1.0 enable the definition of DQ metrics that are dynamically parameterized using the data to be measured. Such metrics could not be defined by existing developments that rely on static parameterization, i.e., use fixed values. Based on our contributions, possibilities for new DQ metrics that could not have been defined otherwise arise.

FAIRness. The FAIR principles are a set of guiding principles that ensure the Findability, Accessibility, Reusability, and Interoperability of (meta)data [36]. As mentioned in [29], using DQD for the representation of DQ definitions is beneficial to the fulfillment of the FAIR principles. In particular, the introduction of our novel method to DQD facilitates the interoperability and reusability for (generic) DQ metric definitions that can be contextualized using parameters.

For example, the "GreaterThanMetric" described Section 5 and shown in Figure 4 could easily be reused in other use cases. Only the adaption of the paths that are passed as parameters and refer to the parameter values to be used is necessary. Using DQD for the definition of the metric ensures that the semantics of the definition, including the parameters, remain clear at any time. Thus, interoperability and reusability of the metric definition can be guaranteed.

Integration with other Ontologies. Since DQD is the only ontology that allows to perform DQ measurements on parts of datasets such as instances of a particular class, it is used for the implementation of the dynamic KG-based parameterization. However, as described in Section 2, there exist also other ontologies for DQ, which mostly focus on whole datasets. Integrating dynamic parameterization with those ontologies therefore seems to be challenging and opens up potential for future work. First ideas of how solutions to this can look like are mentioned as part of the Data Quality Management (DQM) Vocabulary [15]. However, DQM is only capable of computing Boolean measurements.

Further Data Models. One motivation for this paper was the reuse of context provided by KGs. Nonetheless, it seems promising to transfer the dynamical parameterization to further data models, such as the relational model. As mentioned in Section 2, existing frameworks for measuring DQ on relational data include *great expectations* and *deequ*. However, upon investigation, these frameworks support only static parameterization, not dynamic. Thus, adding our concept of dynamic parameterization could greatly improve the expressive power of such a framework. For example, the values of parameters could be, e.g., based on the values of certain attributes. Nevertheless, the method then loses some of its power, since semantics are not present in the relational data model.

Data Quality versus Product Quality. Product and data quality are two related but different concepts. Data quality provides insights into whether data has been measured and processed correctly, while product quality refers to whether an entity has been manipulated correctly. Product quality can be high even though data quality is low, but without a high level of data quality, it is impossible to determine product quality [4]. In this paper, we focussed on data quality, although metrics for product quality could be defined using the DQD version 1.0 as well.

7 CONCLUSION AND OUTLOOK

In this paper, we introduced a novel method for the dynamic parameterization of data quality (DQ) metrics based on knowledge graphs (KGs). By providing data in combination with context, KGs are an ideal basis for DQ measurements. However, existing ontologies for DQ definitions do not make use of context so far. To overcome this limitation of previous works, we introduced a new method that allows DQ metrics to be dynamically parameterized based on KGs. By including context in the arguments of a DQ metric, the reusability of a DQ metric across different use cases is increased. The novel method is implemented as part of a new version of the Data Quality Definition Ontology (DQD)[4] and has been applied to an example with manufacturing data in order to show its relevance to Industry 4.0 use cases.

In terms of future research, we identified (1) the integration of dynamic parameterization with other ontologies, (2) the transfer of dynamic parameterization to other data models, and (3) the development of a sound concept for sharing metric definitions to further increase interoperability as promising further tasks that need to be investigated. Another suggestion for future work is (4) the investigation of the large-scale computation of DQ measurement results based on KGs and dynamically parameterized metrics. To achieve this, a dedicated software framework could be developed.

ACKNOWLEDGMENTS

The research reported in this paper has been funded by the Federal Ministry for Innovation, Mobility and Infrastructure (BMIMI), the Federal Ministry for Economy, Energy and Tourism (BMWET), and the State of Upper Austria in the frame of the SCCH competence center INTEGRATE (FFG grant no. 892418) in the COMET - Competence Centers for Excellent Technologies Programme managed by Austrian Research Promotion Agency FFG.

REFERENCES

[1] Riccardo Albertoni, David Browining, Simon J D Cox, Alejandra Gonzales Beltran, Andrea Perego, and Peter Winstanley. 2024. Data Catalog Vocabulary. https://www.w3.org/ns/dcat#
[2] Riccardo Albertoni and Antoine Isaac. 2016. Data on the Web Best Practices: Data Quality Vocabulary. https://www.w3.org/TR/vocab-dqv/
[3] Thomas Baker and Eric Prud'hommeaux. 2019. Shape Expressions (ShEx) 2.1. Primer. https://shex.io/shex-primer/index.html
[4] Donald Ballou, Richard Wang, Harold Pazer, and Giri Kumar Tayi. 1998. Modeling Information Manufacturing Systems to Determine Information Product Quality. *Management Science* 44 (1998), 462–484. Issue 4. https://doi.org/10.1287/mnsc.44.4.462
[5] Cristóbal Barba-González, Ismael Caballero, Ángel Jesús Varela-Vaca, José A. Cruz-Lemus, María Teresa Gómez-López, and Ismael Navas-Delgado. 2024. BIGOWL4DQ: Ontology-driven approach for Big Data quality meta-modelling, selection and reasoning. *Information and Software Technology* 167 (2024), 107378. https://doi.org/10.1016/j.infsof.2023.107378
[6] Carlo Batini and Monica Scannapieco. 2016. *Data and Information Quality: Dimensions, Principles and Techniques.* Springer, Cham, Switzerland. https://doi.org/10.1007/978-3-319-24106-7
[7] Lukas Budach, Moritz Feuerpfeil, Nina Ihde, Andrea Nathansen, Nele Noack, Hendrik Patzlaff, Felix Naumann, and Hazar Harmouch. 2022. The Effects of Data Quality on Machine Learning Performance. http://arxiv.org/abs/2207.14529
[8] Andrea Cimmino, Alba Fernández-Izquierdo, and Raúl García-Castro. 2020. Astrea: Automatic Generation of SHACL Shapes from Ontologies. In *The Semantic Web*, Andreas Harth, Sabrina Kirrane, Axel-Cyrille Ngonga Ngomo, Heiko Paulheim, Anisa Rula, Anna Lisa Gentile, Peter Haase, and Michael Cochez (Eds.). Springer, Cham, Switzerland, 497–513. https://doi.org/10.1007/978-3-030-49461-2_29
[9] E. F. Codd. 1970. A Relational Model of Data Large Shared Data Banks. *Commun. ACM* 13, 6 (1970), 377–387.

[10] Richard Cyganiak, David Wood, and Markus Lanthaler. 2014. RDF 1.1 Concepts and Abstract Syntax. https://www.w3.org/TR/rdf11-concepts/

[11] Jeremy Debattista, Christoph Lange, and Sören Auer. 2014. daQ, an Ontology for Dataset Quality Information. In *LDOW2014*, Vol. 1184. CEUR Workshop Proceedings, Seoul, Korea. https://ceur-ws.org/Vol-1184/ldow2014_paper_09.pdf

[12] Lisa Ehrlinger, Bernhard Werth, and Wolfram Wöß. 2018. Automated Continuous Data Quality Measurement with QuaIIe. *International Journal on Advances in Software* 11, 3 & 4 (2018), 400–417. https://www.iariajournals.org/software/soft_v11_n34_2018_paged.pdf

[13] Christina Feilmayr and Wolfram Wöß. 2016. An analysis of ontologies and their success factors for application to business. *Data & Knowledge Engineering* 101 (2016), 1–23. https://doi.org/10.1016/j.datak.2015.11.003

[14] Norman E. Fenton and James Bieman. 2015. *Software metrics: a rigorous and practical approach* (3 ed.). CRC Pr, Boca Raton, FL, USA.

[15] Christian Fürber and Martin Hepp. 2011. Towards a vocabulary for data quality management in semantic web architectures. In *Proceedings of the 1st International Workshop on Linked Web Data Management*. ACM, Uppsala, Sweden, 1–8. https://doi.org/10.1145/1966901.1966903

[16] Sandra Geisler, Christoph Quix, Sven Weber, and Matthias Jarke. 2016. Ontology-Based Data Quality Management for Data Streams. *Journal of Data and Information Quality* 7, 4 (2016), 1–34. https://doi.org/10.1145/2968332

[17] Tom Haegemans, Michael Reusens, Bart Baesens, Wilfried Lemahieu, and Monique Snoeck. 2017. Towards a Visual Approach to Aggregate Data Quality Measurements. In *MIT International Conference on Information Quality*. Little Rock, AR, USA. https://ualr.edu/informationquality/p04-iciq2017-towards-a-visual-approach-2/

[18] Steve Harris and Andy Seaborne. 2013. SPARQL 1.1 Query Language. https://www.w3.org/TR/sparql11-query/#propertypaths

[19] Rachel Heery and Manjula Patel. 2000. Application Profiles: Mixing and Matching Metadata Schemas. *Ariadne* 25 (2000). http://www.ariadne.ac.uk/issue/25/app-profiles/

[20] Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard de Melo, Claudio Gutierrez, Sabrina Kirrane, José Emilio Labra Gayo, Roberto Navigli, Sebastian Neumaier, Axel-Cyrille Ngonga Ngomo, Axel Polleres, Sabbir M. Rashid, Anisa Rula, Lukas Schmelzeisen, Juan Sequeda, Steffen Staab, and Antoine Zimmermann. 2021. Knowledge Graphs. *Synthesis Lectures on Data, Semantics, and Knowledge* 12, 2 (2021), 1–257. https://doi.org/10.2200/S01125ED1V01Y202109DSK022

[21] IEEE. 1998. *Standard for a Software Quality Metrics Methodology*. Technical Report 1061-1998. Institute of Electrical and Electronics Engineers.

[22] Gertrude Kappel, Christian Brecher, Matthias Brockmann, and István Koren. 2022. Internet of production: entering phase two of industry 4.0. *Commun. ACM* 65, 4 (2022), 50–51. https://doi.org/10.1145/3514093

[23] Holger Knublauch and Dimitris Kontokostas. 2017. Shapes Constraint Language (SHACL). https://www.w3.org/TR/shacl/

[24] Heiner Lasi, Peter Fettke, Hans-Georg Kemper, Thomas Feld, and Michael Hoffmann. 2014. Industry 4.0. *Business & Information Systems Engineering* 6 (2014), 239–242. https://doi.org/10.1007/s12599-014-0334-4

[25] Dong Joon Lee, Besiki Stvilia, Fatih Gunaydin, and Yuanying Pang. 2025. Developing a data quality assurance ontology for research data repositories. *Journal of Documentation* 81, 7 (2025), 63–84. https://doi.org/10.1108/JD-09-2024-0212

[26] Kashif Rabbani, Matteo Lissandrini, and Katja Hose. 2022. SHACL and ShEx in the Wild: A Community Survey on Validating Shapes Generation and Adoption. In *Companion Proceedings of the Web Conference 2022* (Virtual Event, Lyon, France) *(WWW '22)*. Association for Computing Machinery, New York, NY, USA, 260–263. https://doi.org/10.1145/3487553.3524253

[27] Rubab Zahra Sarfraz. 2024. Towards Semi-Supervised Data Quality Detection in Graphs. In *VLDB 2024 Workshop: 13th International Workshop on Quality in Databases (QDB'24)*. VLDB Endowment, Guangzhou, China, 6. https://vldb.org/workshops/2024/proceedings/QDB/QDB-2.pdf

[28] Johannes Schrott. 2024. *Constraint-based Measurement and Aggregation of Data Quality*. Master's thesis. Johannes Kepler University Linz, Linz, Austria.

[29] Johannes Schrott, Rainer Meindl, Christian Lettner, Wolfram Wöß, and Lisa Ehrlinger. 2024. DQD: The Data Quality Definition Ontology. In *Metadata and Semantic Research*, Emmanouel Garoufallou and Fabio Sartori (Eds.). Springer, Cham, Switzerland, 291–297. https://doi.org/10.1007/978-3-031-65990-4_27

[30] Laura Sebastian-Coleman. 2013. *Measuring data quality for ongoing improvement: a data quality assessment framework*. Morgan Kaufmann, Waltham, MA, USA.

[31] Flavia Serra, Verónika Peralta, Adriana Marotta, and Patrick Marcel. 2024. Use of Context in Data Quality Management: A Systematic Literature Review. *Journal of Data and Information Quality* 16, 3 (2024), 1–41. https://doi.org/10.1145/3672082

[32] Rudi Studer, V.Richard Benjamins, and Dieter Fensel. 1998. Knowledge engineering: Principles and methods. *Data & Knowledge Engineering* 25, 1-2 (1998), 161–197. https://doi.org/10.1016/S0169-023X(97)00056-6

[33] Thomas Schrader. 2013. Image and Data Quality Assessment Ontology. https://bioportal.bioontology.org/ontologies/IDQA

[34] W3C OWL Working Group 2012. OWL 2 Web Ontology Language Document Overview. https://www.w3.org/TR/owl2-overview/

[35] Richard Y. Wang and Diane M. Strong. 1996. Beyond Accuracy: What Data Quality Means to Data Consumers. *Journal of Management Information Systems* 12, 4 (1996), 5–33. http://www.jstor.org/stable/40398176

[36] Mark D. Wilkinson, Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, Jan-Willem Boiten, Luiz Bonino da Silva Santos, Philip E. Bourne, Jildau Bouwman, Anthony J. Brookes, Tim Clark, Mercè Crosas, Ingrid Dillo, Olivier Dumon, Scott Edmunds, Chris T. Evelo, Richard Finkers, Alejandra Gonzalez-Beltran, Alasdair J.G. Gray, Paul Groth, Carole Goble, Jeffrey S. Grethe, Jaap Heringa, Peter A.C 't Hoen, Rob Hooft, Tobias Kuhn, Ruben Kok, Joost Kok, Scott J. Lusher, Maryann E. Martone, Albert Mons, Abel L. Packer, Bengt Persson, Philippe Rocca-Serra, Marco Roos, Rene van Schaik, Susanna-Assunta Sansone, Erik Schultes, Thierry Sengstag, Ted Slater, George Strawn, Morris A. Swertz, Mark Thompson, Johan van der Lei, Erik van Mulligen, Jan Velterop, Andra Waagmeester, Peter Wittenburg, Katherine Wolstencroft, Jun Zhao, and Barend Mons. 2016. The FAIR Guiding Principles for scientific data management and stewardship. *Scientific Data* 3, 1 (2016), 160018. https://doi.org/10.1038/sdata.2016.18

[37] Cemre Yılmaz, Çetin Cömert, and Deniz Yıldırım. 2024. Ontology-Based Spatial Data Quality Assessment Framework. *Applied Sciences* 14, 21 (2024), 10045. https://doi.org/10.3390/app142110045