Aspects of RDF Data Constraints in the Social, Behavioural, and Economic Sciences

Thomas Bosch*, Benjamin Zapilko*, Joachim Wackerow*, Kai Eckert†
*GESIS - Leibniz Institute for the Social Sciences, Mannheim, Germany
Email: {firstname.lastname}@gesis.org
†Stuttgart Media University, Stuttgart, Germany
Email: eckert@hdm-stuttgart.de

***Abstract*—For research institutes, data libraries, and data archives, RDF data validation according to predefined constraints is a much sought-after feature, particularly as this is taken for granted in the XML world. Based on our work in the DCMI RDF Application Profiles Task Group and in cooperation with the W3C Data Shapes Working Group, we identified and published by today 81 types of constraints that are required by various stakeholders for data applications. In this paper, we formulate 115 constraints on three different vocabularies (DDI-RDF, QB, and SKOS) and classify them according to their severity level and whether their type is expressible by different types of constraint languages. We evaluate the data quality of 15,694 data sets (4.26 billion triples) of research data for the social, behavioural, and economic sciences obtained from 33 SPARQL endpoints. Based on the results, we formulate several hypotheses to direct the further development of constraint languages.**

I. Introduction

references to constraint languages

The social, behavioural, and economic sciences (*SBE*) require high-quality data for their empirical research. For more than a decade, members of the *SBE* community have been developing and using a metadata standard, composed of almost twelve hundred metadata fields, known as the *Data Documentation Initiative (DDI)*, an XML format to disseminate, manage, and reuse data collected and archived for research [1]. In XML, the definition of schemas containing data constraints and the validation of data according to these constraints is commonly used to ensure a certain level of data quality. With the rise of the Web of Data, data professionals and institutions are very interested in having their data be DDI-RDFvered and used by publishing their data directly in RDF or at least publish accurate metadata about their data to facilitate data integration. Therefore, not only established vocabularies like SKOS are used; recently, members of the *SBE* and Linked Data community developed with the *DDI-RDF DDI-RDFvery Vocabulary (DDI-RDF)*[1] a means to expose *DDI* metadata as Linked Data.

For constraint formulation and validation of RDF data, several languages exist or are currently developed, like *Shape Expressions*, *Resource Shapes* or *Description Set Profiles*. *OWL 2* is also used as a constraint language under a closed world assumption. With its direct support of validation via SPARQL, SPINis very popular and certainly plays an important role for future developments in this field. It is particularly interesting as a means to validate arbitrary constraint languages by mapping them to SPARQL [2]. Yet, there is no clear favorite and none of the languages is able to meet all requirements raised by data practitioners. Further research and development therefore is needed.

In 2013, the W3C organized the RDF Validation Workshop[2], where experts from industry, government, and academia discussed first use cases for RDF constraint formulation and RDF data validation. In 2014, two working groups on RDF validation have been established to develop a language to express constraints on RDF data: the W3C RDF Data Shapes working group[3] and the DCMI RDF Application Profiles task group[4] which among others bundles the requirements of data institutions of the cultural heritage and *SBE* sector and represents them in the W3C group.

Within the DCMI task group, a collaboratively curated database of RDF validation requirements has been created which contains the findings of the working groups based on various case studies provided by data institutions [3] . It is publicly available and open for further contributions[5]. The database connects requirements to use cases, case studies and implementations and forms the basis of this paper. We distinguish 81 requirements to formulate RDF constraints; each of them corresponding to an RDF constraint type.

Thomas: bibliography error

To gain a better understanding about the role of certain requirements for data quality and in order to direct the further development of constraint languages, we collected constraints for commonly used vocabularies in the *SBE* domain, either from the vocabularies themselves or from domain and data experts. All in all, this lead to 213 constraints of 53 different types on three vocabularies. We let the experts classify the constraints according to the severity if they are violated. Furthermore, we classified the type of each constraint (corresponding to a requirement) based on its complexity ranging from types commonly found in vocabulary specifications (e.g., domain, range, and cardinality restrictions), over types that are simply stated using common constraint languages (e.g., language tag cardinality restrictions) to complex types that

[1] http://rdf-vocabulary.ddialliance.org/DDI-RDFvery.html

[2] http://www.w3.org/2012/12/rdf-val/

[3] http://www.w3.org/2014/rds/charter

[4] http://wiki.dublincore.org/index.php/RDF-Application-Profiles

[5] Online at http://purl.org/net/rdf-validation

involve complex data structures or need more sophisticated languages to be easily expressible (e.g., constraints on graph-based structures) (Section IV).

As we do not want to base our conclusions on the evaluations of vocabularies and constraint definitions alone, we conducted a large-scale experiment. For 115 constraints, we evaluated the data quality of 15,694 data sets (4.26 billion triples) of *SBE* research data on three common vocabularies in *SBE* sciences (DDI-RDF, QB, SKOS) obtained from 33 SPARQL endpoints. Based on the evaluation results, we formulated several hypotheses to direct the further development of constraint languages. To make valid general statements for all vocabularies, however, the hypotheses still have to be verified or falsified by evaluating the quality of data represented by more than three vocabularies (Section V).

In this paper, we discuss constraints on RDF data in general. Note that the data represented in RDF can be data in the sense of *SBE* sciences, but also metadata about published or unpublished data. We generally refer to both simply as RDF data and only distinguish between data and metadata in the data set descriptions and in the case that it matters for the purpose of this paper. The underlying *semantics* for RDF validation is UNA/CWA. RDF validation requires that different names represent different objects (*unique name assumption (UNA)*), whereas, *OWL 2* is based on the *non-unique name assumption (nUNA)*. Reasoning in *OWL 2* is based on the *open-world assumption (OWA)*, i.e., a statement cannot be inferred to be false if it cannot be proved to be true. On the other hand, RDF validation scenarios require the *closed-world assumption (CWA)*, i.e., a statement is inferred to be false if it cannot be proved to be true. Classical constraint languages are based on the CWA where constraints need to be satisfied only by named individuals. This ambiguity in semantics is one of the main reasons why *OWL 2* has not been adopted as a standard constraint language for RDF validation in the past. In case we use *OWL 2* constructs in terms of constraints, we adopt the same semantics (CWA) that is for RDF validation in general.

The data most often used in research within *SBE* sciences is *person-level data* (or more generally *record-unit data*, i.e., data collected about individuals, businesses, and households) in form of responses to studies or taken from administrative registers (such as hospital records, registers of births and deaths). The range of person-level data is very broad - including census, education, health data and business, social, and labor force surveys. This type of research data is held within data archives or data libraries after it has been collected, so that it may be reused by future researchers.

By its nature, person-level data is highly confidential and access is often only permitted for qualified researchers who must apply for access. Researchers typically represent their results as aggregated data in form of multi-dimensional tables with only a few columns; so-called *variables* such as *sex* or *age*. Aggregated data, which answers particular research questions, is derived from person-level data by statistics on groups or aggregates such as frequencies and arithmetic means. The purpose of publicly available aggregated data is to get a first overview and to gain an interest in further analyses on the underlying person-level data. Aggregated data is published in form of CSV files, allowing to perform calculations on the data.

For more detailed analyses, researchers refer to person-level data including additional variables needed to answer subsequent research questions like the comparison of studies between countries. A *study* represents the process by which a data set was generated or collected. Eurostat[6], the statistical office of the European Union, provides research findings in form of aggregated data (downloadable as CSV files) and its metadata at European level that enable comparisons between countries. The variable *formal childcare*[7] captures the measured availability of childcare services in percent over the population in European Union member states by the variables *year*, *duration* (in hours per week), *age* of the child, and *country*. Variables are constructed out of values (of one or multiple datatypes) and/or code lists. The variable *age*, e.g., may be represented by values of the datatype *xsd:nonNegativeInteger* or by a code list of age clusters (e.g., '0 to 10' and '11 to 20').

A very important and representative RDF validation case study within *SBE* sciences is the comparison of variables between data collections of different countries. Several vocabulary-specific constraints on RDF data are checked for each data collection to determine if variables measuring *age* - collected for different countries (age_{DE}, age_{UK}) - are comparable: (1) variable definitions must be available, (2) for each code a human-readable label has to be specified, (3) code lists must be structured properly, and (4) code lists must either be identical or at least similar. If a researcher only wants to get a first overview over the comparability of variables (use case 1), covering the first three constraints may be sufficient, i.e., the violation of the first three constraints is more serious than the violation of the last constraint. If the intention of the researcher is to perform more sophisticated comparisons (use case 2), however, the user may raise the severity level of the last constraint.

II. Related Work

For data archives, research institutes, and data libraries, RDF validation according to predefined constraints is a much sought-after feature, particularly as this is taken for granted in the XML world. DDI-XML documents, e.g., are validated against diverse XSDs[8]. As certain constraints cannot be formulated and validated by XSDs, so-called secondary-level validation tools like *Schematron*[9] have been introduced to overcome the limitations of XML validation. *Schematron* generates validation rules and validates XML documents according to them. With RDF validation, one can overcome drawbacks

[6] http://ec.europa.eu/eurostat

[7] Aggregated data and its metadata is available at: http://ec.europa.eu/eurostat/web/products-datasets/-/ilc_caindformal

[8] http://www.ddialliance.org/Specification/

[9] https://msdn.microsoft.com/en-us/library/aa468554.aspx

when validating XML documents[10]. It cannot be validated, e.g., if each code of a variable's code list is associated with a category (*R-86*). Additionally, it cannot be validated that if an element has a specific value, then certain child elements must be present (*R-71*). A comprehensive comparison of XML and RDF validation, however, is not within the scope of this paper.

A well-formed *RDF Data Cube* is an a RDF graph describing one or more instances of *qb:DataSet* for which each of the 22 integrity constraints[11], defined within the QB specification, passes. Each integrity constraint is expressed as narrative prose and, where possible, a SPARQL ASK query or query template. If the ASK query is applied to an RDF graph then it will return true if that graph contains one or more QB instances which violate the corresponding constraint [4]. Mader, Haslhofer, and Isaac investigated how to support taxonomists in improving SKOS vocabularies by pointing out quality issues that go beyond the integrity constraints defined in the SKOS specification [5].

III. Common Vocabularies in SBE Sciences

We took all well-established and newly developed *SBE* vocabularies into account and defined constraints for six vocabularies commonly used in or developed for the *SBE* sciences which are briefly introduced in the following. For three of them, we analyzed actual data according to constraint violations, as for these vocabularies large data sets are already published.

The *RDF Data Cube Vocabulary (QB)*[12] is a W3C recommendation for representing *data cubes*, i.e. multi-dimensional aggregated data, in RDF [6]. A *qb:DataStructureDefinition* contains metadata of the data collection. The variable *formal childcare*, e.g., is modeled as *qb:measure*, since it stands for what has been measured in the data collection. The variables *year*, *duration*, *age*, and *country* are *qb:dimensions*. Data values, i.e., the availability of childcare services in percent over the population, are collected in a *qb:DataSet*. Each data value is represented inside a *qb:Observation* which contains values for each dimension[13].

Physical Data Description (PHDD)[14] is a vocabulary to represent data in tabular format in RDF enabling further aggregations and calculations. The data could be either represented in records with character-separated values (CSV) or fixed length. *Eurostat* provides a CSV file, a two-dimensional table (*phdd:Table*) about *formal childcare* which is structured by a table structure (*phdd:TableStructure*, *phdd:Delimited*) including information about the character set (*ASCII*), the variable delimiter (,), the new line marker (*CRLF*), and the first line where the data starts (*2*). The table structure is related to table columns (*phdd:Column*) which are described by column descriptions (*phdd:DelimitedColumnDescription*). For the column containing the actual cell values in percent, the column position (*5*), the recommended data type (*xsd:non NegativeInteger*), and the storage format (*TINYINT*) is stated.

For more detailed analyses we refer to the person-level data collected for the series *EU-SILC (European Union Statistics on Income and Living Conditions)*[15]. Where data collection is cyclic, data sets may be released as *series*, where each cycle produces one or more data sets. The aggregated variable *formal childcare* is calculated on the basis of six person-level variables (e.g., *Education at pre-school*) for which detailed metadata is given (e.g., code lists) enabling researchers to replicate the results shown in aggregated data tables. The *DDI-RDF Discovery Vocabulary (DDI-RDF)* is a vocabulary to represent metadata on person-level data in RDF. The series (*DDI-RDF:StudyGroup*) *EU-SILC* contains one study (*DDI-RDF:Study*) for each year (*dcterms:temporal*) of data collection. *dcterms:spatial* points to the countries for which the data has been collected. The study *EU-SILC 2011* contains eight person-level data sets (*DDI-RDF:LogicalDataSet*) including person-level variables (*DDI-RDF:Variable*) like the six ones needed to calculate the aggregated variable *formal childcare*.

The *Simple Knowledge Organization System (SKOS)* is reused multiple times to build *SBE* vocabularies. The codes of the variable *Education at pre-school* (number of education hours per week) are modeled as *skos:Concepts* and a *skos:OrderedCollection* organizes them in a particular order within a *skos:member List*. A variable may be associated with a theoretical concept (*skos:Concept*). *skos:narrower* builds the hierarchy of theoretical concepts within a *skos:Concept Scheme* of a series. The variable *Education at pre-school* is assigned to the theoretical concept *Child Care* which is the narrower concept of *Education* - one of the top concepts of the series *EU-SILC*. Controlled vocabularies (*skos:ConceptScheme*), serving as extension and reuse mechanism, organize types (*skos:Concept*) of descriptive statistics (*DDI-RDF:SummaryStatistics*) like minimum, maximum, and arithmetic mean. *XKOS*[16] is a SKOS extension to describe formal statistical classifications like the International Standard Classification of Occupations (*ISCO*). and the Statistical Classification of Economic Activities in the European Community *NACE*. *DCAT* enables to represent data sets inside of data collections like portals, repositories, catalogs, and archives which serve as typical entry points when searching for data.

IV. Classification of Constraint Types and Constraints

To gain better insights into the role that certain types of constraints play for the quality of RDF data, we use two simple classifications: on the one hand, we classify RDF constraint types whether they are expressible by different types of constraint languages and on the other hand, we classify constraints formulated for a given vocabulary according to the perceived severity of its violation.

[10]http://www.xmlmind.com/xmleditor/_distrib/doc/xmltool/xsd_structure_limitations.html

[11]http://www.w3.org/TR/vocab-data-cube/#wf

[12]http://www.w3.org/TR/vocab-data-cube/

[13]The complete running example in RDF is available at: https://github.com/boschthomas/rdf-validation/tree/master/data/running-example

[14]https://github.com/linked-statistics/physical-data-description

[15]http://www.gesis.org/missy/eu/metadata/EU-SILC

[16]https://github.com/linked-statistics/xkos

Within the working groups, we identified by today 81 requirements to formulate RDF constraints (e.g., *R-75: minimum qualified cardinality restrictions*); each of them corresponding to an RDF constraint type[17]. Within a technical report, we explain each requirement/constraint type in detail and give examples for each expressed by different constraint languages [7]. We provide mappings to representations in Description Logics (DL) [8] to logically underpin each requirement and to determine which DL constructs are needed to express each constraint type. The classification of the 115 concrete constraints on three vocabularies according to severity levels is published in a technical report [9]. In the following, we summarize the results for the purpose of our evaluation.

A. Classification of Constraint Types according to the Expressivity of Constraint Languages

According to the expressivity of constraint languages, the complete set of constraint types encompasses three not disjoint sets of constraint types:

1) *RDFS/OWL Based*
2) *Constraint Language Based*
3) *SPARQL Based*

The modeling languages RDFS and OWL are typically used to formally specify vocabularies. *RDFS/OWL Based* denotes the set of constraint types which can be formulated with RDFS/OWL constructs using CWA/UNA semantics without reasoning and which are therefore commonly found within formal specifications of vocabularies. We determined this set based on the evaluation of the three vocabularies. *RDFS/OWL Based* constraints generally can be seen as a basic level of constraints ensuring that the data is consistent with the formally and explicitly specified intended semantics as well as the integrity of vocabularies' conceptual models about data.

Constraints of the constraint type *minimum qualified cardinality restrictions* (*R-74*), e.g., guarantee that individuals of given classes are connected by particular properties to at least n different individuals/literals of certain classes or data ranges. For *DDI-RDF*, a *minimum qualified cardinality restriction* can be obtained from a respective OWL axiom which ensures that a *disco:Questionnaire* has (*disco:question*) at least one *disco:Question*:

```
[   a owl:Restriction ; rdfs:subClassOf Questionnaire ;
    owl:minQualifiedCardinality 1 ;
    owl:onProperty question ;
    owl:onClass Question ] .
```

Constraint Language Based and *SPARQL Based* constraints are in contrast to *RDFS/OWL Based* constraints usually not (yet) explicitly defined within formal specifications of vocabularies. Instead, they are often defined within textual descriptions of vocabularies. Additionally, we let our domain and data experts define constraints when they agreed that violating the constraint would affect the usefulness of the data.

We further distinguish *Constraint Language Based* as the set of constraint types that can be expressed by common classical declarative high-level constraint languages like ShEx, ReSh, and DSP. There is a strong overlap between *RDFS/OWL Based* and *Constraint Language Based* constraint types as in many cases constraint types are expressible by classical constraint languages and OWL. SPARQL, however, is considered as a low-level implementation language in this context. In contrast to SPARQL, high-level constraint languages are comparatively easy to understand and constraints can be formulated more concisely. Declarative languages may be placed on top of SPARQL when using it as an implementation languages. For these *Constraint Language Based* constraints, we expect a straight-forward support in future constraint languages.

Context-specific exclusive or of property groups (*R-11*) is a constraint type which can be formulated by a high-level constraint language. Constraints of this type restrict individuals of given classes to have properties defined within exactly one of multiple mutually exclusive property groups. *skos:Concept*s, e.g., can have either *skos:definition* (when interpreted as theoretical concepts) or *skos:notation* and *skos:prefLabel* properties (when interpreted as codes), but not both:

```
ShEx: Concept {
    ( definition string ) |
    ( notation string , prefLabel string ) }
```

The set *SPARQL Based* encompasses constraint types that are not expressible by RDFS/OWL or common constraint languages but by plain SPARQL. For assessing the quality of thesauri, e.g., we concentrate on the graph-based structure and apply graph- and network-analysis techniques. An example of such constraints of the constraint type *structure* is that a thesaurus should not contain many orphan concepts, i.e., concepts without any associative or hierarchical relations, lacking context information valuable for search. As the complexity of this constraint is relatively high, it is only expressible by SPARQL and not directly understandable:

```
SELECT ?concept WHERE {
    ?concept a [rdfs:subClassOf* skos:Concept] .
    FILTER NOT EXISTS { ?concept ?p ?o .
        FILTER ( ?p IN ( skos:related, skos:relatedMatch,
                         skos:broader, ... ) ) . } }
```

SPARQL Based constraint types are today in most cases only expressible by plain SPARQL. Depending on their usefulness, a support in constraint languages should be considered.

B. Classification of Constraints according to the Severity of Constraint Violations

A concrete constraint is instantiated from one of the 81 constraint types and is defined for a specific vocabulary. It does not make sense to determine the severity of constraint violations of an entire constraint type, as the severity of the violation of a constraint depends on the individual context and vocabulary. *SBE* experts determined the default *severity level*[18] for each of the 115 constraints to indicate how serious the violation of the constraint is. We use the classification system of log messages in software development like *Apache Log4j*

[17]Constraint types and constraints are uniquely identified by alphanumeric technical identifiers like *R-71-CONDITIONAL-PROPERTIES*

[18]The possibility to define severity levels in vocabularies is in itself a requirement (*R-158*).

2 [10], the *Java Logging API*[19], and the *Apache Commons Logging API*[20] as many data practitioners also have experience in software development and software developers intuitively understand these levels. We simplify this commonly accepted classification system and distinguish the three severity levels *informational*, *warning* and *error*. Violations of *informational constraints* point to desirable but not necessary data improvements to achieve RDF representations which are ideal in terms of syntax and semantics of used vocabularies. *Warnings* are syntactic or semantic problems which typically should not lead to an abortion of data processing. *Errors* are syntactic or semantic errors which should cause the abortion of data processing. examples for each in the subsequent section?

Note that there is a correlation between the severity of a constraint and the classification according to the expressivity of constraint languages of its type: *RDFS/OWL Based* constraints are in many cases classified with an *error* level as they typically represent basic and important constraints; there is a reason why they have been included in the vocabulary specification. Although we provide default severity levels for each constraint, validation environments should enable users to adapt the severity constraints' levels according to their individual needs.

C. Examples

***RDFS/OWL Based* constraint types.** It is a common requirement to narrow down the value space of properties by an exhaustive enumeration of valid values (*R-30/37: allowed values*): *disco:CategoryStatistics*, e.g., can only have *disco:computationBase* relationships to the values *valid* and *invalid* of the datatype *rdf:langString* (default severity level: *error*). Consider the following *DL knowledge base* $\mathcal{K}$[21]:

$$\mathcal{K} = \{ \begin{array}{l} \texttt{CategoryStatistics} \equiv \forall\ \texttt{computationBase}. \\ \{\texttt{valid,invalid}\} \sqcap \texttt{langString}, \\ \texttt{Variable} \equiv \exists\ \texttt{concept.Concept}, \\ \texttt{Catalog} \sqsubseteq \forall\ \texttt{dataset.Dataset}, \\ \exists\ \texttt{isStructuredBy}.\top \sqsubseteq \texttt{Table}, \\ \top \sqsubseteq \forall\ \texttt{belongsTo.Concept}\ \} \end{array}$$

Existential quantifications (*R-86*) enforce that instances of given classes must have some property relation to individuals/literals of certain types. Variables, e.g., should have a relation to a theoretical concept (*informational*). The variable *Education at pre-school* is associated with the theoretical concept *Child Care*. The default severity level of this constraint is weak, as in most cases research can be continued without having information about the theoretical concept of a variable.

A *universal quantification* (*R-91*) contains all those individuals that are connected by a property only to individuals/literals of particular classes or data ranges. Only *dcat:Catalogs*, e.g., can have *dcat:dataset* relationships to *dcat:Datasets* (*error*). *Property domains* (*R-25, R-26*) and *property ranges* (*R-28, R-35*) constraints restrict domains and ranges of properties: Only *phdd:Tables*, e.g., can have *phdd:isStructuredBy* relationships (*error*) and *xkos:belongsTo* relations can only point to *skos:Concepts* (*error*).

It is often useful to declare a given (data) property as the *primary key* (*R-226*) of a class, so that a system can enforce uniqueness and build URIs from user inputs and imported data. In DDI-RDF, resources are uniquely identified by the property *adms:identifier*, which is therefore inverse-functional ($\texttt{funct identifier}^-$), i.e., for each *rdfs:Resource* x, there can be at most one distinct resource y such that y is connected by *adms:identifier*$^-$ to x (*error*). Keys, however, are even more general than *inverse-functional properties* (*R-58*), as a key can be a data property, an object property, or a chain of properties [11]. Thus and as there are different sorts of key, and as keys can lead to undecidability, DL is extended with the construct *keyfor* (`identifier keyfor Resource`) [12] which is implemented by the *OWL 2 hasKey* construct.

***Constraint Language Based* constraint types.**

Depending on property datatypes, two different literal values have a specific ordering with respect to operators like $<$(*R-43: literal value comparison*). Start dates (*disco:startDate*), e.g., must be before ($<$) end dates (*disco:endDate*).

In many cases, resources must be *members of controlled vocabularies* (*R-32*). If a QB dimension property, e.g., has a *qb:codeList*, then the value of the dimension property on every *qb:Observation* must be in the code list (*error*).

Default values (*R-31, R-38*) for objects/literals of given properties are inferred automatically when properties are not present in the data. The value *true* for the property *disco:isPublic* indicates that a *disco:LogicalDataSet* can be accessed by anyone. Per default, however, access to data sets should be restricted (*false*) (*informational*).

***SPARQL Based* constraint types.** *Data model consistency* constraints ensure the integrity of the data according to the intended semantics of vocabularies. Every *qb:Observation*, e.g., must have a value for each dimension declared in its *qb:DataStructureDefinition* (*error*) and no two *qb:Observations* in the same *qb:DataSet* can have the same value for all dimensions (*warning*). If a *qb:DataSet* D has a *qb:Slice* S, and S has an *qb:Observation* O, then the *qb:DataSet* corresponding to O must be D (*warning*).

Objects/literals can be declared to be ordered for given properties (*R-121/217: ordering*). Variables, questions, and codes, e.g., are typically organized in a particular order. If codes (*skos:Concept*) should be ordered, they must be members (*skos:memberList*) in an ordered collection (*skos:OrderedCollection*), the variable's code list (*informational*).

It is useful to declare properties to be *conditional* (*R-71*), i.e., if particular properties exist (or do not exist), then other properties must also be present (or absent). To get an overview over a series/study either an abstract, a title, an alternative title, or links to external descriptions should be provided. If an abstract and an external description are absent, however, a title or an alternative title should be given (*warning*). In

[19]http://docs.oracle.com/javase/7/docs/api/java/util/logging/Level.html

[20]http://commons.apache.org/proper/commons-logging/

[21]A *DL knowledge base* is a collection of formal statements which correspond to *facts* or what is known explicitly. For simplicity reasons, we do not state namespace prefixes.

case a variable is represented in form of a code list, codes may be associated with categories, i.e., human-readable labels (*informational*). The variable *Education at pre-school*, e.g., is represented as ordered code list without any categories.

For data properties, it may be desirable to restrict that values of predefined languages must be present for determined number of times (*R-48/49: language tag cardinality*): (1) It is checked if literal language tags are set. Some controlled vocabularies, e.g., contain literals in natural language, but without information what language has actually been used (*warning*). (2) Language tags must conform to language standards (*error*). (3) Some thesaurus concepts are labeled in only one, others in multiple languages. It may be desirable to have each concept labeled in each of the languages that are also used on the other concepts, as language coverage incompleteness for some concepts may indicate shortcomings of thesauri (*informational*) [5].

V. Evaluation

In this section, we describe our findings based on an automatic constraint checking of a large data set. Despite the large volume of the data set in general, we have to keep in mind that this study only uses data for three vocabularies. As described in Section III, for other vocabularies there is often not (yet) enough data openly available to draw general conclusions. The three vocabularies, however, are representative, cover different aspects of *SBE* data and are also a mixture of established vocabularies (QB, SKOS) and a vocabulary under development (DDI-RDF). Due to this limitation, we will present several hypotheses together with a rationale based on the results for the three vocabularies. If the hypotheses hold generally is still to be determined, nevertheless they should provide valuable insights for future developments of constraint languages.

A. Experimental Setup

On the three vocabularies (DDI-RDF, QB, SKOS), we identified and classified 115 constraints[22] which we implemented for the data validation. We ensured that the implementation of the constraints is equally distributed over the classes and vocabularies we have. We then evaluated the data quality of 15,694 data sets (4.26 billion triples) of *SBE* research data using these 115 constraints, obtained from 33 SPARQL endpoints.

QB and SKOS are well-established vocabularies which are widely adopted and accepted. DDI-RDF is a newly developed vocabulary which will be published in 2015.

Table I lists the number of data sets and the overall sizes in terms of triples for each of the vocabularies. We validated, i.a., (1) QB data sets published by the *Australian Bureau of Statistics*, the *European Central Bank*, and the *Organisation for Economic Co-operation and Development*, (2) SKOS thesauri like the *AGROVOC Multilingual agricultural thesaurus*, the *STW Thesaurus for Economics*, and the *Thesaurus for the Social Sciences*, and (3) DDI-RDF data sets provided by the *Microdata Information System*, the *Data Without Boundaries DDI-RDFvery Portal*, the *Danish Data Archive*, and the *Swedish National Data Service*.

TABLE I: Evaluated Data Sets for each Vocabulary

Vocabulary	Data Sets	Triples
QB	9,990	3,775,983,610
SKOS	4,178	477,737,281
DDI-RDF	1,526	9,673,055

In a technical report, we describe the evaluation in further detail [13]. Furthermore, we published the evaluation results for each data set in form of one document per SPARQL endpoint[23].

We use *SPIN*, a SPARQL-based way to formulate and check constraints, as basis to develop a validation environment[24] to validate RDF data according to constraints expressed by arbitrary constraint languages[25] [2].

The *RDF Validator* can directly be used to validate arbitrary RDF data for the three vocabularies. Additionally, own constraints on vocabularies can be defined using several constraint languages.

The SPIN engine checks for each resource if it satisfies all constraints, which are associated with its assigned classes, and generates a result RDF graph containing information about all constraint violations. There is one SPIN construct template for each constraint type. A SPIN construct template contains a SPARQL CONSTRUCT query which generates constraint violation triples indicating the subject and the properties causing constraint violations and the reason why constraint violations have been raised. A SPIN construct template creates constraint violation triples if all triple patterns within the SPARQL WHERE clause match.

B. Evaluation Results and Formulation of Hypotheses

Tables II and III show the results of the evaluation, more specifically the constraints and the constraint violations, which are caused by these constraints, in percent. The first numbers indicate the absolute amount of constraints and violations. The constraints and their raised violations are grouped by vocabulary, which type of language the constraints are expressed with, and their severity level. The numbers of evaluated triples and data sets differ between the vocabularies as we evaluated 3.8 billion QB, 480 million SKOS, and 10 million DDI-RDF triples. To be able to formulate hypotheses which apply for all vocabularies, we only use normalized relative values representing the percentage of constraints and violations belonging to the respective set of constraint types.

There is a strong overlap between *RDFS/OWL* and *Constraint Language Based* constraint types as in many cases

[22] https://github.com/boschthomas/rdf-validation/tree/master/constraints

[23] Available at: https://github.com/boschthomas/rdf-validation/tree/master/evaluation/data-sets/data-cube

[24] Online demo available at: http://purl.org/net/rdfval-demo, source code at: https://github.com/boschthomas/rdf-validator

[25] SPIN mappings available at: https://github.com/boschthomas/rdf-validation/tree/master/SPIN

constraint types are expressible by classical constraint languages and OWL. This is the reason why the percentage values of constraints and violations grouped by the classification of constraint types according to the expressivity of constraint languages do not accumulate to 100%.

As the evaluation is based on three vocabularies, we cannot make valid general statements for all vocabularies, but we can formulate several hypotheses to direct the further development of constraint languages. As these hypotheses cannot be proved yet, they still have to be verified or falsified by evaluating the quality of data represented by further well-established and newly developed vocabularies.

TABLE II: Constraints and Constraint Violations (1)

	DDI-RDF		**QB**	
	C	CV	C	CV
	78	3,575,002	20	45,635,861
SPARQL	29.5	34.7	**60.0**	**100.0**
CL	**64.1**	**65.3**	40.0	0.0
RDFS/OWL	**66.7**	**65.3**	40.0	0.0
info	**56.4**	**52.6**	0.0	0.0
warning	11.5	29.4	15.0	**99.8**
error	32.1	18.0	**85.0**	0.3

C (constraints), CV (constraint violations)

TABLE III: Constraints and Constraint Violations (2)

	SKOS		***Total***	
	C	CV	C	CV
	17	5,540,988	115	54,751,851
SPARQL	**100.0**	**100.0**	**63.2**	**78.2**
CL	0.0	0.0	34.7	21.8
RDFS/OWL	0.0	0.0	35.6	21.8
info	**70.6**	41.2	**42.3**	31.3
warning	29.4	**58.8**	18.7	**62.7**
error	0.0	0.0	**39.0**	6.1

C (constraints), CV (constraint violations)

Almost 2/3 of all constraints, nearly 1/3 of the DDI-RDF, 60% of the QB, and all SKOS constraints are *SPARQL Based*. For well-established vocabularies, the most formulated constraints are *SPARQL Based* (80%). For newly developed vocabularies, however, the most expressed constraints are *RDFS/OWL Based* (2/3). Nearly 80% of all violations are caused by *SPARQL*, 1/5 by *Constraint Language*, and 1/5 by *RDFS/OWL Based* constraints.

Hypothesis 1 *The facts that 80% of all violations are raised by SPARQL Based constraints and that 2/3 of all constraints are SPARQL Based, increases the importance to formulate constraints, which up to now can only be expressed in SPARQL, using high-level constraint languages. Data quality can be significantly improved when suitable constraint languages are developed which enable to define SPARQL Based constraints in an easy, concise, and intuitive way. Thereby, the more elaborate a vocabulary is, the more sophisticated and complex constraints are specified using SPARQL.*

These constraints are of such complexity that up to now in most cases they can only be expressed by plain SPARQL. It should be an incentive for language designers to devise languages which are more intuitive than SPARQL in a way that also domain experts, which are not familiar with SPARQL, can formulate respective constraints.

Hypothesis 2 *The fact that only 1/5 of all violations result from RDFS/OWL Based constraints, even though 1/3 of all constraints are RDFS/OWL Based, indicates good data quality for all vocabularies with regard to their formal specifications.*

Hypothesis 3 *As 1/3 of all constraints are RDFS/OWL Based, the first step to make progress in the further development of constraint languages is to cover the constraint types which can already be formulated using RDFS and OWL.*

While 2/3 of the DDI-RDF violations result from *RDFS/OWL Based* constraints, QB and SKOS violations are only raised by *SPARQL Based* constraints.

Hypothesis 4 *For well-established vocabularies, RDFS/OWL Based constraints are almost completely satisfied which indicates generally a very impressive data quality, at least in the SBE domain and for the basic requirements. For newly developed vocabularies, however, data quality is poor as RDFS/OWL Based constraints are not fulfilled.*

For DDI-RDF, data providers still have to understand the vocabulary and of course, data can not have high quality if the specification is not yet stable. It is likely that a newly developed vocabulary is still subject of constant change and that early adopters did not properly understand its formal specification. Thus, published data may not be consistent with the current draft of its conforming vocabulary. In case newly developed vocabularies turn into well-established ones, data providers are experienced in publishing their data in conformance with these vocabularies and formal specifications are more elaborated. As a consequence, *RDFS/OWL Based* constraints are satisfied to a greater extend which leads to better data quality. The reason why we only defined *SPARQL Based* constraints for assessing the quality of thesauri is that literature and practice especially concentrate on graph-based structures of thesauri by applying graph- and network-analysis techniques which can only be implemented by SPARQL.

Almost 40% of all constraints are *error*, more than 40% are *informational*, and nearly 20% are *warning* constraints. *Informational* constraints caused almost 1/3 and *warning* constraints narrowly 2/3 of all violations.

Hypothesis 5 *Although 40% of all constraints are error constraints, the percentage of severe violations is very low, compared to about 2/3 of warning and 1/3 of informational violations. This implies that data quality is high with regard to the severity level of constraints and that proper constraint languages can significantly improve data quality beyond fundamental requirements.*

85% of the QB constraints are *error* constraints. More than 50% of the DDI-RDF and SKOS constraints, however, are *informational* constraints. 1/6 of the DDI-RDF violations are caused by *error* constraints and almost all QB violations and 59% of the SKOS violations are caused by *warning* constraints.

Hypothesis 6 *For well-established vocabularies, data quality is high as serious violations rarely appear. For newly developed vocabularies, however, data quality is worse as serious violations occur partially.*

Especially for newly developed vocabularies, constraint languages should be used to a larger extend to define appropriate constraints in order to detect and solve severe violations.

80% of the violations which are raised by either *RDFS/OWL* or *Constraint Language Based* constraints are caused by constraints with the severity level *informational* (see Table IV) and almost all (94%) of the violations which are caused by *SPARQL Based* constraints are raised by *warning* constraints. Approx. 1/2 of all constraints are *informational* constraints regardless how their types are classified according to the expressivity of constraint languages.

TABLE IV: Language Expressivity and Severity Level

	RDFS/OWL		***CL***		***SPARQL***	
	C	CV	C	CV	C	CV
info	**52.5**	**79.64**	**55.2**	**79.60**	**45.1**	4.39
warning	18.0	20.28	15.5	20.27	19.6	**94.17**
error	29.5	0.08	29.3	0.13	35.3	1.43

C (constraints), CV (constraint violations)

Hypothesis 7 *Whatever language is used to formulate constraints, 1/2 of all constraints are informational, 1/3 are error, and 1/5 are warning constraints. Violations caused by constraints expressed by RDFS/OWL or high-level constraint languages are of low severity, whereas the violation of constraints formulated by SPARQL is more serious. There is a significant demand for languages that support the expression of SPARQL Based constraints as 94% of all violations are caused by these constraints whose severity level is warning.*

SPARQL seems to be used to formulate constraints whose violation is of higher severity than the violation of constraints expressed by either RDFS/OWL or high-level constraint languages. This is a surprising finding as the violation of *RDFS/OWL Based* constraints should be of higher severity than *informational* since these constraints are expected to be basic constraints as they are in many cases part of vocabularies' formal specifications.

VI. Conclusion and Future Work

We identified and published by today 81 types of constraints that are required by various stakeholders for data applications. In close collaboration with several domain experts for the social, behavioural, and economic sciences (*SBE*), we formulated 115 constraints on three different vocabularies (DDI-RDF, QB, and SKOS) and classified them according to their severity level and whether their type is expressible by different types of constraint languages - RDFS/OWL, high-level constraint languages, and SPARQL. We evaluated the data quality of 15,694 data sets (4.26 billion triples) of research data for the *SBE* sciences obtained from 33 SPARQL endpoints.

Based on the evaluation results, we formulated several hypotheses to direct the further development of constraint languages. We regard to the results as hypotheses, as their general applicability beyond the examined vocabularies and for other domains is still to be confirmed. The main hypotheses are:

1) *Data quality can be significantly improved when suitable constraint languages are developed which enable to define constraints, which up to now can only be expressed by plain SPARQL, in an easy, concise, and intuitive way. Thereby, the more elaborate a vocabulary is, the more sophisticated and complex constraints are specified using SPARQL.*
2) *The fact that only 1/5 of all violations result from RDFS/OWL Based constraints, even though 1/3 of all constraints are RDFS/OWL Based, indicates good data quality for all vocabularies with regard to their formal specifications.*
3) *Although 40% of all constraints are error constraints, the percentage of severe violations is very low, compared to about 2/3 of warning and 1/3 of informational violations. This implies that data quality is high with regard to the severity level of constraints and that proper constraint languages can significantly improve data quality beyond fundamental requirements.*
4) *Whatever language is used to formulate constraints, 1/2 of all constraints are informational, 1/3 are error, and 1/5 are warning constraints. Violations caused by constraints expressed by RDFS/OWL or high-level constraint languages are of low severity, whereas the violation of constraints formulated by SPARQL is more serious. There is a significant demand for languages that support the expression of SPARQL Based constraints causing 94% of all violations.*

We have been really impressed by the high quality of the QB and SKOS data. This is in contrast to the sometimes heard rumour that Linked Open Data lacks quality. We are actively involved in the further development and implementation of constraint languages and will use the results presented in the paper to set priorities on features where we expect the highest impact on the data quality of real-life data in the *SBE* domain.

References

[1] M. Vardigan, P. Heus, and W. Thomas, "Data Documentation Initiative: Towards a Standard for the Social Sciences," *International Journal of Digital Curation*, vol. 3, no. 1, pp. 107–113, 2008. [Online]. Available: http://www.ijdc.net/index.php/ijdc/article/view/66

[2] T. Bosch and K. Eckert, "Towards Description Set Profiles for RDF using SPARQL as Intermediate Language," in *Proceedings of the DCMI International Conference on Dublin Core and Metadata Applications (DC 2014)*, Austin, Texas, USA, 2014.

[3] ——, "Requirements on RDF Constraint Formulation and Validation," in *Proceedings of the DCMI International Conference on Dublin Core and Metadata Applications (DC 2014)*, Austin, Texas, USA, 2014, http://dcevents.dublincore.org/IntConf/dc-2014/paper/view/257. [Online]. Available: http://dcevents.dublincore.org/IntConf/dc-2014/paper/view/257

[4] R. Cyganiak and D. Reynolds, "The RDF Data Cube Vocabulary," W3C, W3C Recommendation, January 2014, http://www.w3.org/TR/vocab-data-cube/. [Online]. Available: http://www.w3.org/TR/vocab-data-cube/
[5] C. Mader, B. Haslhofer, and A. Isaac, "Finding Quality Issues in SKOS Vocabularies," in *Proceedings of the Second International Conference on Theory and Practice of Digital Libraries*, ser. TPDL'12. Berlin, Heidelberg: Springer-Verlag, 2012, pp. 222–233. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-33290-6_25
[6] R. Cyganiak, S. Field, A. Gregory, W. Halb, and J. Tennison, "Semantic Statistics: Bringing Together SDMX and SCOVO," in *Proceedings of the International World Wide Web Conference (WWW 2010), Workshop on Linked Data on the Web*, ser. CEUR Workshop Proceedings, C. Bizer, T. Heath, T. Berners-Lee, and M. Hausenblas, Eds., vol. 628, 2010. [Online]. Available: http://ceur-ws.org/Vol-628/ldow2010_paper03.pdf
[7] T. Bosch, A. Nolle, E. Acar, and K. Eckert, "RDF Validation Requirements - Evaluation and Logical Underpinning," *Computing Research Repository (CoRR)*, vol. abs/1501.03933, 2015, http://arxiv.org/abs/1501.03933. [Online]. Available: http://arxiv.org/abs/1501.03933
[8] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, Eds., *The Description Logic Handbook: Theory, Implementation, and Applications*. New York, NY, USA: Cambridge University Press, 2003.
[9] T. Bosch, B. Zapilko, J. Wackerow, and K. Eckert, "RDF Constraints to Validate Rectangular Data and Metadata on Person-Level Data, Aggregated Data, Thesauri, and Statistical Classifications," *Computing Research Repository (CoRR)*, vol. abs/1504.04479, 2015, http://arxiv.org/abs/1504.04479. [Online]. Available: http://arxiv.org/abs/1504.04479
[10] Apache Software Foundation, "Apache Log4j 2 v. 2.3 User's Guide," Apache Software Foundation, Tech. Rep., May 2015, http://logging.apache.org/log4j/2.x/log4j-users-guide.pdf. [Online]. Available: http://logging.apache.org/log4j/2.x/log4j-users-guide.pdf
[11] M. Schneider, "OWL 2 Web Ontology Language RDF-Based Semantics," W3C, W3C Recommendation, October 2009, http://www.w3.org/TR/2009/REC-owl2-rdf-based-semantics-20091027/. [Online]. Available: http://www.w3.org/TR/2009/REC-owl2-rdf-based-semantics-20091027/
[12] C. Lutz, C. Areces, I. Horrocks, and U. Sattler, "Keys, Nominals, and Concrete Domains," *Journal of Artificial Intelligence Research*, vol. 23, no. 1, pp. 667–726, Jun. 2005. [Online]. Available: http://dl.acm.org/citation.cfm?id=1622503.1622518
[13] T. Bosch, B. Zapilko, J. Wackerow, and K. Eckert, "An Evaluation of Metadata and Data Quality on Person-Level, Aggregated, Thesauri, Statistical Classifications, and Rectangular Data Sets," *Computing Research Repository (CoRR)*, vol. abs/1504.04478, 2015, http://arxiv.org/abs/1504.04478. [Online]. Available: http://arxiv.org/abs/1504.04478