ABSTRACT
The Web of Linked Data grows rapidly and already contains data originating from hundreds of data sources. The quality of data from those sources is very diverse, as values may be out of date, incomplete or incorrect. Moreover, data sources may provide conflicting values for a single real-world object.

In order for Linked Data applications to consume data from this global data space in an integrated fashion, a number of challenges have to be overcome. One of these challenges is to rate and to integrate data based on their quality. However, quality is a very subjective matter, and finding a canonic judgement that is suitable for each and every task is not feasible.

To simplify the task of consuming high-quality data, we present Sieve, a framework for flexibly expressing quality assessment methods as well as fusion methods. Sieve is integrated into the Linked Data Integration Framework (LDIF), which handles Data Access, Schema Mapping and Identity Resolution, all crucial preliminaries for quality assessment and fusion.

We demonstrate Sieve in a data integration scenario importing data from the English and Portuguese versions of DBpedia, and discuss how we increase completeness, conciseness and consistency through the use of our framework.

Categories and Subject Descriptors
H.4 [Information Systems Applications]: Miscellaneous;
H.2.5 [Information Systems]: Database Management—
Heterogeneous databases

Keywords
Linked Data, RDF, Data Integration, Data Quality, Data Fusion, Semantic Web

1. INTRODUCTION
The Web of Linked Data has seen an exponential growth over the past five years[1]. From 12 Linked Data sets catalogued in 2007, the Linked Data cloud has grown to almost 300 data sets encompassing approximately 31 billion triples, according to the most recent survey conducted in September 2011 [10].

The information contained in each of these sources often overlaps. In fact, there are approximately 500 million explicit links between data sets [10], where each link indicates that one data set ‘talks about’ a data item from another data set. Further overlapping information may exist, even though no explicit links have been established yet. For instance, two data sets may use different identifiers (URIs) for the same real world objects (e.g. Bill Clinton has an identifier in the English and the Portuguese DBpedia). Similarly, two different attribute identifiers may be used for equivalent attributes (e.g. both foaf:name and dbprop:name contain the name ‘Bill Clinton’).

Applications that consume data from the Linked Data cloud are confronted with the challenge of obtaining a homogenized view of this global data space [8]. The Linked Data Integration Framework (LDIF) was created with the objective of supporting users in this task. LDIF is able to conflate multiple identifiers of the same object into a canonical URI (identity resolution), while mapping equivalent attributes and class names into a homogeneous target representation (schema mapping).

As a result of such a data integration process, multiple values for the same attribute may be observed – e.g. originating from multiple sources. For attributes that only admit one value (e.g. total area or population of a city) this represents a conflict for the consumer application to resolve.

With the objective of supporting user applications in dealing with such conflicts, we created Sieve - Linked Data Quality Assessment and Data Fusion.

Sieve is included as a module in LDIF, and can be customized for user applications programmatically (through an open source Scala API) and through configuration parameters that describe users’ task-specific needs. Sieve includes a Quality Assessment module and a Data Fusion module. The Quality Assessment module leverages user-selected metadata as quality indicators to produce quality assessment scores through user-configured scoring functions. The Data Fusion module is able to use quality scores in order to perform user-configurable conflict resolution tasks.

In this paper we demonstrate Sieve through a data integration scenario involving the internationalized editions of DBpedia, which extracts structured data from Wikipedia.
In this scenario we consume data from the English and Portuguese DBpedia editions and wish to obtain data about Brazilian municipalities that is more complete, concise, consistent and up to date than in their original sources.

In Section 2 we describe the LDIF architecture, placing the newly created Sieve modules in the larger context of data integration. Subsequently, we explain the Quality Assessment (Section 3) and Data Fusion (Section 4) modules in more detail. In Section 5 we describe our demonstration setup and in Section 6 we discuss the results. Finally, in Section 7 we make final comments and point to future directions.

2. LDIF ARCHITECTURE

The Linked Data Integration Framework (LDIF) [13] offers a modular architecture to support a wide range of applications in producing a homogenized view over heterogeneous data originating from diverse sources. The architecture of LDIF is displayed in Figure 1.

Data sets are imported into LDIF through Web Data Access Modules. Currently supported data access modules include the Triple/Quad Dump Import, which loads files encoded in all major RDF formats from the disk or the Web, a Crawler Import which relies on the dereferenceability of URIs to obtain RDF by navigating through the Web of Linked Data, and finally the SPARQL Import, which allows SQL-like queries to import data from SPARQL servers [7] on the Web.

As data is imported, its provenance or lineage is also recorded. Provenance data contains information about the history of data items, for example their origins. For provenance tracking, LDIF relies on the Named Graphs data model [6]. Information is stored as quadruples (quads) in the form <subject, predicate, object, graph>.

The fourth element of the quad is interpreted as a graph identifier, a way to refer to a group of triples. LDIF is agnostic to the provenance model used, allowing users to attach arbitrary provenance metadata to their named graphs. While performing import jobs, LDIF also automatically adds its own provenance metadata that record time of import, URL source, etc. as displayed in Figure 2. Please note that URIs in the figure were shortened and the second line was wrapped due to space restrictions.

Since data from multiple sources may use different vocabularies to describe overlapping information, LDIF includes a Schema Mapping step, which relies on the R2R Framework [2]. R2R alleviates schema heterogeneity by translating source schema element names, structure and values into a user-configured target representation. Through its R2R Mapping Language, the framework supports simple or more complex (1-to-n and n-to-1) transformations. Moreover, it is able to normalize different units of measurement, perform string transformations or manipulate data types and language tags.

Furthermore, since data may also include multiple identi-
The quality of the data for some intended use. There may be a choice of several scoring functions for producing a score based on a given indicator. Depending on the quality dimension to be assessed and the chosen quality indicators, scoring functions range from simple comparisons, like “assign true if the quality indicator has a value greater than X”, over set functions, like “assign true if the indicator is in the set Y”, aggregation functions, like “count or sum up all indicator values”, to more complex statistical functions, text-analysis, or network-analysis methods.

An Aggregate Metric is a user-specified aggregate assessment metric built out of individual assessment metrics. These aggregations produce new assessment values through the average, sum, max, min or threshold functions applied to a set of assessment metrics. Aggregate assessment metrics are better visualized as trees, where an aggregation function is applied to the leaves and combined up the tree until a single value is obtained. The functions to be applied at each branch are specified by the users.

### 3.1 Sieve Quality Assessment Module

In Sieve, users have the flexibility of defining relevant indicators and respective scoring functions for their specific quality assessment task. A number of scoring functions are provided with Sieve which can be configured for a specific task through an XML file. Moreover, users can extend the current list of scoring functions and configuration options by implementing a simple programmatic interface that takes in a list of metadata values and outputs a real-valued score.

The complete list of supported scoring functions is available from the Sieve specification website.

- **TimeCloseness** – measures the distance between the input date from the provenance graph to the current date, with more recent data receive scores closer to 1.
Through the Sieve Quality Assessment Specification Language, users can define AssessmentMetric elements that use a specific ScoringFunction implementation to generate a score based on a given indicator and parameters provided in the Input element. Multiple AssessmentMetric elements can be composed into an AggregatedMetric element, yielding a score which is an aggregation of the component scores.

Listing 1 shows a quality assessment policy that outputs scores for the dimensions “Recency” and “Reputation.” In the Recency dimension, Sieve will use the TimeCloseness function in order to measure the distance between two dates: a date input as an indicator, and the current date. In this case, the configuration used the property lastUpdated as indicator (line 6), and a range parameter (in days) to normalize the scores. The function outputs a score between 0 and 1, where values closer to 1 indicate that the two dates are very close. The computed score will be output as value for the sieve:recency property. Consequently, the more recently updated graphs are ranked higher by this function.

In the Reputation dimension, Sieve will take a parameter list (line 11) with a space-separated list of graphs to trust. These graphs will be scored by the function Preference from higher to lower priority in order of appearance in the list parameter. Consequently, in the example at hand, values originating from the Portuguese DBpedia version will take higher priority over those originating from the English version (line 12).

### Listing 1: Sieve Data Quality Assessment Specification Example

```
<AssessmentMetric id="sieve:recency">
    <Param name="timeSpan" value="7"/>
</AssessmentMetric>
<AssessmentMetric id="sieve:reputation">
</AssessmentMetric>
```

In contrast to our framework, we provide a stricter separation of data quality assessment functions and fusion functions. In our framework, the function `TrustYourFriends` combines two aspects: One aspect of quality assessment: the assignment of higher ‘reputation’ scores to some sources; and one aspect of data fusion: prefer values with highest scores in a given indicator (in this case, reputation). Similarly, the fusion function `KeepUpToDate` can be expressed in our framework by preferring values with higher scores in the “Recency” indicator.

In Sieve, fusion functions are basically of two types. Filter Functions (deciding strategies) remove some or all values from the input, according to some quality metric. One example filter function is: keep the value with the highest score for a given metric. Transform Functions (mediating strategies) operate over each value in the input, generating a new list of values built from the initially provided ones. A wide application on the part of the user can be built by using several transformations in a row. In the following example, a new metric is created from existing ones (e.g. KeepUpToDate, which takes the most recent value), or mediates the creation of a new value from the existing ones (e.g. Average).

```
<AssessmentMetric id="sieve:recency">
    <Param name="timeSpan" value="7"/>
</AssessmentMetric>
<AssessmentMetric id="sieve:reputation">
</AssessmentMetric>
```

4. DATA FUSION

In the context of data integration, Data Fusion is defined as the “process of fusing multiple records representing the same real-world object into a single, consistent, and clean representation” [5]. Data Fusion is commonly seen as a third step following schema mapping and identity resolution, as a way to deal with conflicts that either already existed in the original sources or were generated by integrating them. For instance, by mapping two equivalent attributes from different schemata, the system may generate one canonical attribute with two different values. Similarly, the identity resolution step may collapse two object identifiers into one, requiring that applications deal with the multiple attribute values originating from each data source.

Our data fusion framework is inspired by the work of Bleiholder and Naumann [3]. They described a framework for data fusion in the context of relational databases that includes three major categories of conflict handling strategies:

- Conflict-ignoring strategies, which defer conflict resolution to the user. For instance, the strategy `PassItOn` simply relays conflicts to the user or application consuming integrated data.
- Conflict-avoiding strategies, which apply a unique decision to all data. For instance, the strategy `TrustYourFriends` prefers data from specific data sources.
- Conflict-resolution strategies, which decide between existing data (e.g. `KeepUpToDate`, which takes the most recent value), or mediate the creation of a new value from the existing ones (e.g. `Average`).

```
<AssessmentMetric id="sieve:recency">
    <Param name="timeSpan" value="7"/>
</AssessmentMetric>
<AssessmentMetric id="sieve:reputation">
</AssessmentMetric>
```

In our framework, the function `TrustYourFriends` combines two aspects: One aspect of quality assessment: the assignment of higher ‘reputation’ scores to some sources; and one aspect of data fusion: prefer values with highest scores in a given indicator (in this case, reputation). Similarly, the fusion function `KeepUpToDate` can be expressed in our framework by preferring values with higher scores in the “Recency” indicator.

```
<AssessmentMetric id="sieve:recency">
    <Param name="timeSpan" value="7"/>
</AssessmentMetric>
<AssessmentMetric id="sieve:reputation">
</AssessmentMetric>
```
Figure 4: Illustration of the Data Fusion process. Example data originating from the DBpedia Extraction from the English and Portuguese Wikipedia editions are sent through Sieve, generating a cleaner representation. A set of metadata quads (containing quality information), a set of entity descriptions containing the properties to be fused and a data fusion specification.

Listing 2: Sieve Data Fusion Specification Example

```
<Sieve>
  <Fusion>
    <Class name="dbpedia:Settlement">
      <Property name="rdfs:label"/>
      <FusionFunction class="PassItOn"/>
      <Property name="dbpedia-owl:areaTotal">
        <FusionFunction class="KeepSingleValueByQualityScore" metric="sieve:reputation"/>
      </Property>
      <Property name="dbpedia-owl:populationTotal">
        <FusionFunction class="KeepSingleValueByQualityScore" metric="sieve:recency"/>
      </Property>
    </FusionFunction>
  </Fusion>
</Sieve>
```

The complete list of supported fusion functions is available from the Sieve specification website.

### 4.1 Sieve Data Fusion Module

Similarly to the quality assessment module, the data fusion module can also be configured through XML. The Sieve Data Fusion Specification language takes a property-centric perspective. The user has the flexibility of deciding what action to perform for each property of a class that requires data fusion. Actions range from ignoring conflicts (e.g. PassItOn) to filtering out information (e.g. Filter) based on quality indicators, and can include also value transformations (e.g. Average).

The quality indicators used for deciding on which data fusion operation to perform are provided by the quality assessment module. The data fusion component takes as input a set of metadata quads (containing quality information), a set of entity descriptions containing the properties to be fused and a data fusion specification.

<table>
<thead>
<tr>
<th>Property</th>
<th>English Value</th>
<th>Portuguese Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>areaTotal</td>
<td>1437 km²</td>
<td>1436.850 km²</td>
</tr>
<tr>
<td>elevation</td>
<td>678 m</td>
<td>null</td>
</tr>
<tr>
<td>last edit</td>
<td>05/Oct/2011</td>
<td>08/Nov/2011</td>
</tr>
<tr>
<td>changes last year</td>
<td>17</td>
<td>167</td>
</tr>
</tbody>
</table>

range of fusion functions have been proposed [4], and are being implemented in Sieve, including:

- **Filter** – removes all values for which the input quality assessment metric is below a given threshold
- **KeepSingleValueByQualityScore** – keeps only the value with the highest quality assessment
- **Average, Max, Min** – takes the average, chooses the maximum, or minimum of all input values for a given numeric property
- **First, Last, Random** – takes the first, last or the element at some random position for a given property
- **PickMostFrequent** – selects the value that appears most frequently in the list of conflicting values

The quality indicators used for deciding on which data fusion operation to perform are provided by the quality assessment module. The data fusion component takes as input a set of metadata quads (containing quality information), a set of entity descriptions containing the properties to be fused and a data fusion specification.

Listing 2 shows a data fusion specification that acts on every instance of `dbpedia:Settlement` and consolidates information for the properties `areaTotal` and `populationTotal` from the DBpedia Ontology. Both properties use the fusion function `KeepSingleValueByQualityScore`, which removes all but the highest ranked value, where the ranking is determined by a quality indicator calculated in the previous step. The fusion for the `areaTotal` property takes the value with the highest `ldif:reputation` (line 9), while `populationTotal` takes the value with the highest `ldif:recency` (line 13). Values for `rdfs:label` are simply repeated to the output, as multiple values for this property are often useful for language customiza-
In order to demonstrate the capabilities of Sieve, we have collected data about municipalities in Brazil from different Wikipedia editions, namely the English- and Portuguese-language Wikipedia editions. According to the Brazilian Institute for Geography and Statistics, there are 5,565 municipalities in Brazil. However, this information may be absent from DBpedia due to incomplete or missing Wikipedia infoboxes, missing mappings in the DBpedia Mapping Wiki, or irregularities in the data format that were not resolved by the DBpedia Extraction Framework. Furthermore, a particular Wikipedia edition may be out of date or incorrect. See Figure 4 for the schematics of our demonstration.

In this section we show how such problems can be alleviated by fusing data from multiple sources. We start by describing the data sets employed, and subsequently present the results obtained for (slight modifications of) the specifications presented in Listing 1 and Listing 2.

5.1 Data Sets
The data sets employed in our use case were obtained from the English and Portuguese dumps of DBpedia 3.7. We collected mapping-based properties, inter-language links, instance types and provenance information, which were then fed as quads into Sieve through LDIF.

Target attributes.
For the sake of simplicity of explanation, we will focus our discussion on a few properties. Namely, we have collected the total population, total area and the founding date of each municipality. Table 4.1 shows the distribution of values that occur only in DBpedia English but not in DBpedia Portuguese (second column: only en), and vice-versa (third column: only pt). The table also shows properties that occurred in both DBpedia dumps (fourth column: redundant) with the same values, and properties that occurred in both data sets, but with different values (fifth column: conflicting).

Inter-language Links.
Wikipedia pages offer links to other language editions in the left-hand side menu in each article. We collected those links and represented them as owl:sameAs links that were provided as input to the LDIF engine. Through the identity resolution module (Silk) and the URI translation module, these identity links will be used to merge object descriptions into one URI per object.

Instance Types.
Wikipedia editors commonly create so-called “list pages”, which are used to organize collections of links to pages matching certain criteria. We have used a page listing all Brazilian municipalities to derive an extra source of instance type and country location from the Wikipedia page. We used this set as our target universe, that is, the complete set of URIs to be obtained after integration.

Pre-processing provenance.
The DBpedia Extraction Framework tracks the source article from which each property was extracted. For the purpose of this demonstration, we normalized the provenance information from the extraction framework to associate every property value extracted to its original source page in Wikipedia. For each source page, we collected the last update date from the Wikipedia dump, and included it in the LDIF provenance graph.

All of the data collected was serialized in the NQuads format and fed into LDIF for processing, resulting in one integrated NQuads output file. The next section discusses the results of this integration process.

6. RESULTS
Data Integration is commonly applied in order to increase data quality along at least three dimensions: completeness, conciseness and consistency. In the use case described in this paper, the task required retrieving 3 attributes (areaTotal, foundingDate, populationTotal) for all 5565 objects (Brazilian municipalities). According to Bleiholder and Naumann, the extensional completeness (data level), can be measured in terms of the proportion of target URIs found in the output (Equation 1), while the intensional completeness (schema level) can be measured by the proportion of target properties found in the output (Equation 2).

\[
\text{extensional completeness} = \frac{|\text{uniq. obj. in data set}|}{|\text{all uniq. obj. in universe}|} \tag{1}
\]

\[
\text{intensional completeness} = \frac{|\text{uniq. attr. in data set}|}{|\text{all uniq. attr. in universe}|} \tag{2}
\]

\[\text{http://dbpedia.org/Downloads37}\]
Table 2: Impact of the data integration process in quality indicators.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>areaTotal</td>
<td>7.31%</td>
<td>71.28%</td>
<td>71.32%</td>
<td>-10.20%</td>
<td>-9.52%</td>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>foundingDate</td>
<td>4.22%</td>
<td>1.06%</td>
<td>5.27%</td>
<td>+0.34%</td>
<td>-</td>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>populationTotal</td>
<td>7.58%</td>
<td>71.32%</td>
<td>71.41%</td>
<td>+10.49%</td>
<td>+9.31%</td>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

For the original data sets, the English DBpedia contained 2097/5565 municipalities, which means that the extensional completeness was 37% before the integration process, while DBpedia Portuguese contained 3979/5565 municipalities (extensional completeness of 71%). After integration, 3979/5565 municipalities were found, increasing extensional completeness in 36% and 0.5% for the DBpedia English and DBpedia Portuguese respectively.

In order to provide a more fine-grained analysis of the impact of the integrated data set with regard to the original sources, we have defined another measure of completeness that takes into consideration the instantiations of properties. That is, it measures the proportion of objects that contain a value for a given property in relation to the universe of objects (Equation 3).

\[
\text{Completeness}(p) = \frac{||\text{obj. with property } p \text{ in data set}||}{||\text{all uniq. obj. in universe}||}
\]

The Completeness(p) for both English and Portuguese-language DBpedia editions are shown on Table 2 for each property in our use case (areaTotal, foundingDate and populationTotal). The percentages shown represent the completeness of DBpedia English before integration (en), DBpedia Portuguese before integration (pt), and completeness after integration (final). As expected, the integration process increased completeness for both data sets, with particularly high increase (more than 9x) for DBpedia English in the properties areaTotal and populationTotal. The property foundingDate was actually more complete in DBpedia English, and provided an increase of roughly 4% in completeness for DBpedia Portuguese.

Conciseness.

On the schema level, a data set is concise if it does not contain redundant attributes (two equivalent attributes with different names). On the data (instance) level, a data set is concise if it does not contain redundant objects (two equivalent objects with different identifiers). The extensional conciseness measures the number of unique objects in relation to the overall number of object representations in the data set [5]. Similarly, intensional conciseness measures the number of unique attributes of a dataset in relation to the overall number of attributes in a target schema [5]. The intensional conciseness in our use case was 1, since both datasets used the same schema. The extensional conciseness was also 1, since both DBpedia editions only contained one URI per object. Similarly to the case of completeness, we have defined a finer grained conciseness metric for a given property p to measure the proportion of objects that do not contain more than one identical value for p (redundant), with regard to the universe of unique property values (Equation 4).

\[
\text{Conciseness}(p) = \frac{||\text{obj. with uniq. values for } p \text{ in data set}||}{||\text{all uniq. obj. with } p \text{ in dataset}||}
\]

7. CONCLUSION

We have described Sieve, a Linked Data Quality Assessment and Data Fusion module. Sieve is employed by the Linked Data Integration Framework (LDIF) after the Schema Mapping and Identity Resolution steps. Sieve’s role is to assess the quality of the integrated data and subsequently decide on which values to keep, discard or transform according to user-configured quality assessment metrics and fusion functions. Sieve is agnostic to provenance and quality vocabularies, allowing users to configure which metadata to read, and which functions to apply via a declarative specification language.

Through a use case that imported data about Brazilian municipalities from international DBpedia editions, we have demonstrated the usage of Sieve in a simple scenario that yielded an integrated data set that was more complete, concise, consistent and up to date than the original sources. The English DBpedia, although considered the most mature of the international DBpedia editions, did not have a particularly high coverage of Brazilian municipalities, and benefited from the higher coverage offered by the Portuguese DBpedia.
for those particular items. Furthermore, as the population of a city is prone to change, it is important to keep the most recent values. Blending values from multiple DBpedia editions allowed us to include the most recent value among the sources. However, identity resolution and schema mapping introduced multiple values for the same properties (values originating from different sources). The application of data fusion through Sieve allowed us to remove redundant and conflicting values, increasing the conciseness and the consistency of the data set.

Future work includes the development of more quality assessment scoring functions and data fusion functions, as well as performance and scalability experiments.

8. ACKNOWLEDGEMENTS

The authors would like to thank Andreas Schultz, Robert Isele and Anja Jentzsch for their valuable comments on this manuscript. We also thank them, as well as Andrea Matteini and Christian Becker for their work on other components of LDIF.

This work was supported by the EU FP7 grants LOD2 - Creating Knowledge out of Interlinked Data (Grant No. 257943) and PlanetData - A European Network of Excellence on Large-Scale Data Management (Grant No. 257641).

9. REFERENCES


