QUAL: A Provenance-Aware Quality Model

CHRIS BAILLIE, University of Aberdeen, UK
PETER EDWARDS, University of Aberdeen, UK
EDOARDO PIGNOTTI, University of Aberdeen, UK

In this paper we present a model for quality assessment over linked data. This model has been designed to align with emerging standards for provenance on the Web to enable agents to reason about data provenance when performing quality assessment. The model also enables quality assessment provenance to be represented, thus allowing agents to make decisions about re-use of existing assessments. We also discuss the development of an OWL ontology as part of a software framework to support reasoning about data quality and assessment re-use. Finally, we evaluate this framework using two real world case studies derived from transport and invasive species monitoring applications.

Categories and Subject Descriptors: H.1.1 [**Models and Principles**]: Systems and Information Theory

General Terms: Management, Measurement, Reliability

Additional Key Words and Phrases: data quality, Semantic Web, Linked Data, ontology, provenance

ACM Reference Format:
Special issue of the ACM Journal of Data and Information Quality (ACM JDIQ) on Provenance, Data and Information Quality. *ACM JDIQ* V, N, Article A (Y), 23 pages.
DOI:http://dx.doi.org/10.1145/0000000.0000000

1. INTRODUCTION

Assessing the quality of data has been identified as essential if agents (human or machine) are to identify reliable datasets for use in tasks such as decision-making or planning [Baillie et al. 2012]. This issue is illustrated in the following quote from Vint Cerf:

> "The problem is - we don't know whether the information we find [on the Web] is accurate or not. We don't necessarily know what its provenance is. So we have to teach people how to assess what they've found [...] there' so much juxtaposition of the good stuff and not-so-good stuff and flat-out-wrong stuff or deliberate misinformation or plain ignorance." [Wolly 2010]

Data quality is commonly defined as the "fitness for use" of data [Batini and Scannapieco 2006] and can be quantified through the production of quality scores using a number of quality metrics. Such metrics examine the context around data [Waterman and Hendler 2013] to produce scores for particular criteria, known as quality dimensions [Wand and Wang 1996]. Examples of such quality dimensions include *accuracy* (data are correct and reliable) and *relevance* (data are applicable and useful for the task

The research described here is supported by the award made by the RCUK Digital Economy programme to the dot.rural Digital Economy Hub; award reference: EP/G066051/1.
Author's addresses: C. Baillie, P. Edwards and E. Pignotti dot.rural and Computing Science, University of Aberdeen, AB24 5UA, UK

DOI:http://dx.doi.org/10.1145/0000000.0000000

at hand). While the metrics deployed in existing quality assessment frameworks examine the context around data, the majority do not consider data provenance, a record of the *entities*, *agents*, and *activities* involved in data derivation. We have argued elsewhere [Baillie et al. 2012] that examining data provenance is critical in assessing data quality, as such information has been identified as an essential step to support users to better understand, trust, reproduce, and validate data on the Web [Miles et al. 2009]. Provenance therefore has a key role to play in data quality by describing data sources, the creation method, and how data has been transformed, e.g. who had access to the data, who processed it, and how it has been assessed previously.

Our work is grounded in a number of real-world application scenarios. These were selected due to the availability of data, the potential for quality issues to arise, and the need for such issues to be identified within application services. In this article we describe two example scenarios. The first features data contributed by public transport users via their mobile phones. This data is interpreted (and transformed in some cases) before being presented to other users as real-time passenger information describing, for example, when the next bus will arrive at their local stop. In this scenario, poor quality data can result from inaccurate devices, processing artefacts, and malicious users. The second scenario, invasive species monitoring, examines data produced by humans (acting as citizen sensors) that describe sightings of a particular animal species in the wild. For this scenario, attribution provenance is used to add metadata describing who was responsible for generating the report. Here, data quality issues arise as a result of human error such as incomplete reports and users mistakenly reporting different species.

We propose a novel contribution to the quality literature in the form of a quality assessment model and an associated software framework capable of assessing quality by examining the context around data (some of which may be provenance, while the rest is other contextual metadata) and producing provenance descriptions of the quality assessment activity.

The remainder of this article describes our application scenarios before exploring related work. We then describe our model for quality assessment (Qual), a generic software framework based on this model, and its deployment. We conclude by presenting an evaluation of our work and a discussion of future directions.

2. CASE STUDIES

This section provides a detailed insight into our two case studies along with a summary of the requirements for provenance and quality assessment identified in each case.

2.1. Real-Time Passenger Information

The Passenger Information scenario includes GPS latitude and longitude, and the GPS error margin. This data, published by a Web service, is used to place icons indicative of vehicle location on a map within a smartphone app that users can then interpret in order to make decisions about their journey. Assessing data quality is important in this scenario to prevent users being presented with erroneous data that leads them to make incorrect decisions possibly resulting in failed or delayed journeys.

How might poor data be introduced? Firstly, due to inaccurate mobile devices. Secondly, through the behaviour of malicious users who provide irrelevant observations by using the app when they are not using public transport. Thirdly, as a result of delays in the mobile network resulting in out-of-date data. Finally, if new data is not received it is possible that the application will present old data. Based on these cases, we have developed a set of quality requirements for the passenger information scenario. These are:

— **PI1** Observations should be less than one minute old (*timeliness*).
— **PI2** Observations should have an error margin less than 50 metres (*accuracy*).
— **PI3** There should be less than 30 seconds between an observation being produced and published (*availability*).
— **PI4** Observations should be less than 500 metres from the expected route (*relevance*).

2.2. Invasive Species Monitoring

Users in the Invasive Species Monitoring scenario produce data describing sightings of an invasive animal species. Sightings are reported using a Web interface that asks for information describing the location, time, number of animals spotted, and whether they were dead or alive. Quality assessment in this scenario is critical as reports of animals can trigger a response which involves sending a team to investigate. As many of these teams are made up of volunteers and are supported by limited resources they do not want to respond to unreliable reports.

Quality issues in this scenario are caused by human errors rather than a deficiency in technology. For example, users will occasionally omit certain reporting fields, such as the river name. Many users will make reports immediately while others submit large batches of reports at once, anywhere up to two weeks after the original sighting. The animals being monitored can travel long distances in a relatively short time and so reports should be acted upon promptly. Finally, inexperienced users often report the incorrect species and so higher quality scores should be given to reports made by trained volunteers or professionals. As with the passenger information scenario, we have used these cases to produce a number of quality requirements for this scenario. These are:

— **ISM1** An observation must be associated with a river (*completeness*).
— **ISM2** Reports should be made within two weeks of the sighting (*availability*).
— **ISM3** A report should be acted upon within a week (*timeliness*).
— **ISM4** Reports made by professionals are of higher quality than those made by volunteers and members of the public (*reputation*).

2.3. A Case for Provenance

Most of the quality requirements identified for these case studies can be satisfied by implementing metrics that simply examine the characteristics of the data. However, does this mean that examining the data alone is always sufficient? We argue this is not the case as quality problems can often exist in data provenance and are impossible to identify without metrics capable of examining such records. For example, consider a location sensor observation that places a public transport vehicle directly on a bus route. If this observation was produced by a map-matching algorithm [Velaga et al. 2012] triggered when an inaccurate sensor observation indicates a vehicle is not travelling on its correct route then the observation's provenance is critical. Map-matching calculates a new latitude and longitude for the observation based on the closest point on the route corresponding to the original location observation. The result of this is a new derived observation that would score high on relevance. In this example, examining the provenance of the map-matched observation would allow a quality assessment metric to identify the source observation and thus determine that the original observation is much less relevant and should be given a lower quality score. To illustrate further consider the various methods used to produce location observations in the first instance. If GPS is unavailable, mobile devices can attempt to ascertain their location via cellular network details, or via the Internet access point to which they are currently connected. Each of these is less accurate than the other. Considering data provenance would enable a quality assessment mechanism to identify when these less accurate sensing methods had been used and thus alert the agent to potential quality issues. Finally (and as highlighted in ISM4 above) it is often necessary to explore data attribution metadata to identify the agent responsible for data creation. In this specific instance, it is the group to which they belong, i.e. volunteers or professionals, that influences quality. Overall, although quality assessment is possible without provenance, it is clear that examining data provenance is critical to making better informed (and thus more reliable) decisions regarding quality.

3. RELATED WORK

The literature agrees that data quality is a multi-dimensional construct and, to date, there have been a number of attempts at defining a complete set of quality dimensions. For example, Wand and Wang [1996] attempt to describe a set of dimensions that includes many of the ones discussed in this paper (such as accuracy, relevancy, and timeliness). Jarke et al. [1999] refined this list by removing many of Wang and Strong's dimensions and introducing dimensions to measure traceability and metadata evolution. These additions clearly require provenance to be present, for example, traceability would require a record of how data has changed and who applied these transformations. More recently, Schaal et al. [2012] proposed a set of dimensions designed for quality assessment on the Social Web that is significantly larger than the previous collections. It is our position that the vast number of publications dealing with the selection of quality dimensions is evidence that no consensus can be achieved. Ultimately, though the definition of quality dimensions is an important task, we argue that dimension selection rests with the end user and the application domain.

There is also the question of how best to represent quality metrics as practice here also varies. Caruso et al. [2000] describe an approach to quality assessment used to identify duplicate keys within a database. Their tool utilises machine learning techniques over duplicate records to produce a set of quality metrics, which are then applied to the remaining dataset. Each time new metrics are introduced, a new training period is required. Batini and Scannapieco [2006] describe quality assessment within relational databases where the activity of assessment involves creation of a view over the database. Knight and Burn [2005] propose a quality assessment framework based on software that crawls the Web identifying low quality data. This framework is based on the IQIP approach: **I**dentify the elements involved (the *user*, the *environment*, and the *task* at hand); **Q**uantify by selecting the quality dimensions to be assessed; **I**mplement the crawling software; and **P**erfect the algorithm by feeding results back into the software's design. The authors suggest that a single set of quality metrics can be produced that will be applicable for every possible quality assessment. Through our case studies discussion above we have shown that this is not the case: users need to be able to define their own metrics based on their intended use for data. Although metrics could be refined in the '*Perfect*' stage, it may be that such refinements make the metrics less suitable for another agent. WIQA [Bizer and Cygniak 2009] demonstrates how quality assessment could be introduced to the Web of Linked Data. This framework expresses quality metrics using a policy language (WIQA-PL) that extends SPARQL. The result of this is that anyone familiar with SPARQL can easily learn WIQA-PL to define quality metrics. However, as demonstrated by Furber and Hepp [2011a] SPARQL is sufficiently expressive to characterise quality metrics when used with a SPIN-SPARQL reasoner and, therefore, it is not necessary to learn WIQA-PL's SPARQL extensions.

Data provenance has long been viewed as important in quality assessment, though few examples exist of quality assessment using provenance. Wang and Madnick [1990] represents one early example of explicitly tracing the origins of data and its intermediate sources. However, there is no description of how this metadata can be utilised as part of quality assessment. Bizer and Cygniak [2009] describe how their WIQA framework can handle provenance information to manage data attribution. We view this as an important example, but would note that provenance is about much more than attribution, for example, describing how data has been transformed (derivation provenance). Hartig [2009] demonstrates how provenance can be used to quantify timeliness. Again, this is as an important example but only evaluates one dimension. Geerts et al. [2014] address two data quality issues: transforming data using schema mappings and fixing data conflicts and inconsistencies. The example mapping provided describes a source database that contains data on *Treatments* and *Physicians* and a second target database detailing *Prescriptions* and *Doctors*. A mapping can then associate *Treatments* with *Prescriptions* and *Physicians* with *Doctors*. While this is an important aspect of data quality it only really considers one dimension: *what is the quality of the mapping*? There are a number of other quality issues not addressed by this work including *who created the original prescription data and are they reliable/trustworthy?* and *is the prescription selection appropriate given the patient's symptoms?* Moreover, there is little discussion of how this approach can document the assessments performed. For example, Geert et al's *conflict resolution module* selects a 'preferred value' (e.g. '*Doctor*') to translate a column name in a source database from a set of candidates, e.g. {'*Doctor*', *Physician*', '*Practitioner*'}. Here, provenance would be useful in investigating the range of options from which the resolution module made its selection and why one particular value was selected and the others rejected. Chalamalla et al. [2014] describe how data identified as low quality can be repaired after isolating the causes of the quality issue. The example given describes shops run by managers and employees and a quality metric stating that, per shop, managers must have larger salaries. Their framework can then identify which employee's salary is too large or which manager's salary is too low in order to raise the quality of the entire database. We interpret these as *reasons why quality issues exists*, which are what our quality metrics are designed to identify. If we contrast this with the passenger information scenario we identify the issue: *the vehicle is too far from the route* and identify possible causes: *the vehicle is actually taking a different route from normal*, *the sensor is on the correct vehicle but is inaccurate*, or *the sensor is owned by a malicious user and does not represent the location of a vehicle*. Is it possible to identify which of these possible reasons is the correct cause? In this scenario, map-matching could be considered as an example of one of Chalamalla et al's 'repairs' but even though this makes the data appear more reliable (inaccurate data now appears on the bus route) it is still only an approximation. A 'repair' suggests that the data should now be fixed but this may not be the case at all.

There are a number of existing quality assessment implementations. One example [Lynnes et al. 2010] describes a quality assessment framework for remote sensing data. The authors state that quality annotations produced by their framework should be re-used within different communities. However, it is unclear how these communities can make re-use decisions about these annotations as little information is recorded about the quality assessment process. Another quality framework, the ARM Data Quality Assurance Program [Peppler et al. 2008], monitors data produced by climate sensors. However, the framework does not consider data provenance. Exploring provenance would allow an agent to determine which sensor certain data were produced by and then discover, for example, who maintains that sensing device and how it is calibrated. Moreover, quality assessment results could be annotated with a description of the analysts and scientists involved in the assessment, facilitating audit of such assessments. Berti-Equille et al. [2011] provide a framework for capturing and exploring quality assessment in terms of dimensions, measures and identified problems.

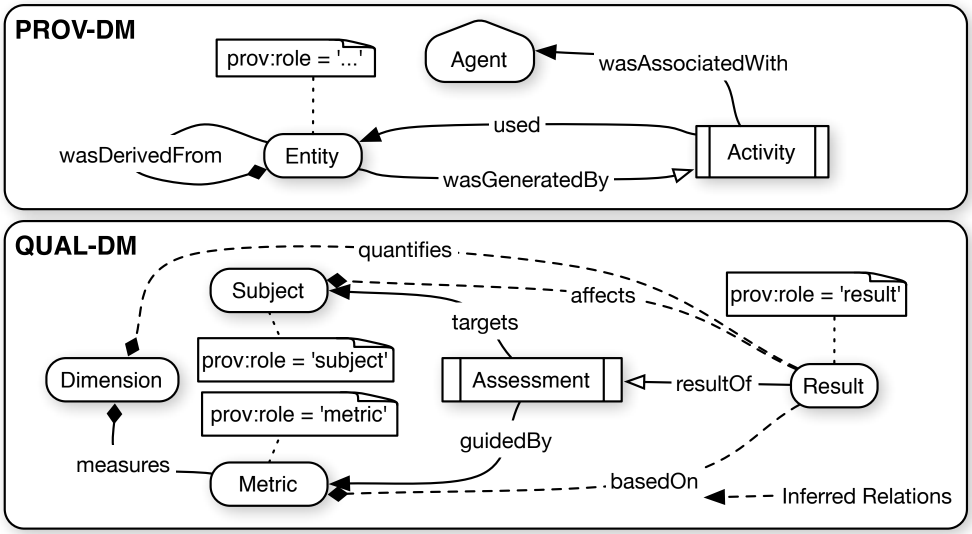


Fig. 1: The Qual Data Model (Qual-DM).

However, the authors do not discuss how to represent quality metrics and provide only a superficial description of *why* quality assessment was performed.

4. A MODEL FOR QUALITY ASSESSMENT

Based on our exploration of the literature and analysis of the requirements derived from the application use cases, we have developed Qual-DM, a data model capable of describing quality assessment activities, and their provenance. In this section, we describe both Qual-DM and its ontological realisation, Qual-O, using OWL 2.0.

4.1. Qual-DM: The Quality Data Model

We began developing Qual-DM (Figure 1) by identifying the minimal set of concepts required to describe quality assessment. To ensure that Qual-DM is capable of documenting the provenance of quality assessment, we have aligned our model with PROV-DM , the conceptual model underpinning the W3C provenance family of specifications. PROV is defined in terms of three main concepts: *Entity* (a physical, digital, conceptual, or other kind of thing with some fixed aspects), *Activity* (something that occurs over a period of time and acts upon or with agents), and *Agent* (something that bears some form of responsibility for an activity). Additionally, an *Entity* can be associated with a *role*, which describes the function an *Entity* performs in an *Activity*. In Qual-DM, we take the view that an Assessment is an *Activity* that acts upon one *Entity* in the role of *Subject*, one *Entity* in the role of *Metric*, to generate one *Entity* in the role of *Result*.

We use the following notation [Baader et al. 2003] throughout this article: $x \sqsubseteq y$ denotes subsumption of one set (x) by another (y), $x \times y$ denotes the cross product of two sets (x, y), $x \sqcap y$ denotes the logical conjunction between two sets (x and y), and $x \in y$ denotes that x is a member of y.

Definition 4.1 (*Subject*). A kind of role indicating that an activity acted upon an Entity to assess its quality. $Subject \sqsubseteq Role$

Definition 4.2 (*Metric*). A kind of role indicating that an Entity defines how an Activity performs quality assessment. $Metric \sqsubseteq Role$

Definition 4.3 (*Result*). A kind of role indicating that an Activity generated an Entity to describe the outcome of quality assessment. $Result \sqsubseteq Role$

In addition to these roles, we require a description of the Dimension assessed by a Metric (e.g. accuracy) and the relationship between these two sets.

Definition 4.4 (*Dimension*). A kind of Entity representing the aspect of quality measured by a Metric, e.g., accuracy. $Dimension \sqsubseteq Entity$

Definition 4.5 (*measures*). The usage of a Metric Entity by an Activity to produce a Result Entity for a particular Dimension. $measures \sqsubseteq (Activity \times (Entity \sqcap \exists hadRole.Metric)) \times Dimension$

We can now describe an Assessment as a PROV Activity that uses one Entity as a Subject, one Entity as a Metric, and generates one Entity as a Result.

Definition 4.6 (*Assessment*). A minimal set containing the Entities and Activities necessary to perform quality assessment. $Assessment \sqsubseteq Activity \times ((Entity \sqcap \exists hadRole.Metric) \times (Entity \sqcap \exists hadRole.Subject) \times (Entity \sqcap \exists hadRole.Result) \times Dimension)$

It is necessary to place some constraints upon the Entities within an Assessment so that an Entity does not perform more than one role in the Activity. These are:

$\forall Assessment \nexists Entity \in (Entity \sqcap \exists hadRole.Subject) \wedge Entity \in (Entity \sqcap \exists hadRole.Metric)$

(For all assessments, there is no entity with a subject role and a metric role)

$\forall Assessment \nexists Entity \in (Entity \sqcap \exists hadRole.Subject) \wedge Entity \in (Entity \sqcap \exists hadRole.Result)$

(For all assessments, there is no entity with a subject role and a result role)

$\forall Assessment \nexists Entity \in (Entity \sqcap \exists hadRole.Metric) \wedge Entity \in (Entity \sqcap \exists hadRole.Result)$

(For all assessments, there is no entity with a metric role and a result role)

To better illustrate how these Qual-DM concepts can be used to represent quality assessment, we now describe an example in terms of the passenger information case study. An *activity* can act upon an instance of a vehicle GPS observation (and its associated provenance) as an *entity* in a *subject role*. One of the quality requirements from this scenario (e.g. PI2) would then be considered as an *Entity* in a *metric role*. An *assessment* would then apply the PI2 *metric* to the *subject* to produce a *result*. For example, applying a metric derived from PI2 to a *subject* with an error margin of 25 metres would produce an accuracy *result* with a score of 0.5.

We also present a number of inference rules to better describe the relationship between a *Result* and the *Metric* and *Subject* used to derive it. The first of these *assesses* denotes that an *Activity* used a *Subject Entity*; next, *guidedBy* denotes that an *Activity* used a *Metric Entity* in an *Assessment*; *resultOf* denotes that a *Result Entity* is the output of an *Activity* in an *Assessment*. Further inferences can then be constructed using these: *basedOn*, denotes a *Result Entity* being derived from some *Metric Entity*; *annotates*, denotes a *Result Entity* being derived from a *Subject Entity*; and finally, *quantifies* denotes a *Result* providing a value for a *Dimension*.

Definition 4.7 (*assesses*). The usage of an Entity in the role of Subject by an Activity.

$$\frac{(\mathit{Activity} \times (\mathit{Entity} \sqcap \exists \mathit{hadRole.Subject})) \sqsubseteq \mathit{Assessment}}{\mathit{Assessment} \vdash (\mathit{Activity} \times (\mathit{Entity} \sqcap \exists \mathit{hadRole.Subject})) \sqsubseteq \mathit{assesses})}$$

Definition 4.8 (*guidedBy*). The usage of an Entity in the role of Metric by an Activity.

$$\frac{(\mathit{Activity} \times (\mathit{Entity} \sqcap \exists \mathit{hadRole.Metric})) \sqsubseteq \mathit{Assessment}}{(\mathit{Assessment} \vdash (\mathit{Activity} \times (\mathit{Entity} \sqcap \exists \mathit{hadRol.Metric})) \sqsubseteq \mathit{guidedBy}}$$

Definition 4.9 (*resultOf*). The generation of an Entity in a Result role by an Activity.

$$\frac{((\mathit{Entity} \sqcap \exists \mathit{hadRole.Result}) \times \mathit{Activity}) \sqsubseteq \mathit{Assessment}}{(\mathit{Assessment} \vdash ((\mathit{Entity} \sqcap \exists \mathit{hadRole.Result}) \times \mathit{Activity}) \sqsubseteq \mathit{resultOf}}$$

Definition 4.10 (*basedOn*). The generation of an Entity in a Result role by an Activity, for a particular Metric.

$$\frac{\begin{array}{c}(\mathit{resultOf} \sqsubseteq (\mathit{Entity} \sqcap \exists \mathit{hadRole.Result}) \times \\ \mathit{activity}) \wedge (\mathit{guidedBy} \sqsubseteq \mathit{Activity} \times (\mathit{Entity} \sqcap \exists \mathit{hadRole.Metric}))\end{array}}{\mathit{Assessment} \vdash ((\mathit{Entity} \sqcap \exists \mathit{hadRole.Result}) \times (\mathit{Entity} \sqcap \exists \mathit{hadRole.Metric})) \sqsubseteq \mathit{basedOn}}$$

Definition 4.11 (*annotates*). The generation of an Entity in a Result role by a Metric, for a particular Subject.

$$\frac{\begin{array}{c}(\mathit{resultOf} \sqsubseteq (\mathit{Entity} \sqcap \exists \mathit{hadRole.Result}) \times \mathit{Activity}) \wedge (\mathit{assesses} \sqsubseteq \\ (\mathit{Entity} \sqcap \exists \mathit{hadRole.Subject}))\end{array}}{\mathit{Assessment} \vdash (\exists \mathit{hadRole.Result} \times \exists \mathit{hadRole.Subject}) \sqsubseteq \mathit{annotates}}$$

Definition 4.12 (*quantifies*). The construction of a new Result by an Activity, for a particular Dimension.

$$\frac{\begin{array}{c}(\mathit{basedOn} \sqsubseteq (\mathit{Entity} \sqcap \exists \mathit{hadRole.Result})) \wedge \\ (\mathit{measures} \sqsubseteq (\mathit{Entity} \sqcap \exists \mathit{hadRole.Metric}) \times \mathit{Dimension})\end{array}}{\mathit{Assessment} \vdash ((\mathit{Entity} \sqcap \exists \mathit{hadRole.Result}) \times \mathit{Dimension}) \sqsubseteq \mathit{quantifies}}$$

Describing quality assessment in terms of Qual-DM and PROV delivers a number of benefits. First, it facilitates a description of the quality assessment activity, which can provide information on what performed the assessment (e.g. some computational service) and when the assessment took place (e.g. using the *startedAtTime* and *endedAtTime* properties defined in the PROV documentation). Second, the agent associated with assessment can be used to describe whether the activity was performed by a human or computational Agent, or even one Agent acting on behalf of another (using *actedOnBehalfOf* in PROV). Third, we can use a model of agent intent such as that proposed in Pignotti et al. [2010] to describe an Agent's motivation for performing an Activity. This model characterises intent as a set of goals and constraints. For example, a *Goal* "decide whether to take an umbrella" and a *Constraint* "only use weather data produced within the last half hour". We will demonstrate later how this model of intent can be used to support decision making about quality assessment re-use.

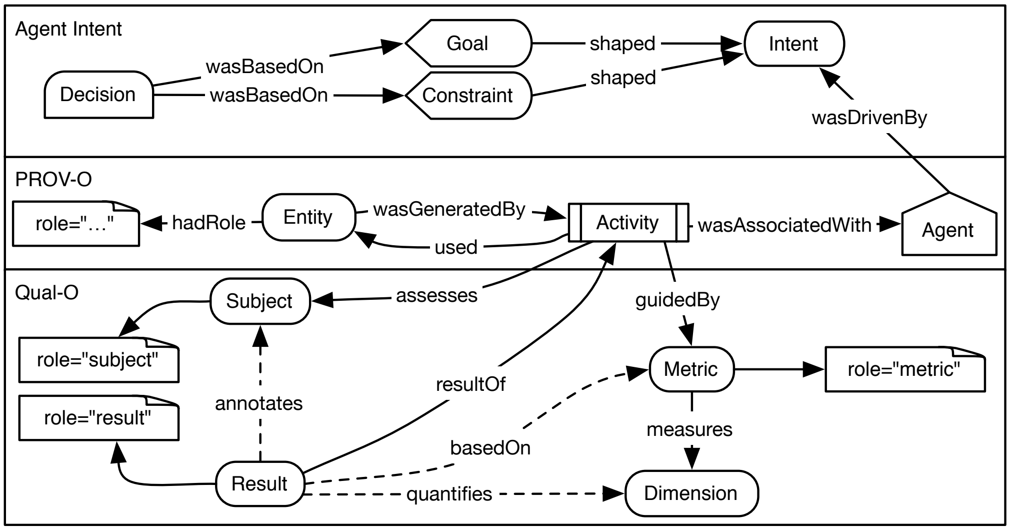


Fig. 2: The Qual Ontology (Qual-O)

4.2. Qual-O: An OWL2 Binding of Qual-DM

To operationalise the Qual-DM model we have produced an OWL2 binding, Qual-O[1] (Figure 2), an extension of PROV-O. We define three subclasses of *prov:Role*: *subject*, *metric*, and *result* to describe the roles that a *prov:Entity* can play in a quality assessment. We also created three subclasses of *prov:Entity*: *Subject*, *Metric*, and *Result* to represent a *prov:Entity* that fulfills either a *subject*, *metric*, or a *result* role in a *prov:Activity*. Although these classes are not sets within Qual-DM, we believe that it is more intuitive for a user to create individuals of type *Subject*, for example, than to define an entity with a *subject* role. Moreover, using OWL cardinality restrictions we can specify that an individual of type *Subject* must have exactly one *prov:Role* (*subject*), a *Metric* must have exactly one *prov:Role* (*metric*), and so on. Finally, *Dimension* is a subclass of *prov:Entity* and has no associated role as it does not perform a particular function as part of an assessment, but rather describes the aspect of quality examined by a particular assessment.

As in Qual-DM, we define a number of properties in Qual-O to describe the relationships between concepts: *assesses* and *guidedBy* are sub-properties of *prov:used* where a *prov:Activity used* a *Subject* and a *Metric*, respectively; *resultOf* is a sub-property of *prov:wasGeneratedBy* where an *Activity* generates a *Result*. Finally, *measures* is a sub-property of *prov:wasDerivedFrom* that describes the relationship between a *Metric* and a *Dimension*.

Inference rules are described using sub-properties of *owl:TransitiveProperty*. Firstly, *basedOn* is a sub-property of *owl:TransitiveProperty* and *prov:wasDerivedFrom* with property chain axioms [*resultOf*, *guidedBy*]. If a *Result* is the *resultOf* some *Activity*, and that *Activity* was *guidedBy* a *Metric*, infer that the *Result* was *basedOn* that *Metric*. A second *owl:TransitiveProperty*, *annotates*, is a sub-property of *prov:wasDerivedFrom* with property chain axioms [*resultOf*, *assesses*]. If a *Result* is the *resultOf* some *Activity*, and that *Activity assesses* a *Subject*, infer that the *Result anno-*

[1] http://purl.org/qual/qual-o

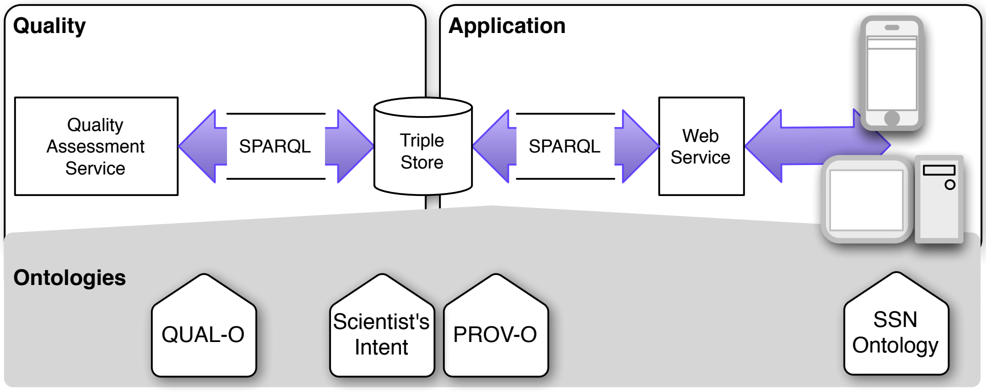


Fig. 3: Overview of an example deployment of our quality assessment framework.

tates that *Subject*. A final *owl:TransitiveProperty*, *quantifies*, is a further sub-property of *prov:wasDerivedFrom* with property chain axioms [*basedOn*, *measures*]. If a *Result* is *basedOn* some *Metric* and that *Metric measures* a particular *Dimension* then the *Result quantifies* the *Dimension*.

This ontology allows agents to define quality assessments to be executed by extending the logic of a *Metric* using a suitable rule language, such as SPIN (SPARQL Inferencing Notation). SPIN is now the de-facto industry standard for representing SPARQL rules and constraints on Semantic Web models and allows the calculation of property values based on other properties and standalone sets of rules to be executed under certain conditions. For example, if a GPS sensor observation is 30 seconds old, generate a quality result for timeliness with a score of 0.5. Furthermore, an Agent, in addition to executing quality assessments, can document the provenance of the assessment process. For example, *Subject*, *Metric*, and *Result* are defined in OWL as subclasses of *prov:Entity* and each has a *prov:Role* (*subject*, *metric*, and *result* respectively). SPIN can then infer that the provenance of the *Result* is that it was generated by some assessment *Activity*, which used two *Entities*, one with a *subject* role and one with a *metric* role. We will demonstrate later how this provenance information can be used to make decisions about the re-use of existing quality assessment results.

5. A QUALITY ASSESSMENT FRAMEWORK

In order to facilitate quality assessment using Qual-O, we have developed a generic software framework[2] using the Java Standard Edition version 1.6. This framework (Figure 3) can be extended to implement quality assessment across different domains.

The system is underpinned by a set of ontologies as follows: Qual-O is used to specify how quality assessment should be performed; PROV-O is used to document the provenance of quality assessment as well as describing the provenance of resources the system assesses; finally, the Scientist's Intent ontology[3] [Pignotti et al. 2010] is used to describe why an Agent is performing quality assessment. The data contained within the triple store (see Figure 3) can be described using any ontology relevant to the domain in which the framework is deployed. The Semantic Sensor Network (SSN)

[2] http://purl.org/qual/framework
[3] http://purl.org/policygrid/ontologies/intent

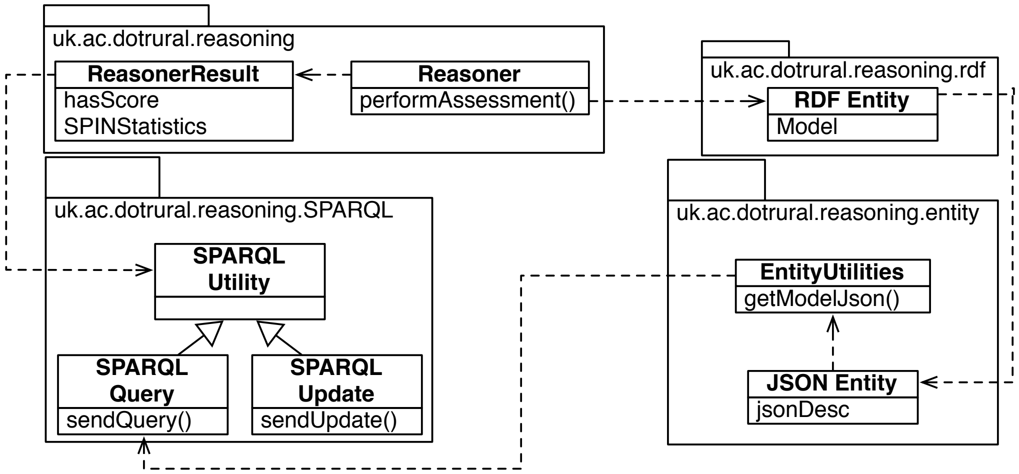


Fig. 4: The quality assessment framework software architecture.

ontology[4] [Compton et al. 2012] is also shown in Figure 3 to illustrate how other ontologies can be integrated with the framework, in this case, sensor data.

At the core of this framework lies a quality assessment service, a software component that can perform quality assessment on the data stored in a triple store. The data to be assessed is retrieved from the triple store using SPARQL and stored in a JENA model. The assessment service can then load the quality metrics expressed against the Qual-O ontology and perform quality assessment using a reasoner. Web services within the application component can then make use of the quality assessment service as required. We now provide a detailed description of the design of the quality assessment service.

5.1. Quality Assessment Service

A generic software architecture that defines a number of classes, which can be used or extended to facilitate quality assessment. The software, as illustrated in Figure 4, is organised into a number of packages, namely: entity, RDF, reasoning, and SPARQL. We now describe the functionality of each of these and how they can be extended to support different application domains.

5.2. Entity

The quality assessment service uses an *Entity* class to represent the resources that are to be the subject of quality assessment. Such a resource can be characterised as a Java object (using an instance of *Entity*). However, in our deployments, data is only accessible via a number of Web services using JSON[5] and so we extend *Entity* to produce a *JSONEntity* with fields representative of the keys contained within a JSON string. Of course, *Entity* could be extended to support any data format. To enable quality assessment any *Entity* (or *JSONEntity*) must be converted to an instance of *RDFEntity* that produces a JENA model representing the *Entity* to be assessed.

[4] http://purl.oclc.org/NET/ssn/ssnx
[5] http://www.json.org

5.3. RDF

RDFEntity must be extended to provide methods that convert the fields in a *JSONEntity* (or any other *Entity*) into a JENA model containing RDF statements describing the *Entity*. This model can then be passed to an instance of *Reasoner* (in the *reasoning* package), which provides access to the quality assessment reasoner.

5.4. Reasoning

The *reasoning* package contains classes that enable quality assessment (through the *Reasoner* class) and the production of quality assessment results (using the *ReasonerResult* class). *Reasoner* begins by loading the ontologies required to perform quality assessment, the assessment *Subject*s (instances of *RDFEntity*) and a set of quality metrics expressed against the Qual-O ontology. Calling the *performAssessment* method in *Reasoner* then performs quality assessment using these models to produce an instance of *ReasonerResult*. This construct stores a model containing the triples inferred by the reasoner (describing the results of quality assessment) and metadata describing, for example, how long each quality metric took to execute and the time at which each metric was invoked.

5.5. SPARQL

This package defines a number of helper classes that enable the execution of SPARQL queries and updates. These can be used in order to produce instances of *RDFEntity* by querying a SPARQL endpoint and can also facilitate persistent storage of *ReasonerResults* by sending SPARQL updates to a SPARQL 1.1 endpoint. As described earlier, this framework was designed to be generic so that it can be adapted for multiple domains by specialising the *Entity* classes and the rules that guide the reasoner. The following section describes two domain specific implementations of the quality assessment framework to demonstrate its operation.

6. DEPLOYMENT EXAMPLES

6.1. Passenger Information

As described earlier, the passenger information case study involves the generation of sensor observations describing the location of passengers and vehicles. There are a number of potential data quality issues. For example, if an observation was produced some time ago then it will not represent the vehicle's current location. Therefore, in order to provide reliable passenger information data quality assessment is critical in this scenario.

Observations in the passenger information case study are represented using an extension of the Semantic Sensor Network (SSN) ontology. This ontology was designed to describe sensing devices, their observations, and other relevant contextual data such as the feature of interest (an abstraction of real world phenomena such as a person or an event) and the observed property (an observable quality of an event or object such as temperature or acceleration). Figure 5 (top) shows the extensions made to the SSN ontology (identifiable by the pi namespace) and how they can be used to describe sensor observations in the passenger information case study.

To enable assessments of quality within the passenger information scenario a number of extensions have been made to the generic quality assessment framework described earlier. Firstly, to retrieve location observations we create an interface to query a web service for JSON objects representing observations. In order to interpret this observation, the *JSONEntity* class within the framework was extended to convert the JSON fields (e.g. *observationSamplingTime*, *latitude*, and *longitude*) to a Java object with these fields. Similarly, *RDFEntity* must be extended to handle conversion of this

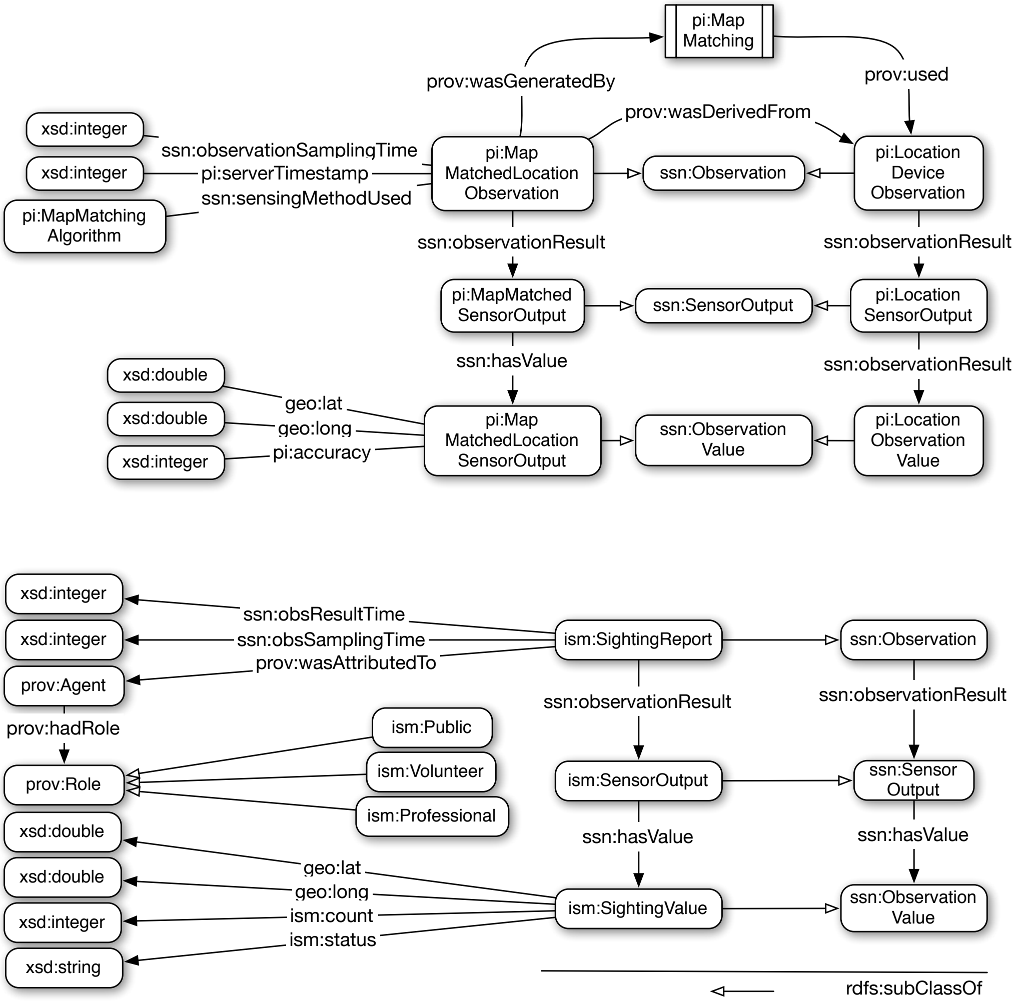


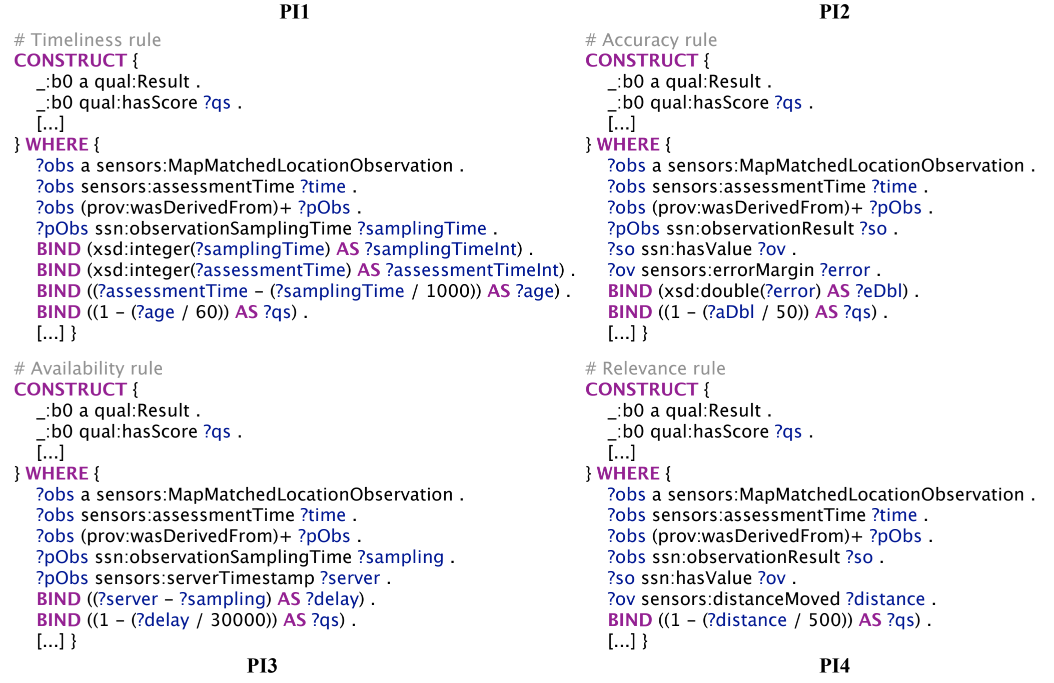
Fig. 5: Example data models from our case studies - passenger information (top) and invasive species monitoring (bottom).

Java object into an RDF model, expressed using JENA. This location observation RDF model can then be passed to a SPIN reasoner that can apply a number of SPIN rules representing quality metrics. Example rules for this scenario (corresponding to the quality requirements in section 2.1) are shown in Figure 6.

6.2. Invasive Species Monitoring

Sighting reports in this scenario are also modelled as observations using extensions to the SSN ontology. Each observation is attributed to the user that created it using PROV. Each user also plays a particular role in the reporting Activity. Figure 5 (bottom) presents an overview of the data model used for this scenario.

Reports are described using instances of *ism:SightingReport*; this class is an extension of *ssn:Observation* and can be used to describe when the sighting originally oc-



PI1

```
# Timeliness rule
CONSTRUCT {
   _:b0 a qual:Result .
   _:b0 qual:hasScore ?qs .
   [...]
} WHERE {
   ?obs a sensors:MapMatchedLocationObservation .
   ?obs sensors:assessmentTime ?time .
   ?obs (prov:wasDerivedFrom)+ ?pObs .
   ?pObs ssn:observationSamplingTime ?samplingTime .
   BIND (xsd:integer(?samplingTime) AS ?samplingTimeInt) .
   BIND (xsd:integer(?assessmentTime) AS ?assessmentTimeInt) .
   BIND ((?assessmentTime - (?samplingTime / 1000)) AS ?age) .
   BIND ((1 - (?age / 60)) AS ?qs) .
   [...] }
```

PI2

```
# Accuracy rule
CONSTRUCT {
   _:b0 a qual:Result .
   _:b0 qual:hasScore ?qs .
   [...]
} WHERE {
   ?obs a sensors:MapMatchedLocationObservation .
   ?obs sensors:assessmentTime ?time .
   ?obs (prov:wasDerivedFrom)+ ?pObs .
   ?pObs ssn:observationResult ?so .
   ?so ssn:hasValue ?ov .
   ?ov sensors:errorMargin ?error .
   BIND (xsd:double(?error) AS ?eDbl) .
   BIND ((1 - (?aDbl / 50)) AS ?qs) .
   [...] }
```

```
# Availability rule
CONSTRUCT {
   _:b0 a qual:Result .
   _:b0 qual:hasScore ?qs .
   [...]
} WHERE {
   ?obs a sensors:MapMatchedLocationObservation .
   ?obs sensors:assessmentTime ?time .
   ?obs (prov:wasDerivedFrom)+ ?pObs .
   ?pObs ssn:observationSamplingTime ?sampling .
   ?pObs sensors:serverTimestamp ?server .
   BIND ((?server - ?sampling) AS ?delay) .
   BIND ((1 - (?delay / 30000)) AS ?qs) .
   [...] }
```

PI3

```
# Relevance rule
CONSTRUCT {
   _:b0 a qual:Result .
   _:b0 qual:hasScore ?qs .
   [...]
} WHERE {
   ?obs a sensors:MapMatchedLocationObservation .
   ?obs sensors:assessmentTime ?time .
   ?obs (prov:wasDerivedFrom)+ ?pObs .
   ?obs ssn:observationResult ?so .
   ?so ssn:hasValue ?ov .
   ?ov sensors:distanceMoved ?distance .
   BIND ((1 - (?distance / 500)) AS ?qs) .
   [...] }
```

PI4

Fig. 6: Example rules for the passenger information scenario.

curred (*ssn:observationSamplingTime*) and was reported (*ssn:observationResultTime*). Although instances of *ssn:Observation* are intuitively associated with hardware sensors, the definition of a *Sensor* in the SSN documentation includes the notion of humans as observers. Indeed, the definition of *Observation* is “a Situation in which a Sensing method has been used to estimate or calculate a value of a *Property* of a *FeatureOfInterest*”, for example, the presence of an animal. Here, sightings are modelled using the SSN ontology and are associated with people reporting the sighting using PROV. Figure 7 presents a set of example metrics for the invasive species monitoring scenario, corresponding to the quality requirements in section 2.2.

6.3. Quality Assessment Results

In both deployments, quality assessment is triggered when a user taps a vehicle on their mobile device (passenger information) or clicks on a report (invasive species). Once complete, the results are displayed to the user using a series of colour-coded smiley faces beside the quality dimension the face quantifies. A high quality result (score ≥ 0.66) is denoted by a green smiley face, a medium quality result ($0.33 \geq$ score < 0.66) is a yellow neutral face, while a low quality score (score < 0.33) is represented by a red sad face. These representations were the result of co-design sessions with users who believed our initial interfaces were much too complicated to understand quickly and easily. Further discussion of this design process is include in section 7.5.

7. EVALUATION

In this section, we discuss a multi-faceted evaluation of our quality assessment framework. This is organised as follows: 1) an analysis of our formal model; 2) an analysis

ISM1

```
CONSTRUCT {
   _:b0 a qual:Result .
   _:b0 qual:hasScore ?qs .
   [...]
} WHERE {
   ?obs a ism:SightingObservation .
   ?obs qual:assessmentTime ?time .
   ?obs ssn:observationSamplingTime ?sTime .
   ?obs ssn:observationResultTime ?aTime .
   BIND (xsd:integer(?sTime) AS ?sTimeInt) .
   BIND (xsd:integer(?aTime) AS ?aTimeInt) .
   BIND ((?aTimeInt – round((?sTimeInt / 1000))) AS ?deltaT) .
   BIND ((1 – (?deltaT / 604800)) AS ?score) .
   [...] }
```

ISM2

```
CONSTRUCT {
   _:b0 a qual:Result .
   _:b0 qual:hasScore ?qs .
   [...]
} WHERE {
   ?obs a ism:SightingObservation .
   ?obs qual:assessmentTime ?time .
   ?obs ssn:featureOfInterest ?foi .
   ?foi ism:name ?riverName .
   BIND (IF((?riverName = "RiverNull"), 0, 1) AS ?qs) .
   [...] }
```

```
CONSTRUCT {
   _:b0 a qual:Result .
   _:b0 qual:hasScore ?qs .
   [...]
} WHERE {
   ?obs a ism:SightingObservation .
   ?obs qual:assessmentTime ?time .
   ?obs ssn:observationSamplingTime ?sTime .
   ?obs qual:assessmentTime ?aTime .
   BIND (xsd:integer(?sTime) AS ?sTimeInt) .
   BIND (xsd:integer(?aTime) AS ?aTimeInt) .
   BIND ((?aTimeInt – round((?sTimeInt / 1000))) AS ?deltaT) .
   BIND ((1 – (?deltaT / 604800)) AS ?qs) .
   [...]_}
```

ISM3

```
CONSTRUCT {
   _:b0 a qual:Result .
   _:b0 qual:hasScore ?score .
   [...]
} WHERE {
   ?obs a ism:SightingObservation .
   ?obs qual:assessmentTime ?time .
   ?obs prov:wasAttributedTo ?agent .
   ?agent prov:hadRole ?role .
   BIND (IF((?role = ism:GeneralPublic), 0.33,
      IF((?role = ism:Volunteer), 0.66,
         IF((?role = ism:Professional), 1, 0))) AS ?qs) .
   [...] }
```

ISM4

Fig. 7: Example rules for the invasive species monitoring scenario.

of the ability of our model to represent the quality requirements identified in our case studies; and 3) an empirical study into the performance of the framework.

7.1. Qual-DM Validity

We begin our evaluation by demonstrating that the semantics of Qual-DM are compatible with those defined in PROV-DM. To be a valid extension of PROV, Qual-DM's semantics should not violate any of the constraints defined within PROV. Those relevant to Qual-DM are: (i) an entity must have been generated before it is used (ii) an entity must exist before another entity is derived from it (iii) there can be no cycles in the provenance graph.

The first of these refers to the *used* property in PROV-DM. Graphs 1 and 2 in Figure 8 show the equivalent *used* properties within Qual-DM (*guidedBy* and *assesses*). Definition 4.6 in Qual-DM states that for an *Assessment* to be valid, it must contain an *Activity*, a *Metric*, a *Subject*, a *Result*, and a *Dimension*. Therefore, it is not possible for an *Assessment* to exist with an *Activity* but no *Metric* or *Subject*; thus the *Metric* or *Subject* must exist before the *Assessment Activity* uses either. Graph 3 in Figure 8 shows an equivalent *wasGeneratedBy* property (*resultOf*). Definition 4.6 also states that for an *Activity* to generate a *Result*, the *Metric* and *Subject* must exist before the *Result* is generated.

Graphs 4 and 5 in Figure 8 show the equivalent derivation relations in Qual-DM (*basedOn*, *annotates*, and *quantifies*). In order for a *Result* to be derived from a *Metric* or *Subject* then these entities must already exist before the derivation occurs as the *Activity* must already be *guidedBy* the *Metric* to assess a *Subject*. The constraints in Qual-DM prevent cycles from occurring in an *Assessment* by stating that an *Entity* cannot perform more than one *role* in any *Activity*. It is permissible, however, for entities

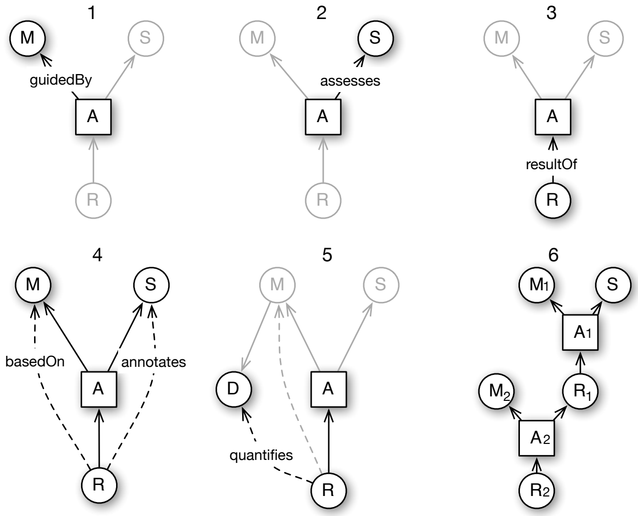


Fig. 8: The PROV constraints relevant to Qual-DM.

to perform different roles in different activities. Graph 6 provides an example where *Result R1* (the output of *Activity A1*) is a *Subject* in *Activity A1*. This is an example of assessment of a previous quality result.

Table I: Comparison between Qual-DM and existing quality assessment models.

	Who	**What?**	**When?**	**Why?**	**How?**	**Which?**
WIQA [Bizer and Cygniak 2009]		✔			✔	
IQ [Missier et al. 2006]		✔			✔	✔
DQM [Furber and Hepp 2011b]		✔	partial	partial	✔	✔
AIR [Jacobi et al. 2011]	✔	✔			✔	✔
Qual-DM	✔	✔	✔	✔	✔	✔

7.2. Comparison with Other Models

A number of models exist that are already capable of describing quality assessment and, therefore, to justify the creation of Qual-DM it is necessary to identify deficiencies in these existing models or advantages introduced through the development of Qual-DM. To do this, we began by considering how each model represents the provenance of quality assessment. An effective model should express this process in terms of as many of the 7 Ws of provenance [Goble 2002] as possible. For our comparison (Table I) we interpret the 7 Ws as: *Who* performed the quality assessment? *What* was the subject of the assessment? *When* was the assessment performed? *Why* was the assessment performed? *How* was the assessment performed? *Which* dimensions of quality were assessed? *Where* was the assessment performed? As the location for an assessment is likely to be quality service, we have decided to omit *where* from our analysis of the various models. More important is *how* the assessment was performed, on *what*, and *why*.

All models that we investigated were capable of capturing the subject of quality assessment (Table I, column 2). However, only Qual-DM and AIR [Jacobi et al. 2011] are capable of describing the agent that performed quality assessment (column 1). Qual-DM was the only model capable of expressing when quality assessment was performed, though DQM [Furber and Hepp 2011b] can identify when quality results were produced (column 3). In terms of explaining why quality assessment was performed, only Qual-DM is capable of providing a full description of an agent's intended goal(s) using the Scientist's Intent ontology, although DQM can describe the task that required quality assessment (column 4). Qual-DM, Air, DQM, the IQ Model [Missier et al. 2006] and WIQA [Bizer and Cygniak 2009] are all capable of capturing how quality assessment was performed (column 5). WIQA, for example, uses the WIQA policy language (an extension of SPARQL), whilst DQM and Qual-DM use SPIN. The final category, which quality dimensions were assessed, is only satisfied by by IQ, DQM, AIR, and Qual-DM (column 6). Based upon this analysis of provenance dimensions, we argue that Qual-O provides a more complete description of quality assessment than alternative models.

7.3. Representing Quality Metrics

In the Case Studies section we introduced a number of quality requirements for each of the application scenarios. A further evaluation criterion for our quality model is therefore its ability to represent the requirements in these case studies as quality metrics. In this section, we describe the characterisation of each of these requirements as logical extensions of *Metric* using SPIN. To compare our SPIN rules to the scenario requirements, we began by identifying the properties or concepts within each data model (such as *sensors:errorMargin* in the passenger information scenario) that could be used to assess each quality requirement. We then identified a formula (the quality metric) that calculates quality scores based on these properties or concepts. Finally, we examined SPIN's ability to represent these formulae for each requirement.

7.3.1. Passenger Information. The SPIN rules used to assess quality in the passenger information scenario are shown in Figure 6. In each instance, the rule applies to instances of type *sensors:MapMatchedLocationObservation* and examines the provenance of the observation to find the original data submitted by a passenger. PI1 assesses the timeliness of observations by examining the value associated with the *ssn:observationSamplingTime* property (?samplingTime) against the value of *sensors:assessment* (?time) (i.e. the time at which the assessment is being performed). The rule then states that as the observation's age approaches 60 seconds, its quality in terms of timeliness decreases. For example, a 30-second-old observation is given a quality score of 0.5. PI2 assesses observation accuracy using the value associated with the *sensors:errorMargin* property (?accuracy). The rule states that as an error margin approaches 50 metres, its quality decreases. PI3 assesses observation availability based on the values of the *ssn:observationSamplingTime* (?sampling) and *sensors:serverTimestamp* (?server). This rule states that as the delay between these two times increase toward 30 seconds, the observation's availability score decreases. Finally, PI4 assesses relevance by examining the value associated with the *sensors:distanceMoved* (?distance). As this value approaches 500 metres, observation quality decreases. From these examples, it is clear that it is possible to characterise the quality requirements identified for the passenger information scenario using Qual-O and SPIN.

7.3.2. Invasive Species Monitoring. The SPIN rules used to assess quality in the invasive species monitoring scenario are shown in Figure 7. The first three rules do not consider any provenance information, as there are no derived observations in this scenario. Instead, each metric simply assesses instances with type *SightingObservation*. ISM1 assesses observation availability using the values of the *ssn:observationSamplingTime* (the time at which the sighting occurred - ?sTime) and *ssn:observationResultTime* (the time at which the sighting was reported - ?aTime). As the delay between these times increases to seven days the quality, in terms of availability, decreases. ISM2 assesses completeness by ensuring that the observation is not associated with a river described by the system as *RiverNull*. This value is substituted by the system when a user submits a sighting without river information. Of course, an equivalent query could check for observations without a triple containing an *ism:name* property. ISM3 assesses timeliness, and checks the difference between the *ssn:observationSamplingTime* and *assessmentTime* values. As this difference approaches seven days the quality of the observation, in terms of timeliness, decreases. ISM4 is the only metric for the species monitoring scenario that considers provenance. This rule assesses reputation based on the role performed by the agent within an organisation. If the value of *prov:hadRole* is *ism:GeneralPublic* the observation is given a score of 0.33, if its value is *ism:Volunteer* a score of 0.66, and if its value is *ism:Professional* then a score of 1.

These examples demonstrate that, as with the passenger information scenario, it is possible to characterise all of the invasive species monitoring quality metrics using Qual-O and SPIN. We now continue our evaluation with an empirical evaluation of quality assessment performance.

7.4. Quality Assessment Framework Performance

In order to investigate how our framework performs we designed a pair of experiments to measure the reasoning time required to perform quality assessment. Experiment one investigated the effects of considering data provenance as part of a quality assessment activity. Experiment two compared the time taken to reason about the re-use of existing quality assessment results, versus the time required to perform a new quality assessment. Experiment three then investigated whether quality result re-use queries could be executed in less time than performing new assessments. Each experiment was performed on a Sun Fire X4100 M2 with two dual-core AMD Opteron 2218 CPUs and 32GB of RAM. Additionally, this platform ran CentOS 5.8, Java SE1.6 and required JENA version 2.10 and SPIN version 1.3.

7.4.1. Experiment 1. This experiment investigated how the different kinds of metadata examined by quality assessment affected the required reasoning time. For both scenarios, we created a set of 350 observations containing metadata describing the location of a bus or the sighting of an animal, depending upon the scenario, and the provenance of this data. In the passenger information scenario this provenance described the map-matching activities applied to an observation, and in the invasive species monitoring scenario it describes the attribution of data to the agent that made the sighting. Each observation was randomly selected and assessed ten times with the results averaged to produce a timing value for each observation.

In the Invasive Species Monitoring scenario (Fig. 9) the only quality metric that used provenance is *Reputation*. Thus, for this scenario, considering provenance metadata enables new kinds of quality assessment that would not be possible without provenance. The average time taken to perform *reputation* assessments (158 milliseconds) is comparable to the other metrics measured and so should not adversely affect application performance. For the passenger information scenario, considering data provenance, enables the reasoner to assess data quality in a different way. This results in two sets of metrics: one set that uses provenance (Fig. 10, dashed line) and another that does not (Fig. 10, solid line). From these results, it is clear that considering provenance is more costly. However, in the passenger information scenario, not considering provenance is more efficient but will result in agents making incorrect decisions.

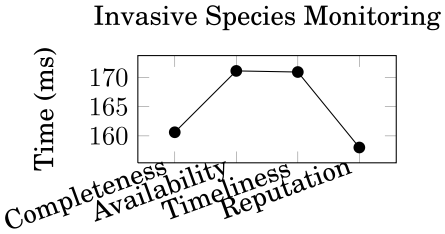


Fig. 9: Average reasoning time required in the ISM scenario.

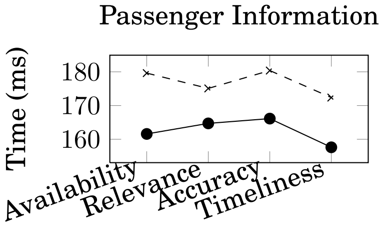


Fig. 10: Average reasoning time required in the PI scenario with provenance (dashed line) and without (solid line).

7.4.2. Experiment 2. Experiment two investigated how the provenance of existing quality assessment results can be used to make decisions about quality result re-use. Such decisions can be made using metrics similar to those used in performing new quality assessments. However, instead of assessing the quality of data they assess the 'fitness for use' of other quality results. Table II presents an example of the metrics used in experiment two for the passenger information scenario. This experiment shows (Figure 11 & 12) that for both scenarios reuse (dashed lines) requires less time than performing new assessments (solid lines).

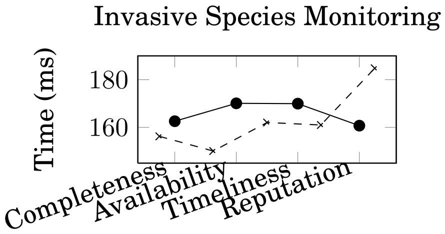


Fig. 11: Average reasoning time to perform new assessments (solid line) and make re-use decisions (dashed line).

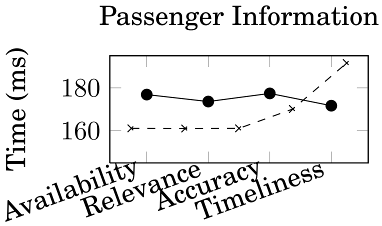


Fig. 12: Average reasoning time to perform new assessments (solid line) and make re-use decisions (dashed line).

7.4.3. Experiment 3. Our final experiment examined how the complexity of quality metrics affected the required reasoning time. By complexity, we mean the number of statements matched in the *WHERE* pattern in an equivalent *SELECT* query based on a quality metric. As the number of matched statements increases, so does the complexity of the metric. Each observation was randomly selected and assessed ten times with the results averaged to produce the graph shown in Figure 13. Although the actual experimental results (Fig 13, solid line) are difficult to interpret the line of best fit (Fig 13, dashed line) indicates that as metric complexity increases the required reasoning time also increases. Although this seems obvious, we argue that the results do function as a guide for metric authors interested in how metrics may perform.

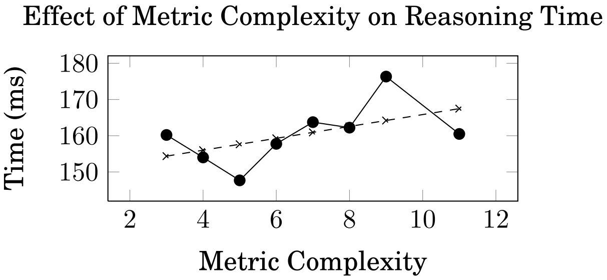


Fig. 13: Average reasoning time required by metrics of different complexity. Dashed line shows line of best fit for these data.

Table II: The quality result re-use metrics used in the passenger information scenario.

Name	**Re-use a result if...**
Metric (Simple)	it was produced using a *pi:RelevanceMetric*.
Agent	it is attributed to a *pi:Chris* a *prov:Agent*.
Time	it is less than 10 minutes old.
Intent	it is attributed to an agent with the same intent as *pi:Chris*.
Metric (Complex)	it requires the observation's error margin to be less than 50m.

7.5. Lessons Learnt

In this section we illustrate some of the lessons learnt during the deployment of our quality framework with different case study scenarios.

While deploying Qual for the passenger information scenario it quickly became clear that examining data provenance would be critical. The vast majority of sensor observations in this scenario had been subject to map-matching to some extent. As a result, any quality assessment that does not examine provenance would be unable to identify that this data had been transformed, resulting in erroneous quality scores.

During Qual's development we hypothesised that quality assessment would be too costly to perform in real-time due to the number of inferences being produced. However, our results clearly indicate that, for these scenarios and metrics, quality assessment can be performed with little additional overhead. As a consequence, quality assessment can be performed as and when agents require, resulting in more representative measures of temporal dimensions such as *timeliness* and *availability*. For example, in the passenger information scenario new data is streamed every minute and we are able to perform assessments on demand ensuring that quality scores consider the latest available information.

We did, however, identify one factor that increases the reasoning time required by quality assessment: inferring quality assessment provenance as the assessment is performed. This increased the number of inferences (provenance or otherwise) exponentially resulting in quality assessments that, on average, took around 2 minutes rather than 0.2 seconds to complete which, we argue, *would* negatively affect application performance particularly for time-critical situations, e.g. when deciding when to leave to catch a bus. One workaround to this was to encode provenance within the quality metrics so they generate PROV statements in addition to QUAL statements.

An issue present throughout Qual's development was the way in which metrics were authored. As researchers with experience in Semantic Web programming we are familiar with tools for authoring SPIN-SPARQL rules. However, these tools are not necessarily intuitive to ordinary users of the Web and so it became clear that some concerted effort would be required to develop a tool to allow users to author metrics without concerning themselves with SPARQL syntax. To encourage widespread adoption of our framework, this user interface must be as intuitive as possible and possibly give insight into the results a quality metric will yield via simulation before an assessment is performed.

In discussions with users about the deployment of Qual we learnt that being able to view and understand the provenance of quality assessment was important for users to trust the results of such assessments. Users were especially concerned about authorship of quality metrics, how the metrics actually guided the assessment process, and how long ago the results had been generated. Therefore, aligning Qual-O with a provenance model was the correct decision as this allows quality assessment provenance to be documented. However, the issue remains how best to display provenance information to users in a way that is accessible and understandable to them.

As discussed in section 6, we spent a considerable amount of time with users working on our representation of quality assessment results. Initially, data quality was presented in terms of numerical scores to test that the framework operated as expected. An alternative mode employed a bar-chart system whereby individual charts represented the scores for each quality dimension. A red unfilled bar indicated low data quality, a half-filled amber bar medium data quality (a quality score of around 0.5), and green high data quality (a quality score approaching 1). Users still found this representation too complex and we therefore settled upon yet another representation (using the series of faces outlined in the deployment section of this article).

8. CONCLUSIONS & FUTURE WORK

In this paper we have demonstrated how Qual-DM, our quality assessment data model, is compatible with PROV and that its OWL binding, Qual-O, is capable of representing a number of different quality metrics drawn from two illustrative scenarios. We have also demonstrated that our model captures more aspects of assessment provenance than existing models. Through the exploration of these scenarios, we have highlighted the benefits of using provenance as part of quality assessment but also of re-using existing quality assessment results. For example, through the passenger information scenario we have demonstrated that assessing quality by examining sensor data alone is insufficient, as quality problems can be hidden in observation provenance. Additionally, the Invasive Species Monitoring example shows how the presence of provenance information can enable new quality assessments, e.g. data attribution.

A series of empirical experiments have shown how considering data provenance affects overall reasoning time. Although reasoning time, on average, increased by a significant margin when considering provenance information, the actual scale of this increase is not large enough to affect overall application performance. We have also recorded data that demonstrate how the reasoning time required by quality metrics increases as the complexity of the metric increases. This can provide useful insight to metric authors about the likely performance of their own metrics. Finally, we have demonstrated that reasoning about quality assessment re-use is (for the use cases considered) on average more efficient than performing a new assessment.

In terms of future work, a number of issues remain to be explored. The first of these relates to the use of SPIN rules to represent quality metrics, which in turn requires knowledge of SPARQL. *Can a user-interface be developed to enable someone with no knowledge of SPARQL to author quality metrics*? There is also the question of whether

SPIN is the most suitable language to specify quality metrics. Answering this would require a comparison between SPIN and other rule languages in terms of required reasoning time as well as the expressivity of the languages.

At present, Qual-O encodes the provenance of quality assessment using the constructs defined in Qual-O alone. A further step is required to translate these Qual-O constructs into PROV provenance using a reasoner such as OWL-RL. This raises further questions: *Should the provenance of quality assessment be generated as the assessment is performed and if so, what is the effect on reasoning time*? or *Is it sufficient to infer the provenance after an assessment is complete*?

In conclusion, using Qual enables agents to examine data provenance as part of quality assessment and also to document the provenance of the assessment process. We have shown that although considering provenance during quality assessment increases the time to perform an assessment, consideration of this metadata can be critical in making correct decisions, and also enables new types of assessment. Moreover, re-using existing quality assessment results can be faster than performing new quality assessments.

REFERENCES

F. Baader, D. McGuinness, and D. Nardi. 2003. *Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press.

Chris Baillie, Peter Edwards, and Edoardo Pignotti. 2012. Quality Reasoning in the Semantic Web. In *The Semantic Web - ISWC 2012 (Lecture Notes in Computer Science)*, Vol. 7650. 383–390.

Carlo Batini and Monica Scannapieco. 2006. *Data Quality Concepts, Methodologies and Techniques*. Springer-Verlag, Heidelberg.

Laure Berti-Equille, Isabelle Comyn-Wattiau, Mireille Cosquer, Zoubida Kedad, Sylvaine Nugier, Veronika Perala, Samira Si-Said Cherfi, and Virginie Thion-Goasdoue. 2011. Assessment and analysis of Information Quality: A Multidimensional Model and Case Studies. *International Journal of Information Quality* 2, 4 (2011), 300–323.

Christian Bizer and Richard Cygniak. 2009. Quality-Diven Information Filtering Using the WIQA Policy Framework. *Journal of Web Semantics* 7 (2009), 1–10.

Francesco Caruso, Munir Cochinwala, Uma Ganapathy, Gail Lalk, and Paolo Missier. 2000. Telcordia's Database Reconciliation and Data Quality Analysis Tool. In *Proc. of the 26th International Conference on Very Large Databases*. 615–618.

Anup Chalamalla, Ihab Iiyas, Mourad Ouzzani, and Paolo Papotti. 2014. Description and Prescriptive Data Cleaning. In *International Conference on Management of Data*. 445–456.

Michael Compton, Payam Barnaghi, Luis Bermudez, Raul Garcia-Castro, Oscar Corcho, Simon Cox, John Graybeal, Manfred Hauswirth, Cory Henson, Arthur Herzog, Vincent Huang, Krzysztof Janowicz, W. David Kelsey, Danh Le-Phouc, Laurent Lefort, Myriam Leggieri, Holger Neuhaus, Andriy Nikolov, Kevin Page, Alexandre Passant, Amit Sheth, and Kerry Taylor. 2012. *The SSN Ontology of the W3C Semantic Sensor Network Incubator Group*. Vol. 17. Web Semantics: Science, Services and Agents on the World Wide Web, 25–32.

Christian Furber and Martin Hepp. 2011a. SWIQA - A Semantic Web Information Quality Assessment Framework. In *Proc. of the European Conference on Information Systems*. Paper 76.

Christian Furber and Martin Hepp. 2011b. Towards a Vocabulary for Data Quality Management in Semantic Web Architectures. In *Proc. of the 1st International Workshop on Linked Web Data Management*. 1–8.

Floris Geerts, Giansalvatore Mecca, Paolo Papotti, and Donatello Santoro. 2014. Mapping and Cleaning. In *International Conference on Data Engineering*. 232–243.

C. Goble. 2002. Position Statement: Musings on Provenance, Workflow and (Semantic Web) Annotation for Bioinformatics. In *Proc. of Workshop on Data Derivation and Provenance*.

Olaf Hartig. 2009. Provenance Information in the Web of Data. In *Proc. of the Linked Data on the Web Workshop at WWW*.

Ian Jacobi, Lalana Kagal, and Ankesh Khandelwal. 2011. Rule-Based Trust Assessment on the Semantic Web. In *Proc. of the 5th Int. Conf. on Rule-Based Reasoning, Programming, and Applications*. 227–241.

Matthias Jarke, Manfred Jeusfeld, Christoph Quix, and Panos Vassiliadis. 1999. Architecture and Quality in Data Warehouses: an Extended Repository Approach. *Information Systems* 24 (1999), 229–253.

Shirlee Knight and Janice Burn. 2005. Developing a Framework for Assessing Information Quality on the World Wide Web. *Informing Science Journal* 8, 5 (2005), 159–172.

Christopher Lynnes, Edward Olsen, Peter Fox, Bruce Vollmer, Robert Wolfe, and Shahin Samadi. 2010. A Quality Screening Service for Remote Sensing Data. In *Proc. of the International Symposium on High Performance Distributed Computing*. 554–559.

Simon Miles, Paul Groth, Steve Munroe, and Luc Moreau. 2009. PrIME: A Methodology for Developing Provenance-Aware Applications. *ACM Transactions on Software Engineering and Methodology* 20, 3 (June 2009), 39–46.

Paolo Missier, Suzanne Embury, Mark Greenwood, Alun Preece, and Binling jin. 2006. Quality Views: Capturing and Exploiting the User Perspective on Data Quality. In *Proc. of the 32nd International Conference on Very Large Data Bases*. 977–988.

RA Peppler, CN Long, DD Sisterson, CP Turner, CP Bahrmann, SW Christensen, KJ Doty, RC Eagan, TD Halter, MD Ivey, NN Keck, KE Kehoe, JC Liljegren, MC Macduff, JH Mather, RA McCord, JW Monroe, ST Moore, KL Nitschke, BW Orr, RC Perez, BD Perkins, SJ Richardson, KL Sonntag, JW Voyles, and R Wagener. 2008. An Overview of ARM Program Climate Research Facility Data Quality Assurance. *The Open Atmospheric Science Journal* 2 (2008), 192–216.

Edoardo Pignotti, Peter Edwards, Nick Gotts, and Gary Polhill. 2010. Enhancing Workflow with a Semantic Description of Scientific Intent. *Journal of Web Semantics* 9 (2010), 222–244.

Markus Schaal, Barry Smyth, Roland M. Mueller, and Rutger MacLean. 2012. Information Quality Dimensions for the Social Web. In *Proc. of the Internation Conference on Management of Emergent Digital EcoSystems*. 53–58.

N R Velaga, J Nelson, S Sripada, P Edwards, D Corsar, N Sharma, and M Beecroft. 2012. Development of a Map-Matching Algorithm for Rural Passenger Information Systems via Mobile Phones and Crowd-sourcing. In *91st Annual Meeting of the Transportation Research Board of National Acadmies*.

Yair Wand and Richard Y. Wang. 1996. Anchoring Data Quality Dimensions in Ontological Foundations. *Commun. ACM* 39 (1996), 86–95.

Richard Y. Wang and Stuart E. Madnick. 1990. A Polygen Model for Heterogeneous Database Systems: The source tagging perspective. In *Proceedings of the 16th VLDB Conference*. Brisbane, Australia, 519–538.

K. Waterman and J. Hendler. 2013. Getting the Dirt on Big Data. *Big Data* 1 (September 2013), 137–140.

Brian Wolly. 2010. Vinton Cerf on Where the Internet Will Take Us. *Smithsonian Magazine* (August 2010). http://www.smithsonianmag.com/specialsections/40th-anniversary/Vinton-Cerf-on-Where-the-Internet-Will-Take-Us.html