International Journal of Computer Science and Applications
© Technomathematics Research Foundation
Vol. 11 No. 3, pp. 1 - 20, 2014

A Semantic-Enhanced Quality-based Approach to Handling Data Sources in Enterprise Service Bus*

Trinh Hoang Nguyen+, Kamyar Rasta++, Yohanes Baptista Dafferianto Trinugroho+, Andreas Prinz+

Department of Information and Communication Technology
University of Agder, Grimstad, Norway
+{*trinh.h.nguyen, dafferianto.trinugroho, andreas.prinz*}*@uia.no*
++*kamyar.rasta@gmail.com*

Data quality plays an important role in success of organizations. Poor data quality might significantly affect organizations' businesses since wrong decisions can be made based on data with poor quality. It is therefore necessary to make data quality information available to data users and allow them to select data sources based on their given requirements. Enterprise Service Bus (ESB) can be used to tackle data integration issues. However, data sources are maintained out of the ESB's control. This leads to a problem faced by users when it comes to selecting the most suitable data source among available ones. In this article, we present an approach to handling data sources in ESB based on data-quality and semantic technology. This introduces a new level of abstraction that can improve the process of data quality handling with the help of semantic technologies. We evaluate our work using three different scenarios within the wind energy domain.

Keywords: Quality-based; data source handling; data source combination; data quality

1. Introduction

Service-oriented architecture (SOA) has changed the way of designing software. SOA provides the possibility of integrating and distributing services in a loosely-coupled manner [Zhang (2010); Barry (2003)]. There are several SOA topologies such as peer-to-peer network, hub & spoke, pipeline, and enterprise service bus [Delia and Borg (2008)]. Among them, Enterprise Service Bus (ESB) is gaining more and more attention from industries. The idea of the ESB concept is that all the applications that are connected to a bus can share, produce, and consume information on the bus [Chappell (2009)].

ESB is an emerging technology derived from the combination of SOA and event-driven architecture (EDA) [Maréchaux (2006)]. EDA means that any event that happens inside or outside a business disseminates immediately to all interested

*This article is a revised and expanded version of a paper entitled, A Framework for Data Quality Handling in Enterprise Service Bus, presented at INTECH 2013 conference, 29 - 31 August, London, UK. ISBN: 978-1-4799-0047-3 parties that subscribe to the event [Michelson (2006)]. An example of this could be using a publish/subscribe mechanism in EDA to enable real-time monitoring of offshore wind farms. Whenever new data arrives, it is immediately published to a channel and notifications about it will be sent to subscribers instead of letting subscribers query for information every once in a while.

ESB provides a number of benefits such as loosely coupled architecture (i.e. whatever changed in an application will not affect other applications in the whole system), increased flexibility (i.e. easier to change as requirements change, standardized platform for integration), sharing common services (security, error management, reporting, etc.), and supporting a large number of communication patterns over different transport protocols [Papazoglou amd Van Den Heuvel (2007)]. ESB has been used in different areas such as eHealth [Trinugroho *et al.* (2012)], and oil & gas [OLF (2008)].

Although ESB handles communication between applications, it does not support a way to select the most suitable data source among several available ones. Assume that a user requests for wind speed in the North Sea for the last few months. The user wants to have data with accuracy and completeness in a certain interval. There are probably several available sources of wind speed data in the North Sea. How can we provide the most suitable data source to the user based on the data quality criteria? How can a system respond to the user's request if the requested data source is not available or the user's requirements are not met? Is there any way to combine the available data sources to produce an improved data source? Fig. 1 depicts an overview of the problem that we are targeting in this work.

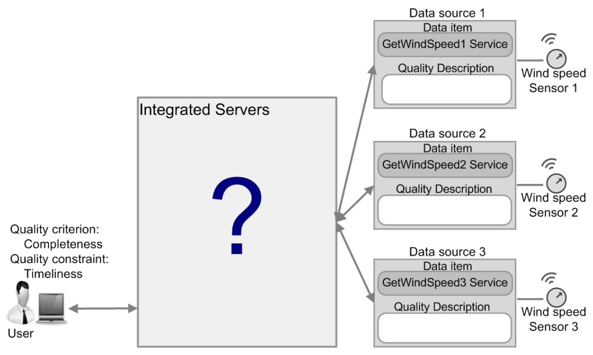

Fig. 1. An abstract view of the problem

This article answers the stated questions by proposing a quality-based approach to handling resources in ESB with semantic technologies. The rest of the article is organized as follows. Section 2 presents the main concepts of data sources and challenges in data source handling. Section 3 discusses data quality and its dimensions. Section 4 describes our proposed approach to tackle challenges in handling data sources. The implementation of the proposed approach is presented in section 5 along with the details of the system components and how they perform the desired functionalities. Related work and discussion are presented in section 6. Finally, section 7 concludes the paper.

2. Data source

This section explains the definition of data source in the context of our work. Challenges in handling data sources are discussed afterwards.

2.1. *Data source definition*

Data sources are sensor sources, web services that provide data generated by sensors or other appliances, and existing databases. A data source consists of three components: data stream, data source description, and quality description. Fig. 2 illustrates the components of a data source.

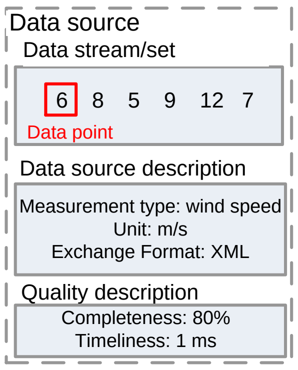

Fig. 2. Components of a data source

Data stream is a stream of signals coming from a source over time, while a data set is a set of data points coming from a static source, e.g., existing relational databases. Each signal in the stream is a data point, for instance, 6 (m/s) for wind speed at a specific date and time. An example of data stream is a set of 4000 wind speed records from the 1^{st} of May to the 1^{st} of June. Data streams are provided by data providers or by having direct connection to sensors. Data providers can make the data accessible via services.

Data source description contains information about measurement type, measurement unit, data exchange format. For instance, the description of a data source could be: (1) measurement type: wind speed, (2) unit: m/s, and (3) data exchange format: XML.

Quality description is a description of the quality of the data source. For example, the quality description of a data source states that the completeness is 80% and the timeliness is $1ms$.

2.2. *Type of data sources*

In this work, data sources are classified into *real* and *virtual* data sources. *Real* data sources are sensors or services that have access to sensors' measurements. Data quality of these data sources can be obtained either from devices' manufacturers or by computation using reference data sources. A *reference* (also known as benchmark in some literature [Ge and Helfert (2006)]) is a data source that can be either a previous measurement of the same data source or a mathematical model [Manyonge *et al* (2012); de Jesus Rubio *et al* (2011)] of that data. Reference data source are considered as real data sources.

In contrast, *virtual* data sources have no direct connection to sensors. These data sources are computed by combining more than one *real* data source. It is possible to combine data sources which measure different physical phenomena provided that there is a mathematical relationship between these physical phenomena. If the data sources provide data in different units (Fahrenheit and Celsius), a unit converter will be employed before combining.

Since virtual data sources are generated by combining real data sources, they also consist of three parts: data stream, data source description, and quality description. The virtual data stream is described by a formula, e.g., average of data streams from the real data sources. The quality description of the virtual data source is computed by combining the quality descriptions of the real data sources.

2.3. *Challenges in data source handling*

There are many challenges in handling data sources in ESB related to data quality, service selection, and data source integration. Here, we address those challenges that we try to overcome in this work.

2.3.1. *Availability of data quality description*

Typically, data are provided without any quality description attached to them. It is not clear what the accuracy or completeness of the data is. There are also cases where data quality is available but the service that makes data accessible does not provide any method to access the information. A way to compute data quality and make it available is therefore important.

A widespread issue in data integration is the management of data with insufficient quality. For example in offshore wind energy, a couple of sensors are deployed on a windmill and they frequently measure and deliver the data to the users and applications by means of services. As sensors are prone to failures, their results might be inaccurate, incomplete, and inconsistent [Snyder and Kaiser (2009)]. Therefore, the data quality issues should be handled in such a way that users and applications can specify the desired quality level of the data. Only when the data source has the requested quality descriptions it would be used for further processing.

2.3.2. *Data source selection based upon users' requests*

Assume a user can access data sources with data quality information available. How can a system fulfill the user's requests for data with given constraints on data quality? The user only cares about the requested data and its quality and does not care from which data source the data is selected. Thus, answering the request by giving a list of possible data sources is not a good answer. The aforementioned question can be formulated as follows: how to select the best suited data source among available data sources based upon user's defined quality criteria?

2.3.3. *Data source combination*

Another issue is that sometimes none of the available data sources has the required quality. In this case, it is possible to improve the quality of data to meet the user's requirement by combining existing data sources. A virtual data source is the result of the combination.

2.3.4. *Semantic inconsistency of data sources in sensor networks*

Sensors are becoming one of the main data sources since they are used intensively in many areas such as oil & gas, eHealth, smart grid, and smart cities. However, sensor data sources are described differently by their manufacturers. This leads to a semantic inconsistency issue when it comes to integrating different sensors into a system. It therefore makes the process of data source selection and combination more difficult.

2.4. ***Approaches to selecting data sources***

There are several ways of selecting data sources such as content-based filtering [Dumais *et al* (1988)], social information filtering [Shardanand and Maes (1995)], agent-based selection [Sreenath and Singh (2004)], and quality-based selection [Mihaila *et al* (2000)].

Content-based filtering is a traditional and static way of selecting a data source out of a list of available data sources. This approach filters data sources based on users' keywords. When users send requests for data, the content-based filtering statically selects the data source with more relevant description. This approach might not solve the selection problem if there are data sources with the same descriptions.

Another approach is the social information filtering. It refers to a sort of techniques to provide personalized recommendations for users according to the similarities of their interests. This approach is common in sites, e.g., Amazon and LinkedIn.

The third category of selection methods is the agent-based approach. Agents evaluate data sources by communicating, cooperating, and rating each other. Each agent can make decision and work autonomously as well.

The quality-based approach takes into consideration the importance of data quality. A data source is selected based on data quality dimensions given in users' defined requirements. Our work is based on this approach. We also use semantic technology to solve inconsistency issues and enable semantic description for sensor networks.

3. Data quality

Data quality plays an important role in the success of organizations [Wang (1998)]. Poor data quality might significantly affect organizations' businesses since wrong decisions can be made based on data with poor quality [Strong *et al* (1997); Huang *et al* (1998); Ge and Helfert (2006)].

In this section we define the basic terms and concepts in the article. Although there is no consensus on the definition of data quality [Klein (1998)], the data quality concept is defined as the extent to which attributes of data are suitable for their use. It is usually evaluated from a consumer point of view [Huang *et al* (1998)].

3.1. *Data quality dimensions*

Data quality is a multidimensional notion. There are more than 17 data quality dimensions which have been mentioned in literature, e.g., accuracy, consistency, confidence, interpretability, completeness, relevancy, timeliness. Some of the studies such as [Wang and Strong (1996); Lee *et al* (2002); DeLone and McLean (1992)] support several quality dimensions. These work often address data quality frameworks that encompass a rich set of data quality dimensions. On the other hand, studies like [Eppler (2006); Baumgartner *et al* (2010); Geisler *et al* (2011)] only cover a small number of dimensions because they are applying the data quality concept to a specific application domain.

3.2. *Selected data quality dimensions*

In this work, we select the most commonly used quality dimensions in the literature, i.e. *accuracy*, *completeness*, and *timeliness* to demonstrate the proposed approach. These data quality dimensions are defined differently in the literature. Here we define the terminologies based on the existing definitions and our understanding. Table 1 shows the notation that we use in the following definitions.

Accuracy is defined as how close the observed data are to reality. According to ISO 5725 standard [ISO (1994)], accuracy consists of precision and trueness.

- *Precision* is the closeness of agreement within individual results.
- *Trueness* is defined as the mean value of the difference of data source to the reality.

Table 1. Table of notation

Symbol	**Explanation**
D	Data source
R	Reference data source (reality)
N_D	total number of data points in D
N_R	total number of data points in R
$N_{D_{cons}}$	total number of consistent data points in D
d_i	a single data point in D
r_i	real value corresponding to d_i
x_i	d_i - r_i
$t(r_i)$	the moment when the data point i is due
$t(d_i)$	the moment when the data point i is available

We assume that the sensors are calibrated, meaning that the trueness is very close to zero. Therefore, we only consider precision as the accuracy in our system. A statistical measure of the precision for a series of repetitive measurements is the standard deviation.

Let x_i denote the difference between the measured data (d_i) and the reality (r_i) and μ denote the trueness. Thus, the accuracy of data source D can be obtained using Eq. (1).

$$Acc(D) = \sqrt{\frac{1}{N_D}\sum_{i=1}^{N_D}(x_i - \mu)^2} \tag{1}$$

Given $\mu = 0$ and $x_i = d_i - r_i$, Eq. (1) can be rewritten as follows.

$$Acc(D) = \sqrt{\frac{1}{N_D}\sum_{i=1}^{N_D}(d_i - r_i)^2} \tag{2}$$

Completeness is defined as the ratio of the number of successful received data points to the number of expected data points. The completeness of the data source D can be calculated using Eq. (3):

$$Compl(D) = \frac{N_D}{N_R} \tag{3}$$

For example, if a user requests for wind speed from a data source and he expects to get 60 data points in 1 hour. If the user has received only 40 data points in 1 hour, the completeness of the wind data source is 67%.

Timeliness is the average time difference between the moment a data point has been successfully received and the moment it is expected to be received. The timeliness of data source D is calculated using Eq. (4):

$$Time(D) = \frac{\sum_{i=1}^{N_D}(t(d_i) - t(r_i))}{N_D} \tag{4}$$

4. Semantic-enhanced quality-based data source handling

This section describes the proposed approach to handling data sources. By handling we mean that the approach offers ways to manage data sources, to insert a new data source, and to provide the best suited data source to users. An overview of our approach is illustrated in Fig. 3.

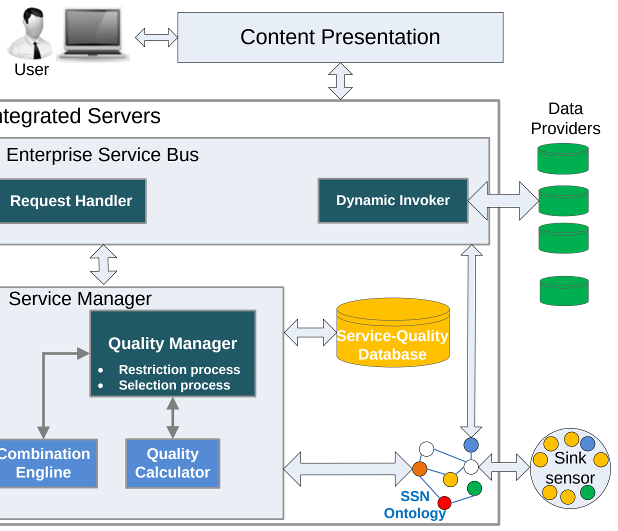

Fig. 3. An overview of the proposed approach

The *Request Handler* is an ESB-dependent module. If the ESB platform is changed, this module needs to be changed. It is used to parse and analyze requests from users. The *Dynamic Invoker* is responsible for invoking web services that describe data sources.

The *Service Manager* is in charge of finding proper data sources. It consists of three sub modules: quality manager, combination engine and quality calculator.

- The *Quality Manager* is used to select the most appropriate data sources in terms of data quality dimensions and constraints from the *Service-quality database*. The *Quality Manager* uses two processes to find the best data source for the user: restriction process and selection process. The purpose of the restriction process is to detect and filter out outliers from the results according to the given quality dimensions and constraints. The restriction process prepares a list of available *good* data sources and hands in this list to the selection process. The basic idea of selection is to find the *best* data source among *good* data sources according to selection quality dimension. If there are more than one data sources that meets users' quality requirements, the *Quality Manager* selects one of the *good* data sources randomly.

- The *Combination Engine* is used to combine existing data sources in order to generate virtual data sources. The module is called when there is no data source that meets the users' defined quality requirements. The combination process includes combination of data points, data source descriptions, and quality descriptions. Section 5.2.1 describes an example of using the *Combination Engine*.
- The *Quality Calculator* is used in case the data quality description of a data source is not available. The computation is done using equations mentioned in section 3.2. After computing the quality dimension the quality calculator can store the information in the *Service-quality Database*. An example of the use of the *Quality Calculator* is shown in section 5.2.2.

The *Quality Manager* generates a list of suitable data sources. The list contains information such as the data source address, parameters, and values. In case the *Combination Engine* is involved, the list will also contain information about a combination method. *Service Manager* passes the list to *Dynamic Invoker*. *Dynamic Invoker* then access data sources by invoking web services dynamically.

The *Service-quality Database* stores metadata of all available services and the *Semantic Sensor Network Ontology* specifies metadata of sensors such as location, type of sensor, unit, and sensors' quality. Details of the database and the ontology are presented in the next section.

4.1. *Integration & management of data sources*

Data can come directly from sensors or from services made available by service providers. Hence, data sources can be sensors or services. Metadata of (web) services are stored in a relation database while metadata of sensors are described and stored in a semantic sensor network ontology. Metadata of real-time data can be accessed through SSN and metadata of historical data can be accessed through relational database.

4.1.1. *The Service-quality Database*

The *Service-quality Database* is a relational database that stores metadata of all available services. It does not store measurement data. The database is divided into two parts. The first part contains information about services and their descriptions. The second part stores information about data quality of the services. Based on input from users, the *Quality Manager* uses the database to select the most suitable data source. The result of a query is typically a data source or a list of available data sources. The result is then delivered to the *Dynamic Invoker* service for subsequent binding invocation.

4.1.2. *A semantic sensor network ontology*

Sensors are being used intensively in different systems. This creates vast amount of data. However, a big portion of data cannot be transformed to knowledge due to the lack of integration and communication between sensor networks. In order to tackle this issue, the W3C Semantic Sensor Network Incubator group took the initiative in enabling SSN by proposing an SSN ontology [Compton *et al* (2012)]. The semantic sensor network (SSN) was introduced based on Sensor Web Enablement (SWE) standards proposed by the Open Geospatial Consortium (OGC) [Sheth *et al* (2008)] and the Stimulus-Sensor-Observation ontology design pattern [Janowicz and Compton (2010)]. Based on the SSN ontology, a number of developments have been reported in several work such as [O'Byrne *et al* (2010); Bröring *et al* (2011)]. In this section, we use SSN ontology to overcome the challenge posed in section 2.3.4.

Even though accuracy has been mentioned in the SSN ontology, data quality is not only about accuracy. Many work have reported that data quality should be defined beyond accuracy [Huang *et al* (1998)]. We therefore have extended the SSN ontology by adding some quality dimensions that are described in section 3.2.

The developed ontology contains spatial attributes (i.e. information about sensors' location), temporal attributes (i.e. information about timestamp), thematic attributes (i.e. information about sensor type, measurement, units), and quality attributes (e.g. accuracy, timeliness, completeness). Fig. 4 depicts a partial view of the developed ontology. The ontology describes two type of sensors: temperature sensor and wind speed sensor. Each sensor's value is classified in different level, e.g., high or low. The quality of measurements is defined by the *MeasurementProperty* class.

4.2. *Data source selection*

The data source selection answers the question of how to select the most suitable data source from available data sources. This section presents our solution to overcome the challenge posed in section 2.3.2. Based on a user request, the selection process requires a set of quality constraints and a selection dimension. The mandatory constraints describe conditions to be met by the data sources, and the optional selection dimension describes which dimension to use for finding the best data source. A scenario described in section 5.2.2 explains our point.

What if the requested data source is not available? The next section discusses the combination methods that are used to tackle the problem.

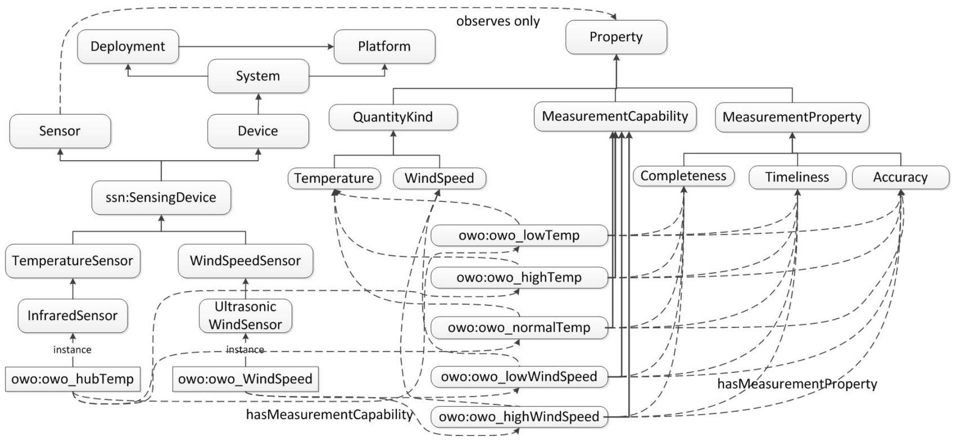

Fig. 4. A partial view of the quality-enhanced SSN ontology

4.3. *Data source combination*

In order to improve the data quality to meet users' requirements, it is possible to combine different data sources. This section presents a solution to the challenge mentioned in section 2.3.3. For simplicity, we consider only those data sources that measure the same physical phenomena. Let us consider a case where we need to combine two data sources D1 and D2. There are several methods to combine data sources, but looking deeply into combination methods is out of scope of this work. Let us examine three simple combination methods as follows.

- D1 (A) D2: taking a conventional average of the data sources D1 and D2.
- D1 $\bigoplus$ D2: use data points from D1 if available, otherwise use D2.
- D1 (E) D2: pick up the earliest received data point from either D1 or D2.

Assume that data points in data sources $D1$ and $D2$ are independent normally distributed random variables. Let $Acc(D1)$ and $Acc(D2)$ be the accuracy (precision) of $D1$ and $D2$ respectively. Let P(D1) denote the probability of the event *"D1 having data available"* and P(D2) denote the probability of the event *"D2 having data available"*. These two events are independent. Let $\overline{P(X)} = 1 - P(X)$ denote complementary of the event *"data source X having data available"*.

We also assume that the timeliness $Time(D1)$ and $Time(D2)$ of $D1$ and $D2$ are two independent exponentially distributed random variables as shown in Eq. 5. That said, $Time(D1) = \frac{1}{\lambda_1}$ and $Time(D2) = \frac{1}{\lambda_2}$.

$$f(t;\lambda) = \begin{cases} \lambda e^{-\lambda t}, & \text{if } t \geq 0 \\ 0, & \text{if } t < 0 \end{cases} \tag{5}$$

where λ is a rate parameter.

4.3.1. *D1 (A) D2*

Completeness: The completeness of the virtual data source is calculated by Eq. (6). This is also the result for the cases D1 $\bigoplus$ D2 and D1 (E) D2. In three cases, the value of the completeness of the virtual data source is improved.

$$Compl(D1\ (A)\ D2) = \overline{\overline{P(D1)}\ .\ \overline{P(D2)}} \tag{6}$$

Accuracy: Since Acc(D1) and Acc(D2) are normally distributed, sum of them is also normally distributed. As we mentioned in section 3.2, the sensors are calibrated and thus the trueness is equal to zero. The accuracy is then equal to the precision which can be obtained by Eq. (7):

$$Acc(D1\ (A)\ D2) = \sqrt{\frac{Acc(D1)^2 + Acc(D2)^2}{4}} \tag{7}$$

Timeliness: The timeliness of the virtual data source is the maximum value of the set which consists of $Time(D1)$ and $Time(D2)$, i.e. $max(Time(D1), Time(D2))$.which is not exponential. However, if $Time(D1)$ and $Time(D2)$ are identical and have rate parameter λ, we can easily obtain the probability density function (PDF) of the maximum as follows:

$$f_{tmax}(t;\lambda) = 2\lambda e^{-\lambda t}(1 - e^{-\lambda t}),\ t \geq 0 \tag{8}$$

The expected value is obtained via Eq. (9).

$$Time[D1\ (A)\ D2] = E[tmax] = \int_0^\infty t f_{tmax}(t;\lambda)dt = \frac{3}{2\lambda} \approx \frac{3}{2}Time(D1) \tag{9}$$

4.3.2. *D1 $\bigoplus$ D2*

Accuracy: The accuracy of virtual data source is calculated using weighted average as shown in Eq. (10):

$$Acc(D1 \bigoplus D2) = \frac{P(D1) * Acc(D1) + \overline{P(D1)} * P(D2) * Acc(D2)}{P(D1) + \overline{P(D1)} * P(D2)} \tag{10}$$

Timeliness: The timeliness of the virtual data source is calculated using Eq. (11):

$$Time(D1 \bigoplus D2) = \frac{P(D1) * Time(D1) + \overline{P(D1)} * P(D2) * Time(D2)}{P(D1) + \overline{P(D1)} * P(D2)} \tag{11}$$

4.3.3. *D1 (E) D2 method*

Accuracy: Let α be the probability of the event a data point $D1_i$ arrives before a data point $D2_i$. In other words, it is the probability that t(D1) is smaller than or equal to t(D2). α is obtained via Eq. (12)

$$\alpha = 1 - \frac{\lambda_2}{\lambda_1 + \lambda_2} \tag{12}$$

The accuracy of virtual data source is then calculated using Eq. (13):

$$Acc(D1\ (E)\ D2) = \alpha Acc(D1) + \overline{\alpha} Acc(D2) \tag{13}$$

Timeliness: In this case, the timeliness of the virtual data source is the minimum value of the set which consists of $t(D1)$ and $t(D2)$, i.e. $min(t(D1), t(D2))$. Given $t \geq 0$, the distribution function of the timeliness $min(t(D1), t(D2))$ is also exponential [Marshall and Olkin (1967)]. It can be obtained by considering complementary cumulative distribution function as shown in Eq. (14).

$$f_{tmin}(t;\lambda) = (\lambda_1 + \lambda_2)e^{-(\lambda_1+\lambda_2)t} \tag{14}$$

The timeliness is then obtained via Eq. (15)

$$Time[D1\ (E)\ D2] = \int_0^\infty t(\lambda_1 + \lambda_2)e^{-(\lambda_1+\lambda_2)t}dt = \frac{1}{\lambda_1 + \lambda_2} \tag{15}$$

The Eq. (15) can be rewritten through $Time(D1)$ and $Time(D2)$ as follows.

$$Time[D1\ (E)\ D2] = \frac{1}{\lambda_1 + \lambda_2} = \frac{Time(D1) * Time(D2)}{Time(D1) + Time(D2)} \tag{16}$$

4.3.4. *Summary of combination methods*

By combining data quality dimensions, we aim to generate a virtual data source with better data quality. The three combination methods have different effects on data quality dimensions. A method can increase or decrease the quality depends on the receiving data. Table 2 shows the results of applying combination methods on two independent data sources $D1$ and $D2$.

Table 2. The quality of the combined data source

Combination method	Completeness	Accuracy	Timeliness
D1 (A) D2	✓	✓	×
D1 $\bigoplus$ D2	✓	–	–
D1 (E) D2	✓	–	✓

✓: It can be better than both of D1 and D2.
–: It is not decidable. It varies from case to case.
×: It is worse than both of D1 and D2.

According to this table, all three methods can increase the completeness. By using average method, the combined data source would have better accuracy. This method also improves the accuracy. However, it makes the timeliness become worse. For the $\bigoplus$ method, both the accuracy and timeliness of the combined data source varies from case to case.The (E) method helps to increase the completeness and timeliness, but not accuracy. If the timeliness is the critical choice, (E) method is recommended to use.

5. Implementation

As a proof of concept, we have developed a prototype system based on the overview described in Fig. 3. This section discusses the internal implementation details of the prototype.

5.1. *Prototype description*

We use the client-server architecture to develop our prototype system. The client is the web-based client application that allows users to make requests. The client is also in charge of data visualization in terms of graphs. The integrated server (IS) is responsible for request handling and communicating with data providers.

Data providers make services available as web services and describe the services using Web Service Description Language (WSDL). Both Restlet and JAX-WS (Java API for XML Web Services) clients are employed to send data using SOAP protocol and REST architecture style.

5.1.1. *The client side*

The client side consists of a user interface where users can choose the measurement type, the quality dimensions, the constraints, and one quality dimension as selection dimension. We use Ajax (Asynchronous JavaScript and XML) technology to handle the messages from the server and display information on a graph. Flot, a JavaScript plotting library is used to produce graphical plots on a web browser [Laursen (2012)].

5.1.2. *The integrated servers*

The IS consists of following main components:

MuleESB[a]: is an open source enterprise service bus framework and we use it as the communication backbone of the system. It is easy to use Mule ESB, compared with other open source ESB frameworks such as PEtALS ESB, ServiceMix, and Open ESB [Ueno and Tatsubori (2006)]. Mule ESB is not based on JBI (Java Business Integration), but it provides seamless support for JBI containers [Rademakers and Dirksen (2008)]. Hence, it allows components of other ESBs, e.g., ServiceMix, which are based on the JBI model, to be used alongside MuleESB. Mule ESB is provided together with an Integrated Development Environment, Mule studio which makes the process of flow design much easier. Mule ESB also provides many connectors and transports, for instance REST, SOAP, JMS (Java Message Service) [Ziyaeva *et al* (2008)].

[a]http://www.mulesoft.org

Service Manager: is a .NET web service that is deployed on Internet Information Services (IIS)[b] web server.

Dynamic Invoker: is also a Java-based web service that dynamically binds and invokes services for gathering the selected data sources.

Service-quality Database: is developed using Microsoft SQL Server[c].

Request handler: is a Java web service deployed on the MuleESB. This service receives the information from web-based client application and forwards it to the *Service Manager*.

Quality calculator: The quality calculator is .NET web service which has a separate function for each quality dimension. It takes a data source and a reference data source, then computes the quality value.

Combination Engine: is a .NET web service. It takes a list of data sources and combines them according to the corresponding combination formula, e.g., the one presented in section 4.3 and stores a new virtual data source in the *Service-quality database*.

5.2. *Use case scenarios*

We use three offshore wind scenarios to demonstrate the use of the prototype system. A number of wind sensors (e.g., wind speed sensor) are attached to a wind turbine to monitor and control the wind turbine [Akhmatov and Knudsen (2002)]. Due to the weather conditions, the sensors are subject to the moisture and corrosion [Snyder and Kaiser (2009)]. Consequently, the quality of the data produced by them can be negatively influenced [Desholm (2003)]. It is therefore needed to have multiple sensors to measure the same physical phenomena.

5.2.1. *Scenario 1*

Description: There are three real wind speed data sources in the system. Each data source has two quality dimensions available, e.g., completeness and timeliness. Fig. 5 shows scenario 1 where the data is mainly provided by wind sensors through a number of wind services.

User's request: The user sends a request for wind speed. The completeness is chosen for the restriction process and its value is set to 75%. The timeliness is selected as the criteria for data source selection.

Results: First the restriction process is executed, data sources 2 and 3 are selected. The selection process based on timeliness is then executed. As the result, data source 3 is selected since it has better timeliness compared to data source 2.

[b]http://www.iis.net
[c]http://www.microsoft.com/en-us/sqlserver/default.aspx

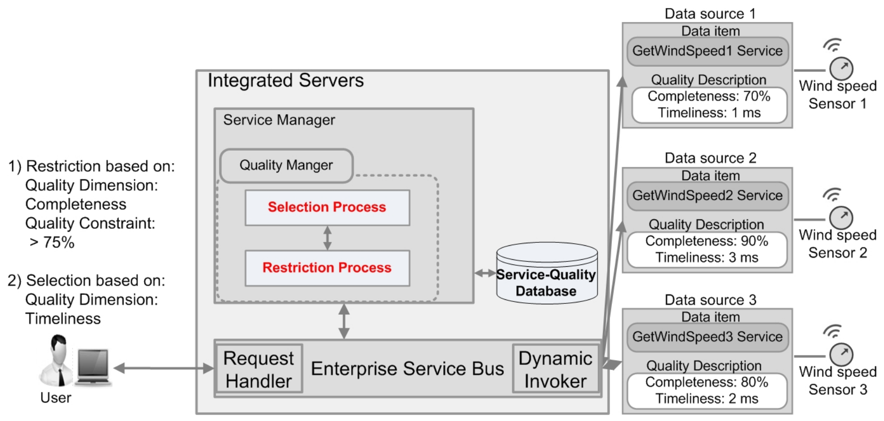

Fig. 5. Scenario 1. Data sources: three real data sources. Quality dimensions: completeness and timeliness. Request: completeness $\geq$ 75% and the selection quality dimension is timeliness.

5.2.2. *Scenario 2*

Description: There are two real wind speed data sources. Each data source has one available quality dimension, e.g., completeness. Fig. 6 shows the scenario where the data is mainly provided by wind sensors through a number of wind services.

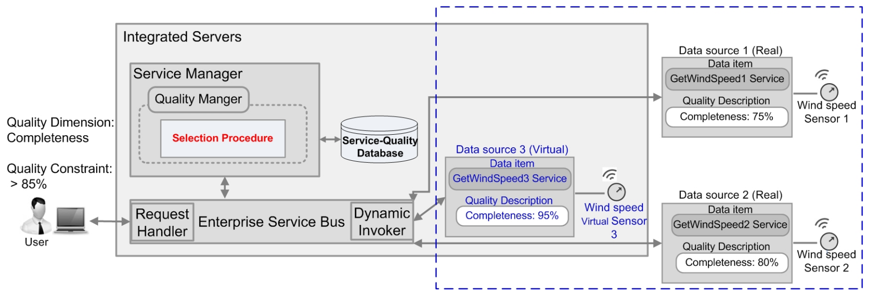

Fig. 6. Scenario 2. Data sources: two real data sources. Quality dimension: completeness. Request: completeness $\geq$ 85%.

User's request: The user issues a request for wind speed whose the completeness is more than 85%.

Results: The completeness of data sources 1 and 2 are 75%, 80% respectively. It means that none of the given data sources meets the user's requirement. The system therefore to combine these two data sources. As the result, a virtual data source whose the completeness is 95% is produced. The virtual data source that is selected by *Quality Manager* and its graph is shown to the user. The reason is that data sources 1 and 2 have missed some parts of the data but when they are composed, the missing parts are filled.

5.2.3. *Computation of data quality scenario*

Description: There are three real wind speed data sources in the system. The quality descriptions of data sources 1 and 2 are not available. The reference data source has one available data quality dimension, e.g., completeness. Figure 7 illustrates the scenario.

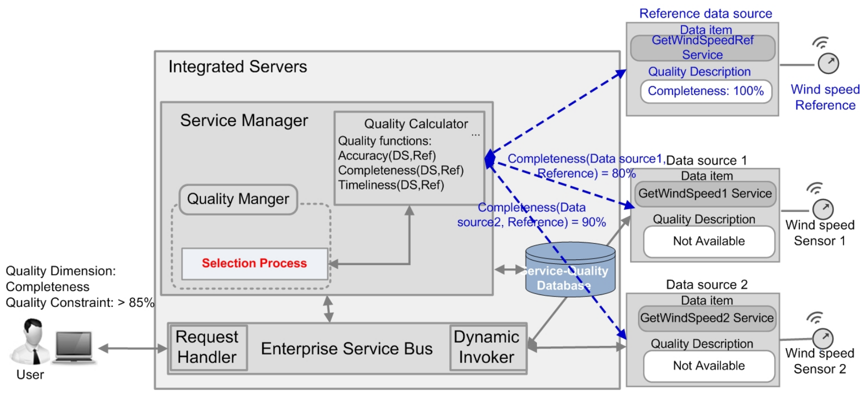

Fig. 7. No data quality descriptions available for data source 1 and 2. The reference data source has the completeness available. Request: completeness $\geq 85\%$.

User's request: The user issues a request for wind speed whose the completeness is more than 85%.

Results: By using the reference, the *Quality Calculator* computes the completeness and the results are 80% and 90% for data source 1 and 2, respectively. Data source 2 is selected and returned to the user.

6. Related work and discussion

In order to support intelligent handling of data sources in ESB, some efforts have been reported. For example, authors in [Ziyaeva *et al* (2008)] proposed a multi-layered framework to support content-based intelligent routing path construction and message routing. Their approach facilitates the data source selection based on the message content. Besides handling data sources in ESB, there is an attempt to combine data sources based on data quality as reported in [Mihaila *et al* (2000)]. There has also been some research within the area of multi-sensor data fusion that employed different techniques to generate a better virtual sensor, for instance, artificial intelligence, pattern recognition, statistical estimation are used to fuse multi-sensor data [Hall and Llinas (1997); Le and Hauswirth (2009)].

Different from these work, we have taken into consideration data quality dimensions as criteria for filtering and selecting the best suitable data source. We have also implemented semantic technologies to describe services and sensor networks.

In this work, we have only discussed combination methods for data sources that measure the same physical phenomena, e.g., wind speed. In reality, sometimes we need to combine two or more measurement quantities in order to fulfill users' requests. The combination can be based on mathematical relations between the quantities, e.g., from rotor speed, power output of a wind turbine can be easily derived provided power coefficient of the generator is given and the pitch angle is constant.

7. Conclusions

Data quality is one of data consumers' concerns. It is an important feature of any business organization. Data consumers normally need to be aware of data providers and the quality of data that they get. Indeed, it is complicated for data consumers, data integrators, and even data providers when the number and diversity of data sources increase. ESB is proposed to make the data integration easier. This will reduce the details that data consumers have to know in order to get access to quality data. However, there is a lack of unambiguous ways to manage data sources in current available open source ESB platforms. In this work, we have presented a quality-based approach to managing, selecting, and providing the most suitable data source for users based upon their quality requirements. The approach gives users possibility of getting the most suitable data source from the available ones. It also increases the chance to find the requested data source by combining multiple data sources so that the users' quality requirements are met. We have also enhanced the data quality-based approach by implementing semantic technology to describe sensor networks in an unambiguous manner.

Acknowledgement

This work has been (partially) funded by the Norwegian Centre for Offshore Wind Energy (NORCOWE) under grant 193821/S60 from the Research Council of Norway (RCN). NORCOWE is a consortium with partners from industry and science, hosted by Christian Michelsen Research.

References

Akhmatov, V., & Knudsen, H. (2002). An Aggregate Model of a Grid-Connected, Large-Scale, Offshore Wind Farm for Power Stability Investigations - Importance of Windmill Mechanical System. *International Journal of Electrical Power & Energy Systems* **24** , no. 9, 709–717.

Baumgartner, N., Gottesheim, W., Mitsch, S., Retschitzegger, W., and Schwinger, W. (2010). Improving Situation Awareness In Traffic Management. In Proc. Intl. Conf. on Very Large Data Bases.

Bai, X., Xie, J., Chen, B., and Xiao, S. (2007). Dresr: Dynamic Routing in Enterprise Service Bus. In: e-Business Engineering, ICEBE 2007. IEEE International Conference on,pp.528–531, IEEE.

Barry, D. K. (2003). *Web services and service-oriented architectures: the savvy manager's guide*. Morgan Kaufmann.

Bröring, A., Maué, P., Janowicz, K., Nüst, D., & Malewski, C. (2011). Semantically-enabled sensor plug & play for the sensor web. *Sensors*, 11(8), 7568-7605.

Compton, M., Barnaghi, P., Bermudez, L., García-Castro, R., Corcho, O., Cox, S., Graybeal, J., et al. (2012). The SSN ontology of the W3C semantic sensor network incubator group. *Web Semantics: Science, Services and Agents on the World Wide Web*, **17**, 25–32.

Chappell, D.A. (2009). *Enterprise Service Bus*. O'Reilly media.

Delia, P., & Borg, A. (2008). *Mule 2: A Developers Guide*. Apress.

de Jesus Rubio, J., Figueroa, M., Pacheco, J., & Jimenez-Lizarraga, M. (2011). Observer design based in the mathematical model of a wind turbine. *International Journal of Innovative Computing Information and Control*, 7(12), 6711-6725.

Desholm, M. (2003). Thermal Animal Detection System (TADS): Development of a Method for Estimating Collision Frequency of Migrating Birds at Offshore Wind Turbines. Tech. report, *National Environmental Research Institute*.

Dumais, S.T., Furnas, G.W., Landauer, T.K., Deerwester, S., and Harshman, R. (1988). Using latent semantic analysis to improve access to textual information. In: Proceedings of the SIGCHI conference on Human factors in computing systems, pp. 281–285, ACM.

DeLone, W.H., and McLean, E.R. (1992). Information Systems Success: The Quest for the Dependent Variable. *Information Systems Research*, **3**, no. 1, 60–95.

Eppler, M.J. (2006). *Managing Information Quality: Increasing The Value of Information in Knowledge-intensive Products and Processes*. Springer.

Ge, M., and Helfert, M. (2006). A Framework to Assess Decision Quality using Information Quality Dimensions. In Proceedings of the 11th International Conference on Information Quality - ICIQ, v. 6, p. 10–12.

Geisler, S., Weber, S., and Quix, C. (2011, November). An ontology-Based Data Quality Framework for Data Stream Applications. 16th International Conference on Information Quality, pp. 145–159.

Hall, D.L., and Llinas, J. (1997). An introduction to multisensor data fusion. *Proceedings of the IEEE*, **85**, no. 1, pp. 6–23.

Huang, K.T., Lee, Y.W., and Wang, R.Y. (1998). Quality Information and Knowledge. *Prentice Hall PTR*.

ISO (1994). ISO 5725-2: 1994: Accuracy (Trueness and Precision) of Measurement Methods and Results-Part 2: Methods for the Determination of Repeatability and Reproductibility.

Janowicz, K., and Compton, M. (2010). The Stimulus-Sensor-Observation Ontology Design Pattern and its Integration into the Semantic Sensor Network Ontology. 3rd International Workshop on Semantic Sensor Networks.

Klein, B.D. (1998). Data Quality in The Practice of Consumer Product Management: Evidence from the Field. *Data Quality*, **4**, no. 1.

Karastoyanova,D., Wetzstein, B., Lessen, T.V., et al. Semantic Service Bus: Architecture and Implementation of A Next Generation Middleware. Data Engineering Workshop, IEEE 23rd International Conference on, pp. 347–354, IEEE.

Laursen, O. (2012). Flot - Attractive Javascript plotting for jQuery - Google Project Hosting. Accessed: 23/05/2013.

Lee, Y.W., Strong, D.M., Kahn, B.K., and Wang, R.Y. (2002). AIMQ: A Methodology For Information Quality Assessment. *Information and Management*, **40**, no. 2, 133–146.

Manyonge, A. W., Ochieng, R. M., Onyango, F. N., & Shichikha, J. M. (2012). Mathematical Modelling of Wind Turbine in a Wind Energy Conversion System: Power Coefficient Analysis. *Applied Mathematical Sciences*, 6(91), 4527-4536.

Maréchaux, J. L. (2006). Combining service-oriented architecture and event-driven architecture using an enterprise service bus. *IBM Developer Works*, 1269–1275.

Michelson, B. M. (2006). Event-driven architecture overview. *Patricia Seybold Group, 2.*

Marshall, A.W., and Olkin, I. (1967). A multivariate exponential distribution. *Journal of the American Statistical Association*, **62**, no. 317, 30–44.

Mihaila, G., Raschid, L., and Vidal, M.E. (1999). Querying quality of data metadata. In Third IEEE META-DATA Conference.

Mihaila, G., Raschid, L., and Vidal, M.E. (2000). Using quality of data metadata for source selection and ranking. WebDB, pp. 93–98.

O'Byrne, D., Brennan, R., and O'Sullivan, D. (2010). Implementing the draft W3C semantic sensor network ontology. Pervasive Computing and Communications Workshops (PERCOM Workshops), 8th IEEE International Conference on,pp. 196–201.

The Norwegian Oil Industry Association (2008). Reference architecture of IT systems for OLFs IO G2 (2008). Technical report.

Le P.D, and Manfred Hauswirth, M. (2009). Linked open data in sensor data mashups. Proceedings of the 2nd International Workshop on Semantic Sensor Networks.

Papazoglou, M. P., and Van Den Heuvel, W. J. (2007). Service oriented architectures: approaches, technologies and research issues. *The VLDB journal*, 16(3), 389–415.

Rademakers, T., and Dirksen, J. (2008). *Open-source esbs in action*, Manning.

Sheth, A., Henson, C., and Sahoo, S.S. (2008). Semantic sensor web. *Internet Computing*, **12**, no. 4, 78–83, IEEE.

Snyder, B., and Kaiser, M.J. (2009). Ecological and Economic Cost-Benefit Analysis of Offshore Wind Energy. *Renewable Energy*, **34**, no. 6, 1567–1578.

Strong, D.M., Lee, Y.W., and Wang, R.Y. (1997). Data Quality in Context. *Communications of the ACM*, **40**, no. 5, 103–110.

Shardanand, U., and Maes, P. (1995). Social information filtering: algorithms for automating word of mouth. Proceedings of the SIGCHI conference on Human factors in computing systems, pp. 210–217, ACM Press/Addison-Wesley Publishing Co.

Sreenath, R.M., and Singh, M.P. (2004). Agent-based service selection. *Web Semantics: Science, Services and Agents on the World Wide Web*, **1**, no. 3, 261–279.

Trinugroho, Y.B.D., Rasta, K., Nguyen T.H., Fensli, R.W., and Reichert, F. (2012). A Real-Time Web-Based Health Monitoring System Based on Enterprise Service Bus. In: Proceedings of 2012 11th IADIS International Conference on WWW/Internet, p. 165–172.

Ueno, K., and Tatsubori, M. (2006). Early capacity testing of an enterprise service bus. Web Services, ICWS'06. International Conference on, pp. 709–716, IEEE.

Wang, R.Y., and Strong, D.M. (1996). Beyond Accuracy: What Data Quality Means to Data Consumers. *Journal of Management Information Systems*, 5–33.

Wang, R. Y. (1998). A product perspective on total data quality management. *Communications of the ACM*, 41(2), 58-65.

Ziyaeva, G., Choi, E., and Min, D. (2008). Content-Based Intelligent Routing and Message Processing in Enterprise Service Bus. Convergence and Hybrid Information Technology, 2008. ICHIT'08. International Conference on, pp. 245–249, IEEE.

Zhang, Y. (2010). Dependable ESB Routing in Hybrid Service Execution Environment. *Advances in Information Sciences and Service Sciences (AISS)*, 2(1), 83–93.