Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2016) 24: 1099 – 1116

doi:10.3906/elk-1304-79

A classification of semantic conflicts in heterogeneous Web services at message level

Ibrahim Ahmed AL-BALTAH*, Abdul Azim Abdul GHANI, Wan Nurhayati Wan AB RAHMAN, Rodziah ATAN
Department of Information Systems, Faculty of Computer Science and Information Technology, University Putra Malaysia, Serdang, Malaysia

Received: 10.04.2013 • **Accepted/Published Online:** 13.02.2014 • **Final Version:** 23.03.2016

Abstract: Since the last decade, semantic conflicts have been considered as a critical problem in establishing seamless message exchange between Web services. To provide the essential step for solving this problem, semantic conflicts have to be defined and classified properly. Most of the existing classifications that attempt to classify semantic conflicts lack completeness and accuracy. Therefore, this paper aims at proposing a new complete and accurate semantic conflicts classification, whose purposes are identifying and classifying all potential conflicts that may arise in heterogeneous Web services at the message level. This classification has three main classes based on the cause of conflicts, which are representation class, interpretation class, and structure class. Furthermore, the implementation of the classification starts with proposing three objective design criteria and ends with evaluating the classification against the proposed criteria. Three real scenarios from different domains are used to evaluate the proposed classification. The evaluation result shows that the proposed classification is the most complete and accurate when compared with other classifications.

Key words: Web services, semantic conflicts, conflicts classification

1. Introduction

Web services have become the main components for the Web, which add a new level of functionality to the Web [1]. The communication between Web services is based on the message exchange mechanism [2], and this communication should preserve the syntax and semantics of the message that is being exchanged. Thus, Web services should agree on both the syntax and semantics of the data before communication can take place [3]. SOAP is the common protocol that provides a basic messaging framework to enable Web services to exchange messages [4]. In this respect, it is worth pointing out that SOAP does not convey the semantics of the data [5,6]. However, in terms of syntax, Web services are perhaps able to exchange even heterogeneous messages successfully [7], despite the data being incorrect or meaningless to the message receiver [8]. Therefore, an agreement between the messages of Web services in terms of the meaning of the exchanged data is required in order to achieve sound communication at the semantic level as well. In contrast, inadequate semantics make data exchange impossible [9]. Thus, sharing the same understanding of the exchanged data between Web services is the fundamental condition to achieve semantic interoperability [10].

Semantic conflict is the natural result from the heterogeneities that exist among Web service messages, and it could arise at any domain [11]. So far, this problem has not been solved efficiently and effectively,

*Correspondence: abou_amel@yahoo.com

even though considerable efforts have been proposed as general solutions to solve semantic conflicts in various domains such as e-government [12] and geospatial analysis [13]. Despite all the works that have been done in this field, semantic conflict is still a gray area [14], which needs further investigation to identify the origin of each semantic conflict type and then relate every semantic conflict type to its specific cause in order to give clear insight for solving this problem. Consequently, it is very important to classify these conflicts completely and accurately in order to provide a clear guideline to facilitate the process of detecting and/or solving semantic conflicts.

In the Web service literature, there is a lack of a complete and accurate classification that is able to classify all likely semantic conflicts at the message level of Web service. Furthermore, there are some conflicts between the existing classifications in terms of the causes and definitions of the conflicts. In order to address this gap, we propose a new classification that aims at classifying all likely semantic conflicts that manifest themselves in Web services at the message level. As a consequence, the focus of this work is on semantic conflicts classification. The main classes of this classification are interpretation, representation, and structure classes, which are different from the other classifications' classes. This makes these classes be disjointed where every semantic conflict belongs only to a single class. In this respect, our classification also provides a common understanding of the semantic conflicts in terms of their causes and definitions. Moreover, the significance of the proposed classification, in terms of detecting and/or solving semantic conflicts, includes the following:

- The classification is an important and useful tool, which can be used as a guideline for any techniques that aim at detecting and/or solving the semantic conflicts problem.
- The classification is helpful to divide the whole semantic conflict problem into subproblems.
- The classification organizes the conflicts into main classes, in which every class consists of related conflicts based on a specific cause of the conflict.

Our evaluation results confirm that the proposed classification fulfills its design criteria, and it is the most complete and accurate classification compared with the two selected classifications from the literature. This paper has the following specific contributions:

- A new comprehensive classification that aims at classifying semantic conflicts of Web services at the message level, as well as unifying semantic conflicts in terms of their causes and definitions.
- A new novel evaluation process framework for a comparative evaluation of the proposed classification.

2. Related work

The handling of semantic conflict has received massive attention from researchers due to its existence in different fields. However, semantic conflict is still the crucial problem that has to be solved to enhance the interoperability between different systems, databases, Web services, etc. In practice, the essential step to solve semantic conflicts is by classifying these conflicts properly in order to facilitate their detection and reconciliation. Thus, researchers have proposed several classifications in different domains. In fact, our classification is inspired by various areas, namely database, XML, and Web services. Thus, this paper considers only the related work with respect to the mentioned areas.

However, semantic conflict essentially originates from problems in multiple-database designs [15]. Thus, we reviewed only the classifications that our classification was inspired by, and they were used as foundations for proposing the other classifications.

In the area of databases, a comprehensive classification for semantic conflicts in multiple-database systems was proposed by Naiman and Ouksel [16], which includes three main dimensions of naming, abstraction, and level of heterogeneity. Naming and abstraction are identified as the inherent dimensions of semantic conflicts, which are combined with the third dimension (level of heterogeneity), and these dimensions can exist at entity, instance, and attribute levels. The level of heterogeneity is included in order to be used as an assistant for schematic mapping between the corresponding databases [16]. The reasons for these are the use of different representations or interpretations of the information [17]. Additionally, six desirable properties are suggested to be incorporated in the classification.

Another conflict classification was provided by Kashyap and Sheth [18], in which five broad classes of heterogeneities are defined: domain incompatibility, entity definition incompatibility, data value incompatibility, abstraction level incompatibility, and schematic discrepancies. On the other hand, Ceruti and Kamel [19] proposed a classification based on information granularity with only two broad categories: data and schema. Similarly, Park and Ram [20] classified semantic conflicts into data level and schema level. Data-level conflicts occur due to the use of different representations and interpretations of the same data, while schema-level conflicts happen when multiple logical structures are used for the same application domain. This classification is considered as a well-known classification for information systems to facilitate semantic interoperability [21].

From a different perspective, Pokraev et al. [10] identified two different levels of conflicts as value-level and type-level conflicts. The former level arises when different representations are used to represent the same value, while the latter arises when different entity types are used to interpret the same value.

Nowadays, most business data are published to the Web as XML documents [22,23] to get the advantage of being XML, the well-known accepted standard to exchange data on the Web [23,24]. In this case, semantic conflicts can also be found in the context of XML data [23]. A comprehensive classification of conflicts that arise in heterogeneous XML data sources was proposed by Pluempitiwiriyawej and Hammer [25]. This classification has three main classes, which are data conflict, structural conflict, and domain conflict. Data conflict refers to contradictions related to similar values, structural conflict refers to the overlapping of schema representing the data, and domain conflict refers to the use of different exhibits for the semantics of data resources.

In another case, Kyong-Ha et al. [26] proposed a completely different classification of conflicts. This classification defines two classes of conflicts based on the components of XML schema, which would be the reason for the conflicts in XML schema during the integration process. The first class is schematic conflict, which occurs due to heterogeneous schema definition. The second class is semantic conflict, which occurs due to discrepancies in meaning between multiple schema definitions. Furthermore, information granularity was identified as a conflict in [26], while it was the base for the classification proposed in [19].

The semantic conflict that is manifested in federated databases is now undoubtedly revealed in the way of federating Web services [27]. Therefore, some studies have tried to classify the conflicts that are likely to occur between heterogeneous Web services based on some classifications from database areas. For example, Aragão and Fernandes [27] classified the conflicts that are likely to occur in Web service federations. This classification was inspired by [28,29] from the database area, and it identifies two levels of heterogeneities, which are at description level and value level. Description-level heterogeneities arise in the case where one or some of the message elements are described differently between the sender and receiver Web services. Value-level heterogeneities arise in the case where message values are defined differently between the communicated Web services. However, this classification does not cover all semantic conflicts that might occur in the Web service context, such as abstraction-level conflict, which occurs at the schema level.

On the other hand, Nagarajan et al. [8] proposed a specific classification for message-level heterogeneities in Web services. This classification was adapted from the database area [18,28]. They classified message-level heterogeneities into three levels: the attribute level, entity level, and abstraction level. The semantic conflicts that arise at the attribute level are due to the multiple descriptions that are used for semantically similar attributes. Entity-level conflicts refer to the multiple descriptions that are used for semantically similar entities. The abstraction-level conflicts might occur at both the attribute and the entity level when two entities or attributes are represented at different levels of abstractions where they are semantically similar. Although this classification is very specific to the scope of this study, some conflicts at the attribute level were not included.

As it can be observed in this section, different conflict classifications with different or the same classes have been proposed by different authors. These classifications have been identified either by sharing a common basis or by adapting previous classifications. Furthermore, similar conflicts are classified into different conflict types by different classifications. For example, the conflict that arises due to the use of different data type of the same data was considered in [8] as a representation type of conflict, while in [27] it was considered as a data type of conflict. Thus, the intention of our proposed classification is to unify the previous classifications into a unique classification that captures all possible conflicts at the message level of Web services.

3. The design criteria

From our point of view, the success of any technique that tends to detect and/or to solve semantic conflicts relies on the classification that has been used during the implementation of the technique. Therefore, such a classification should be designed in the way that makes it acceptable and trustable to be used. This can be accomplished by designing the classification based on some criteria that can ensure its quality.

As mentioned above, this classification provides the vital step to facilitate the automation (fully or partially) of the detection and/or solving of the semantic conflicts process. Thus, the classification should be designed according to some criteria to meet the requirements of its design goal. Hence, the evaluation of the classification can be done against the design criteria. In this work, we propose three design criteria for designing a classification for semantic conflicts at the message level of heterogeneous Web services. These criteria were derived based on analysis of the semantic conflicts problem and investigation of some real Web services scenarios. These design criteria are as follows:

- ***Minimality:*** In this context, minimality refers to the degree of the absence of the redundant classes in a classification. In other words, the classification is minimal when it has no redundant classes and every element of the classes appears only once. Therefore, there is no chance to reduce any class or element from the minimal classification. Furthermore, reducing any class or element from the classification will make it unable to classify all the encountered semantic conflicts, and then the result of the classification will be incomplete because some semantic conflicts will remain unclassified. Nevertheless, minimality does not guarantee the completeness of the classification; it rather guarantees that the classification has no redundancies.
- ***Completeness:*** The classification should completely cover all conflicts at the message level in the Web service. In this context, completeness is defined as a ratio of coverage of the classification. The classification is complete if it is able to classify all the conflicts. Otherwise, it will be considered as an incomplete classification. Nonetheless, the completeness does not guarantee the minimality of the classification; rather, it guarantees the coverage of the classification regardless of whether the classification has redundancies or not.

- ***Accuracy:*** The classification should be accurate. In this context, accuracy means that if any conflict is encountered in Web services, it should be classified by the classification accurately. In other words, any possible conflict that might be encountered in Web services should correspond to only one conflict in the classification.

4. Semantic conflicts classification

Our proposed classification classifies the conflicts from the perspective of when the conflicts occur. There are three main situations when semantic conflicts would occur. Specifying these situations is very important in order to classify the related conflicts for every situation. Semantic conflicts arise due to different interpretations and representations of the same data as well as due to different logical structures of the same application domain [20–30]. In the proposed classification, each situation is formed as a class, and thus this classification contains three main classes. Each class is a set of conflicts grouped together on the basis of sharing the same reason for the conflicts. The main classes are representation class, interpretation class, and structure class, which are disjointed. This means that a conflict that belongs to a particular class will not be in other classes. The classes being disjointed gives our classification an important advantage while categorizing the conflicts. This advantage prevents any contradiction from taking place between the classes. Figure 1 shows the proposed classification that resulted from exploring and analyzing different conflict classifications identified in the literature. The aim is to produce a complete and accurate classification. The following subsections describe each class in the proposed classification. For the purpose of presenting illustrative examples of Web services messages we will use the following syntax:

Message i: Entity (attribute 1, attribute 2,. . . .., attribute n), where (message i) indicates the message number.

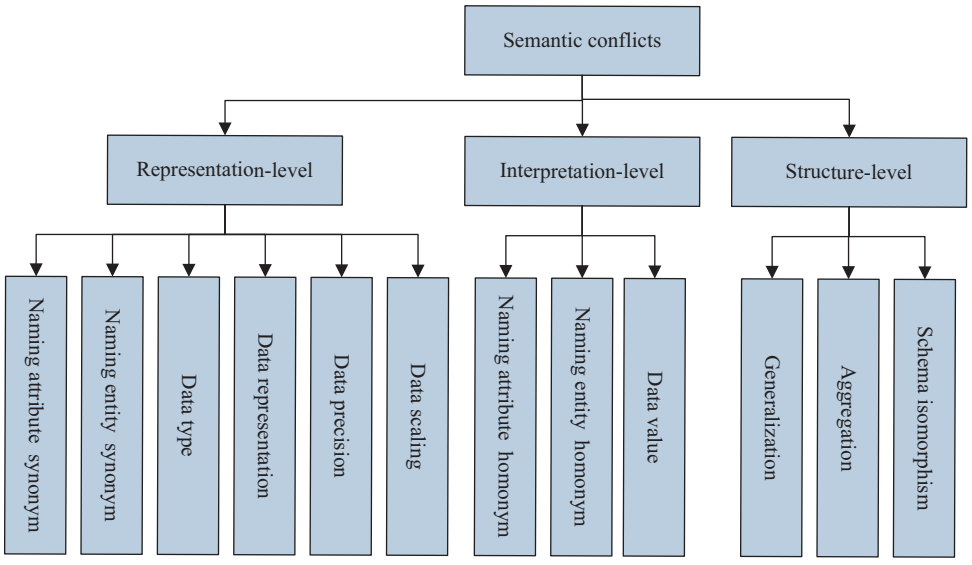

Figure 1. Classification of message-level semantic conflicts in Web services.

4.1. Representation class

This class contains all likely conflicts that arise due to the use of different representations of the same data. The relevant conflicts for this class are naming entity synonym, naming attribute synonym, data representation, data type, data precision, and data scaling.

4.1.1. Naming entity synonym

The naming entity synonym arises due to using multiple names to represent multiple semantically similar (synonyms) entities. Consider the below two messages:

Message 1: **Grade**(stdID, course, **grade**, semester)

Message 2: **Score**(stdID, course, **score**, semester)

Grade and **Score** are semantically equivalent entities, but are represented differently. Thus, the conflict between these two entities is naming entity synonym conflict.

4.1.2. Naming attribute synonym

This conflict arises due to using multiple names to represent multiple semantically similar attributes synonyms. For example, the **grade** attribute from **Message 1** and the **score** attribute from **Message 2** are semantically equivalent attributes, but are represented differently. Therefore, this conflict is naming conflict synonym at the attribute level.

4.1.3. Data representation

This type of conflict occurs at the attribute level when the same data are represented differently as multiple formats of the same data [21]. Consider the following two messages:

Message 3: Student(stdID, name, **date**, semester, grade)

Message 4: Student(stdID, firstName, lastName, date, **birthDate**, score)

In ***Message 3***, **date** is represented using (dd/mm/yyyy) format, while **birthDate** in ***Message 4*** is represented using (mm-dd-yyyy). Both attributes refer to the student birth date, but they are represented differently.

4.1.4. Data type

Data type conflict arises at the attribute level when semantically related attributes are defined with different data types [31], or when different data type precisions are used. An example of this type of conflict can be found between the **grade** attribute from **Message 1** and the **score** attribute from **Message 2**. The data type of the attribute **grade** is string, while the data type of the attribute **score** is float.

4.1.5. Data precision

This conflict arises when different precisions are used to express the same data values. For example, student grade is expressed on an A–F scale in the attribute **grade** in **Message 1**, while in the attribute **score** in **Message 2** it is expressed on a 1–100 scale.

4.1.6. Data scaling

Data scaling conflict arises when two similar attributes are represented using two different units and measures [18]. In the Web services context, using different units to measure the same concepts is common due to the policy of Web service providers. Consider the following messages:

Message 5: Course(coursId, name, department, **cost**)

Message 6: Course(coursId, name, department, **cost**)

The attribute **cost** in **Message 5** is represented in RM (Ringgit Malaysia), while attribute **cost** in **Message 6** is represented in YR (Yemen Riyal).

4.2. Interpretation class

This class contains all likely conflicts that arise due to the use of different interpretations of the same data. The relevant conflicts for this class are naming entity homonym, naming attribute homonym, and data value.

4.2.1. Naming entity homonym

This type of conflict occurs when the same name is used to represent two semantically unrelated entities (homonyms). Consider the following messages:

Message 7: **Program**(ProId, name, department)

Message 8: **Program**(ProId, name, department)

The entity **Program** from **Message 7** refers to the specialization (software engineering, multimedia,. . .), while in **Message 8,** it refers to the academic level (bachelor degree, master degree, or PhD). Both entities are represented similarly, while they are semantically unrelated.

4.2.2. Naming attribute homonym

Naming attribute homonym conflict arises due to using same name to represent two semantically unrelated attributes (homonyms). For example, the attribute ProId in **Message 7** refers to the specialization identification, while the attribute ProId in **Message 8** refers to academic level identification. These similar attributes are representing two semantically different attributes.

4.2.3. Data value

According to Halevy [32], the heterogeneity that occurred in the actual data values is known as data value conflict. This type of conflict arises when the same data value has different interpretations for its meaning, and it always arises at the attribute level. Consider the following two messages:

Message 9: Person(name**, age_category**), where the person is considered to be an adult if the age is over 18 years

Message 10: Person(name, **age_category**), where the person is considered to be an adult if the age is over 17 years

Let us assume that the data value for **age_category** is "*adult*"; in this case, the same data value "*adult*" has two different interpretations, which might be interpreted in **Message 9** as age over 18 years or in **Message 10** as age over 17 years.

4.3. Structural class

As we mentioned previously, structural class conflicts arise due to the use of multiple logical structures for the same application domain. In our classification context, this class arises when different logical structures are used to model the input and output messages of the Web service. The possible conflicts in this class are schema isomorphism, aggregation, and generalization conflict.

4.3.1. Schema isomorphism

This type of conflict occurs when the same entity is represented by different sets of attributes. To exemplify this type of conflicts, consider the following messages:

Message 11: **StudentAddress**(Id, AptNo, streetName, ZipCode, City, State)

Message 12: **StudentAddress**(Id, address)

As can be seen from **Message 11** and **Message 12**, both of them represent the same concept, "student address", but with different numbers of attributes.

4.3.2. Generalization

This conflict results when there are multiple choices for modeling the entity. For example, let us assume that there are two choices for modeling the "Staff" entity at different abstraction levels. In one Web service the "Staff" entity is modeled in the faculty as an entity for all academic and nonacademic staff, while another Web service may have two distinct representations as "Academic" and "Nonacademic". Thus, the "Staff" entity exists at two different abstraction levels, which leads to the generalization conflict. The following messages were used to illustrate this type of conflict:

Message 13: Nonacademic(StaffID, name, department)

Message 14: Staff(StaffID, name, department)

As can be seen, **Message 14** represents the staff entity at a more general level than **Message 13**.

4.3.3. Aggregation

This type of conflict may occur due to mapping an entity in one Web service to a group of entities in another Web service. According to [7], this conflict arises when semantically similar entities are represented at different levels of generalization in two Web services. The following messages' schemas were used to illustrate this type of conflict:

Message 15: **Course**(courseID, name, department)

Message 16: **Teacher** (Id, name, courseId, department)

In **Message 15**, a set of course entities is a teacher entity in **Message 16**.

It is worth pointing out that, in some situations, more than one conflict type may occur between two attributes of messages. The cause of such situations is the existence of multiple differences in the attributes, which leads to the arising of different conflict types. For example, as shown in **Message 1** and **Message 2**, there is a naming attribute synonym conflict between the attributes of grade and score, and at the same time there is another conflict between these attributes called a data type conflict. This situation does not contradict the accuracy of the classification, as every conflict will correspond to only one conflict type in the classification based on the cause of the conflict.

5. Evaluation

In this section, we present the evaluation of the proposed classification based on its design criteria. A close look at the current semantic conflict classifications shows that none of them have been quantitatively evaluated, unlike our classification. In our case, the proposed classification was evaluated against three real scenarios. We compare the proposed classification with the most relevant classifications in the area of Web services.

5.1. Evaluation planning

5.1.1. Scenario selection

For evaluation purposes, three real scenarios are selected from different publications, and every scenario represents different conflicts from different domains. Furthermore, the conflicts in these scenarios capture most of the potential conflicts that are likely to occur at the message level. The selected scenarios are described below.

Scenario 1 (travel booking): The first scenario considers a person who plans a trip from Europe to Japan and plans to rent a car during his stay in Japan, as described in [33]. This scenario includes three Web services, namely, Flight-booking, Car-rental, and Addition Web services. Flight-booking is provided by a European branch of the company, and thus it computes the price in euro with a scale factor of 1 and adheres to European notation to represent the date and time (dd.mm.yyyy and 12:00 AM/PM). Car-rental computes the price using yen with a scale factor of 1000, and it adheres to Japanese notation to represent the date and time (yy.mm.dd and 24:00). Addition is used to calculate the total cost for the flight ticket and the car rental. This scenario starts when a person specifies the departure date, return date, and number of persons, which are the input for Flight-booking. Some data of the output message from Flight-booking will be used as an input message for Car-rental, such as DepartureDate and ReturnDate, and some data of the output messages from both Flight-booking and Car-rental will be used as input messages for the Addition Web service to calculate the total cost, such as price of the flight ticket and price of the car rental. Some conflicts prevent establishing a successful process between those Web services, as shown in Table 1.

Table 1. Semantic *conflicts* found in the travel booking scenario.

Web service	Input message's elements	Output message's elements	Conflicts type	#conflict
Flight-booking	DepartureAirportCode DepartureDate DestinationAirportCode ReturnDate NumberOfPersons	DepartureAirportCode DepartureDate DepartureTime DestinationAirportCode ReturnDate ReturnTime FlightNumber FlightCategory NumberOfPersons Price	1- Data presentation between DepartureDate and ReturnDate (Flight-booking uses dd.mm.yyyy and 12:00 AM/PM notation, while Car-rental uses yy.mm.dd and 24:00 notation to represent data and time) 2-Value heterogeneity between the scale factors (one scale factors of 1 and the other of 1000) 3-Data unit conflict between the prices (Flight-booking computes prices in euro, while Car-rental computes prices in yen) 4-Structural heterogeneities between prices concepts are different from one agency to another 5-Synonym conflict between "VAT-Included" and "TVAIncluse", which denotes whether or not Value-Added Taxes are included in a price	5
Car-rental	DepartureDate ReturnDate	RentalIDnumber StartDate/StartTime EndDate/EndTime TypeOfCar ClassOfCar CarIDnumber Price		
Addition	Price(from Flight-booking Web service) Price(from Car rental Web service)	Total Price		

Scenario 2 (lookup information): This scenario was described in [7], which considered the process of sending mails to customers of a company by using their phone numbers. Two Web services are involved in this scenario, the PhoneLookup service and the Geocode service, which are provided by different providers. The input of PhoneLookup is the listed phone number to provide reverse phone lookup information. Geocode

provides the demographic and logistical information for the provided address (from PhoneLookup). This process has to call these two Web services to get the full address for each customer based on their phone numbers. The PhoneLookup service will be called first by using phone numbers to return the contact information as an output message. Second, the Geocode service will be called using some data of the output message that is given by the PhoneLookup service to return the full address that is required from the company to send the mails as an output message. Exchanging messages between these Web services is difficult due to the semantic conflicts between the output message of PhoneLookup and the input message of Geocode. According to Nagarajan et al. [7], this scenario comprises several conflicts, as shown in Table 2.

Table 2. Semantic conflicts found in the lookup information scenario.

Web service	Input	Output	Conflicts	#conflicts
GeoPhone	TelephoneNumber	ListingName FirstNam LastName Address City State PostalCode PhoneNumber Published	Naming conflict at attribute level between: 1-Address and AddressLine1, AddressLine2 2-City, State, PostalCode and city_state_zip 3-Naming conflict at entity level between Listing and GetGeoCodeUSA Generalization conflict between: 4-Address VS AddressLine1, AddressLine2 5-City, State, PostalCode and city_state_zip 6-Schema isomorphism between: Listing and GetGeoCodeUSA	6
USGerocoding5	AddressLine1 AddressLine2 City_state_zip	CensusTract StateNumber CountryNumber BlockNumber BlockGroup Latitude longitude		

Scenario 3 (PET virtual enterprise): This scenario was reported in [27]. This scenario has three Web services (PDP, ET, and TS), which are provided by specialist providers. These Web services are orchestrated to computationally capture a Virtual Enterprise VE called PET. PET starts when a request for a new television program is received by the PDP. This request will be sent to the PDP as an input message. Once the PDP receives this request, it invokes the ET to produce the typeface and send the output message for the requested program, and then the PDP invokes TS to produce the title sequence for the requested program using the output message of the ET. Some conflicts manifest themselves in this scenario, especially between PDP and ET Web services. Table 3 demonstrates those conflicts as identified and reported in [27].

5.1.2. Comparison

For the purpose of comparison with our classification (we code this as C1), two other different classifications were selected. The selected classifications are Web service heterogeneity classification [27] and message-level heterogeneity classification [8]. These two classifications were selected because they classify the conflicts at the message level, which is the subject of this work. Table 4 shows the overview of the classifications.

Table 3. Semantic conflicts found in PET scenario.

Web service	Input	Output	Conflicts	#conflicts
PDP	prodSpecs	typeFaceSpecs varieties serif	1-Naming conflict synonym between *varieties* from PDP and *styles* from Et 2-Naming conflict homonym between *price* from PDP (which records price only) and *price* from ET (which records price and tax) 3-Data type between *Serif* from PDP (Boolean, which is built-in XML schema type) and *Serif* from ET (Choices, which is user-defined type (Yes or No)) 4-Arity heterogeneity between *typeface(styles, serif, LowerCase)* from ET and *typefacespace(varieties, serif)* from PDP 5-Representational heterogeneity, PDP represents the style bold with *bold* while ET represents it with *bd* 6-Interpretation heterogeneity, PDP may interpret *size* to be in pica while ET may interpret it to be in points	6
ET	typeface serif styles lowrCase	typeFace		

Table 4. Description of the selected classifications.

Name	Code	Source	Main levels	Conflict type
Message-level heterogeneity classification	C2	[8]	Attribute level	Naming, data representation, data scaling
			Entity level	Naming, schema isomorphism
			Abstraction level	Generalization, aggregation, attribute entity
Web services heterogeneity classification	C3	[27]	Description level	Naming conflict, type conflict, cardinality
			Value level	Interpretational, representational

5.1.3. Measures

As we have mentioned previously, our classification is evaluated against its design criteria, which are minimality, completeness, and accuracy. Below are details on how to measure the design criteria.

Minimality: Minimality is widely used to measure the quality of the subject under investigation based on redundant aspects in the subject's requirements. For example, the UML schema is minimal if the schema has no redundant aspect of the requirements [34]; in schema integration, minimality is used to measure the quality of the schema based on the degree of the absence of the redundant elements in the schema [35]. Nevertheless, Naiman and Ouksel [16] considered minimality as one of the desirable properties of a classification of semantic conflicts; therefore, the classification is minimal if the set of classes is enough to represent semantic conflicts in the messages under consideration. In this work, minimality of the classification is straightforward, in which the classification is minimal if it has no redundant classes and all the classes are disjointed; otherwise, it will be considered as not minimal. The minimality of the classification is measured as:

$$\text{Minimality} = \left(1 - \frac{\sum_{i=0}^{n} R\,(Ci)}{\#\,\text{all classes}}\right) \times 100\%, \tag{1}$$

where R(Ci) is the redundant classes in the classification, minimality = 1 if R(Ci) = 0 and all classes are disjointed, and minimality $<$ 1 otherwise.

Completeness: The classification must be suitable for classifying message-level conflicts in the Web service domain. Consequently, these conflicts should be completely covered by the classification. Otherwise, the classification will be considered as incomplete. Therefore, completeness is used as a measure of coverage of the classification [36]. Thus, the classification is complete if it is able to classify all likely conflicts, and it will be considered as incomplete if at least one conflict remains unclassified. In this work, completeness is defined as the percentage of the number of conflicts that have been correctly classified, divided by the total number of the relevant conflicts (see Section 6.2 for more details about correct and incorrect classified conflicts, and for the relevant conflicts). The completeness of the classification is measured as:

$$\text{Completeness} = \frac{\#\text{correct classified conflicts}}{\#\text{total relevant conflicts}} \times 100\%. \tag{2}$$

Accuracy: Accuracy is one of the design criteria of our classification to ensure that all message-level conflicts are classified accurately. In this context, the classification is accurate if it is able to classify every conflict to exactly one corresponding conflict from the classification. In this work, accuracy is defined as the fraction of the number of conflicts that have been correctly classified, divided by the total classified conflicts. The total classified conflicts are the sum of correct classified conflicts and incorrect classified conflicts. The accuracy of the classification is measured as:

$$\text{Accuracy} = \frac{\#\,\text{correct classified}}{\#\text{total classified conflicts}} \times 100\% \tag{3}$$

Table 5 illustrates the conflicts that damaged the accuracy and completeness of C2 and C3.

Table 5. The conflicts that damaged the accuracy and completeness of C2 and C3.

Scenario	Conflicts	
	C2	C3
Scenario 1	1-Value heterogeneity between the scale factors (one scale factors of 1 and the other of 1000) 2-Structural heterogeneity between price concepts differs from one agency to another	1-Value heterogeneity between the scale factors (one scale factors of 1 and the other of 1000) 2-Data unit conflict between the prices (Flight-booking computes prices in euro, while Car-rental computes prices in yen) 3-Structural heterogeneities between price concepts are different from one agency to another
Scenario 2	-	1-Schema isomorphism between: Listing and GetGeoCodeUSA Generalization conflict between: 2-Address VS AddressLine1, AddressLine2 3-City, State, PostalCode and city_state_zip
Scenario 3	1- Data type between *Serif* from PDP (Boolean, which is built-in XML schema type) and *Serif* from ET (Choices, which is user-defined type (Yes or No)) 2-Arity heterogeneity between *typeface(styles, serif, LowerCase)* from ET and *typefacespace(varieties, serif)* from PDP 3-Interpretation heterogeneity, PDP may interpret *size* to be in pica while ET may interpret it to be in points	1-Arity heterogeneity between *typeface(styles, serif, LowerCase)* from ET and *typefacespace(varieties, serif)* from PDP 2-Interpretation heterogeneity, PDP may interpret *size* to be in pica while ET may interpret it to be in points

5.1.4. Threats to validity

The aim of this section is to discuss the threats to validity for this evaluation. We used real scenarios, which are relatively complete, from previous studies. Even though the size of these scenarios is quite small in terms of the number of semantic conflicts, their size is adequate and reasonable to make the evaluation realistic. Despite the size of the scenarios, they implement different conflicts from different domains, which are commonly encountered in practice. Therefore, the size of these scenarios does not reduce the significance of the evaluation. As for the classifications under evaluation, the selected classifications are the closest (that concern classifying semantic conflicts of Web services at message level) to our classification, and they are the only available classifications reported in the literature at the time of the evaluation. Finally, the process of extracting the semantic conflicts from the scenarios was done manually, which could possibly bias the results. To avoid this bias, we adopted the extracted conflicts that are reported in the related literature [7,27,33].

5.2. Evaluation execution process

The general overview of the evaluation execution process is depicted in Figure 2. The process is started by selecting one classification to classify all conflicts found in each scenario. To allow the selected classification to take place and classify every conflict that belongs to each scenario, all the scenario's conflicts were extracted and stored in a list called the relevant conflicts list. This list stores only one scenario's conflicts at a time, and then after the selected classification has classified every conflict from the relevant list, this list will be reset to store the next scenario's conflicts. The result from performing the selected classification for every scenario will be grouped into two lists. The first list is called the correct classified list, which contains all classified conflicts that matched the appropriate conflict reason from the semantic conflict reasons list that triggers this conflict. For example, if the selected scenario has a conflict type called data scaling conflict, then this conflict will be added to the correct classified list if and only if the conflict reason that caused this conflict was due to using different units and measures.

The second list is called the incorrect classified list, which contains the classified conflicts that did not match the appropriate conflict reason from the semantic conflict reasons list that causes this conflict. For example, if the selected scenario has a conflict type called data type conflict, then this conflict will be added to the incorrect classified list if and only if this conflict has been classified using the selected classification but the reason that caused this conflict was other than using different data types. However, the selected classification may not be able to classify all the conflicts stored in the relevant conflicts list for the selected scenario. As an illustration, this situation was clearly found when C3 was selected to classify the lookup information scenario's conflicts, as this scenario has generalization conflicts, and C3 does not have this type of conflict. Therefore, this conflict remained unclassified using C3.

Subsequently, the calculation process for completeness and accuracy of the classification will take place after ensuring that the selected classification has been performed for each conflict in the relevant conflict list of the selected scenario. The completeness of the selected classification will be calculated using Eq. (2) based on the selected scenario's conflicts, in which the total numbers of conflicts stored in the relevant conflict list and in the correct classified list represent the total relevant conflicts and the correct classified conflicts in Eq. (2), respectively. Similarly, the completeness of the selected classification will be calculated using Eq. (3) based on the selected scenario's conflicts, in which the number of conflicts stored in the correct classified list represents the number of correctly classified conflicts in the formula, while the total classified conflicts are obtained from the addition of the correct classified list and the incorrect classified list.

This process will be repeated using the selected classification until no more scenarios are left. The same

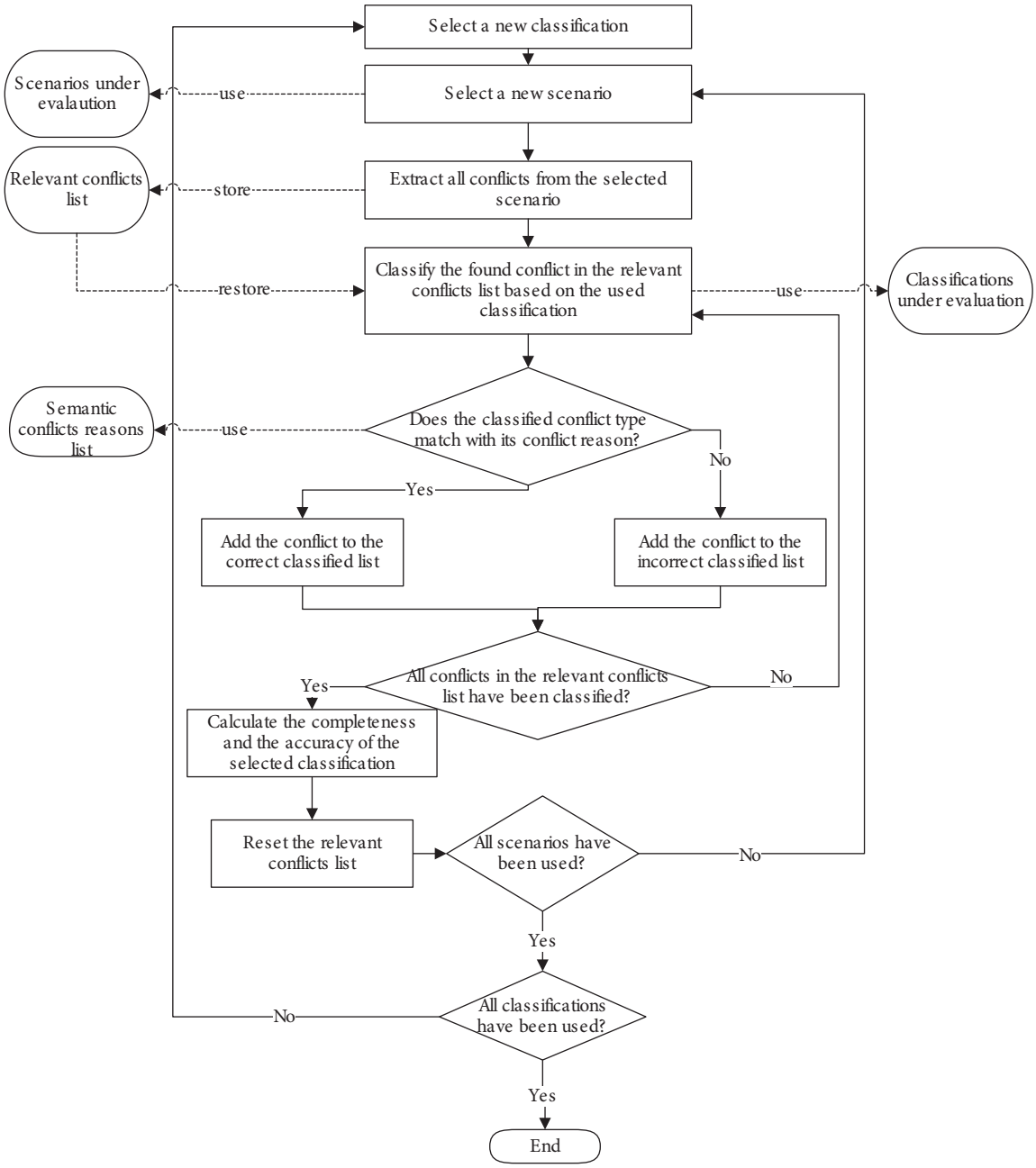

Figure 2. Evaluation execution process framework.

process will be performed using another classification. This process will be repeated until no more classifications under evaluation are left.

6. Results and discussion

The importance of evaluating our classification against its design criteria is to determine whether the classification has met its design goal. Similarly, a quantitative evaluation helps to reveal to what extent this classification has met its design criteria and to what extent this classification can be used to classify the semantic conflicts between heterogeneous messages.

Regarding the minimality, there is no difference between the classifications. This is due to the fact that the classifications have no redundant classes, which means R(Ci) = 0 for any conflict class Ci. Thus, all the classifications have the same results in terms of minimality (100%). This indicates that the classes of every classification are disjointed with no chance for reducing any class.

In terms of the completeness, the results show that our classification (C1) is the most complete among the three scenarios. As can be seen from Figure 3, the completeness results of each classification vary from one scenario to another, in which the classification is complete in some scenarios and incomplete in other scenarios. For example, message-level heterogeneity classification (C2) is complete at 100% in scenario 2; in contrast, it is incomplete in scenario 1 (60%) and scenario 3 (50%). This is because scenario 1 has value heterogeneity and structural heterogeneity, which are not included in C2, and thus these conflicts remained unclassified using C2. Similarly, C2 could only classify exactly half of the conflicts found in scenario 3 and the second half (data type, arity heterogeneity, and interpretation conflicts) remained unclassified, because these conflict types are not included in C2. Therefore, C2 can be considered as an incomplete classification in some scenarios as can be concluded from the completeness results, which means it does not capture all likely conflict types in some domains. On the other hand, Web services heterogeneity classification (C3) scores almost half of our classification results in the three scenarios. This is because C3 does not capture all possible conflict types in the scenarios such as data unit conflicts, data scaling, generalization, aggregation, schema isomorphism, and data value conflicts. It is thought that data unit conflict is likely to occur due to the variety of the measurement units; for example, different countries may use different currencies [21]. However, according to the completeness results depicted in Figure 3, C3 is the least complete classification compared to C1 and C2.

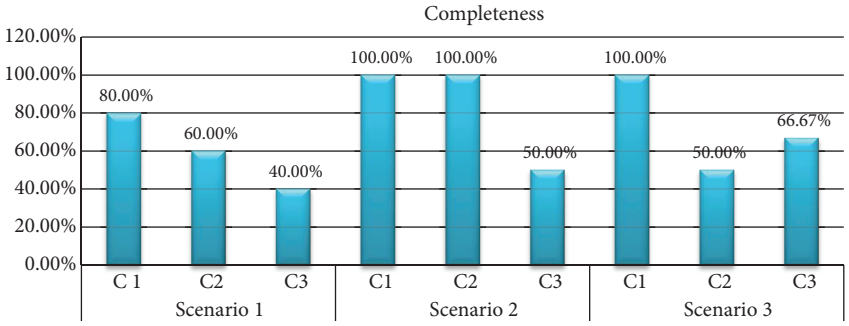

Figure 3. The completeness results.

In the same manner, the comparison results depicted in Figure 4 indicate that our classification is the most accurate classification compared with the others using the selected scenarios. Similarly, C2 seems to be an accurate classification in scenarios 1 and 2, but the value of its accuracy in scenario 3 is only 75%, which indicates that in some scenarios the accuracy of C2 is below 100%. For example, C2 considers data representation conflict and date type conflict as one conflict type only, i.e. data representation conflict. However, in practice, they cannot be considered as one conflict type, because data type conflict arises due to the use of different data types [31], while data representation arises due to the use of different representations (formats) to represent the same data [21]. On the other hand, the accuracy value of C3 is between 40% and 75%, unlike the accuracy of our classification, at least in the selected scenarios. This is due to the fact that some conflicts were incorrectly classified using C3. However, C2 is relatively as accurate as C1, as shown in the accuracy results (see Figure 4).

To deal with semantic conflicts at the message level of Web services, classifications must expose and

categorize all possible conflicts in a logical manner. One of the appropriate approaches is to investigate the problem from real scenarios in different domains.

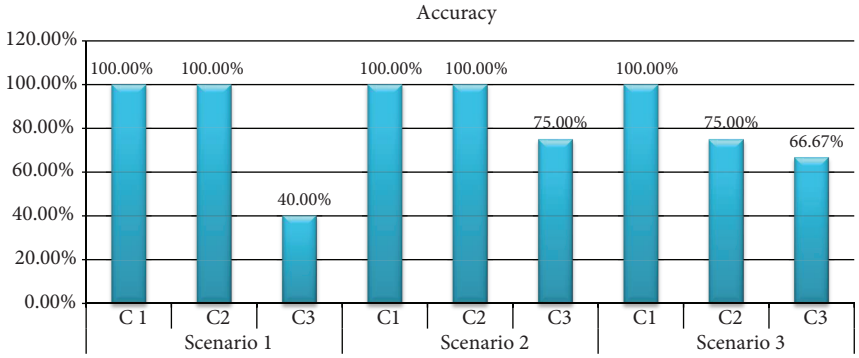

Figure 4. The accuracy results.

The evaluation results reported in this study indicate that our classification is able to achieve better results in terms of completeness and accuracy than other classifications. This is due to the fact that our classification is able to classify most of the likely conflicts completely and accurately from the messages of Web services. Furthermore, the implementation of our classification was based on analyzing and understanding the origin of semantic conflicts, which makes our classification more realistic.

Our results have several implications for proposing semantic conflict classifications. First, to the best of our knowledge, there is no such quantitative evaluation that aims at evaluating the classifications in the area of semantic conflicts (see, e.g., [16,25,29]); therefore, the results would be used as evaluation benchmarks with respect to completeness and accuracy. Second, the results give hints regarding the design criteria of the classification, in which the classification might be complete but on the other hand not accurate, and vice versa. This is to say that, it is not necessary for the complete classification to be accurate, and it is not necessary for the accurate classification to be complete.

To detect and/or resolve semantic conflicts, we suggested use of ontology, in which the implementation of this ontology should be based on the proposed classification. For detection purposes, every element of the input and the output messages should be mapped into the ontology, and some detection rules should be applied to these mappings to detect the conflicts between the elements of the messages. For resolving purposes, we recommend to use the data mediation technique, as it is the common technique that considers resolving semantic conflicts between heterogeneous Web services [37]. Mapping the elements of the input and the output messages to the ontology should be done to allow data meditation to take place.

7. Conclusion

The classification of semantic conflicts is a vital tool that facilitates the process of detecting and resolving semantic conflicts, thus improving the communication between heterogeneous Web services. In this paper, we have proposed a new classification that aims at classifying semantic conflicts, which appear in heterogeneous Web services at the message level. This new classification gives a common understanding of the semantic conflicts problem by providing a clear definition for every conflict type. This classification has three main disjointed classes, namely interpretation, representation, and structure. Our classification was designed based on three design criteria: minimality, completeness, and accuracy. In the evaluation process, we used these criteria to evaluate our classification by executing the evaluation process framework that we have proposed in this paper.

It can be seen from the evaluation results that our classification successfully fulfills its design criteria and obtained better results in terms of completeness and accuracy compared with the considered classifications using different scenarios selected from different domains. Furthermore, the results confirm the applicability of our classification in different domains.

As future work, the evaluation should be extended to include more sophisticated scenarios in order to ensure the applicability of the proposed classification in such scenarios. Currently we are also investigating a systematic way to encode the classification into ontology or rules in order to be used in any technique that concerns detection and/or resolution of these conflicts.

References

[1] Kopecky J, Vitvar T, Bournez C, Farrell J. SAWSDL: Semantic annotations for WSDL and XML schema. IEEE Internet Comput 2007; 11: 60–67.

[2] Kona S, Bansal A, Simon L, Mallya A, Gupta G, Thomas DH. USDL: A service-semantics description language for automatic service discovery and composition. Int J Web Serv Res 2009; 6: 20–48.

[3] Kannan G, Arindam B. HP Web services architecture overview. In: W3C Workshop on Web Services; 11–12 April 2001; San Jose, CA, USA.

[4] Albreshne A, Fuhrer P, Pasquier-Dorthe J. Web Services Technologies: State of the Art: Definitions, Standards, Case Study. Fribourg, Switzerland: University of Fribourg, 2009.

[5] Wang H, Huang JZ, Qu Y, Xie J. Web services: Problems and future directions. Web Semantics 2004; 1: 309–320.

[6] Diamadopoulou V, Makris C, Panagis Y, Sakkopoulos E. Techniques to support Web service selection and consumption with QOS characteristics. J Netw Comput Appl 2008; 31: 108–130.

[7] Nagarajan M, Verma K, Sheth A, Miller J. Ontology driven data mediation in Web services. Int J Web Serv Res 2007; 4: 104–126.

[8] Nagarajan M, Verma K, Sheth A P, Miller J, Lathem J. Semantic interoperability of Web services - challenges and experiences. In: International Conference on Web Services; 18–22 September 2006; Chicago, IL, USA. New York, NY, USA: IEEE. pp. 373–382.

[9] Muthaiyah S, Kerschberg L. Achieving interoperability in e-government services with two modes of semantic bridging: SRS and SWRL. Journal of Theoretical and Applied Electronic Commerce Research 2008; 3: 52–63.

[10] Pokraev S, Reichert M, Steen M, Wieringa R. Semantic and pragmatic interoperability: a model for understanding. In: Open Interoperability Workshop on Enterprise Modeling and Ontology for Interoperability; 13–14 June 2005; Porto, Portugal. pp. 1–5.

[11] Reiter T, Altmanninger K, Bergmayr A, Schwinger W, Kotsis G. Models in conflict-detection of semantic conflicts in model-based development. In: 3rd International Workshop on Model-Driven Enterprise Information Systems (MDEIS'07); 12–13 June 2007; Funchal, Madeira. pp. 29–40.

[12] Guijarro L. Semantic interoperability in eGovernment initiatives. Comp Stand Inter 2009; 31: 174–180.

[13] Vaccari L, Shvaiko P, Pane J, Besana P, Marchese M. An evaluation of ontology matching in geo-service applications. Geoinformatica 2012; 16: 31–66.

[14] Khattak AM, Pervez Z, Sarkar AMJ, Young-Koo L. Service level semantic interoperability. In: 10th IEEE/IPSJ International Symposium on Applications and the Internet (SAINT'10); 19–23 July 2010; Seoul, Korea. New York, NY, USA: IEEE. pp. 387–390.

[15] Wang TW, Murphy KE. Semantic heterogeneity in multidatabase systems: a review and a proposed meta-data structure. J Database Manage 2004; 15: 71–87.

[16] Naiman CF, Ouksel AM. A classification of semantic conflicts in heterogeneous database systems. Journal of Organizational Computing 1995; 5: 167–167.

[17] Moise G, Netedu L. Ontologies for interoperability in the eLearning systems. Petroleum-Gas University of Ploiesti Bulletin Mathematics - Information - Physics Series 2009; 81: 75–88.

[18] Kashyap V, Sheth A. Semantic and schematic similarities between database objects: a context-based approach. VLDB J 1996; 5: 276–304.

[19] Ceruti MG, Kamel MN. Preprocessing and integration of data from multiple sources for knowledge discovery. Int J Artif Intell T 1999; 8: 157–177.

[20] Park J, Ram S. Information systems interoperability: What lies beneath? ACM T Inform Syst 2004; 22: 595–632.

[21] Peristeras V, Loutas N, Goudos SK, Tarabanis K. A conceptual analysis of semantic conflicts in pan-European e-government services. J Inf Sci 2008; 34: 877–891.

[22] Shanmugasundaram J, Shekita E, Barr R, Carey M, Lindsay B, Pirahesh H, Reinwald B. Efficiently publishing relational data as xml documents. VLDB J 2001; 10: 133–154.

[23] Kim H, Park S. Semantic integration of heterogeneous XML data sources. In: Bellahsène Z, Patel D, Rolland C, editors. Object-Oriented Information Systems. Berlin, Germany: Springer-Verlag, 2002. pp. 771–775.

[24] Näppilä T, Niemi T. An approach for developing a schemaless XML dataspace profiling system. J Inf Sci 2012; 38: 234–257.

[25] Pluempitiwiriyawej C, Hammer J. A Classification Scheme for Semantic and Schematic Heterogeneities in XML Data Sources. Technical Report TR00-004. Gainesville, FL, USA: University of Florida, 2000.

[26] Lee KH, Kim MH, Lee KC, Kim BS, Lee MY. Conflict classification and resolution in heterogeneous information integration based on XML schema. In: The 2002 IEEE Region 10 Technical Conference on Computers, Communications, Control and Power Engineering; 28–31 October 2002; Beijing, China. New York, NY, USA: IEEE. pp. 93–96.

[27] Aragão V, Fernandes A. Conflict resolution in Web service federations Web services. In: Jeckle M, Zhang LJ, editors. ICWS-Europe 2003. Berlin, Germany: Springer-Verlag, 2003. pp. 109–122.

[28] Kim W, Choi I, Gala S, Scheevel M. On resolving schematic heterogeneity in multidatabase systems. Distrib Parallel Dat 1993; 1: 251–279.

[29] Kim W, Seo J. Classifying schematic and data heterogeneity in multidatabase systems. Computer 1991; 24: 12–18.

[30] Sudha R, Jinsoo P. Semantic conflict resolution ontology (SCROL): an ontology for detecting and resolving data and schema-level semantic conflicts. IEEE T Knowl Data En 2004; 16: 189–202.

[31] Arch-Int N, Sophatsathit P. A semantic information gathering approach for heterogeneous information sources on www. J Inf Sci 2003; 29: 357–374.

[32] Halevy A. Why your data won't mix. Queue 2005; 3: 50–58.

[33] Mrissa M, Ghedira C, Benslimane D, Maamar Z, Rosenberg F, Dustdar S. A context-based mediation approach to compose semantic Web services. ACM T Internet Techn 2007; 8: 4.

[34] Cherfi S, Akoka J, Comyn-Wattiau I. Conceptual modeling quality - from EER to UML schemas evaluation. Lect Notes Comp Sci 2503: 414–428.

[35] Moraes MM, Salgado AC. Minimality quality criterion evaluation for integrated schemas. Journal of Information Assurance and Security 2007; 2: 275–287.

[36] Fettke P, Loos P. Classification of reference models: a methodology and its application. Lect Notes Bus Inf 2003; 1: 35–53.

[37] Alamgir M, Mohayidin M. Data mediation to message level conflict in heterogeneous Web services. In: International Conference on IT to Celebrate S. Charmonman's 72nd Birthday; 30 March 2009; Bangkok, Thailand. pp. 41.1–41.7.