Redundancy Detection in Service-Oriented Systems

Peep Küngas
Institute of Computer Science
University of Tartu
50409 Liivi 2
Tartu, Estonia
peep.kungas@ut.ee

Marlon Dumas
Institute of Computer Science
University of Tartu
50409 Liivi 2
Tartu, Estonia
marlon.dumas@ut.ee

ABSTRACT

This paper addresses the problem of identifying redundant data in large-scale service-oriented information systems. Specifically, the paper puts forward an automated method to pinpoint potentially redundant data attributes from a given collection of semantically-annotated Web service interfaces. The key idea is to construct a service network to represent all input and output dependencies between data attributes and operations captured in the service interfaces, and to apply centrality measures from network theory in order to quantify the degree to which an attribute belongs to a given subsystem. The proposed method was tested on a federated governmental information system consisting of 58 independently-maintained information systems providing altogether about 1000 service operations described in WSDL. The accuracy of the method is evaluated in terms of precision and recall.

Categories and Subject Descriptors

D.2.8 [**Software Engineering**]: Metrics; D.2.12 [**Software Engineering**]: Interoperability; H.2.5 [**Database Management**]: Heterogeneous Databases

General Terms

Algorithms, Design, Experimentation, Measurement

Keywords

Data model redundancy, metrics, Web services, federated information systems, semantic Web services

1. INTRODUCTION

A major issue in large-scale information systems management is that of avoiding data redundancy, that is, ensuring that each fact is stored in a single location [12]. Data redundancy does not originate exclusively from duplicated records within a database, but perhaps more frequently, from a common practice to store partially overlapping entries in multiple databases or information (sub-)systems. For instance, it often happens that supplier contact addresses are stored in the procurement, billing, logistics and technical support subsystems, as opposed to storing this address at one subsystem and having the other subsystems retrieve it from this primary location. The reasons for such redundancy may range from performance, reliability or security concerns, to miscommunications between system architects, lack of documentation of existing systems, or lack of cooperation between independent business units. In some cases, data redundancy is deliberate and controlled, while in others it is highly problematic and may lead to inconsistency and poor data quality.

The practical relevance of data redundancy management has been highlighted in several previous works. Moody & Shanks [7] report on a technical review of a repository of data models of a large information system. This technical review surfaced a high degree of overlap between different application data models. Closer inspection showed that different project teams had independently decided to represent the same data in different ways, resulting in data redundancy and duplicated development effort. In a similar vein, Ventrone & Heiler [10] point to several cases where data model overlap in large federated information systems was up to 80%.

In this paper, we propose and evaluate a method for detecting potentially redundant attributes in a service-oriented information system by exploiting the metadata stored in Web service interfaces. The proposed method relies on a representation of a service-oriented information system as a network structure in which the nodes denote either data attributes (XML attributes or leaf XML elements such as "supplier business name" or "supplier address") or operations that take certain attributes as input and produce other attributes as output. Centrality measures[1] are then applied to analyze the resulting service network in order to detect sources of potential redundancy. The result is a set of data attributes that appear in multiple information systems, a diagnostic of which of these information systems is the primary location of the attribute, and a diagnostic of whether or not the attribute in question is a key (primary or foreign key). With this information, we diagnose an attribute as potentially redundant if it occurs in multiple information systems and it is not a key. In this case, we are able to pinpoint in which information system the attribute should be stored (the primary location) and in which information systems the attribute is probably redundant.

The proposed method does not intend to provide fully reliable diagnostics, nor is it able to assert if the redundancy it detects has been introduced on purpose or whether it is desirable or not. In this respect, what we provide are heuristics

WWW 2010, April 26–30, 2010, Raleigh, North Carolina, USA.
ACM 978-1-60558-799-8/10/04.

[1] In network theory, a centrality measure is a measure of the relative importance of a node within a graph.

for identifying potential sources of redundancy as opposed to exact methods. On the other hand, the proposed method is able to detect potential redundancy despite heterogeneous naming conventions (synonyms). To this end, the method is not directly applied on the raw WSDL interfaces, but on WSDL interfaces that have been semantically annotated using the method outlined in [4].

We have validated our proposal on a service-oriented information system consisting of 58 independently-maintained information systems providing altogether about 1000 data services described by means of WSDL. The results obtained by applying the method were compared to the results obtained from a manual inspection of the service interfaces. The results show that the proposed heuristics achieve a high precision and recall.

The rest of the paper is structured as follows. In Section 2 we define basic concepts and we frame the research problem. Section 3 describes the proposed method while Section 4 presents its empirical evaluation. Section 5 reviews related work and Section 6 concludes the paper.

2. PRELIMINARIES

This section introduces basic definitions used in the rest of the paper, formulates the research questions and introduces a running example.

2.1 Basic Definitions

The input of the redundancy detection method proposed in this paper is a collection of semantically annotated service interfaces. Specifically, we assume that we are given a collection of Web service interfaces described using WSDL and XML Schema, and a collection of semantic annotations on these interfaces. Semantic annotations are encoded as SA-WSDL *model references*. A model reference is an URI that refers to a concept in a semantic model. For example, a model reference may refer to a class or a property in an OWL ontology, but equally well it may refer to a class or attribute in an UML class diagram. In this paper, we do not deal with the issue of obtaining the semantic annotations. For the purpose of validating the redundancy detection method, we relied on a method for semi-automated annotation of Web services presented in our previous work [4], but other annotation methods could be employed instead.

If multiple elements in an XML Schema are annotated with the same model references, these elements are deemed to encode the same datum. For example, if two XML Schema elements "client_address" and "customer_address" refer to the same class or property in an OWL ontology, they are considered to represent the same datum.[2]

The purpose of the method proposed in this paper is to identify *redundant entity attributes*. By *entity attribute* we mean an atomic unit of information about an entity, like for example the address of a supplier or the salary of an employee. In the context of a service-oriented information system, an entity attribute corresponds to an XML element or an XML attribute that appears in the schema of one of the messages produced or consumed by a Web service. We abstract away from the choice of granularity of an attribute. For example, one could either take "supplier address" to be an entity attribute, or "supplier address's street name" to be an entity attribute. The lower the granularity, the finer-grained will be the detection of redundant entity attributes, but having too low granularity may lead to large numbers of entity attributes being reported as redundant.

We define a (service-oriented) information system as a collection of service operations. As an alternative, we could have defined an information system as a set of services, each one providing a set of service operations, but the intermediate level of grouping (the "service") turns out not to be needed in our proposal. A service operation takes as input a set of entity attributes and produces as output another set of entity attributes. We write $\mathit{input}(so)$ and $\mathit{output}(so)$ to denote the set of inputs and the set of outputs of service operation so.

For a given information system IS, we define the set of attributes of IS as: $\mathit{atts}(IS) = \{d \in \mathcal{A} \mid \exists so \in IS \; d \in \mathit{input}(so) \cup \mathit{output}(so)\}$, where $\mathcal{A}$ is the set of all possible attributes. In other words, the set of attributes of an information system IS is composed of all attributes that appear at least once as input or output of an operation in IS.

A federated (service-oriented) information system is a set of information systems whose schema are semantically annotated using a common vocabulary (either in OWL, UML or any other modeling language) or a reference system. Given a federated information system FIS, the set of attributes of a federated information system FIS is the union of the set of attributes of its contained information systems, i.e. $\mathit{atts}(FIS) = \cup_{IS \in FIS}\, \mathit{atts}(IS)$.

An attribute d may appear in multiple information systems within a federated information system. For a given FIS and a given attribute d, we define $\mathit{occurs}(FIS, d)$ as the number of information systems in which d appears, i.e. $\mathit{occurs}(FIS, d) = |\{IS \in FIS \mid d \in \mathit{atts}(IS)\}|$.

Ideally, each entity attribute is maintained in one information system and retrieved from other information systems if and when required. The information system in which an informed system architect would most likely place an attribute is called the *primary location* of the attribute. In some cases, replicas of the attribute exist in other information systems (and these replicas may or may not be maintained synchronized). Information systems where replicas of an attribute exist are called secondary locations of the attribute in question. The concept of primary location is purposefully left subjective since it is largely application-dependent. For example, an attribute *businessAddress* might appear in two information systems: the *Business Registry* and the *Tax and Customs Information System*. Intuitively, this attribute belongs primarily in the business registry. We know this because we have some understanding of the functional scope of these two information systems. In some cases, the primary location of an attribute might be less clear-cut. For example, one might argue whether the primary location of an attribute *annualTurnover* should be the *Business Registry* or the *Tax and Customs Information System*. We adopt the view that analysts and system architects, based on their knowledge of the application domain, are sole judges of the primary location of an attribute. Later in the paper we will define a metric that can serve as an indicator of the primary location of an attribute. In other words, we try to

[2] Other notions of semantic equivalence between elements could be employed. For example, we could consider that two elements are equivalent if these elements are annotated with concepts that subsume one another according to a given ontology. For practical purposes, the notion of equivalence used to compare model entities is orthogonal to the techniques proposed in this paper.

approximate a subjective notion by means of a metric. The situation is akin to the concept of *relevance* in the context of document retrieval. A search engine returns a list of ranked documents based on the degree of match between a query and the documents indexed by the search engine. This degree of match is intended to reflect a subjective notion of *relevance* that only users of the search engine can judge.

Some attributes are used to link entities across multiple information systems. For example, a customer identifier can be used in one information system in order to refer to a customer entity in another information system. Such an attribute is called a *reference attribute*. The concept of reference attribute is akin to the concept of "key" in the database world. A reference attribute is a "(primary) key" from the perspective of the information system that is the primary location of the attribute, and a "foreign key" from the perspective of other information systems.

In the context of databases, keys are determined at design-time based on functional dependencies. This approach does not scale in the context of large-scale service-oriented systems, and in particular in the context of systems with large numbers of legacy services. In this context, it is impractical for analysts to define all possible functional dependencies. Accordingly, later in this paper we will define a metric that can serve as indicator of whether or not a given attribute is a reference attribute, without assuming that an analyst has identified all possible functional dependencies.

The key intuition of our redundancy detection method is the following: If an attribute appears in multiple information systems and it is not a reference attribute, then this attribute is redundant in some information systems. Reference attributes link different entities together so it is normal that they appear in multiple information systems. Since the definition of redundancy is based on two subjective definitions, it is itself subjective. It is also up the analysts and architects of a system to judge whether a given occurrence of an attribute in multiple information systems constitutes a redundancy or not. Our redundancy detection criterion is meant to approximate this subjective judgment.

In order to define the metric used for redundancy detection, we start by abstracting a federated information system as a network. This network is constructed by introducing an arc between each service operation and its inputs and outputs. An arc exists from an attribute to an operation if the attribute appears in the inputs of the operation, and an arc exists from an operation to an attribute if the attribute in question appears in the outputs of the operation. It is important to note that in the context of service networks, if we talk about inputs and outputs we mean conceptual representations of inputs and outputs. Formally:

Definition 1 (Service network). *A service network is a graph $\{E, N\}$, where E and N represent respectively a set of edges and a set of nodes in the graph. Set N consists of service operation nodes N_s and entity attribute nodes N_d. Set E is defined as:*

$$E = \bigcup_{IS \in FIS} \big(\{(d, so) \mid so \in IS \wedge d \in inputs(so)\} \cup \{(so, d) \mid so \in IS \wedge d \in outputs(so)\} \big)$$

In the rest of the paper, we use the following definitions.

Definition 2 (Degree). *The degree $deg(d)$ of an attribute node d in a service network $\mathcal{N} = (N, E)$ is the number of edges incident to that node (counting both incoming and outgoing edges), i.e. $deg(d) = |\{so \mid so \in N \wedge [(d, so) \in E \vee (so, d) \in E]\}|$. Meanwhile, the* degree of an attribute node in an information system *IS is the number of edges incident to d whose source or target is an operation in IS, i.e. $deg(d, IS) = |\{so \mid so \in IS \wedge [(d, so) \in E \vee (so, d) \in E]\}|$.*

Definition 3 (Indegree). *The indegree $deg^-(d)$ of an attribute node is the number of edges targeting d, i.e. $deg^-(d) = |\{so \mid so \in N \wedge (so, d) \in E\}|$. Meanwhile, the* indegree of an attribute node in an information system *IS is the number of edges targeting d whose source is an operation in IS, i.e. $deg^-(d, IS) = |\{so \mid so \in IS \wedge (so, d) \in E\}|$.*

Definition 4 (Outdegree). *The outdegree $deg^+(d)$ of an attribute node is the number of edges emanating from d, i.e. $deg^+(d) = |\{so \mid so \in N \wedge (d, so) \in E\}|$. Meanwhile, the* outdegree of an attribute node in an information system *IS is the number of edges emanating from d whose target is an operation in IS, i.e. $deg^+(d, IS) = |\{so \mid so \in IS \wedge (d, so) \in E\}|$.*

2.2 Problem Statement & Example

In order to detect potential redundancy, we are looking for answers to the following questions:

1. Given a federated information system FIS, can we find a classification function $C(IS, d)$ that takes as input an information system $IS \in FIS$ and an attribute $d \in atts(FIS)$, and returns T (true) if IS is the primary location of d, and F (false) otherwise?

2. Given a federated information system FIS, can we find a classification function $C'(d)$ that takes as input an attribute $d \in atts(FIS)$ and returns T if d is a reference attribute in FIS, and F otherwise?

Let us consider a federated information system comprising subsystems for Customer Relationship Management (CRM), invoicing, logistics and after-sales service. All these subsystems refer to customers and need to deal with contact addresses (billing address, delivery address, legal address). One would expect that the CRM maintains the customer address(es), therefore $C(CRM, customerAddress) = T$. The invoicing, logistics and after-sales services need to retrieve customer data (including addresses) from the CRM system. To do so, they need to refer to a particular customer through an identifier. Accordingly, one would expect that there would be an attribute *custId* so that $C'(custId) = T$. Note that these attributes *customerAddress* and *custId* are conceptual entities rather than concrete XML elements. It is possible that different information systems will use different concrete representations of this attribute. For example, in the invoicing system this attribute might be represented as an element called *customerIdentifier* while in the logistics subsystem it might be represented as an element *addresseeIdentifier*. The semantic annotations attached to the XML schemas will allow us to relate these two elements back to the same concept.

As a second example, let us consider three information systems of the Estonian federated governmental information system: the Tax and Customs Board service, the Business Registry service and the Register of Economic Activities. Figs. 1, 2, 3 and 4 show message schema fragments for each of these services. For the sake of understandability, element names have been translated to English and some irrelevant fragments have been deleted. The Business Registry service provides information about companies – their registration numbers (details_company/businessregistrycode in Fig. 3 and detailedQuery/businessregistrycode in Fig. 2)[3], names (details_business_name/content in Fig. 3 and detailedQuery/businessname in Fig. 2), and contact details (details_contact_medium/content in Fig. 3 and detailedQuery/address in Fig. 2). At the same time the Register of Economic Activities provides also business registration number (generalinfoBaseType/code in Fig. 4), name (generalinfoBaseType/name in Fig. 4) and contact details (generalinfoBaseType/{tel,fax,email,web} in Fig. 4). Furthermore, the Tax and Customs Board service also provides business registration numbers (employerTaxQueryAnswer/businessregistrycode in Fig. 1) and business names (employerTaxQueryAnswer/employername in Fig. 1).

In this example, it is clear that the business registry number is a reference attribute. Business names and business contact details fit most naturally in the Business Registry (i.e. this is their primary location). Therefore, elements referring to company names and contact details at the Register of Economic Activities and at the Tax and Customs Board service are redundant. However, the business registration number, which is stored in all three information systems is not redundant since it is required in order to link company data stored across these information systems.

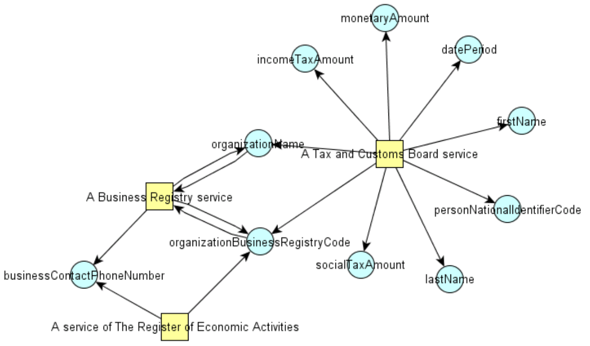

monetaryAmount
incomeTaxAmount
datePeriod
firstName
organizationName
A Tax and Customs Board service
A Business Registry service
personNationalIdentifierCode
organizationBusinessRegistryCode
socialTaxAmount
lastName
businessContactPhoneNumber
A service of The Register of Economic Activities

Figure 5: Service network constructed from service interface fragments in Fig. 1, Fig. 2, Fig. 3 and Fig. 4.

Based on SA-WSDL references in Fig. 1, Fig. 2, Fig. 3, Fig. 4 we can construct a service network as seen in Fig. 5. Rectangular nodes in the figure represent services whose interface fragments were annotated with SA-WSDL references, while ellipsoidal nodes represent data attributes, which were annotated.

3. REDUNDANCY DETECTION METHOD

In order to detect redundancy we start by constructing clusters representing entity attributes in different information systems (IS). Each cluster represents entity attributes within an information system, whereas entity attributes are collected from service descriptions of particular IS according to definitions in Section 2. An example of clusters and their overlappings is visualized in Fig. 6. The figure uses cluster map technology to represent overlappings of entity attribute clusters. The highlighted central area represents entity attributes that are potentially redundant, since they appear in multiple clusters at the same time.

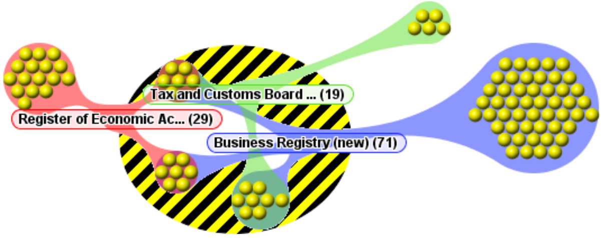

Tax and Customs Board ... (19)
Register of Economic Ac... (29)
Business Registry (new) (71)

Figure 6: A cluster map of data entities in different information systems.

After clusters have been formed, we analyze in how many clusters an entity attribute occurs in. If an entity attribute occurs only in a single cluster, it is clearly not redundant. For instance, in Fig. 1 *socialtax* (paid social taxes) and *incometax* (paid income tax) are in this respect not redundant.

In the case of attributes occurring in multiple information systems, we start by determining the primary location. Primary location of an entity attribute is determined by measuring its degree in the constructed service network. An IS for which the entity attribute degree is highest, is most probably the attribute's primary location. The justification is based on the tendency that the majority of data processing services are normally provided at the same information system where the data originates from. Accordingly, the primary location classifier $C(IS, d)$ for information system IS and entity attribute d is defined as follows:

$$C(IS,d) = \begin{cases} T, & \textbf{if } S_r(IS,d) - S_m(d) \geq \rho \\ F, & \textbf{otherwise} \end{cases},$$

where relative score $S_r(IS, d) = deg(d, IS)/deg(d)$, average score $S_m(d) = 1/occurs(FIS, d)$ and $\rho \in [0, 1]$ is a threshold that can be used to tune the classifier.

We can interpret $deg(d, IS)/deg(d)$ as a metric indicating the "relative attachment" of attribute d to IS. A relative attachment of 1 means that the attribute exclusively belongs in that information system, an attachment of 0 means that the attribute does not appear at all in IS. The higher the attachment of an attribute to an IS, the higher the chances that this is the primary location of the attribute. If an attribute appears in multiple information systems (say n), and the attribute appears an equal amount of times in each system, then its relative attachment to each system is $1/n$. Thus a relative attachment above $1/n$ shows that an attribute is proportionally more strongly than average linked to an IS. In this light $S_m(d)$ can be interpreted as the "average attachment" of d to the information systems in which it is used.

When an attribute appears more times in one information system than in others, then the difference between relative attachment $S_r(IS, d)$ and $1/n$ becomes higher. For example, if an attribute d appears in two information systems X and

[3] We use XPath-style references to refer to specific fragments of schema.

```
<xsd:complexType name="employerTaxQueryAnswer">
  <xsd:all>
    <xsd:element name="nationalidcode" nillable="true" type="xsd:string"
     sawsdl:modelReference="http://onto.soatrader.com/ontology/NationalOntology.owl#personNationalIdentifierCode"
/>
    <xsd:element name="personname" nillable="true" type="xsd:string"
     sawsdl:modelReference="http://onto.soatrader.com/ontology/NationalOntology.owl#firstName"
     sawsdl:modelReference="http://onto.soatrader.com/ontology/NationalOntology.owl#lastName"/>
    <xsd:element name="period" nillable="true" type="xsd:string"
     sawsdl:modelReference="http://onto.soatrader.com/ontology/TimeOntology.owl#datePeriod"/>
    <xsd:element name="businessregistrycode" nillable="true" type="xsd:string"
     sawsdl:modelReference="http://onto.soatrader.com/ontology/NationalOntology.owl#organizationBusinessRegistryCode"/>
    <xsd:element name="employername" nillable="true" type="xsd:string"
     sawsdl:modelReference="http://onto.soatrader.com/ontology/NationalOntology.owl#organizationName"/>
    <xsd:element name="sum" nillable="true" type="xsd:decimal"
     sawsdl:modelReference="http://onto.soatrader.com/ontology/FinanceOntology.owl#monetaryAmount"/>
    <xsd:element name="socialtax" nillable="true" type="xsd:decimal"
     sawsdl:modelReference="http://onto.soatrader.com/ontology/FinanceOntology.owl#socialTaxAmount"/>
    <xsd:element name="incometax" nillable="true" type="xsd:decimal"
     sawsdl:modelReference="http://onto.soatrader.com/ontology/FinanceOntology.owl#incomeTaxAmount"/>
  </xsd:all>
</xsd:complexType>
```

Figure 1: A Tax and Customs Board service—output message content fragment.

```
<xsd:complexType name="detailedQuery">
  <xsd:sequence>
    <xsd:element name="businessname" type="xsd:string" minOccurs="0"
     sawsdl:modelReference="http://onto.soatrader.com/ontology/NationalOntology.owl#organizationName"/>
    <xsd:element name="businessregistrycode" type="xsd:int" minOccurs="0"
     sawsdl:modelReference="http://onto.soatrader.com/ontology/NationalOntology.owl#organizationBusinessRegistryCode"/>
    <xsd:element name="address" type="xsd:string" minOccurs="0"/>
    <xsd:element name="relatedpersonfirstname" type="xsd:string" minOccurs="0"/>
    <xsd:element name="relatedpersonlastname" type="xsd:string" minOccurs="0"/>
    <xsd:element name="relatedpersonbirthdate" type="xsd:date" minOccurs="0"/>
    <xsd:element name="relatedpersonnationalidcode" type="xsd:string" minOccurs="0"/>
     ...
  </xsd:sequence>
</xsd:complexType>
```

Figure 2: A Business Registry service—input message content fragment.

```
<xsd:complexType name="details_company">
  <xsd:sequence>
     //business registry number
    <xsd:element name="businessregistrycode" type="xsd:int"
     sawsdl:modelReference="http://onto.soatrader.com/ontology/NationalOntology.owl#organizationBusinessRegistryCode"/>
    <xsd:element name="generaldata" type="typens:details_general" minOccurs="0"/>
    <xsd:element name="personaldata" type="typens:details_personal" minOccurs="0"/>
  </xsd:sequence>
</xsd:complexType>
...
<xsd:complexType name="details_contact_medium">
  <xsd:sequence>
    <xsd:element name="typecode" type="xsd:string" minOccurs="0" /> //phone, fax, e-mail, ...
    <xsd:element name="typename" type="xsd:string" minOccurs="0"/>
     //contact medium value
    <xsd:element name="content" type="xsd:string" minOccurs="0"
     sawsdl:modelReference="http://onto.soatrader.com/ontology/NationalOntology.owl#businessContactPhoneNumber"/>
    <xsd:element name="enddate" type="xsd:date" minOccurs="0"/>
  </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="details_business_name">
  <xsd:sequence>
    ...
    <xsd:element name="entryno" type="xsd:int" minOccurs="0"/>
     // business name
    <xsd:element name="content" type="xsd:string" minOccurs="0"
     sawsdl:modelReference="http://onto.soatrader.com/ontology/NationalOntology.owl#organizationName"/>
    <xsd:element name="startdate" type="xsd:date" minOccurs="0"/>
    <xsd:element name="enddate" type="xsd:date" minOccurs="0"/>
  </xsd:sequence>
</xsd:complexType>
```

Figure 3: A Business Registry service—output message content fragment.

```
<complexType name="generalinfoBaseType">
  <sequence>
    <element name="name" type="string"/> // business name
// business registry code
    <element name="code" type="string" minOccurs="0"
     sawsdl:modelReference="http://onto.soatrader.com/ontology/NationalOntology.owl#organizationBusinessRegistryCode"/>
// contact phone
    <element name="tel" type="string" minOccurs="0"
     sawsdl:modelReference="http://onto.soatrader.com/ontology/NationalOntology.owl#businessContactPhoneNumber"/>
    <element name="fax" type="string" minOccurs="0"/> //contact fax
    <element name="email" type="string" minOccurs="0"/> // contact E-mail
    <element name="web" type="string" minOccurs="0"/> // contact WWW
  </sequence>
</complexType>
```

Figure 4: A service of The Register of Economic Activities—output message content fragment.

Y and it is used 10 times in X and 5 times in Y, then the attachment of this attribute to X will be $10/15 = 0.66$ and $S_r(X, d) - 1/n = 0.16$. We can then say with some confidence that X is likely to be the primary location of d. Note that for a given attribute d, classifier $C(IS, d)$ might return true for multiple information systems. This may happen for example when the relative attachment of an attribute d is the same in all information systems in which this attribute appears – i.e. $S_r(IS, d) = S_m(d)$ for all IS such that $d \in atts(IS)$. In this case, the classifier is unable to assign attribute d to a single primary location.

To illustrate the primary location classifier, let us consider a selection of entity attributes (business registry code, business name, paid social tax, paid income tax, business contact phone number) from information system descriptions presented in Section 2.2. In Table 1 we summarize degrees of these entity attributes in considered information systems (Tax and Customs Board services (TCB), The Register of Economic Activities (REA), Business Registry (BR)). According to the classifier, the primary location of business registry code and business name is the Business Registry, while the primary location of "paid social tax" and "paid income tax" is the Tax and Customs Board services and the primary location of "business contact phone number" is the Register of Economic Activities.

If an entity attribute appears in multiple information systems, it may be redundant, but only, if it is not a reference attribute. Symmetrically, an entity attribute occurring in more than one information system, is a potential reference attribute. In order to detect such reference attributes we use the following classifier:

$$C'(d) = \begin{cases} T, & \textbf{if } \exists IS : C(IS, d) \wedge \frac{deg^+(d,IS)}{deg^-(d,IS)} - \frac{deg^+(d)}{deg^-(d)} \leq \rho' \\ F, & \textbf{otherwise} \end{cases}$$

where $\rho' \in [-\infty, +\infty]$ is a threshold.

The hypothesis underpinning this definition is that the ratio between the number of times a reference attribute is used as input and the number of times it is produced as output can be used to characterize whether an attribute is a reference attribute. Especially in the attribute's primary location, we would expect that the reference attribute is used many times since such attributes are used to retrieve data about an entity and these data are normally located in the primary location. To illustrate the reference attribute detection classifier, let us elaborate further on the primary location suggestion results in Table 1. In Table 2 we list additional characteristics for entity attributes whose primary location was proposed. According to Table 2 we see that both business registry code and business name serve as reference attributes, which would be used to link company records over multiple informations systems within a federated IS. One may argue that business name is not a reference attribute. If we adopt this view, we have here an example where the findings of the classifier do not always agree with the subjective judgment of informed users.

Given the above two classifiers, we define a third classifier, namely $R(IS, d)$, which determines whether or not an attribute d is redundant in an information system IS:

$$R(IS, d) = \begin{cases} T, & \textbf{if } occurs(d) > 1 \wedge C(IS, d) \wedge C'(d) \\ F, & \textbf{otherwise} \end{cases}$$

In other words, an attribute d is redundant in an information IS if it appears in multiple information systems, IS is not its primary location and d is not a reference attribute.

4. EVALUATION

4.1 Dataset and methodology

We applied the redundancy detection method proposed above to the Estonian governmental Web service repository [3], which contains interfaces of 58 information systems, each one exposed as a Web service described in WSDL. These 58 systems encompass around 1000 Web service operations.

In previous work [4] we introduced a method to semantically annotate WSDL interfaces and we applied it to the above governmental information system. This led us to a collection of SA-WSDL annotations on top of the 60 WSDL files comprising the Web services repository. Altogether, there were 7757 leaf elements in the XML schemas in the repository from which we managed to annotate 5555 leaf elements. The remaining elements were too specialized to be annotated meaningfully, but since they each only occurred in one information system, they do not constitute a source of potential redundancy. The semantic annotations that we constructed refer to classes in an ontology that we built incrementally during the semantic annotation process.

From the semantically annotated Web service interfaces, we constructed a service network consisting of 928 service operation nodes (annotated WSDL operations), 466 entity attribute nodes (forming a unified data model used for covering about 72% of XML Schema leaf node elements across all WSDL interfaces of the federated IS) and 17006 edges.

Table 1: Example of entity attribute primary location detection with $\rho = 0$.

Entity attribute d	**Location** is	$deg(d, is)$	$C(is, d)$
Business registry code	TCB	10	F
Business name	TCB	3	F
Paid social tax amount	TCB	1	T
Paid income tax amount	TCB	6	T
Business registry code	REA	6	F
Business name	REA	7	F
Business contact phone number	REA	5	T
Business registry code	BR	15	T
Business name	BR	13	T
Business contact phone number	BR	1	F

Table 2: Example of reference detection for attributes in Table 1 with $\rho' = 1$.

Entity attribute d	**Location** IS	$deg^-(d, IS)$	$deg^+(d, IS)$	$deg^-(d)$	$deg^+(d)$	**C'(d)**
Paid social tax	TCB	0	1	0	3	F
Paid income tax	TCB	0	6	0	6	F
Business contact phone	REA	0	5	35	73	F
Business registry code	BR	7	8	38	52	T
Business name	BR	6	7	16	33	T

Clusters were built for the 58 information systems forming the federated information system.

For evaluation purposes we use the classical notions of precision and recall defined for statistical classifiers. In a statistical classification task, the *precision* of a classifier for a given class is the number of true positives divided by the sum of true positives and false positives. Meanwhile, the *recall* of a classifier for a given task is defined as the number of true positives divided by the sum of true positives and false negatives.A *precision* score of 1.0 for a class c means that every item labeled by the classifier as belonging to class c does indeed belong to this class, whereas a *recall* of 1.0 means that every item of class c was labeled by the classifier as belonging to c. Finally, the evaluation also relies on the concept of F-score, which is defined as the harmonic mean of the precision and recall.

To evaluate the performance of the classifiers defined in Section 3, we manually inspected each entity attribute and we determined its primary location and whether it is a reference attribute or not. This manual judgement was made by the first author of the paper who is familiar with the overall information system, from involvements in previous projects. Based on these manual judgements, we computed redundancy as defined in Section 2.1 and we compared the resulting judgment to the one obtained with the automated redundancy classifier.

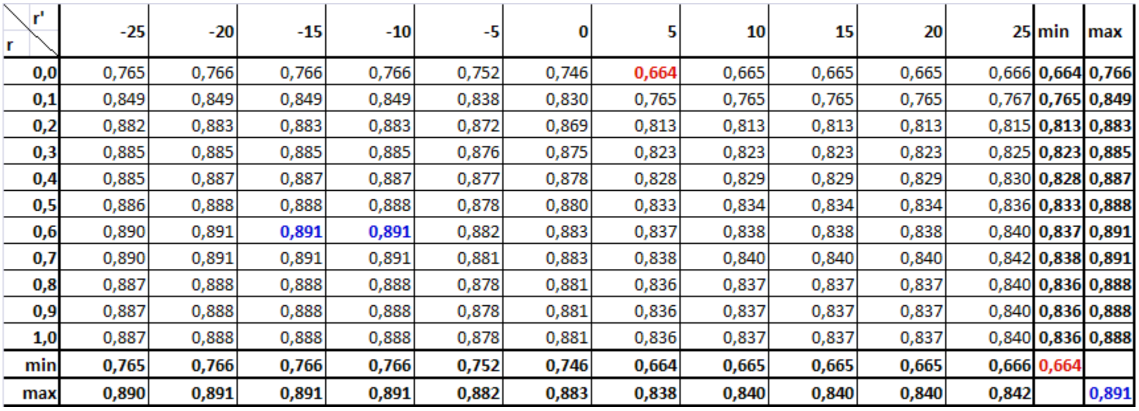
When evaluating the redundancy classifier, we discarded all entity attributes that occurred in a single information system, since these attributes are trivially non-redundant and including them in the evaluation of the redundancy classifier would have led to biased results (i.e. all these attributes would have been correctly classified, in a trivial manner).

The ontology used to annotate the WSDL interfaces had taxonomic relations between classes. However, in order to reduce effects arising from semantic annotations with different granularity (such as "general identifier" vs "person's national identifier code" vs "child's national identifier code") we discarded annotations in the top-level of the taxonomy. In other words, we gave preference to more specific semantic annotations over more general ones. The rationale for this choice is the following: If we compared annotations at a higher level, we would immediately obtain a large number of false positives for the redundancy classifier. For example, every time we find an attribute containing an address we would say that this attribute is redundant. Yet, it is normal that a federated information system contains multiple address types (e.g. personal address versus work address, billing address versus shipping address). Thus, even though all these elements would have been annotated with the concept "address", this annotation was deleted during the preprocessing phase. Without this filtering step, the accuracy results of the statistical classifiers became meaningless.

4.2 Results and discussion

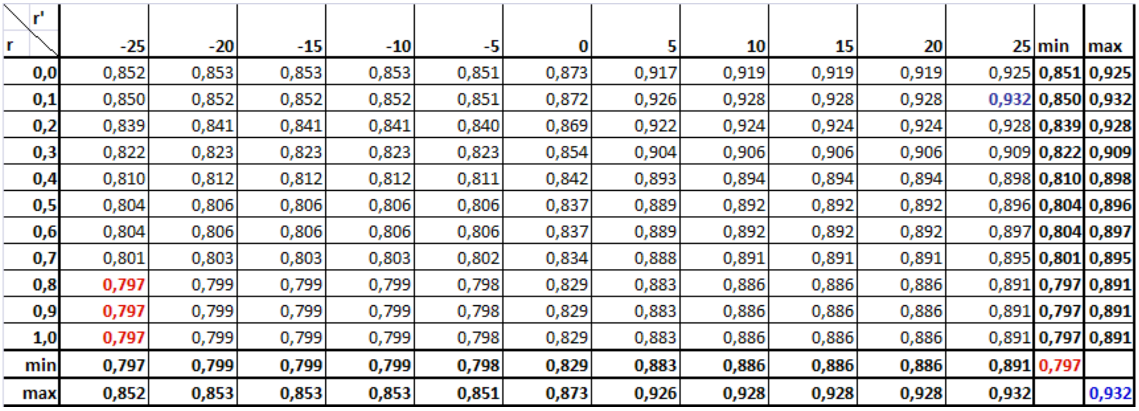
We calculated precision, recall and f-score for every possible setting of parameters ρ and ρ', with ρ ranging from 0 to 1 in steps of 0.1 and ρ' ranging from -25 to 25 in steps of 5. The resulting f-scores for each setting are shown in Table 3, while precision and recall are summarized respectively in Table 4 and in Table 5.

We can note that the F-score is consistently high when $\rho > 0.2$. Parameter ρ' has less influence on the f-score, although there is a trend that the f-score is better for negative values of ρ'. By inspecting the results closer, we noted that the problem when ρ is positive is that the recall drops significantly, meaning that we start getting many false negatives. These false negatives probably stem from the fact that for $\rho' \geq 0$, the method starts misclassifying some attributes as reference attributes and these misclassified attributes are not classified as redundant. It appears that a value of ρ' between -10 and 0 addresses this issue without overly affecting the precision. For values of $\rho' < 10$ we observed that the precision is heavily affected, because the method



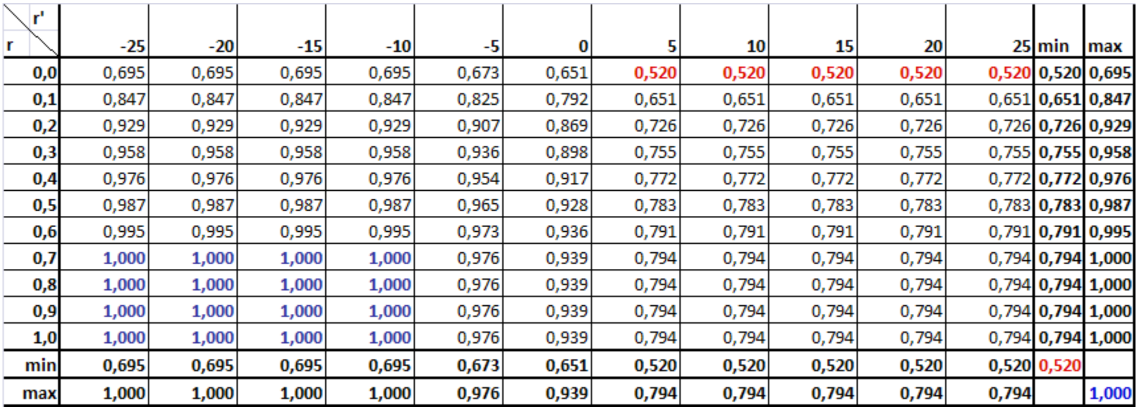
r' r	-25	-20	-15	-10	-5	0	5	10	15	20	25	min	max
0,0	0,765	0,766	0,766	0,766	0,752	0,746	0,664	0,665	0,665	0,665	0,666	0,664	0,766
0,1	0,849	0,849	0,849	0,849	0,838	0,830	0,765	0,765	0,765	0,765	0,767	0,765	0,849
0,2	0,882	0,883	0,883	0,883	0,872	0,869	0,813	0,813	0,813	0,813	0,815	0,813	0,883
0,3	0,885	0,885	0,885	0,885	0,876	0,875	0,823	0,823	0,823	0,823	0,825	0,823	0,885
0,4	0,885	0,887	0,887	0,887	0,877	0,878	0,828	0,829	0,829	0,829	0,830	0,828	0,887
0,5	0,886	0,888	0,888	0,888	0,878	0,880	0,833	0,834	0,834	0,834	0,836	0,833	0,888
0,6	0,890	0,891	0,891	0,891	0,882	0,883	0,837	0,838	0,838	0,838	0,840	0,837	0,891
0,7	0,890	0,891	0,891	0,891	0,881	0,883	0,838	0,840	0,840	0,840	0,842	0,838	0,891
0,8	0,887	0,888	0,888	0,888	0,878	0,881	0,836	0,837	0,837	0,837	0,840	0,836	0,888
0,9	0,887	0,888	0,888	0,888	0,878	0,881	0,836	0,837	0,837	0,837	0,840	0,836	0,888
1,0	0,887	0,888	0,888	0,888	0,878	0,881	0,836	0,837	0,837	0,837	0,840	0,836	0,888
min	0,765	0,766	0,766	0,766	0,752	0,746	0,664	0,665	0,665	0,665	0,666	0,664	
max	0,890	0,891	0,891	0,891	0,882	0,883	0,838	0,840	0,840	0,840	0,842		0,891

Table 3: F-scores for redundancy detection with $\rho = [0.0, 1.0]$ and $\rho' = [-25, 25]$.



r' r	-25	-20	-15	-10	-5	0	5	10	15	20	25	min	max
0,0	0,852	0,853	0,853	0,853	0,851	0,873	0,917	0,919	0,919	0,919	0,925	0,851	0,925
0,1	0,850	0,852	0,852	0,852	0,851	0,872	0,926	0,928	0,928	0,928	0,932	0,850	0,932
0,2	0,839	0,841	0,841	0,841	0,840	0,869	0,922	0,924	0,924	0,924	0,928	0,839	0,928
0,3	0,822	0,823	0,823	0,823	0,823	0,854	0,904	0,906	0,906	0,906	0,909	0,822	0,909
0,4	0,810	0,812	0,812	0,812	0,811	0,842	0,893	0,894	0,894	0,894	0,898	0,810	0,898
0,5	0,804	0,806	0,806	0,806	0,806	0,837	0,889	0,892	0,892	0,892	0,896	0,804	0,896
0,6	0,804	0,806	0,806	0,806	0,806	0,837	0,889	0,892	0,892	0,892	0,897	0,804	0,897
0,7	0,801	0,803	0,803	0,803	0,802	0,834	0,888	0,891	0,891	0,891	0,895	0,801	0,895
0,8	0,797	0,799	0,799	0,799	0,798	0,829	0,883	0,886	0,886	0,886	0,891	0,797	0,891
0,9	0,797	0,799	0,799	0,799	0,798	0,829	0,883	0,886	0,886	0,886	0,891	0,797	0,891
1,0	0,797	0,799	0,799	0,799	0,798	0,829	0,883	0,886	0,886	0,886	0,891	0,797	0,891
min	0,797	0,799	0,799	0,799	0,798	0,829	0,883	0,886	0,886	0,886	0,891	0,797	
max	0,852	0,853	0,853	0,853	0,851	0,873	0,926	0,928	0,928	0,928	0,932		0,932

Table 4: Precision for redundancy detection with $\rho = [0.0, 1.0]$ and $\rho' = [-25, 25]$.



r' r	-25	-20	-15	-10	-5	0	5	10	15	20	25	min	max
0,0	0,695	0,695	0,695	0,695	0,673	0,651	0,520	0,520	0,520	0,520	0,520	0,520	0,695
0,1	0,847	0,847	0,847	0,847	0,825	0,792	0,651	0,651	0,651	0,651	0,651	0,651	0,847
0,2	0,929	0,929	0,929	0,929	0,907	0,869	0,726	0,726	0,726	0,726	0,726	0,726	0,929
0,3	0,958	0,958	0,958	0,958	0,936	0,898	0,755	0,755	0,755	0,755	0,755	0,755	0,958
0,4	0,976	0,976	0,976	0,976	0,954	0,917	0,772	0,772	0,772	0,772	0,772	0,772	0,976
0,5	0,987	0,987	0,987	0,987	0,965	0,928	0,783	0,783	0,783	0,783	0,783	0,783	0,987
0,6	0,995	0,995	0,995	0,995	0,973	0,936	0,791	0,791	0,791	0,791	0,791	0,791	0,995
0,7	1,000	1,000	1,000	1,000	0,976	0,939	0,794	0,794	0,794	0,794	0,794	0,794	1,000
0,8	1,000	1,000	1,000	1,000	0,976	0,939	0,794	0,794	0,794	0,794	0,794	0,794	1,000
0,9	1,000	1,000	1,000	1,000	0,976	0,939	0,794	0,794	0,794	0,794	0,794	0,794	1,000
1,0	1,000	1,000	1,000	1,000	0,976	0,939	0,794	0,794	0,794	0,794	0,794	0,794	1,000
min	0,695	0,695	0,695	0,695	0,673	0,651	0,520	0,520	0,520	0,520	0,520	0,520	
max	1,000	1,000	1,000	1,000	0,976	0,939	0,794	0,794	0,794	0,794	0,794		1,000

Table 5: Recall for redundancy detection with $\rho = [0.0, 1.0]$ and $\rho' = [-25, 25]$.

is unable to properly identify any reference attribute and it reports all reference attributes as redundant. We therefore conclude that good settings can be obtained by simply setting ρ and ρ' to the middle of their ranges, i.e. $\rho = 0.5$ and $\rho' = 0$, although further work on other datasets would be needed to confirm this hypothesis.

The maximum F-score (0.89) was achieved with $\rho = 0.6$ and $\rho' \in (-20, -10)$. More detailed results for $\rho = 0.6$ are plotted in Fig. 7. We can observe from this figure the tradeoff that occurs between precision and recall when ρ' moves from negative to positive territory. Essentially, when $\rho' \in (-20, -10)$, the recall of the classifier is around 99%. In other words, if an attribute could reasonably qualify as redundant, the classifier will find it. At around 80%, the precision is not optimal, but arguably still acceptable. One could argue that higher precision (at close to 100% recall) would be difficult to attain, given the subjectivity underpinning the notion of redundancy.

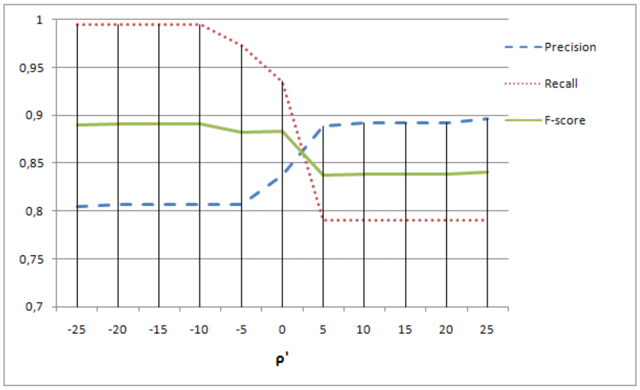


Figure 7: Redundancy detection results for $\rho = 0.6$.

Based on manually classified redundant data items, we analyzed also what percentage of data items occurring at multiple locations are redundant. It turned out that 79% of such data items are redundant, which is consistent with findings of Ventrone & Heiler [10] who point to several cases where data model overlap in large federated information systems was up to 80%.

The following threats to validity apply to our results:

- The evaluation of the classifiers proposed in the paper was made against our own judgment of the primary location of each attribute and its likelihood of it being a reference attribute. Some may argue that these judgments are subjective and possibly biased. To minimize the risk of bias, we made the manual classification of attributes before defining and evaluating the classifiers.

- The redundancy detection technique depends on the quality of the semantic annotations, so the conclusions we made might not be applicable if the quality of the semantic annotations is significantly lower (or higher), or if some semantic annotations are missing.

- There were large amounts of data redundancy in the federated information system considered in this study.

To tackle these issues we plan to engage in another iteration of this evaluation, this time by engaging field experts to get their expert opinion on the results of automated classifiers. Furthermore, we expect semantic annotations of studied information system interfaces of better quality to be available at another iteration. Finally, further evaluation with other federated information systems would help to address the third threat to validity. It is worth noting in this respect that although the level of data redundancy found is high, a large part of this redundancy is likely to be deliberate. Due to privacy concerns and IT governance decisions, information exchange between different information systems in the government sector is sometimes deliberately restricted. For example, the fact that a citizen can give multiple contact details for different engagements with government agencies is considered to be possible in certain scenarios and government agencies are sometimes restricted in their possibilities of exchanging these details.

5. RELATED WORK

Inter-record redundancy has been recognized in the literature for some time [1], and has been addressed in terms of normal forms and normalization by Ling et al [5], who gave one of the first treatments of inter-relational dependencies. However, the focus of this and other works on database normalization is to avoid redudancy within a single database. In contrast, the objective of our work is to avoid redundancy across information systems that, although federated, are independently developed and maintained.

Wadsack et al [11] studied data dependencies among the integrated Web information systems and classified distributed data dependencies. The main driving force behind such activity was a need to understand data dependencies between locally autonomous informations systems created through ad-hoc integration projects. The latter are characterized as projects with no systematic planning and usually produce poor or no documentation at all. Today, industry faces the challenge of maintaining and adapting these systems without complete documentation although many of the systems have become indispensable. The situation is critical in large systems, such as federated governmental systems, which have evolved independently and just recently have started to interact with each-other. This is why automated discovery of data dependencies is so important.

The authors propose three types of inter-schema dependencies such as redundancy, inclusion and constraint dependency. Redundancy dependency characterizes data, which is held and maintained (at least) at two sources. Inclusion dependency means that an (a set of) attribute in one database table holds a part or the same information as an (a set of) attribute of a second database table. Finally, constraint dependency characterizes condition(s) over two or more data dependencies to assign information. Redundancy dependency is then further classified into synonymity, duplication, replication and “real” redundancy. Synonymity occurs when multiple attributes hold the same information but have different names. This can be tackled by using semantic annotations, as we do in our work, rather than working directly on raw schemas or interfaces. Duplication corresponds to the case where an explicit copy of a data entry is made at specific points in time, but the copied data entry is not kept consistent. Finally, replication occurs when an explicit copy of a data entry is made and the copies are kept consistent, i.e., a controlled redundancy. Finally, “real” redundancy occurs when a data entry is stored in multiple locations without any mechanism to keep the copies “in sync”. In this paper, we are interested in “real” redundancy.

Wadsack et al [11] also point out that inclusion dependencies, known from (single) relational databases [2], form the basis for interpreting the semantics of foreign keys. Each foreign key implies an inclusion dependency where the included attribute (set of attributes) is a key of the corresponding data (table). Since the authors analyze SQL queries to identify primary key/foreign key relations we cannot use this approach in this paper since we assume availability of service interface descriptions only.

According to definition by Witt and Simsion [12]: “model contains no redundancy means that each fact is represented in only one place” we are trying to detect “external redundancy”. Moody [6] identifies external redundancy as existence of data model entities, which are duplicated over a set of models or systems. This form of redundancy is a serious problem in most organizations—empirical studies show that there are an average of ten physical copies of each primary data item in medium to large organizations [8]. This finding applies to our case as well where for instance partially overlapping information regarding companies is stored in majority of around 60 informations systems we studied.

In the context of federated databases, Sheth and Larson [9] have analyzed different forms of redundancy between database schemas defined at different levels of a federated database architecture. The authors note that redundancies can arise, for example, between “federated schemas” and “external schemas” defined for different federation users. This type of redundancy however occurs at the level of schemas, because these schemas represent different (and possibly overlapping) views on the underlying databases. This schema redundancy does not entail a redundancy in the underlying databases, so it is not a case of redundancy dependency as defined in this paper.

6. CONCLUSION

We have proposed metrics for enabling discovery of data redundancy from WSDL descriptions of information system interfaces and evaluated them on a federated governmental information system. The results of the evaluation are encouraging, since consistently high precision and recall were achieved, both for identifying redundant attributes and for identifying the primary location of redundant attributes. Moreover, the evaluation unveiled that, although individual information systems might not have a lot of data redundancy, there can be considerable redundancy in federated information systems.

The proposed metrics included two parameters (ρ and ρ'), which have to be fine-tuned for different configurations. Our future research will aim at validating the proposed metrics with further datasets in order to determine to what extent the optimal settings of these parameters are domain-dependent. Further experimentation is also required in order to determine what percentage of the redundancy identified by the proposed metrics is deliberate, and therefore how useful are the proposed metrics in identifying unintended redundancy. Finally, more experiments could be conducted using different notions of semantic equivalence between schema elements.

Acknowledgement

This research is funded by Swedbank and the European Regional Development Fund through the Estonian Centre of Excellence in Computer Science.

7. REFERENCES

[1] E. F. Codd. A relational model of data for large shared data banks. *Commun. ACM*, 13(6):377–387, 1970.

[2] Christian Fahrner and Gottfried Vossen. Transforming relational database schemas into object-oriented schemas according to odmg-93. In *DOOD ’95: Proceedings of the Fourth International Conference on Deductive and Object-Oriented Databases*, pages 429–446, London, UK, 1995. Springer-Verlag.

[3] A. Kalja, A. Reitsakas, and N. Saard. eGovernment in Estonia: Best practices. In *Technology Management: A Unifying Discipline for Melting the Boundaries*, pages 500–506. IEEE Press, 31 July–4 August 2005.

[4] P. Küngas and M. Dumas. Cost-effective semantic annotation of XML schemas and web service interfaces. In *Proceedings of the IEEE International Conference on Services Computing, SCC 2009, Bangalore, India, September 21–25, 2009*, pages 372–379. IEEE Computer Society Press, 2009.

[5] T.-W. Ling, F. W. Tompa, and T. Kameda. An improved third normal form for relational databases. *ACM Transactions on Database Systems*, 6(2):329–346, 1981.

[6] D. L. Moody. Metrics for evaluating the quality of entity relationship models. In *Proceedings of the 17th International Conference on Conceptual Modeling*, volume 1507 of *Lecture Notes in Computer Science*, pages 211–225, London, UK, 1998. Springer-Verlag.

[7] D. L. Moody and G. G. Shanks. Improving the quality of data models: empirical validation of a quality management framework. *Information Systems*, 28(6):619–650, 2003.

[8] C. O’Brien and S. O’Brien. Mining your legacy systems: A data-based approach. In *Asia Pacific DB2 User Group Conference, Melbourne, Australia, November 21-23, 1994*, 1994.

[9] A. P. Sheth and J. A. Larson. Federated database systems for managing distributed, heterogeneous, and autonomous databases. *ACM Computing Surveys* 22(3):183–236, 1990.

[10] V. Ventrone and S. Heiler. Some advice for dealing with semantic heterogeneity in federated database systems. In *Proceedings of the Database Colloquium, San Diego, August 1994, Armed Forces Communications and Electronics Assc. (AFCEA)*, 1994.

[11] J. P. Wadsack, J. Niere, H. Giese, and J. H. Jahnke. Towards data dependency detection in web information systems. In *In Proceedings of the Database Maintenance and Reengineering Workshop (DBMR’2002), Montreal, Canada.*, 2002.

[12] G. C. Witt and G. C. Simsion. *Data Modeling Essentials: Analysis, Design, and Innovation.* The Coriolis Group, 2000.