# Data contracts for cloud-based data marketplaces

## Hong-Linh Truong*

Distributed Systems Group,
Argentinierstrasse 8/184-1, A-1040 Vienna, Austria
E-mail: truong@infosys.tuwien.ac.at
*Corresponding author

## Marco Comerio and Flavio De Paoli

Department of Informatics, Systems and Communication,
University of Milano-Bicocca,
viale Sarca 336/14, 20126, Milano, Italy
E-mail: comerio@disco.unimib.it
E-mail: depaoli@disco.unimib.it

## G.R. Gangadharan

Institute for Development and Research in Banking Technology,
Castle Hills, Road No. 1, Masab Tank,
Hyderabad-500057 India
E-mail: grgangadharan@idrbt.ac.in

## Schahram Dustdar

Distributed Systems Group,
Argentinierstrasse 8/184-1, A-1040 Vienna, Austria
E-mail: dustdar@infosys.tuwien.ac.at

**Abstract:** Currently, rich and diverse data types have been increasingly provided using the dataas- a-service (DaaS) model, a form of cloud computing services and the core element of data marketplaces. This facilitates the on-the-fly data composition and utilisation for several data-intensive applications in e-science and business domains. However, data offered by DaaS are constrained by several data concerns that, if not automatically being reasoned properly, will lead to a wrong way of using them. In this paper, we support the view that data concerns should be explicitly modelled and specified in data contracts to support concern-aware data selection and utilisation. We perform a detailed analysis of current techniques for data contracts in the cloud. Instead of relying on a specific representation of data contracts, we introduce an abstract model for data contracts that can be used to build different types of data contracts for specific types of data. Based on the abstract model, we propose several techniques for evaluating data contracts that can be integrated into data service selection and composition frameworks. We also illustrate our approach with some real-world scenarios and show how data contracts can be integrated into data agreement exchange services in the cloud.

**Keywords:** data marketplaces; cloud computing; data contract; workflows.

**Reference** to this paper should be made as follows: Truong, H-L., Comerio, M., De Paoli, F., Gangadharan, G.R. and Dustdar, S. (2012) 'Data contracts for cloud-based data marketplaces', *Int. J. Computational Science and Engineering*, Vol. 7, No. 4, pp.280–295.

**Biographical notes:** Hong-Linh Truong is a Senior Research Scientist at the Distributed Systems Group, Vienna University of Technology. His research interests focus on understanding of performance, context, and data quality metrics associated with distributed and parallel applications and systems through monitoring and analysis techniques, and on utilising these metrics for the adaptation and optimisation of these applications and systems.

Marco Comerio is a research fellow at University of Milano-Bicocca. His research interests are mainly in the definition of models and techniques to support contract-based service discovery and selection.

Flavio De Paoli is an Associate Professor at the University of Milano-Bicocca. His research interests include models and languages in service-oriented computing.

G.R. Gangadharan is an Assistant Professor at the Institute for Development and Research in Banking Technology. His research interests are mainly in the interface between technological and business perspectives.

Schahram Dustdar is a Full Professor of Computer Science heading the Distributed Systems Group, Vienna University of Technology. His interests are in service-oriented architectures and computing, mobile and ubiquitous computing, complex-, autonomic-, and adaptive systems, and context-aware computing.

This paper is a revised and expanded version of a paper entitled 'On analysing and developing data contracts in cloud-based data marketplaces' presented at Proceedings of 2011 IEEE Asia-Pacific Services Computing Conference (APSCC'11), Jeju, Korea (South), 12–15 December 2011.

## 1 Introduction

Recently, delivering data based on service-oriented and cloud computing techniques is becoming popular. In such a delivery model, data are typically made available for retrieving from web services, mostly implemented using SOAP or REST technologies, deployed in the internet and cloud environments. This model offers extensible and interoperable delivery means in which data can be easily retrieved and business supporting can be easily implemented. Moreover, this model allows incorporating data constraints (e.g., free or commercial usage) and it can be defined as a form of the so-called read-only data-as-a-service (DaaS) (Truong and Dustdar, 2009) which is the core element of cloud-based data marketplaces. Unlike the conventional view on services in which the service provider is the only responsible for all its functions and deliveries, in DaaS, the data provider and the DaaS provider are considered separately. DaaS providers offer the backbone for delivering data while data providers offer data.

While techniques for making data available through DaaS are well-developed, we are interested in the specification of data contractual terms and in the relationship between data contracts and service contracts in the ecosystem of DaaS, which have been neglected in current research. In fact, when a DaaS provides rich types of data, then service contracts cannot be used to specify data contracts as

1 a DaaS offers facilities for multiple data providers

2 a data provider has multiple types of data

3 each type of data can be associated with multiple data contracts.

In this paper, we argue that it is required to define data contracts that can be used separately from service contracts or in combination with service contracts. In particular, we concentrate on data contracts that can support (automatic and on-the-fly) data selection and composition.

Currently, there is a lack of understanding and techniques to deal with data contracts, although data delivered via DaaS is typically associated with human-readable data contracts (often called data agreements or data licences). Our contributions in this paper[1] are

1 the analysis of current data contracts in order to identify relevant data contract properties and methods for DaaS

2 an abstract data contract model for developing data contracts in order to facilitate the right selection and utilisation of data assets in data marketplaces

3 possible methods for evaluating data contract compatibility and possible solutions for making decision in utilising data based on data contracts.

In this paper, we provide an initial implementation of our data contract framework and illustrate some techniques and data contracts related to real-world scenarios to demonstrate the usefulness of our methods and models.

The rest of the paper is organised as follows: Section 2 presents the background, motivation and related work. Section 3 analyses current data contracts in detail. Section 4 presents techniques for developing data contracts. Section 5 presents techniques and guidelines for evaluating data contracts. We describe our experiments in Section 6, followed by conclusions and future work in Section 7.

## 2 Motivation and related work

### *2.1 Background*

The DaaS model is based on the concept that the data can be provided on-demand to the data consumer at anytime and from anywhere, encapsulating the actual platform where data resides. DaaS plays a vital role in emerging data marketplaces in cloud computing environments, such as Microsoft Azure Data Marketplace (https://datamarket.azure.com/) and Infochimps (http://www.infochimps.com/), as well as in Open Data Initiative (http://www.data.gov/opendatasites). In these data marketplaces, several business, statistics, and e-science datasets are provided, and the data can be, on-the-fly, queried by and fed to different computational data-intensive

analysis processes. In DaaS and data marketplaces, data contracts are used to:

- define the extent to which the data can be used, on the basis that any use outside the terms of the contract would constitute an infringement
- have a remedy against the data consumer where the circumstances are such that the acts complained of do not constitute an infringement of the contract
- limit the liability of data and DaaS providers in case of failure of the provided data
- specify information on data delivery, acceptance, and payment.

Currently, most real DaaSs and data marketplaces present data contracts for their offered data assets, often called data agreements or data licenses, in human-readable forms. Typically, data contracts consisting of constraints on data concerns are diverse, rich, and contextual (e.g., depending on geographical regions and publishing purposes).

### 2.2 Motivation

While non-functional properties (NFPs) for services are well-researched and provenance metadata associated with data are well-researched to support service selection and data utilisation, data contract information has not been modelled and associated with DaaS. The lack of well-formed data contract models hinders the data selection and utilisation with respect to data contractual terms, such as data rights, quality of data (QoD), and law enforcements. This triggers calls for consideration of data contracts in data mashup (Fung et al., 2011) and data provisioning (Miller et al., 2008). Our main motivation is that an analogy to a well-researched service contracts but for data assets in DaaS and data marketplaces should be conducted. By doing so, we can answer several questions, e.g.: Are we allowed to use these data? Do the qualities of data delivered via DaaS meet the agreement between data providers and data consumers? Are we allowed to republish the results built based on these data sources? However, such questions require extensible models that are able to capture contractual terms for data contracts and to represent them in a form to be reasoned by automatic techniques. Moreover, certain domain-specific properties of data, such as quality and compliance, make the definition of the methodology to be used for developing data contracts more complicated.

### 2.3 Related work

ODRL (Iannella, 2002) allows specifying data terms but it is not designed for data assets in data marketplaces. ONIX-PL (2011) is another XML-based licenses for digital resources. Our abstract data model is more flexible as we do not propose specifications with all concrete contractual terms; we do not think that a set of pre-defined terms in a specification will be suitable for rich data assets in cloud data marketplaces. Instead, our model is open and includes only common contractual terms that can be reused and composed and allows new terms to be defined and integrated into our model.

In SOA, QoS models for web services have been well-researched and various techniques, methods and tools to support QoS modelling for web services have been proposed (Lee et al., 2003; Ran, 2003; Wang et al., 2006). However, they mainly focus on operational aspects of services like performance, reliability, availability, and security, while the data aspects related to data publishing are largely ignored. On the other hand, much effort has been spent on data quality from database perspectives and many metrics characterising data quality have been proposed (Pipino et al., 2002; Batini et al., 2009). Nevertheless, there is a lack of integration between data contract terms and service contract terms. In fact, no standard model of data contracts that could serve as a basis for the DaaS specification is available so far. Similarly, existing service licensing and service level agreements (SLAs), see e.g., (Gangadharan and D'Andrea, 2006; Keller and Ludwig, 2003), are mainly for 'operational' service APIs and they do not include mechanisms to deal with data contract terms. In specific domains, some data licensing models exist but they are not standards (e.g., see Committee on Licensing Geographic Data and Services, National Research Council, 2004), so they cannot be used in the DaaS model.

To support the composition of data sources in the Internet, especially in the recent Web 2.0 phenomenon, many data composition tools have been developed (Di Lorenzo et al., 2009; Hoyer and Fischer, 2008). In e-science, several workflows have been developed, such as ASKALON (Fahringer et al., 2005), Kepler (Ludäscher et al., 2006), Pegasus (Deelman et al., 2005), Taverna (Turi et al., 2007), and Trident (Simmhan et al., 2009). Many of them provide powerful mechanism to obtain data from different data sources, including DaaS and web services, and to process data in workflows. However, existing techniques mainly focus on selecting data sources based on data structures and on dealing with syntax and semantics of the data, but neglecting data contract terms.

Existing concepts, such as ad-hoc flows (Voorhoeve and van der Aalst, 1997) and web mash-up (Liu et al., 2007), are not integrated with data contracts. Contemporary service selection and combination techniques are built around the QoS, cost, and the semantics of service operations (Ran, 2003; Wang et al., 2006; Blau et al., 2008) without paying attention to data quality and data contracts. Our work does not focus on data composition taking into account data contracts but we support the development of data contracts that can be integrated into existing data discovery and composition tools.

Another related topic is the development of techniques for associating and exchanging data contracts with data. Several works have been introduced to support data licensing, such as Dalheimer and Pfreundt (2009) and Götze et al. (2010). In Truong et al. (2011a), a data agreement exchange service has been developed. In this paper, we do not address the issue of exchanging data contracts.

However, we will illustrate how our data contracts can be used together with a data agreement exchange service. In our previous work, the service selection techniques currently do not deal with the compatibility between different data licence models (Gangadharan et al., 2008) when integrating data from different services. A recent work has supported the evaluation of service contracts, but its support on data-related concerns is limited (Comerio et al., 2009b). In this paper, we present a specific algorithm for data contract compatibility

## 3 Analysis of data contracts

### *3.1 Main data contract terms*

Although data include variety of properties, in this paper, we investigate some of the properties that are considered relevant in the perspective of contracts for DaaS. Our analysis is conducted based on studying of existing data licences and agreements as well as service contracts. Some of the key properties of data that are significant in making a data contract in DaaS are elucidated as follows.

#### *3.1.1 Data rights*

Data rights specify the rights that the provider authorises the consumer to exercise for data in DaaS. They are important for clarifying and assuring intellectual property rights. The set of common data right terms for data assets offered by existing DaaS and data marketplaces are the following:

- *Derivation*: any translation, adaptation, or any other alteration of a data asset or of a substantial part of the data makes a derivative data asset. This derivation includes, but is not limited to, extracting or re-utilising the whole or a substantial part of the data in a new data asset.
- *Collection*: a collective data asset refers to a data asset in unmodified form as part of a collection of independent works in themselves that together are assembled into a collective whole.
- *Reproduction*: from a given data asset, temporary or permanent reproductions can be created by any means and in any form, in whole or in part, including of any derivative data assets or as a part of collective data assets.
- *Attribution:* the data provider may expect attribution (a kind of moral right) for the use of its data.
- *Noncommercial use:* a data asset could be allowed/denied either for non-commercial purposes or for commercial purposes.

#### *3.1.2 Quality of data*

Multiple metrics can be used to describe data quality, such as completeness, reliability, accuracy, consistency, and interpretability (Batini et al., 2009). In existing DaaS, QoD data certification is mentioned, e.g., in certain data assets in http://data.gov. However, it is not clear how to establish data quality certification. In our view, there exist several QoD metrics, each can have a unique name. The interpretation of a QoD metric for a data asset should be based on common agreements established in the domain in which the data asset is created and used. Usually, a QoD term specifies a range of possible values associated with a QoD metric.

#### *3.1.3 Regulatory compliance*

It is important to protect privacy and confidentiality of information published, thus data assets are typically associated with many regulatory compliance. For example, in certain data assets in http://data.gov, data compliance is mentioned (http://explore.data.gov/Law-Enforcement-Courts-and-Prisons/2008-C Some of the common regulatory compliance laws include the Healthcare Insurance Portability and Accountability Act (requiring the securing of patient information), Sarbanes-Oxley (SOX) Act (requiring company financial executives to be culpable for financial reporting), the European Union Data Protection Directive (protecting data privacy for citizens throughout the European Union), and so on. Most of the DaaS providers define specifications on data compliance terms. Most data compliance laws and regulation assume that the liable party controls the infrastructure and the location where the data is stored (Wang et al., 2010). In our view, a compliance term can be specified as a term name and a set of values where values relate to respective compliance specifications. evaluation.

#### *3.1.4 Pricing model*

Data consumers pay data providers for the right to use the data asset subject to the contract by the financial terms. The most common models for data pricing in DaaS and data marketplaces are transaction and subscription-based model. The *transaction model* allows DaaS providers to charge for each use. The *subscription model* allows consumers to purchase data for a fixed term, during which time they automatically receive full support from providers including any upgrades or feature enhancements. For both models, pricing can be applied to the whole DaaS [e.g., Gnip (http://www.gnip.com) supports subscription] or specific data assets (e.g., the pricing model in Microsoft Azure Data Marketplace and Infochimps). In our view, pricing model is typically specified as a set of values per pricing plan which includes cost, usage time and/or maximum number of transactions to be applied to the whole DaaS or a particular data asset.

**Table 1** Example of data contracts in real-world DaaS

| *Contracts* | *Data rights* | | | | | *Quality of data* | | *Compliance* | *Pricing model* | | *Control and relationship* | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | *Derivation* | *Collection* | *Reproduction* | *Attribution* | *Non-commercial use* | *Completeness* | *Accuracy* | | *Transaction* | *Subscription* | *Warranty* | *Indemnity* | *Liability* | *Laws, Jurisdiction* |
| AvianKnowledge.net (AKN, 2011) | | | | + | + | | | | + | + | + | + | + | |
| Building model products (BMP, 2011) | | + | | | + | | | + | | + | + | + | + | + |
| Creative common: attribution-ShareAlike 2.0 Generic (CCAS, 2011) | + | + | + | + | | | | | | | + | + | + | + |
| Consumer expenditure data (CED, 2011) | | + | | | + | | | + | | + | + | + | + | + |
| Freebase data dump (FDD, 2011) | + | + | + | + | | | | | | | + | + | + | + |
| GBIF Data Usage Agreement (GBIF, 2011) | + | + | + | + | | | | + | | | + | + | + | + |
| Infochimps Twitter Census: Stock Twittes (TCST, 2011) | + | + | | | | | | + | + | | + | | + | + |
| Open Data Commons Attribution License (ODCAL, 2011) | + | + | + | + | | | | | | | + | + | + | |
| Open Government License (OGL, 2011) | + | + | + | + | | | | | | | | | + | |
| US Consumer Price Index: 1913 to current (USCPI, 2011) | | + | | | + | | | + | | + | + | + | + | + |

### *3.1.5 Control and relationship*

The control and relationship terms consist of evolution terms, support terms, indemnification, limitation of liability, and audits of contract compliance. Existing data contracts indicate control and relationship terms using similar ways in service contracts. Therefore, in our opinion, control and relationship terms in data contracts could reuse the similar ways of control and relationship terms in service contracts. From the modelling perspective, control and relationship terms can be specified as a set of $tuple(name, value)$ in which $name$ and $value$ have corresponding interpretations. For example, tuples $(LawandJurisdiction, USA)$ and $(LawandJurisdiction, Austria)$ can be used to describe two different laws, $USA$ and $Austria$ to be enforced for data contracts. Here, all terms – $LawandJurisdiction, USA$, and $Austria$ – require concrete interpretation rules in order to understand their semantics.

### *3.2 Analysis of contemporary data contracts*

As mentioned above, the most popular form of data contracts is human-readable textual description of data agreements/licensing. Table 1 presents our analysis of data contracts in real-world data services in which all data contracts are in textual description for human beings, thus they do not foster the incorporation of data contracts in data discovery and composition. Overall, we have not seen a relevant difference between current data contracts/licensing and existing service contract/licensing with respect to the specification of scope of rights, control and relationships (e.g., warranty and liability).

As shown in Table 1, studied data contracts do not cover many aspects of contractual terms related to data. For example, most of the current DaaS contracts do not provide information about QoD, which in fact should be one of the main terms in data contracts. The analysis of data contracts heralds the requirement of new research directions for data contracts because data assets provided by DaaS have different properties, compared to software services. For example, data contract composition is needed when mashup of data from different data providers are performed. This composition consists in

1 retrieving comparable contractual terms from the different data contracts
2 evaluating the new contractual terms for the data mashup applying proper composition rules.

Another example is data contract compatibility evaluation. This activity must be performed, e.g., before conducting a data mashup, to check if terms are compatible or not.

## 4 Developing data contracts

### *4.1 Community view on data contract development*

As we discuss in the previous section, categories of data contract terms are limited. However, contract terms are diverse. In particular, data contract terms are contextual (e.g., based on laws of geographical regions and the domain of data assets). Furthermore, in many cases, data contract term values and their measurement units are also complex and contextual, e.g., one needs to make sure that the value 'Austria' can be interpreted as a sub element of 'European Union' (EU) in some specific contexts. Therefore, we do not expect that a unified specification for data contracts, with pre-defined term names, will be available and sufficient. In order to deal with data contracts in data marketplaces, we propose a different approach centered on a combination of community and people-centric collaboration.

First, we propose to enable community users to participate in defining

1 fundamental elements in data contracts, such as term categories, term names, term values and term units

2 rules for data contracts, such as syntax validation and evaluation rules

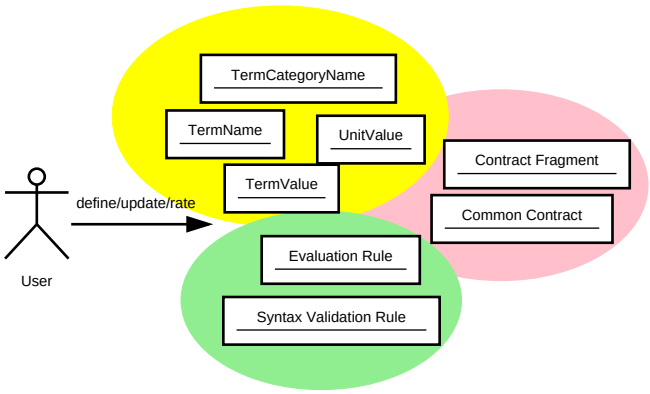
3 common contracts and contract fragments (see Figure 1).

Note that community users should be understood as experts in specific domains who understand contractual terms suitable for data in their domain, not novice users. The combination of community and people-centric collaboration is required to solve the heterogeneity of data contract terms, their values, and their measurement units. Such terms and units are contextual since different terminologies can be used in different domains. In our view, common terminologies and domain-specific knowledge are used by domain experts to define term categories, term names, term values and term units that characterise a particular domain. Then, domain experts utilise these definitions and domain-specific knowledge to provide common contracts and contract fragments as well as customised validation and evaluation rules. This way is similar to the approach carried out in developing the Dublin core (http://dublincore.org/), which results in several fundamental and well-understood terms.

Second, by employing a people-centric approach in establishing and developing data contracts, we propose that data providers and consumers can utilise fundamental elements to define their own contracts and evaluation techniques.

We should note that this approach has been applied well in the development of community-based knowledge. Thus, based on our approach, different communities, such as in astrophysics, biological data, social network data, can collaborate to define contract vocabularies, terms and rules for data contracts in their domains.

**Figure 1** Community contributions in data contracts (see online version for colours)




### *4.2 Representing data contract terms*

The first step in providing abstract data contract models is to determine possible representations for data contract terms. From our analysis, Table 2 presents possible ways to model data contract terms for different categories. Overall, we can represent a data contract term as a tuple of $(termName, termValue)$ in which $termName$ is either common terms established via standards/communities or user-specific terms and $termValue$ are the assigned values for $termName$. As shown in Table 2, $termValue$s can be a set, a single value, or a range. We explain them in the following:

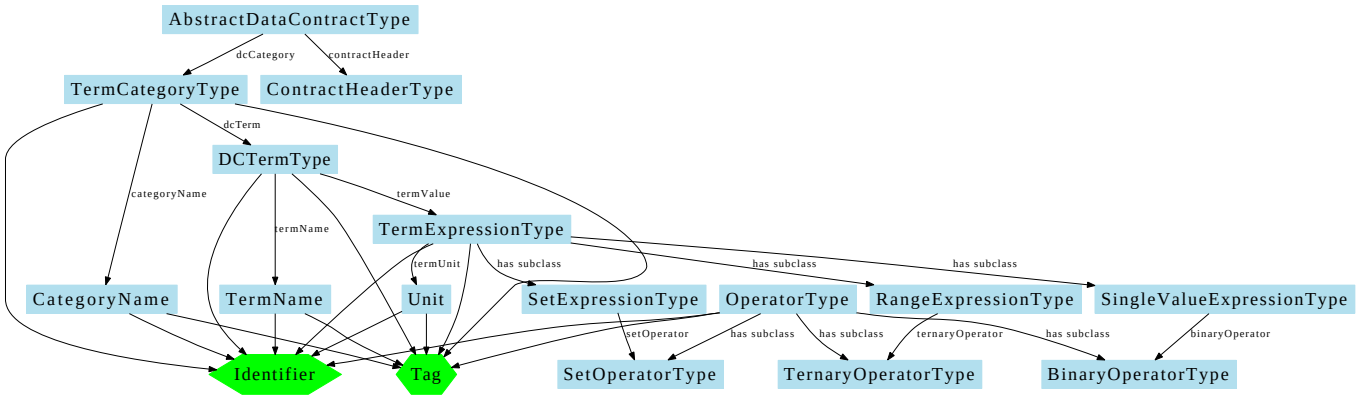
- Data rights: a term name in `DataRight` can be represented as a unique name whose values can be represented by a set of pre-defined values. Both term names and values are pre-defined and their interpretations are known.
- QoD: the value of a QoD term can be represented in a range in $[0, 1]$. The QoD term are predefined and their meanings are known. The semantics of values are also understood by the community.
- Compliance: the values of a compliance term indicate the names of compliance regulations. The regulations are known and pre-defined.
- Pricing model: the value of a pricing model is represented in a generic way in which the cost, the time and the number of transactions are specified.
- Control and relationship: the value of a `ControlRelationship` is described by a name indicating the geographical regions in which `ControlRelationship` terms are applied. The interpretation of a `ControlRelationship` $(termName = val)$ term is as follows: the condition indicated by $termName$ is applied in the geographical region indicated by the $val$.

In all above-mentioned terms and categories, without indicating the value, we can interpret that the data terms are unknown.

**Table 2** Data contract terms and values

| *Category* | *Term representation* | *Examples* |
|---|---|---|
| Data rights | $termName = \{val_1, val_2, \cdots, val_n\}$ | $termName$={`Derivation, Collection, Reproduction, Attribution, Noncommercialuse`}, $val_i$ = {`Undefined, Null, Allowed, Required, True, False`} |
| Quality of data | $val_l \leq termName \leq val_u$ | $termName$={`Accuracy, Completeness, Uptodateness`}, $val_l$ and $val_u \in [0, 1]$ |
| Compliance | $termName = \{val_1, val_2, \cdots, val_n\}$ | $termName$ and $val_i$ are any string, e.g., $termName$={`PrivacyCompliance`} and $termValue$={`Sarbanes-Oxley (SOX) Act`} |
| Pricing model | $termName = (cost = val_1, usagetime = val_2, , maximumuse = val_3)$ | $termName$ is any string, e.g., `MonthlyPayment`; $val_1 \in R$, e.g., $cost = 50$ €, $val_2 = \{(end_t - start_t); UNLIMITED\}$ where $end_t, start_t \in datetime$, e.g., $usagetime = 30$ days; $val_3 \in N$, e.g, $maximumuse = 1,000$ calls |
| Control and relationship | $termName = val$ | $termName$ and $val$ are any string, e.g., $termName$={`Liability, LawandJurisdiction`} and $val = \{US, Austria\}$ |

**Figure 2** (Simplified) abstract structure of data contracts (see online version for colours)




### 4.3 *Structuring data contracts*

Figure 2 presents our abstract data contract structure. By 'abstract' we mean two aspects. First, this structure is not the final form of a data contract as it represents only contractual conditions without specifying on which data assets it is applied to. Second, the structure is not a proposal for final and concrete data contract specification, which can be seen and obtained by data consumers in data marketplaces, but it can be used to make abstract contracts from which concrete specifications will be generated for data consumers and providers. In the following, we explain the proposed structure.

`TermCategoryType` is used to specify one or more elements that categorise the terms specified in the contract. `CategoryName` is used to identify the category. From the analysis of data contract terms in the previous section, we identify that five `CategoryName` – (`DataRight`, `QoD`, `Compliance`, `PricingModel` and `ControlRelationship`) – are available. In principle, a new category name can be defined. A contract consists of a set of `TermCategoryType` elements, each includes a set of data contract terms described by `DCTermType` which covers specific-aspects and it is specified by means of a `termName` and a `termValue`. Each `termValue` is defined by means of a `TermExpressionType` that is specified by an operator and by a set of attributes that depend on the constraint operator used. The specification of both qualitative and quantitative terms is supported. The former are terms defined through single and set expressions of qualitative values with well-defined meanings; the latter use expressions of numeric values, whose measurement units is specified by `Unit`.

All the above-mentioned types are associated with an identifier and set of tags, specified by `Identifier` and `Tag`, respectively. Using `Identifier` we follow the Dublin core model to distinguish well-defined, agreed categories, terms, and values. Using `Tag`, we allow community users to specify tags to support searching terms, values and contracts.

Our abstract data contract model is designed at the technical level, rather than at the business level. Therefore,

it requires to perform certain mappings from business-level terms to technical-level terms. However, the abstract data contract model is capable of representing several contractual terms at the business level. For example, under `QoD`, several terms like `Accuracy`, `Completeness` or `Uptodateness` have their values the range of $[0, 1]$, whose descriptions are the same in real world businesss contracts. Furthermore, the given abstract data model is capable of representing directly many business constraints/conditions or obligations/requirements by using `SetExpressionType`, `RangeExpressionType`, `SingleValueExpressionType`, and `OperatorType`.

## 5 Data contract compatibility evaluation

Several applications for the management of data contracts can be developed by utilising our proposed data contract model. In this section, we focus on the definition of an approach to develop an application for *data contract compatibility evaluation* for data composition (e.g., data mashups). This application is required when we intend to combine multiple data assets, and we need to check whether data contracts associated with these data assets are compatible. Basically, we say that a data contract $c_x$ and a data contract $c_y$ are compatible if each contract term in $c_x$ does not clash with any contract term in $c_y$, and vice versa. A contract term $ct_x$ clashes with a contract term $ct_y$ if they assume distinct values without relations (e.g., *subset, isA, subsumes, partOf, includes*) between them.

Generally, an approach to data contract compatibility evaluation covers the following basic principles:

- For each `DCTermType` $t_j$ in each `TermCategoryType` $tc_i$, we can extract the comparable terms from all the data contracts to be checked. For example, in the category of `DataRight`, comparable terms can be `Derivation`, `Composition` and `Reproduction`.
- Then, we can retrieve from a *rule repository* the evaluation rule associated with the `DCTermType` $t_j$. In cases, such a rule does not exist, we need to define it.
- Finally, we can execute the rule by passing the list of comparable terms extracted from the contracts.

However, when realising the evaluation of data contracts, a particular important issue is the role of the quality of the information provided by each data contract and the quality of the data contracts according to the particular task (e.g., data composition) in which they are used. This has not been investigated so far in related works. For this reason, we propose an approach that merges the basic principles mentioned above with new principles in order to consider the quality of the data contracts along the evaluation. Basically, we provide a comprehensive approach that supports the evaluation of compatibility along with the evaluation of a wide set of data quality dimensions associated with data contracts.

### *5.1 Evaluating the QoD contracts*

To evaluate the QoD contracts, we rely on *reputation, timeliness, consistency* and *completeness* described in Batini and Scannapieco (2006). In our work, they are re-defined as follows:

- *Reputation*: specifies the trustworthiness of a data contract in terms of its sources and contents. This metric is directly inferred from the reputation of the DaaS provider that offers the contract. Statistical measures of DaaS provider reputation are organised and shared by third party services according to conceptual models such as in Maximilien and Singh (2002). The value of $Reputation$ is in $[0, 1]$ where $0$ and $1$ indicates the lowest and the highest trustworthiness, respectively.
- *Timeliness*: has the value in $[0, 1]$ that defines if the age of a contract term is appropriate. This metric is evaluated for each contract term considering its expected validation:

$$Timeliness = max\left(0, 1 - \frac{Age}{Expected\ Validation}\right) \quad (1)$$

  The expected validation represents the average lifetime of a contract term. As an example, the expected validation of a *pricing model* and a *data right* terms could be equal to one month and one year respectively, since the price of a dataset is supposed to change more frequent than its data rights.
- *Consistency*: indicates the degree of contradictions between contract terms. $Consistency$ has a value in $[0, 1]$ in which $0$ indicates no contradiction and $1$ indicates a full contradiction. Examples of contradictions are:

  1 different contractual terms on the same data contract term type in the same contract and under the same conditions (e.g., *payment = Flat Rate* and *payment = Free per use*)

  2 conflicting contract terms in the same contract and under the same conditions (e.g., *payment = Free per use* and *cost = 100 Euro*).

  Contract consistency is evaluated by means of pre-defined rules available in the literature such as in Cambronero et al. (2007).
- *Completeness*: has the value in $[0, 1]$ and represents the ratio between the number of contract terms in a contract and the cardinality of the minimum set of terms that is required for a complete data contract evaluation:

$$Completeness = min\left(1, \frac{\|Contract\ terms\|}{\|Minimum\ term\ set\|}\right) \quad (2)$$

To be notice that the minimum set is strictly related to the domain the data refers to. As an example, the minimum term set for a contract associated with biological data

can be {*derivation, collection, reproduction, accuracy* and *uptodateness*}.

In order to evaluate the quality of individual data contracts. Two main activities are performed:

- The quality that each contract has on its own is evaluated. In our approach, we evaluate a data contract based on $reputation$ information about the DaaS offering the contract. This information can be retrieved from third-party services. Then, we evaluate the $timeliness$ of each contract term.
- We evaluate the $consistency$ of each data contracts in order to verify the presence of contradictions between contract terms within the contract.

By employing the above-mentioned steps, we can decide to accept or eliminate data contracts offered from different DaaS. As a result, a DaaS can be selected or rejected, or its data contract can be renegotiated. When a DaaS is selected, its contract must be evaluated to check if the contract is compatible with other contracts associated with data to be composed in the same application.

### 5.2 Evaluating compatibility among data contracts

Given a set of individual data contracts that have been verified using the method described before, we need to evaluate if there is any incompatible issue in the data composition with respect to contract terms. Three main activities are performed:

- *Matching contract terms*: this step discovers comparable contract terms $ct_x$ and $ct_y$ specified in two data contracts $c_x$ and $c_y$. The results is a set of matching couple ($ct_x$,$ct_y$). Two contract terms $ct_x$ and $ct_y$ are comparable when they are defined as expressions built as a constraint based on the same data contract term type (*DCTermType* in our model). Rule-based mediators, defined as logic programming rules, are exploited to solve semantic mismatches. Rules have the following form:

  $$matchCouple(?ct_x, ?ct_y, TId) :- COND(?ct_x, ?ct_y).$$

  with $TId$ being an identifier of the rule concept target (e.g., *derivation*), and COND representing a set of conditions over $?ct_x$ and $?ct_y$ defining the matching criteria (e.g. membership of $?ct_x$ and $?ct_y$ to specific *DCTermType*).
- *Evaluating contract term compatibility and completeness wrt application needs*: this step evaluates, for each ($ct_x$,$ct_y$) identified in the previous step, if the two terms are compatible or not. According to the approach proposed in Comerio et al. (2009a), mathematical functions and logic programming rules are used to perform the evaluation. The results are in $[0, 1]$ with 0 means that contract terms are completely incompatible and 1 means they are fully compatible. A result in $(0, 1)$ indicates a partial incompatibility. Along with the evaluation of compatibility between contract terms, this step evaluates the *completeness* of each contract $c_x$ involved in the data composition. This metric is strictly related to the task at hand (i.e., contract term compatibility evaluation) and it is evaluated using the formula (2).
- *Making decision in using data*: If two contracts are compatible, we can check the overall $reputation$, $consistency$ and $timeliness$ of the two contracts and decide whether the data should be used or other steps must be done. If any incompatibility has been found, we can try to identify possible remedy solutions, dependent on the $completeness$ and $timeliness$. Possible steps in decision-making after evaluating contract compatibility are shown in Table 3. To observe that the quantification 'LOW' must be considered according to pre-defined thresholds. Examples of these thresholds that can be customised are: 0.5 for reputation and 0.99 for $consistency$ and $completeness$. For what concerns timeliness, different values are associated with different contract term type (e.g., 0.66 for data right terms and 0.5 for payment).

**Algorithm 1** Compatibility evaluation

```
1: for all c_i ∈ C do
2:   for all c_j ∈ C(j ≠ i) do
3:     λ(c_i, c_j) = φ, where λ(c_i, c_j) is a set of
       incompatible contract terms [ct_w, ct_z]
4:     ϒ(c_i, c_j) = φ, where ϒ(c_i, c_j) is a set of
       comparable contract terms [ct_w, ct_z]
5:     ϒ(c_i, c_j) = Matching(c_i, c_j)
6:     for all [ct_1, ct_2]∈ ϒ(c_i, c_j) do
7:       rule = Extract(ct_1.type)
8:       result = CheckCompatibility(rule, ct_1, ct_2)
9:       if result ≠ 1 then
10:        λ(c_i, c_j) = λ(c_i, c_j) ∪ [ct_1, ct_2]
11:      end if
12:    end for
13:    if λ(c_i, c_j) = φ then
14:      CheckReputation(c_i, c_j)
15:      CheckConsistency(c_i, c_j)
16:      CheckTimeliness(c_i, c_j)
17:    else
18:      CheckCompleteness(c_i, c_j, λ(c_i, c_j))
19:      CheckTimeliness(c_i, c_j, λ(c_i, c_j))
20:    end if
21:  end for
22: end for
```

### 5.3 Algorithm for data contract compatibility evaluation

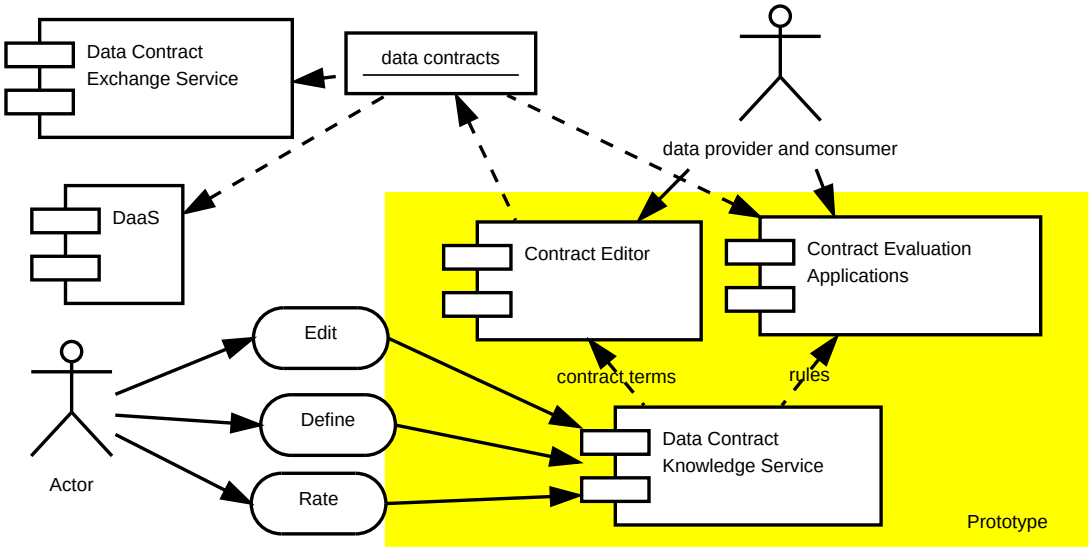
Let $C = \{c_1, c_2, \cdots, c_m\}$ denote the set of contract associated with data to be composed. Each contract is composed of a set of contract terms $\{ct_1, ct_2, \cdots, ct_w\}$.

Our data contract compatibility algorithm is listed in Algorithm 1. The algorithm evaluates the compatibility among all the contracts available in the composition.

**Table 3** Possible steps in making decisions based on contract compatibility evaluation

| | *Action* | *Case* | *Solution* |
|---|---|---|---|
| Compatibility = 1 | Check reputation, consistency and timeliness | Reputation = LOW | Change the data composition. Substitute the data provided by the untrustworthy DaaS. |
| | | Consistency = LOW | Interact, if possible, with the DaaS providers to solve inconsistent contract terms. |
| | | Timeliness = LOW | Interact, if possible, with the DaaS providers to update contract terms. |
| Compatibility $<$ 1 | Check completeness and timeliness | Completeness = LOW | Interact, if possible, with the DaaS providers to have additional contract terms. |
| | | Timeliness = LOW | Interact, if possible, with the DaaS providers to update contract terms. |

**Figure 3** Our prototype for data contract management (see online version for colours)




Line 3 defines $\lambda(c_i, c_j)$ as a set of incompatible contract terms specified in the contracts $c_i$ and $c_j$. The evaluation of $\lambda(c_i, c_j)$ starts in Line 4 defining $\Upsilon(c_i, c_j)$ as a set of comparable contract terms $[ct_1, ct_2]$ specified in $c_i$ and $c_j$. $\Upsilon(c_i, c_j)$ is populated by the `Matching` procedure (Line 5) that applies matching rules.

For each identified couple $[ct_1, ct_2]$ of comparable terms, the algorithm retrieves the related evaluation rule using the procedure `Extract` and specifying the data contract term type (Line 7). The compatibility between $[ct_1, ct_2]$ is evaluated by means of the procedure `CheckCompatibility` specifying the retrieved rule and the two comparable contract terms (Line 8).

The result of the procedure is in $[0, 1]$ with 0 means contractual terms are not compatible and 1 means they are compatible. If $[ct_1, ct_2]$ are not fully compatible, they are saved in $\lambda(c_i, c_j)$ (Line 10).

To support decision-making after the compatibility evaluation, several metrics are checked, starting at Line 13. If no incompatible contractual terms exist between $c_i$ and $c_j$ (i.e., $\lambda(c_i, c_j) = \phi$), the procedures `CheckReputation`, `CheckConsistency` and `CheckTimliness` are invoked to check the accuracy of the evaluation (Lines 14 to 16). Otherwise, the procedures `CheckCompleteness` and `CheckTimeliness` are invoked to check the availability of remedy solutions (Lines 18 and 9) like the ones in Table 3.

## 6 Prototype and experiments

### *6.1 Prototype*

We choose to use the resource description framework (RDF) to represent term categories, term names, term values and term units. As a consequence, we have rules developed atop RDF. Figure 3 describes our prototype. Our community-based term categories, names, values and units can be defined, edited and rated by community users (such as owners of data assets) via different processes. We use Allegro Graph (http://www.franz.com/agraph/allegrograph/)

as our *data contract knowledge service*. By utilising the RDF knowledge, data providers and consumers can edit and evaluate data contracts. The resulting contracts can be extracted into different formats, such as XML, JSON and RDF. These contracts can be associated with data assets, managed by DaaS, stored in other services [such as a data agreement exchange service for data marketplaces (Truong et al., 2011a)], or stored into *data contract knowledge service* as common, shared data contracts. In our current prototype, *Data Contract Knowledge Service* includes common terms, categories, and contracts (based on data contracts in Table 1). In our prototype, we also use SPARQL rules and we develop evaluation applications to implement the algorithm mentioned in Section 5.

### 6.2 Constructing and composing data contract

Let us consider a cloud sustainability governance platform that manages very large sustainability monitoring data, such as the Galaxy platform (PCS, 2011). Using the data and analysis capability in this platform, several summarised data could be provided. In our example, the platform provider would like to combine the real-time total and per capita of $CO_2$ emission of monitored buildings with an open government data asset about the $CO_2$ emission per capita in the national level (such as http://www.apho.org.uk/resource/view.aspx?RID=91904) to show how green these buildings are.

In the first step, the provider decides to utilise Open Data Common terms for building $CO_2$ emission data but the provider wants to include certain QoD and to prevent any derivation of the emission data. Thus, the provider first checks existing common terms in *Data Contract Knowledge Service* in order to reuse these terms. Figure 4 shows examples of existing common categories, term names, operators, units, and expressions as well as open data commons (ODC)-based terms. By utilising this existing knowledge, the provider defines a new data contract named `OpenBuildingCO2`. For this contract, the provider takes all ODC terms except `odcDerivation` (for derivation in data rights) and defines a category `obcQoD` (for QoD) and a new term `obcDerivation` (for derivation). The new `obcDerivation` is defined by combining the common existing `Derivation` term and `NotAllowed` expression in the service. Listing 1 shows an excerpt of `OpenBuildingCO2` with respect to the `DataRight` category and the `Derivation` term. From this abstract data contract, concrete forms of the data contract can be generated in XML, RDF or JSON and then associated with appropriate data and DaaS.

The next step is to combine building $CO_2$ emission data with an open government data asset and an open map data[2]. Because the resulting data is a combination of different data assets controlled by different data contracts, the provider has to check the compatibility and even propose a new data contract for the combined data. In this experiment we assume that the open government data is based on the Open Government License (OGL, 2011) and we create an abstract contract – named `OpenGovernment` – for open government data.

Listing 2 shows an example of rules to detect if data rights are compatible or not. The example illustrates a rule used to check the data rights of `OpenBuldingCO2` and `OpenGovernment` contracts. Part of the evaluation, we need to check the `Derivation` right of `OpenBuildingCO2` – denoted by variable `?varDR1` – the `Collection` right of `OpenGovernment` – denoted by the variable `?varDR2`. In this case, because `OpenBuildingCO2` has derivation right as `NotAllowed` and `OpenGovernment` has collection right as `Allowed`, invoking the rule will result in an incompatibility.

Listing 3 shows the rule used for composing an `Accuracy` term under `QoD` category from two inputs – `varAcc1` and `varAcc2`. This rule considers that `varAcc1` has `SingleValueExpressionType` with `atLeast` operator and `varAcc2` has `RangeExpresionType` with `interval` operator. Due to the operators and expression types, the composite accuracy, denoted by `compositeAccuracy`, will have `RangeExpressionType` and its lower bound value must be $max(varAcc1, varAcc2.lowerBound)$, while its upper bound will be the upper bound of $varAcc2$. Note that depending on different `TermExpressionType` of input variables, we could have different rules for composing two terms under `QoD`. Thus, in principle, several rules can be developed and data contract applications can utilise these rules based on their needs. In our case, since `OpenGovernment` has no QoD term, the rule can take the QoD terms from `OpenBuildingCO2`.

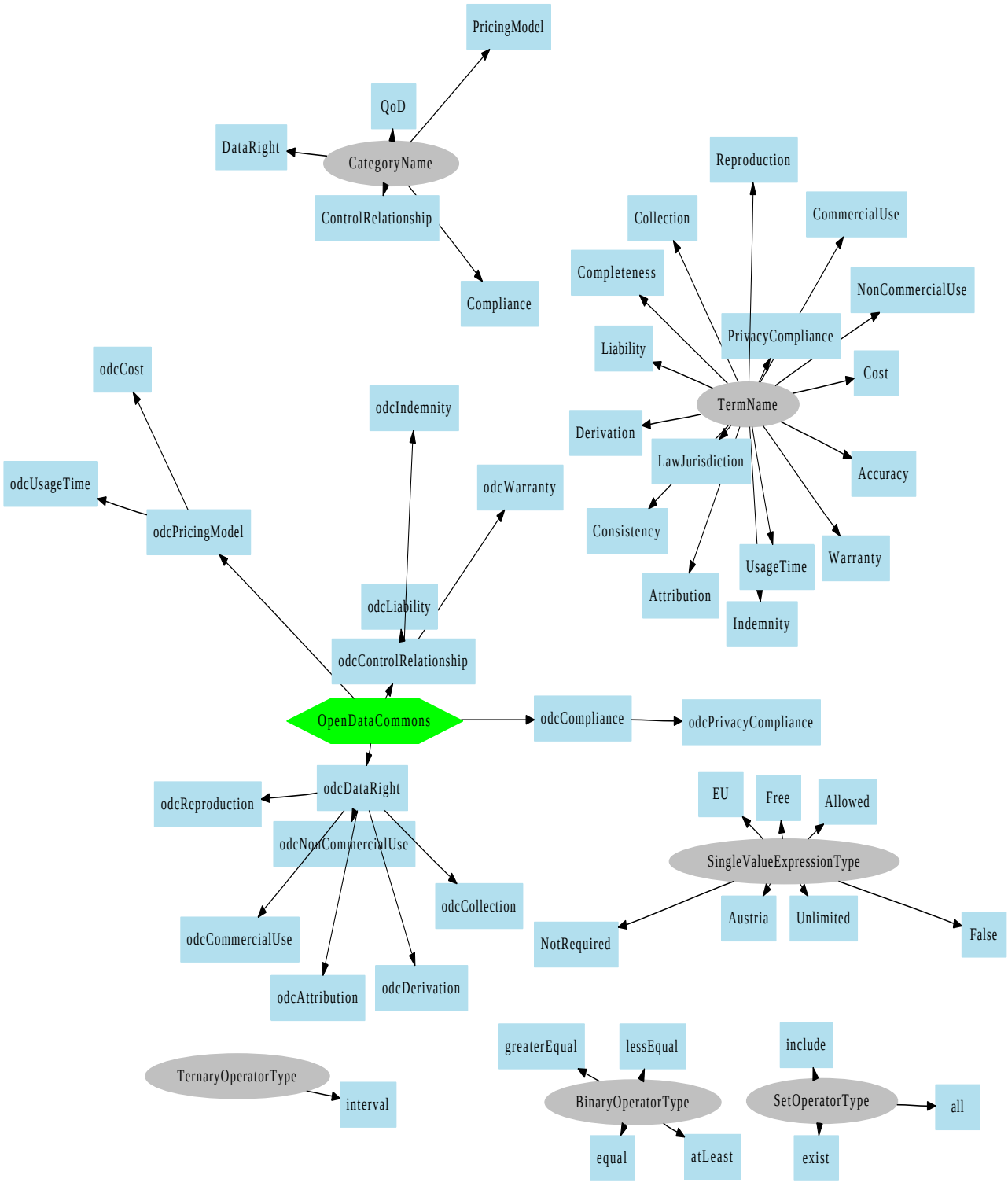
Overall, our experiments illustrate the usefulness of having abstract data contracts being defined by utilising existing categories and terms. The concrete data contracts in XML, JSON or RDF will facilitate the search and composition of data assets.

### 6.3 Exchanging data contracts

Data contracts can be associated with and delivered together with data or can be used to establish the conditions for accessing data. Our data contracts can be integrated with the Data Agreement Exchange Service (DAES) developed in Truong et al. (2011a).

Listing 4 presents an example of how `OpenBuildingCO2` contract can be stored and linked to data[3]. The metadata agreement is defined in Truong et al. (2011a). In this example, the `identification` part is used to specify information about data assets, providers, consumers and DAES. The example illustrates an agreement, whose id is `urn:pcccl:agreement:1`, to allow the consumer `urn:tuwien:infosys` to utilise a data stream indicated by `http://pcccl/dataStream/stream124` which is provided by `http://pcccl`. The agreement is stored in an instance of DAES indicated by the tag `dataAgreementExchangeService`. By using `agreementReference`, the consumer can retrieve the agreement in RDF using the external link in `content`.

**Figure 4** Example of exploring common categories, terms, expressions, operators and values in *data contract knowledge service*, visualised by our prototype which utilises GraphViz (see online version for colours)



**Listing 1** Simplified excerpt of OpenBuildingCO2

```
<Description rdf:about="http://www.infosys.tuwien.ac.at/SOD1/adcm#OpenBuildingCO2">
  <ns1:dcCategory rdf:resource="http://www.infosys.tuwien.ac.at/SOD1/adcm#odcFinancial"/>
  <ns1:dcCategory rdf:resource="http://www.infosys.tuwien.ac.at/SOD1/adcm#
      odcControlRelationship"/>
  <ns1:dcCategory rdf:resource="http://www.infosys.tuwien.ac.at/SOD1/adcm#odcCompliance"/>
  <ns1:dcCategory rdf:resource="http://www.infosys.tuwien.ac.at/SOD1/adcm#obcQoD"/>
  <ns1:dcCategory rdf:resource="http://www.infosys.tuwien.ac.at/SOD1/adcm#obcDataRight"/>
  <rdfs:label xml:lang="en">OpenBuildingCO2</rdfs:label>
  <rdf:type rdf:resource="http://www.infosys.tuwien.ac.at/SOD1/adcm#
      AbstractDataContractType"/>
</Description>
<Description rdf:about="http://www.infosys.tuwien.ac.at/SOD1/adcm#odcDataRight">
  <ns1:dcTerm rdf:resource="http://www.infosys.tuwien.ac.at/SOD1/adcm#odcReproduction"/>
  <ns1:dcTerm rdf:resource="http://www.infosys.tuwien.ac.at/SOD1/adcm#odcDerivation"/>
  <ns1:dcTerm rdf:resource="http://www.infosys.tuwien.ac.at/SOD1/adcm#odcCommercialUse"/>
  <ns1:dcTerm rdf:resource="http://www.infosys.tuwien.ac.at/SOD1/adcm#odcCollection"/>
  <ns1:dcTerm rdf:resource="http://www.infosys.tuwien.ac.at/SOD1/adcm#odcAttribution"/>
  <rdfs:label xml:lang="en">odcDataRight</rdfs:label>
  <rdf:type rdf:resource="http://www.infosys.tuwien.ac.at/SOD1/adcm#TermCategoryType"/>
</Description>
<Description rdf:about="http://www.infosys.tuwien.ac.at/SOD1/adcm#odcDerivation">
  <ns1:termName rdf:resource="http://www.infosys.tuwien.ac.at/SOD1/adcm#Derivation"/>
  <ns1:termValue rdf:resource="http://www.infosys.tuwien.ac.at/SOD1/adcm#NotAllowed"/>
  <rdfs:label xml:lang="en">odcDerivation</rdfs:label>
  <rdf:type rdf:resource="http://www.infosys.tuwien.ac.at/SOD1/adcm#DCTermType"/>
</Description>
```

**Listing 2** Example of compatibility rule for data rights

```
PREFIX
  adcm: <http://www.infosys.tuwien.ac.at/
      SOD1/adcm#>
ASK {
  ?varDR1 rdf:type adcm:DCTermType .
  ?varDR1 adcm:termName adcm:Derivation .
  ?varDR1 rdfs:label 'obcDerivation' .
  ?varDR2 rdf:type adcm:DCTermType .
  ?varDR2 adcm:termName adcm:Collection .
  ?varDR2 rdfs:label 'odcCollection'.
  ?varDR1 adcm:termValue ?value1 .
  ?varDR2 adcm:termValue ?value2 .
 FILTER (regex(str(?value1),str(?value2))).
}
```

**Listing 3** An example of a composition rule for QoD

```
PREFIX
  adcm: <http://www.infosys.tuwien.ac.at/SOD1/adcm#>
CONSTRUCT {
  adcm:compositeAccuracy adcm:lowerBound ?compositeLowerBound .
  adcm:compositeAccuracy adcm:upperBound ?compositeUpperBound .
}
WHERE {
  ?varAcc1 rdf:type adcm:SingleValueExpressionType .
  ?varAcc1 adcm:numericValue ?value .
  ?varAcc1 adcm:binaryOperator adcm:atLeast .
  ?varAcc2 rdf:type adcm:RangeExpressionType .
  ?varAcc2 adcm:lowerBound   ?lowerBound .
  ?varAcc2 adcm:upperBound   ?upperBound .
FILTER (?value <= ?upperBound) .
  LET (?compositeLowerBound := afn:max(?value, ?lowerBound)) .
  LET (?compositeUpperBound :=?upperBound) .
}
```

**Listing 4** Example of metadata about a data agreement

```
<?xml version="1.0" encoding="UTF-8"?>
<ns0:dataAgreement xmlns:ns0="urn:de:icsy:dataagreement" xmlns:xsi="http://www.w3.
    org/2001/XMLSchema-instance">
   <identification>
      <agreementId>urn:pcccl:agreement:1</agreementId>
      <dataAsset>http://pcccl/dataStream/stream124</dataAsset>
      <dataAssetProvider>http://pcccl</dataAssetProvider>
      <dataAssetConsumer>urn:tuwien:infosys</dataAssetConsumer>
      <creationDate>2012-01-19T22:20:00Z</creationDate>
      <dataAgreementExchangeService> http://sod.infosys.tuwien.ac.at:7101/services/
          jersey/DAES</dataAgreementExchangeService>
      <agreementStatus>AGREED</agreementStatus>
   </identification>
   <extension>
      <agreementReference agreementSchema="urn:pcccl:adcm" category="contract">
         <content>http://sod.infosys.tuwien.ac.at:7101/services/jersey/DAES/da/
             references/retrieve/OpenBuildingCO2.rdf</content>
      </agreementReference>
   </extension>
</ns0:dataAgreement>
```

## 7 Conclusions and future work

Although various data marketplaces and DaaS emerge and provide multitude sets of data, data contracts associated with these data so far are mainly written in textual form for human beings. Furthermore, what constitutes data contracts has not been deeply investigated. In this paper, we analyse data contracts in DaaS and data marketplaces in detail. We have developed an initial abstract data contract model that can be used by different communities to specify conditions applied to data provided via DaaS. Our approach for supporting the definition of data contracts that takes into account diverse types of data terms is based on the community model. Based on our data contract model, we have presented some possible methods and defined guidelines to develop an application for data contract compatibility evaluation for data composition.

Our methods and models for specifying and evaluating data contracts surely are just at an early stage. Our future plan is to continue with our prototype and start to test it in a larger setting. Moreover, we are working on the full integration of our data contract framework with the description model for DaaS and data marketplaces (Vu et al., 2012) and into data selection and composition framework.

Finally, we are currently defining guidelines to develop applications for data contract selection and aggregation/composition starting from our previous works on service contracts (Comerio et al., 2009a, 2009b).

## Acknowledgements

This paper is a revised and expanded version of a paper entitled 'On analysing and developing data contracts in cloud-based data marketplaces' presented at APSCC 2011, Jeju, Korea (South), December 12–15, 2011. This work is partially supported by the Vienna Science and Technology Fund (WWTF), project ICT08-032, by the Pacific Controls Cloud Computing Lab, and by the SAS Institute srl (Grant Carlo Grandi).

## References

AKN (2011) 'AKN data sharing policy', available at http://www.avianknowledge.net/content/about/akn-data-sharing-policy (accessed on 25 July 2011).

Batini, C. and Scannapieco, M. (2006) *Data Quality: Concepts, Methodologies and Techniques. Data-Centric Systems and Applications*,Springer-Verlag Berlin Heidelberg.

Batini, C., Cappiello, C., Francalanci, C. and Maurino, A. (2009) 'Methodologies for data quality assessment and improvement', *ACM Comput. Surv.*, Vol. 41, No. 3, pp.1–52.

Blau, B., Michalk, W., Neumann, D. and Weinhardt, C. (2008) 'Provisioning of service mash-up topologies', in *Proceedings of the 16th European Conference on Information Systems (ECIS)*, Galway, Ireland, June.

BMP (2011) 'Building model products', available in Microsoft Azure – https://datamarket.azure.com/dataset/bfa417be-be79-4915-82c7-efae9ced5cb7 (accessed on 21 August 2011).

Cambronero, E., Okika, J.C. and Ravn, A.P. (2007) 'Analyzing web service contracts', in *Proceedings of the International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies, IEEE Computer Society*, Washington, DC, USA, pp.149–154, available at http://portal.acm.org/citation.cfm?id=1339259.1339292.

CCAS (2011) 'Creative common – attribution-sharealike 2.0 generic (cc by-sa 2.0)', available at http://creativecommons.org/licenses/by-sa/2.0/ (accessed on 21 August 2011).

CED (2011) 'Consumer expenditure data', available in Microsoft Azure – https://datamarket.azure.com/dataset/1a89a286-6ff2-4cc1-a215-ea4370259049 (accessed on 21 August 2011).

Comerio, M., De Paoli, F. and Palmonari, M. (2009a) 'Effective and flexible nfp-based ranking of web services', in *Proc. of ICSOC/ServiceWave '09*, Stockholm, Sweden, 23–27 November, pp.546–560.

Comerio, M., Truong, H-L., De Paoli, F. and Dustdar, S. (2009b) 'Evaluating contract compatibility for service composition in the seco2 framework', in *Proc. of ICSOC/ServiceWave '09*, Stockholm, Sweden, 23–27 November, pp.221–236.

Committee on Licensing Geographic Data and Services, National Research Council (Ed.) (2004) *Licensing Geographic Data and Services*, The National Academies Press, USA.

Dalheimer, M. and Pfreundt, F-J. (2009) 'Genlm: license management for grid and cloud computing environments', in Cappello, F., Wang, C-L. and Buyya, R. (Eds.): *CCGRID. IEEE Computer Society*, pp.132–139.

Deelman, E., Singh, G., Su, M-H., Blythe, J., Gil, Y., Kesselman, C., Mehta, G., Vahi, K., Berriman, G.B., Good, J., Laity, A.C., Jacob, J.C. and Katz, D.S. (2005) 'Pegasus: q framework for mapping complex scientific workflows onto distributed systems', *Scientific Programming*, Vol. 13, No. 3, pp.219–237.

Di Lorenzo, G., Hacid, H., Paik, H-y. and Benatallah, B. (2009) 'Data integration in mashups', *SIGMOD Rec.*, Vol. 38, No. 1, pp.59–66.

Fahringer, T., Prodan, R., Duan, R., Nerieri, F., Podlipnig, S., Qin, J., Siddiqui, M., Truong, H-L., Villazon, A. and Wieczorek, M. (2005) 'ASKALON: a grid application development and computing environment', in *6th International Workshop on Grid Computing (Grid 2005)*, IEEE Computer Society Press, Seattle, USA, November.

FDD (2011) 'Freebase data dump', available in Amazon Public Dataset – http://aws.amazon.com/datasets/2320?_encoding=UTF8{{{{{&}}}}}jiveRedirect=1 (accessed on 21 August 2011).

Fung, B.C.M., Trojer, T., Hung, P.C.K., Xiong, L., Al-Hussaeni, K. and Dssouli, R. (2011) 'Service-oriented architecture for high-dimensional private data mashup', *IEEE Transactions on Services Computing* Vol. 99, (preprints).

Götze, J., Fleuren, T., Müller, P. and Schwantzer, S. (2010) 'License4grid: adopting drm for licensed content in grid environments', in Brogi, A., Pautasso, C. and Papadopoulos, G.A. (Eds.): *ECOWS*, IEEE Computer Society, pp.19–26.

Gangadharan, G.R. and D'Andrea, V. (2006) 'Licensing services: Formal analysis and implementation', in *Proc. ICSOC*, Vol. 4294 of Lecture Notes in Computer Science, Springer, pp.365–377.

Gangadharan, G.R., Truong, H.L., Treiber, M., D'Andrea, V., Dustdar, S., Iannella, R. and Weiss, M. (2008) 'Consumer-specified service license selection and composition', in *Proc. ICCBSS*, IEEE Computer Society, pp.194–203.

GBIF (2011) 'Data usage agreement – gbif (global biodiversity information facility)', available at http://data.gbif.org/terms.htm (access on 21 August 2011).

Hoyer, V. and Fischer, M. (2008) 'Market overview of enterprise mashup tools', in *ICSOC '08: Proceedings of the 6th International Conference on Service-Oriented Computing*, Springer-Verlag, Berlin, Heidelberg, pp.708–721.

Iannella, R. (2002) 'Open digital rights language (odrl) version 1.1', available at http://www.w3.org/TR/odrl/ (access on 21 August 2011).

Keller, A. and Ludwig, H. (2003) 'The WSLA framework: Specifying and monitoring service level agreements for web services', *J. Network Syst. Manage.*, Vol. 11, No. 1, pp.57–81.

Lee, K., Jeon, J., Lee, W., Jeong, S-H. and Park, S-W. (Eds.) (2003) 'QoS for web services: requirements and possible approaches', W3C Technical Report, November, available at http://www.w3c.or.kr/kr-office/TR/2003/ws-qos/ (accessed on 21 August 2011).

Liu, X., Hui, Y., Sun, W. and Liang, H. (2007) 'Towards service composition based on mashup', in *Proc. IEEE SCW)*, IEEE Computer Society, pp.332–339.

Ludäascher, B., Altintas, I., Berkley, C., Higgins, D., Jaeger, E., Jones, M.B., Lee, E.A., Tao, J. and Zhao, Y. (2006) 'Scientific workflow management and the Kepler system', *Concurrency and Computation: Practice and Experience*, Vol. 18, No. 10, pp.1039–1065.

Maximilien, E.M. and Singh, M.P. (2002) 'Conceptual model of web service reputation', *SIGMOD Rec.*, December, Vol. 31, pp.36–41.

Miller, P., Styles, R. and Heath, T. (2008) 'Open data commons, a license for open dataCopyright is held by the author/owner(s)', *LDOW2008*, 22 April, Beijing, China, available at http://events.linkeddata.org/ldow2008/papers/08-miller-styles-open-data-commons.pdf.

ODCAL (2011) 'Open data commons attribution license', available at http://opendatacommons.org/licenses/by/ (accessed on 25 July 2011).

OGL (2011) 'Open government license', available at http://www.nationalarchives.gov.uk/doc/open-government-licence/ (accessed on 25 July 2011).

ONIX-PL (2011) Available at http://www.editeur.org/21/ONIX-PL/ (accessed on 21 August 2011).

PCS (2011) 'The pacific controls galaxy', available at http: //pacificcontrols.net/products/galaxy.html (accessed on 8 August 2011).

Pipino, L.L., Lee, Y.W. and Wang, R.Y. (2002) 'Data quality assessment', *Communications of the ACM*, Vol. 45, No. 4, pp.211–218.

Ran, S. (2003) 'A model for web services discovery with qos', *SIGecom Exch.*, Vol. 4, No. 1, pp.1–10.

Simmhan, Y., Barga, R., van Ingen, C., Lazowska, E. and Szalay, A. (2009) 'Building the trident scientific workflow workbench for data management in the cloud', in *Advanced Engineering Computing and Applications in Sciences, ADVCOMP '09, Third International Conference on*, October, pp.41–50.

TCST (2011) 'Twitter census: Stock twittes', available at http://www.infochimps.com/datasets/twitter-censusstock-tweets (accessed on 21 August 2011).

Truong, H.L. and Dustdar, S. (2009) 'On analyzing and specifying concerns for data as a service', in Kirchberg, M., Hung, P.C.K., Carminati, B., Chi, C-H., Kanagasabai, R., Valle, E.D., Lan, K-C. and Chen, L-J. (Eds.): *APSCC*, IEEE, pp.87–94.

Truong, H-L., Dustdar, S., Gotze, J., Fleuren, T., Muller, P., Tbahriti, S-E., Mrissa, M. and Ghedira, C. (2011a) 'Exchanging data agreements in the daas model', in *Services Computing Conference (APSCC), 2011 IEEE Asia-Pacific*, December, pp.153–160.

Truong, H-L., Gangadharan, G., Comerio, M., Dustdar, S. and De Paoli, F. (2011b) 'On analyzing and developing data contracts in cloud-based data marketplaces', in *Services Computing Conference (APSCC), 2011 IEEE Asia-Pacific*, December, pp.174–181.

Turi, D., Missier, P., Goble, C.A., Roure, D.D. and Oinn, T. (2007) 'Taverna workflows: syntax and semantics', in *Third IEEE International Conference on e-Science and Grid Computing (e-Science 2007)*, pp.441–448.

USCPI (2011) 'US consumer price index – 1913 to current', available in Microsoft Azure – https://datamarket.azure.com/dataset/26058d69-5cad-4a7c-9a14-a21a0c40de86 (accessed on 21 August 2011).

Voorhoeve, M. and van der Aalst, W.M.P. (1997) 'Ad-hoc workflow: problems and solutions', in *Proc. DEXA Workshop*, pp.36–40.

Vu, Q.H., Pham, T.V., Truong, H-L., Dustdar, S. and Asal, R. (2012) 'DEMODS: a description model for data-as-a-service', in *The 26th IEEE International Conference on Advanced Information Networking and Applications (AINA-2012)*, IEEE Computer Society Press, 26–29 March.

Wang, C., Balaouras, S., Staten, J., King, O. and Nelson, L. (2010) 'Compliance with clouds: caveat emptor', Tech. rep., Forrester Research Report.

Wang, X., Vitvar, T., Kerrigan, M. and Toma, I. (2006) 'A qos-aware selection model for semantic web services', in *Proc. ICSOC*, Vol. 4294 of Lecture Notes in Computer Science, Springer, pp.390–401.

## Notes

1 This paper substantially extends our previous paper presented in APSCC 2011 (Truong et al., 2011b). In addition to the revised concept for the whole paper, we have detailed the design of our abstract contract model, introduced data contract compatibility evaluation, and possible steps in making decisions on utilising data based on data contracts. In addition, we also extended the experiments to illustrate contract compatibility evaluation and how data contract can be integrated with DaaS and data agreement exchange techniques.

2 E.g., the data in http://www.openstreetmap.org/ is governed by Creative Commons Attribution-ShareAlike 2.0.

3 For the sake of simplicity, we remove real URIs in many cases.