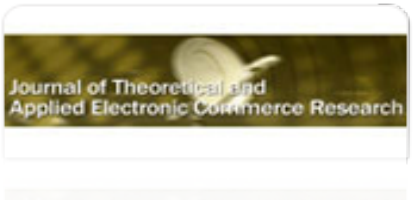

Journal of Theoretical and Applied Electronic Commerce Research

E-ISSN: 0718-1876

ncerpa@utalca.cl

Universidad de Talca

Chile

Hofman, Wout; Rajagopal, Madan

A Technical Framework for Data Sharing

Journal of Theoretical and Applied Electronic Commerce Research, vol. 9, núm. 3, septiembre-, 2014, pp. 45-58

Universidad de Talca

Curicó, Chile

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 9 / ISSUE 3 / SEPTEMBER 2014 / 45-58

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762014000300005

A Technical Framework for Data Sharing

Wout Hofman[1] and Madan Rajagopal[1]

[1]Dutch National Institute of Applied Science, Technical Science Department, P.O. Box 5050, 2600 GB Delft, The Netherlands, ((wout.hofman), (madan.rajagopal))@tno.nl

Received 12 July 2013; received in revised form 31 January 2014; accepted 12 March 2014

Abstract

Open data is receiving considerable attention because of its potential for public and private sector innovation. Various governments have the policy of providing data sets to the public via open data portals. Data sets are published in a format defined by a source, which makes it difficult to discover a useful data set and requires user interpretation of structure and semantics. Meta data insertion and semantic annotation address these problems, but are not yet widely implemented for open data. Also privacy issues and commercial sensitivity have to be addressed, leading to (technical) interventions like access restrictions and billing functions. Users might also want to be informed only of data changes with publish/subscribe functions to increase the quality of decisions based on large data sets. Data transformation, billing, security, monitoring, and publish/subscribe functionality has to be associated with data sets that are available via Application Programming Interfaces. Application Programming Interface management platforms providing this type of functionality are central to implementing open data. This paper analyses required functionality for data sharing considering the above mentioned requirements and matches these requirements with functionality of available platforms.

Keywords: Open data, API management, Linked data, Data plataforms, Big data

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 9 / ISSUE 3 / SEPTEMBER 2014 / 45-58

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762014000300005

1 Introduction

Data is of growing importance to our society and economy. Most literature takes a so-called push approach in which data availability will contribute to public – and private sector innovation [29]. Jetzek et.al [17] have constructed and validated a model for value generation by open government data, where they have defined value from an economical and social perspective. In their approach, data user requirements are not specified, they either stem from innovation, participation, efficiency, and transparency. Data is provided to potential data users as is according a stage model [3]. Such a data push mechanism still has several technical aspects that need to be addressed [17], [29]. Each data provider is autonomous and has its particular implementation of processes, IT solutions, etc., although many processes and solutions of for instance municipalities could be identical as these municipalities manage the same data. Different implementations of similar processes lead to differences in data sets. A data user has to deal with these differences in data sets, e.g. different syntax and (slightly) different semantics, for realizing solutions based on various data sets. Semantics of data sets can also be specified implicitly, e.g. by element names of for instance XML (eXtensible Markup Language) documents. Although a software developer will have no problem of implementing these differences, they may lead to different solutions and thus potential inconsistencies in services provided to end-users. Another issue of a data push mechanism is that a data user has to process large amounts of data (*big data*) of different sources to derive information out of that data. In crisis management applications, for instance, a data user requires a direct overview of potential problems to quickly decide on proper action. A requirement of a data user would be to receive only particular events and/or combine events of different data providers to take proper action.

There are over 150 open data government portals (site 1). These portals publish data sets, either grouped under labels like the data provider, a particular subject and/or the geographic area covered by the data set (site 2). Some of these data sets have additional metadata like the last update provided, the volatility of the data set, etc. Labels and metadata are all types of metadata that can be used for discovery of proper data sets. The Dublin Core Metadata Initiative (DCMI, site 3) provides a set of metadata tags that can be used.

This paper addresses functionality for data publication by data providers and retrieval of those data sets by data users by taking an empirical approach. Functional requirements for open data platforms are formulated addressing technical barriers identified in literature. As the literature on technical barriers is insufficient to identify required functionality, additional literature with respect to interoperability is reviewed to identify technical solutions. Also, the concept *platform* needs further refinement in a data value chain between data providers and - users, since there are many platform variants with different functionality [6]. Esmeijer et.al [10] provide a data value chain identifying data generation and - collection, -preparation, - integration, - storage, - analysis, - visualization, data driven action, and data governance and security. This contribution focusses on the stages from collection to storage and thus will not consider for instance data visualization platforms. The stages considered in this contribution enable system-to-system data sharing, without providing details of the functionality of these systems. Each of these systems can have one of two roles, namely a data provider or – user. A visualization platform is an example of a data user system, but an app running on a smart device or an internal IT system supporting business processes of end-users are examples of systems with either or both the role of data user or - provider. Based on the evaluation of the stage model [5] that shows Application Programming Interfaces (APIs) are the most popular means of implementing open data, API management platforms are considered that have no data analysis and - visualization functionality. Data analysis and – visualization are thus outside the scope of this contribution.

The structure of this paper is as follows. First of all, section 2 describes the literature review on functional requirements for open data functionality and provides criteria for selecting data platforms for further analysis. The functionality identified in literature and provided by available platforms, is the basis for a technical framework for data sharing (section 3). Section 4 applies the framework in analysing the platforms identified in section 2, whereas section 5 identifies potential gaps in required functionality. Finally, the paper will give conclusions and provide steps for further research.

2 Methodology

This section presents a literature review of technical functionality for open data and criteria for selecting platforms that are further analysed. The literature review and functionality of available platforms will be the basis for a technical framework for data sharing as presented in the next section.

2.1 Literature Review

Literature on required platform functionality for open government data is limited; most papers refer to potential technical barriers for implementing open government data. However, data sharing is not limited to open data. In this respect, a large selection of scientific papers and books are available that address technical aspects of interoperability. Also additional literature is available addressing data virtualization. This section briefly elaborates literature addressing technical barriers to open government data, interoperability, and data virtualization in the context of cloud computing.

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762014000300005

Zuiderwijk et.al [29] identify the following technical barriers to open data: difficult to process data sets, lack of good Application Programming Interfaces (APIs), data is not in machine readable format, sophistication required for linking and combining data, difficulties in changing formats, and no integrated tool set to combine data of different data providers. Jetzek et.al [17] mention technical aspects like infrastructure accessibility, storage and aggregation, integration, and analysis. Schema heterogeneity and lack of consistency are hindering value creation, and an increasing need for semantic interoperability with ontologies, data management, and identity resolution with multiple platforms are required. A value chain of storage, access, combination, and analysis is required [16]. From a big data perspective, others identify technical issues for storage and transport of data [18], but also issues like compliance, security, and data ownership. Although Berners-Lee [4] has developed a model for publishing data, another study [5] identifies that this model is not always fully applied, hindering the implementation data combination and – fusion and barriers with respect to data quality, relevance, privacy.

Technical barriers for open data are addressed by interoperability in heterogeneous systems, since interoperability not only deals with semantics like open data does, but interoperability also addresses pragmatics of data sharing [20] Interoperability also addresses schema heterogeneity [12] and proposes various technical solutions. Chituc et.al [8] and Blommestein [6] address for instance seamless interoperability as *the ability to share interoperability metadata before business data is actually shared*. Metadata comprises aspect like semantics, syntax, and communication protocols; they allow communicating systems to establish transformation functions for sharing data. Messaging or web services, supported by for instance APIs, are the most used paradigm for data sharing in Business-to-Business (B2B) interoperability. Erl [9] presents the implementation of a so-called Enterprise Service Bus (ESB) supporting various integration patterns like adapters for transforming one interface into another [12]. To reduce interoperability maintenance costs, a common information model expressing semantics of all interfaces can be used to configure an ESB. Adapters transform between external formats and this common information model, thus reducing the number of transformations. Peristeras et.al [24] identify the need for the development of ontologies in the Ontology Web Language (OWL) for expressing semantics of shared data and actually sharing the data in Resource Description Framework (RDF) format. These ontologies can serve as a common information model for interoperability. However, expressing semantics by ontologies also provides the ability to request whatever data one would like to obtain with a query language for RDF (SPARQL). Ontologies can be the interface to data users: a data user is able to express its particular data requirement in terms of an ontology. By providing a so-called SPARQL endpoint to ones public data sets and providing data in RDF format, a data user is able to retrieve any required data of a set.

With respect to data storage, the cloud computing model [7] allows computation – and data virtualization implying end-users are not aware of processing - and data storage facilities. From a computational perspective, data can be stored on servers of different providers. However, cloud computing solutions have not been developed with interoperability in mind, they are closed systems with respect to data storage. The creation of a common information space for data analysis, processing, and exchange can be facilitated if the cloud computing model is developed following the semantic approach that focuses on semantic interoperability [22]. Loutas et.al [22] argue to construct Cloud APIs for interoperability in this common information space. These types of issues are similar to data sharing from an open government data perspective. Similarly, a high-level market oriented cloud architecture [7] might be considered for open data, that also incorporates functionality like pricing, accounting and event dispatching. The latter is supported by a publish/subscribe mechanism [9].

The technical barriers identified by Zuiderwijk et.al [29] reflect a data user's perspective, namely the current status of available open data as perceived by a potential user. One could conclude that a decoupling of data provider and user is required, where a data set of a data provider should have particular features to be useful for a data user. As literature review learns, these features relate to semantics and syntax of data. In terms of the stage model for data publication introduced by Berners-Lee [4], data must be structured in a syntax and the semantics of the data has to be clear and concise for data processing by IT systems of data users. Using syntaxes like RDF enables the ability for linking and combining data as identified in Zuiderwijk et.al [29] as described also by Berners-Lee [4], but requires data providers to establish and maintain those links instead of duplicating data themselves and combining it with their data. Scheme heterogeneity requiring data transformation is also still an issue that partly requires human intervention [6]. Semantic models represented as ontologies to address schema heterogeneity can be found in open data -, interoperability – and cloud computing literature. Opening data requires still an Application Programming Interface (API) that can be semantically annotated by an ontology, or a SPARQL endpoint that allows a data user to abstract from different data providers with their APIs and express data requirements in terms of ontologies. The latter requires functionality like data matching and – linking [28]. Implementation of these APIs by the Representational State Transfer (REST) protocol that is based on Uniform Resource Identifiers (URIs) of the http-protocol is the most commonly used approach.

2.2 Selection of Platforms for Analysis

There are many interpretations of *platform*. Operating systems and computer hardware are considered platforms for software applications and data storage, but similarly a transport infrastructure can be considered as a platform for vehicles (Site 4). With respect to data sharing, a platform should provide sufficient functionality to data users for accessing data and data providers to publish their data sets. From literature review, data publication should be with Application Programming Interfaces by which data can be obtained. The semantics of this data also needs to be clearly and concisely specified by for instance ontologies (see previous section), with as less as possible different

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762014000300005

interpretations. Basically, a platform has to support data sharing between heterogeneous computer systems of different organizations [8] with APIs. A software application, for instance a visualization platform or an app running on a smart device, are examples of data users. Also specific software tools that are able to gather data from different sources and provide visual analytics like the Data Delivery Engine of Data Market are outside the scope, although a company like Data Market serves as a visualization platform for open data. These software applications are outside the scope of a platform. Furthermore, tools for temporary data storage of data sets like Allegrograph, MonetDB, and MongoDB are not considered, but there could be a requirement to implement these tools as a component of a platform. These types of databases are able to store for instance RDF data and quickly answer queries on these data sets, although Allegrograph also states to support semantics of that data.

API management is therefore one of the basic criteria for selecting a platform. From this perspective, data management platforms like CKAN and Socrata do not meet the criterion, although most open data portals of governments use CKAN. Therefore, this paper also investigates functionality of data management platforms.

Other important criteria for platform selection are:

- *API independent*. A platform should be independent of a particular API and its semantics. A platform like UrbanOS focussing on smart city applications is thus not considered.

- *Software solutions*. Each organization should be able to implement a platform. There are also service providers offering platform functionality, but these are not considered in the context of this paper. A software solution should be available that can be implemented by any organization.

- *Open source*. A platform should be available as open source to ease its adaption to support particular functionality. Platforms like IBM CastIron, Layer 7, Virtuoso, and Talend are therefore not considered in this paper.

- *Data provider and – user support*. A platform should provide support to both data providers and data users and serve as a decoupling point between both of them (although a data provider can also be a data user and vice versa). Adapters like DataTank that can be configured by semantics, are thus not considered.

Considering these criteria, only a few software solutions for implementing a platform are available, namely WSO2 API Management Platform and API Axle. To provide a broader overview of functionality, this paper also investigates functionality of a limited number of other commercial API management platforms. Additionally, an adapter like Datatank is considered while adapters are a well-known technology to implement for instance web-services [9], [14].

3 A Technical Framework for Data Sharing

This section presents a technical framework for data sharing between data providers and – users. The technical framework consists of a number of functions derived from a data value chain between data providers and – users. Firstly, the data value chain is presented,, secondly, the technical framework and finally organizational deployement is discussed. The data value chain assumes a sort of linear process in which a data provider makes data available to a data user. However, the data value chain can have a decoupling point, as will be discussed in the next section.

3.1 Data Value Chain Functionality

Esmeijer et.al [10] present a data value chain for sharing open data between a data user and - provider. It consists of processes like data generation, - collection, - preparation, - integration, - storage, - analysis, - visualization, data driven action, and data governance and security. Combining this data value chain with literature review leads to the following functionality, in which data visualization is considered out of scope (see section 1 of this contribution):

- Data governance and security: the ability to distinguish between open data, data with access restrictions, and data that is not available outside a data source.

- Data source management: available data sets of a data provider with its APIs and quality features expressed by metadata.

- Data preparation: data enhancement and enrichment to increase its usefulness for data users.

- Temporary data storage: a decoupling point between data providers and – users for storing to improve processing on behalf of a data user request.

- Data retrieval: functionality to retrieve required data, which might include integration of data from one or more data sources (also called: data combining, Zuiderwijk et.al [29]).

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 9 / ISSUE 3 / SEPTEMBER 2014 / 45-58

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762014000300005

- Data driven action: the ability to take action upon particular data changes.

- Analytics and support: analysis of the shared data, including accountability.

This functionality is discussed in more detail hereafter, identifying relevant software components. A technical framework of these software components will be given in section 3.2. Whereas data governance and security and data source management are settings for every data set, the other processes support actual data sharing. Data governance and security and data source management is thus only done once for each data set, whereas they provide information to all other processes.

Data governance and security offer functionality to a data provider to control access and use of data. Data governance allows a data provider to qualify data as open, restricted to (roles of) data users, or restricted for internal use. Whereas open data is accessible to anyone, data with access restrictions requires identification, authentication, and authorization mechanisms. Filtering is an example of an intervention supporting data governance and allows to hide specific data to particular data users on attribute level, called attribute-based access control [25]. It allows a data provider to provide a common API for a data set, filtered for instance on commercial - or private sensitive data at data preparation like *price* or *person name*, with different output to different data users. Filtering tailors one common API of a data set to user requirements, thus virtually resulting in a specific API per (role of a) data user. In case a data user is able to formulate query on a semantic model, e.g. by SPARQL, data governance and - security needs further research. A semantic model can potentially express particular access restrictions by a (role of) data user. This semantic model can in its turn be expressed in the semantic model of a data set, thus constructing a networked ontology for authorization. For instance, Dutch Customs Authority is allowed to access data of all trade flows to the Netherlands from a foreign trader, but not those flows of that trader to other countries [26]. Homomorphic encryption [13] is another intervention mechanism that allows querying encrypted data from different data providers, without decrypting that data. These types of interventions require a clear data governance model, which is yet lacking.

Data governance is not only applicable to a data provider. A data user can also implement particular governance mechanism according agreements with one or more data providers, e.g. never to provide data obtained from a data provider to another data user without that provider's consent or to qualify privacy sensitive data obtained from a data provider as restricted to internal use by that data user only. The latter could be applicable to for instance statistics authorities that obtain private or commercial sensitive to produce statistical overviews. A data user should be able to provide proof of his behaviour, e.g. by implementing an audit trail and log of data obtained from various data providers.

Data source management considers a registry of data sets of data providers, the data quality of each set and its accessibility. Central or distributed registries can be composed with different (open source) solutions [9]. For instance, Kademlia (site 4) is used to implement a distributed registry in peer-to-peer networks. Each data provider has to be able to register data sets with their API(s), add quality features to an API by means of metadata, and annotate each API with semantics, preferably via a common information model or ontologies. Quality features are implemented by metadata [6] and represent aspects like volatility and completeness [2], but also the syntax in which data is produced. Metadata insertion is done once for each API and is automatically added to data when it is retrieved during data preparation. Not all data providers will have an API [5], in which case data needs to be uploaded to a temporary data store and an API for that data can be created. A data user can access this data from a temporary store via its API. Data source management can also imply the configuration of transformation, required for actual data sharing (see the next process).

A data user can also utilize a data source management function, but from the perspective of data retrieval (see further).

Data preparation provides functionality to offer the data in a format required for retrieval. Transformation, filtering and cleaning, verification, and metadata insertion are part of data preparation. Data can be *transformed* to a common format, see for instance [17], [24], or a format required by a data user. A data provider can also submit the data as is, implying that potentially a data user will be in need of transforming the data. Transformation is required for matching and combining data of different data sets or providing a generic query interface according a common information model or a set of ontologies. Transformation also validates the correctness of the data format and potentially the data by comparing with allowed data formats and constraints formulated by ontologies. The comparison with allowed data formats can be based on validating against an external source. For instance, time and place are provided in proper formats and contain agreed values like whether a place or address in the Netherlands actually exists. The address is validated by for instance a web service call to an address registry. In case of validation errors, particular metadata with respect to data quality can be set. By APIs provided by conversion algorithms, data transformation is also able to convert data types, e.g. the notation of time can be changed and a conversion between currencies and temperatures can be supported. *Filtering* and *cleaning* are other functions of data preparation and are an intervention mechanism supporting data governance (see before). These functions remove data from source data that is private, commercial, or otherwise sensitive, but also apply any access restrictions to a particular data user (see data governance). Examples are to remove commercial sensitive trading relations, container content data that is not required by a carrier, and person identification. These trading relations and container content data are required by for

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762014000300005

instance customs and will thus not be filtered [26]. Cleaning improves data quality by removing duplicate data, validate data on its consistency, and so on. *Verification* is the validation of completeness and correctness. The latter is also a function of transformation, but verification is required in case transformation is not applied. The last function for data preparation is *metadata insertion*. Whereas data source management also provides metadata for a given data set, the actual metadata can be added to any data retrieved. Metadata comprises for instance the specification of data quality in terms of aspects like completeness, consistency, and velocity [2]. Velocity is of importance to data relevance.

Temporary data storage is required to meet data users' requirements for data combination and fusion or to support particular quality of service of a data provider. In principle, data storage solutions need not to be known by supporting APIs for data access [5]; data is accessed at its source. From a performance, availability, and accessibility perspective, temporary data storage by a platform needs to be considered. Data storage needs to be combined with data management functionality that keeps track of changes (velocity) provides functionality with respect to data quality (e.g. completeness, consistency).

Data retrieval supports data users in accessing and retrieving data. A data user can real time select and initiate an API for a data provider, compose its own API based on available APIs or formulate a query on a common format known by the data provider, or configure the APIs and supporting functionality before actually retrieving data. In case APIs for data retrieval are statically configured, a data user can configure several functions like transformation, data fusion, etc. API composition requires *data fusion, - combination, and – linkage*. Data fusion and combination composes output based on two or more inputs. A data users' API calls two different APIs and combines their results. Data linkage implies the existence of links in data to other data [15]. Based on a link retrieved from data set, data of another set is retrieved, e.g. a trading relation in a data set provided by a trader to customs could serve as a link for customs to retrieve data of that other trading relation or to validate trustworthiness of that other trader against an external data provider. The end result of data linkage might still be the combination of data from two or more resources. Data retrieval needs to consider *precision* and *recall*. Precision indicates that all retrieved results meet a data user's query or API and recall the completeness of the results. Thus, the result only presents data that meets a query (precision) and all available data meeting a query is retrieved (recall) to provide a data user with the best and complete data set meeting its query. Of course, completeness of the retrieved data also relates to data quality: if the available data is incomplete, a data user will never have the optimal data set meeting its query. In case there is insufficient data meeting a query, a ranking algorithm can be applied indicating how the results meet the query [21]. The latter is specifically the case if data semantics of a data user differs from that of a data provider and a real time query is formulated. So-called genetic algorithms can be applied, utilizing for instance databases like Wordnet [1].

Data retrieval can be considered from a historical or real-time perspective, but also a geo-perspective can be taken. Precision and recall can be supported by metadata, e.g. a semantic model of one or more APIs might serve as a means to locate the proper data set and initiate an API to retrieve data from that set, and various algorithms and external data providers providing input to these algorithms. From the perspective of data governance, an audit trail needs to be available (see before).

Data driven action requires a data user to take proper action on data retrieved from one or more data providers. Since potentially many data providers are able to provide data, either static or real time data, the amount of data provided to a data user requires either data analysis or restricted data retrieval based on data changes. Event mechanisms [19] with publish/subscribe functionality [9] can be used to inform data users of any relevant data changes. These mechanisms require implementation of data management by a data provider to detect relevant data changes and generate events to subscribers.

Analytics and support control quality of service and monitor data sharing between a user and one or more data providers, both from a data user and – provider perspective. Control of quality of service is for instance by providing different performance and availability levels to data users, e.g. offering a particular response time for an API call. By monitoring every query or API call, a data provider can gain insight on its usefulness and decide to delete the API in case it is not used. Analytics and support can thus be used for API lifecycle management and provide input to billing and accounting [7]. Analytics and support also provides an audit trail for accountability purposes. Accountability implies that a data provider can prove and also analyse that particular data is shared according access control restrictions and a data user can prove and analyse the data that is obtained from one or more data providers in accordance with agreed governance mechanisms of that user.

3.2 Functional Components for Data Sharing

The data value chain identifies a set of functional components for a data provider and – user. We group these components into *Registration* and *Adapte*r like shown in figure 1. This grouping is identical to the approach taken for web services that distinguishes between registries and adapters (Erl [9]). Registration considers data governance and – source management and can be done once for each data set. The latter registries allow a data provider to register its API and apply intervention mechanisms to support data governance and security mechanisms for a given (group of) user(s). Of course, a data provider can also be a data user and vice versa and data governance rules of a provider can differ from that of a user. Data governance rules can be expressed as behavioural rules agreed and applied by providers and users (see before).

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 9 / ISSUE 3 / SEPTEMBER 2014 / 45-58

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762014000300005

We distinguish a data provider and – user adapter. A provider adapter supports functionality like data transformation, access control and verification which covers filtering, cleansing, and data enhancement by adding metadata if required. The figure also shows a data management function for analysing state changes of data and generating events to those data users that have subscribed to these state changes. A data provider can also generate events itself, those have to be processed by data management to determine the subscription. A publish/subscribe functionality can also be distributed, utilizing existing peer-to-peer protocols.

A data user is able to present data in the format required by a system or an end-user. A user adapter is able to combine and link data from more than one data source and supports query formulation in a language like SPARQL.

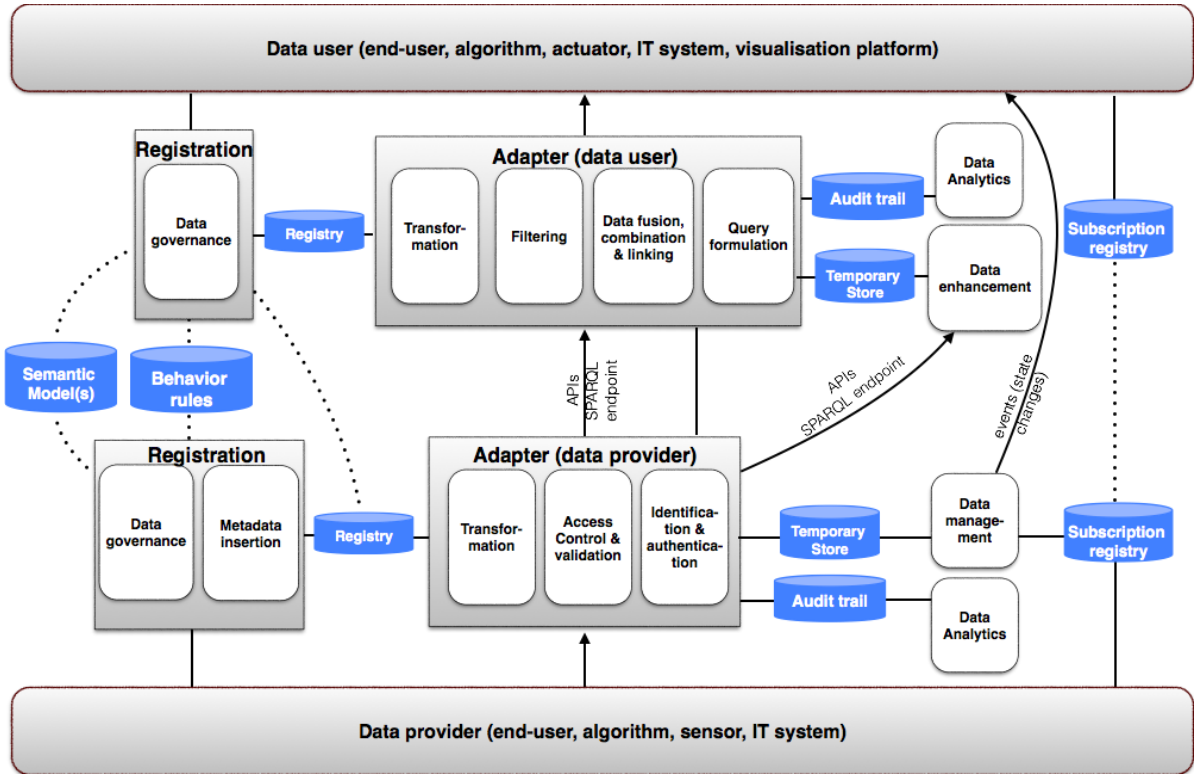

Figure 1: Functionality for sharing open (government) data

In the proposed framework, a data user can also be a data provider and vice versa. It can be a software algorithm computing particular data values and providing these data values as open data, e.g. transformation of degrees Fahrenheit to Celsius, calculating the estimated arrival time based on speed and location of a vehicle, or truck planning based on pick up and drop off locations. These algorithms can be triggered by events or can for instance on timely basis access data providers to retrieve any changes of data triggered by an event. A data user can also be a visualization platform or functionality offered to an end-user on a smart device. The latter is also applicable to a data provider: it can also be an app of an end-user on a smart device or sensor functionality of a smart device providing data.

The previous figure shows a number of data stores that we will explain. Semantic models represented for instance as ontologies are the basis for data transformation, access control and validation, filtering, fusion, linking and combining, query formulation, and data management to offer publish/subscribe functionality. A data provider can have a different semantic model for each data set, but data provider(s) and – user(s) could also agree to use a common model [24]. The same is applicable to behaviour rules: a community can agree to use a set of rules, but a provider and user are also able to formulate their own rules. A Registry contains all details of a data provider, its APIs, data governance - and any quality features for a data set, and data users, the data governance rules they apply, and any composite APIs that they use. A temporary store, an audit trail, and a publish/subscribe registry can be part of an adapter of a data provider and a data user.

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 9 / ISSUE 3 / SEPTEMBER 2014 / 45-58

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762014000300005

3.3 Functionality Deployment

The previous figure does not show a deployment model, e.g. governing functionality of semantics, registries and adapters. Each organization or a platform provider can implement the functionality. In the first case, a peer-to-peer solution based on peer-to-peer protocols between registries and publish/subscribe is realized. In this case, registries, a publish/subscribe, and event mechanism require implementation of protocols like Data Distribution Service (DDS, a standard developed by the Object Management Group used in time critical applications), Advance Message Queueing Protocol (AMQP, supported by for instance Microsoft and open source solutions), and Pubsubhubbub on particular *topics*. APIs provide data representing topics users could subscribe to. There are also services like Pubnub to construct a data ecosystem including pricing strategies, Pusher, and many others.

Any organization can have its registry and a data provider and – user adapter with particular data management and analytics. Agreement between data providers and – users in a peer-to-peer environment could encompass several aspects, e.g. technical interoperability in terms of API protocols, syntactical interoperability of data provided, and semantic interoperability (see also Tolk et.al [28]). REST is for instance an API protocol, utilizing XML or JSON (JavaScript Object Notation) as data sharing syntax. In case there is no agreement on any level, a data user has to deal with protocols and syntaxes offered by a provider. An enterprise like for instance eBay implements its own protocols and publishes them via a registry function. Such an organization provides its APIs based on its particular semantics. In case a data user requires the integration of this functionality in his system, he needs to configure his adapter in the appropriate manner. An organization could also adhere to standards like REST for the API protocol and XML as data sharing syntax. In that particular case, that organization still only has a registry and a data user will have to implement transformation, if required, by interpreting the data structures.

In case organizations agree on common semantics, the semantic models and potentially their syntactical representation needs to be governed. Organizations have to install a central governance organization providing various support activities (see for instance Folmer & Punter [11]). The semantic models can be available by reference and each data provider and – user can use them to configure their adapter.

A platform provider can also provide all functionality: a registry, adapters, and publish/subscribe mechanisms. To optimize its management, such a platform can use a common information model to configure transformations in provider – and user adapter. Data can be stored internally in a common format according the models used. These common models can be based on open, available ontologies. A platform provider can perform harvesting data of different providers and publish particular (composed) APIs on that data to users. A platform provider could also use the harvested data for analysis purposes. The adapters also support API protocol and syntax transformation, thus decoupling data providers and – users. A platform provider needs to address non-functional quality of service requirements.

A platform provider is able to provide a decoupling point between data providers and – users by providing transformations between API protocols, syntax, and semantics. Harvesting data of data providers abstracts from the quality of service offered by a data provider and compose new APIs based on harvested data. Agreed semantic models can also serve as a type of decoupling point, but require the configuration of adapters by each data user and – provider. However, each organization is also able to tailor its adapters to meet particular requirements, thus leading to potentially a large variety of data driven applications.

4 Analysis of Data Platforms

The previous section provided a technical framework for data sharing in which data sets either stored at their source or in a temporary data store of a platform are accessible via an API. Registries supporting API management are the core of a such a data platform and can be implemented distributed or by a platform provider. There is quite a number of API management platforms, a limited set of free source data platforms, and some adapters to retrieve data (see section 2). This section analyzes these platforms from the perspective of the proposed technical framework and, secondly, from an API management approach.

First of all, this section presents the analysis of platforms against the functionality of the data value chain. Secondly, the API management functionality support of platforms is analyzed in more detail.

4.1 Data Value Chain Support by Platforms

The following table presents an overview of available tools and their functionality. The functionality is based on the processes of the data value chain (section 3.1) and functionality offered by solutions, also considering lists of required functionality of so-called data hubs or Data Management Systems (DMS, see for instance Wikipedia). Details are included based on functionality provided by a platform as in some occasions this detailed functionality is common to two or more platforms. Functionality is only listed, if it is supported by more than one platform. Platform

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 9 / ISSUE 3 / SEPTEMBER 2014 / 45-58

This paper is available online at www.jtaer.com
DOI: 10.4067/S0718-18762014000300005

specific functionality is not presented. In case functionality is not clear and requires further research, the functionality is not listed in the tables in this section.

Table 1: Overview of platform support of the technical framework

	API Management Platforms								Data Management Platforms		Adapter
	WSO2 API Manager	3Scale API Management Platform	Apigee	Mashery	API Axle	Mashape	Intel expressway API manager	Vordel API server	Socrata	CKAN	Data Tank
Data Retrieval											
– Modify and enhance				x			x	x	x	x	x
– Combine	x		x								
– Precision and recall	x	x							x	x	x
• Themes	x		x	x					x	x	
• Promote	x	x	x	x					x		
• Geospatial									x	x	
• Data preview									x	x	
• (e.g. social web											
• platforms)									x	x	
– Query formulation - user dashboard	x	x	x	x		x			x	x	
– Events	x										
(publish/subscribe)			x						x	x	
Temporary storage	x	x	x	x							x
Data Preparation											
– Publish	x								x	x	x
– Semantics										?	x
• Metadata											
• Semantic support	x	x	x	x	x			x	x		
Security	x	x	x	x	x	x	x	x	x	x	x
Analytics and support											
– Analytics and reporting	x	x	x			x	x				
– Billing	x										
• end-user billing											
• developer billing	x	x	x	x	x		x	x			
– Performance		x	x					x			
• etc.	x	x	x			x	x		x	x	
• SLA management											
– Developer community	x	x					x				
– API support											
• Any					x				x	x	x
• Limited (e.g. REST or RDF)	x			x			x	x			
– API lifecycle management											

Table 1 shows that all tools have analytics like discussed in section 3. Most API management platforms support information security features and some of them include billing functionality, which makes these tools useful in closed communities or support of commercial data platforms. Since API management platforms do not store data, search and find is on the APIs, e.g. by themes or promote particular APIs. Almost all open data portals of governments use CKAN as data platform, although there are more that provide APIs to their data. It has API management functionality with metadata support, including the ability for end-users to add their metadata. Data Tank, which is an adapter, is the only tool with semantic support, the Data Tank Semantifier, although the functionality is not clear from documentation.

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 9 / ISSUE 3 / SEPTEMBER 2014 / 45-58

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762014000300005

Table 1 shows functionality for supporting non-functional requirements, e.g. flow control, throttling, and management of particular service levels (SLA: Service Level Agreement) reflecting quality of service parameters identified in section 3.1. Furthermore, developer communities can use API management platforms, including lifecycle management of the APIs.

Table 2: API support by API management platforms

	WSO2 API Manager	Apigee	3Scale	Mashery	Apiaxle	Vordex API server	Intel Expressway API Manager
API Discovery							
– Catalog	x	x	x	x		x	x
– Search	x	x	x	x			x
– Provisioning	x	x	x	x		x	
API Security Protocols							
– SSL	x	?	?	?			
– PKI							
– Role Based Access	x	x	x	x		x	
API Identity Management		x					
– API key	x		x	x	x	x	x
– Oauth	x		x			x	x
– SAML	x			x		x	
– Multifactor						x	
– Token translation & mngt					x		x
API Lifecycle Governance							
– Create, publish, delete, update, depreciate	x	x	x	x		x	x
– Versioning	x	?	?	?		x	x
API Analytics & Billing							
– Analytics and monitoring	x	x	x	x	x	x	x
– API billing/invoicing	x	x	x				
API Quality of Service							
– Rate limiting (throttling)	x	x	x	x	x	x	x
– Response caching	x			x	x	x	
– Tier management	x	x	x	x			
Platform Quality of Service							
– Clustering	x						x
– Scalability	x	x	x	x			x
Events, Publish/Subscribe	x						
Open Source	x	?	?		x		

Events supported by publish and subscribe functionality are only supported by one of the API management platforms analyzed, namely WSO2 API Management. This platform offers an Enterprise Service Bus (ESB) with adapters and all types of transformation functions see [9], [12]. Additionally, WSO2 API Management provides business process support functionality. Both functions, ESB and business process management need further study to know if they support data fusion and - linkage respectively.

4.2 Detailed Platform Functionality for API Management

Whereas the previous section provided an analysis of platform functionality in terms of processes in the data value chain, this section takes the perspective of APIs management. Thus, data platforms and adapters are not analyzed in this respect, since these do not provide API management functionality. Table 2 provides an overview of API management support.

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 9 / ISSUE 3 / SEPTEMBER 2014 / 45-58

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762014000300005

Table 2 shows platform analysis by which data value chain processes are reviewed from the perspective of API management. It implies for instance that *data retrieval – precision and recall* is renamed to *API discovery*. In a similar manner, all others are renamed, e.g. *security* is renamed to *API security protocols* reflecting encryption and access control, and *API Identity Management* reflecting identification and authentication of data provider and – user.

Precision and recall required for data retrieval is based on available APIs. Data retrieval cannot be based on common models. In terms of a large amount of data providers accessible via different platforms [17], discovery of the proper data provider can be a burden. The question marks in table 2 indicate that the functionality is available according platform provider documentation and websites, but is not clearly specified.

5 Evaluation and Discussion

The previous sections showed the support of the proposed data value chain and technical framework by available platforms for API management and (temporary) data storage. Table 3 lists platform completeness in terms of data value chain processes, including an indication of open source. Completeness is not only expressed by support of the data value chain, of which details are provided by tables 1 and 2, but also by support provided for installation and deployment, installed base, integration with other components, extendibility with new functionality.

Table 3: Completeness of API management platforms

API platform completeness	WSO2 API Manager	3Scale API Management Platform	Apigee	Mashery	API Axle	Mashape	Intel express way API manager	Vordel API server
Data value chain	x	x	x	x	x	x	x	x
– Data retrieval	x	x	x	x	x	x	x	x
– Data source management	x	x	x	x	x		x	x
– Temporary data storage			x					
– Data preparation	x	x	x	x				
– Security & Identity Management	x	x	x	x	x		x	x
– Analytics	x	x	x	x	x	x	x	x
– Data driven action (events, publish/subscribe)	x							
Support	x	x	x	x				
Installed base	x	x	x	x	x	x	x	x
Extendibility	x							
Open source	x				x			

Table 3 shows that WSO2 API Manager provides the most functionality and is also an open source platform. Extendibility can be used to support temporary data storage. Detailed analysis has learned that the technical framework provided in section 2.2 is not fully supported by any of the platforms, as we will discuss here.

Analysed platforms function as intermediates between a data provider and data user, not necessarily altering data when shared between a provider and user. API management platforms combine the functionality of a registry and monitor (secure) API calls over the platform, data management platforms (temporarily) store data of different sources. Data replication to a platform might potentially lead to a decrease of governance by a data provider and inconsistency with source data. Decrease of data governance is important for sensitive data. As table 3 shows, only security and identity management is supported by platforms. Data governance supported by interventions implemented by a platform can be supported, but are not easy to use. It is up to a platform user to configure for instance a filter as part of transformation functionality for hiding commercial or privacy sensitive data.

Data manipulation functionality like transformation and filtering required for data preparation and - retrieval seem to be implemented only by the ESB and business process management functionality of WSO2 API management platform, but the use of this functionality does not seem to be tailored to open data. The platform does not provide functionality by which a data user or – source is able to (simply) configure particular functionality for data preparation and/or – retrieval. Data link evaluation, data fusion, and data combination are not supported by the analyzed platforms. Possibly, business process management functionality of WSO2 API management can be used for this purpose, but it still requires developers or a platform provider to construct new APIs. Real time API (de)composition by data users is not supported and may also be complex [14]. Data management costs at a source are increased, since each source has to implement a different API for filtering sensitive data. Filtering data of one API providing data to different data users is expected to have lower Total Cost of Ownership (TCO). Further research into the

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762014000300005

application of an ESB combined with business process management functionality is required to determine the ease of use and configuration of these functions. Many ESBs combining these functions, complemented with an Event Driven Architecture, have already been applied for web services [9].

Precision and recall are important for data retrieval [21]. Data searches are either supported by API discovery or portals constructed on top of these platforms with for instance themes and faceted search mechanisms based on a data set classification provided by metadata. Particular functionality to increase precision and recall for data discovery, for instance geo-fencing or filtering, needs to be developed by a data user or a platform provider upon request of data users. Another aspect of precision is support of events with publish/subscribe functionality. Only WSO2 API management platform seems to implement this function, thus allowing a data user to subscribe to data updates of one or more sources. It requires data management functionality of data providers that is able to generate updates. Since most open government data is rather static, these mechanisms are not provided. Particular applications like traffic management might support these mechanisms, but these are mostly not considered as open government data applications. Recall is on the one hand defined by data provided by data providers and on the other hand the quality of that data. Current platforms do not support any functionality in this respect.

Only Mashape supports real time query formulation by data users, although its support in relation to data sources is not clear. There is not yet a platform supporting for instance SPARQL on a semantic model. There are of course SPARQL endpoints on database management systems, like those provided by Oracle and also SPARQL adapters that transform a query formulated on a semantic model into a database query, are under development.

A conclusion would also be the lack of an agreed architecture for large scale data sharing based on (common) semantic models. It is up to developers to construct, publish, search, compose new APIs and implement these APIs in end-user applications by interpreting the semantics. Semantic API annotations based on approaches developed for web services (SA-WSDL: Semantically Annotated Web Service Definition Language) are not yet supported by the analyzed platforms. Yet, APIs seem the best solution to meet data governance requirements of data providers, since data is not replicated and APIs are able to support real time data access by providers.

This contribution analyzed solutions against a technical framework, where a data provider, data user, or platform provider can implement these. As discussed, platforms do not yet provide (easy to use) functionality to data users and – providers, especially in the context of data governance, data transformation, and data linkage and - fusion. A distributed implementation, in which each organization implements the required functionality, requires search mechanisms to increase precision and recall. Semantically enriched APIs or APIs operating on semantic models will improve precision and recall, by allowing data users to search for particular data from different perspectives of a semantic model, e.g. particular objects in place and time [15]. These searches result in appropriate API calls providing the required data. Although a semantic model [17] seems to be required, its added value is yet to be defined for open data. A distributed implementation combined with semantic models, enables in our view large scale decoupling of sources and consumers. It gives full flexibility to data users to specify their data requirements based on (linked) semantic models supported by APIs of different data providers.

A last aspect of importance is security. Open data basically does not require additional security features, but complementing an open data infrastructure with a security infrastructure allows sharing of not only open, publicly available data, but also data with access restrictions. Requirements of such a closed data infrastructure is given by the customs compliance architecture [16]. Security protocols to construct such a closed data infrastructure are specified by Pruksasri et.al [26]. Analyzed platforms implement security mechanisms, but these can only be set by a data provider. Combining a peer-to-peer data sharing architecture complemented with semantic models, (distributed) registries and security protocols enables all types of innovative applications based on real time source data.

6 Conclusions

This contribution builds upon a data value chain model developed by Esmeijer et.al [] and provides a technical framework supporting the data value chain. Platforms are analyzed with respect to their support of the data value chain its supporting technical framework. The data value chain and the technical framework both contribute to technical barriers identified in open data, interoperability, and cloud computing literature with respect to data sharing.

A first conclusion drawn from platform analysis given in this contribution is that an API registry implemented by API management platforms is the minimal required functionality for large scale open data implementation. Implementing APIs provides data governance to data providers, although these data providers cannot prevent the creation of potential new privacy or commercial sensitive data sets based on (filtered) data providers by a data user. There are quite a number of API management platforms available. They offer functions to publish and access data, both in an open and a closed environment, and support billing functionality. The latter enables the implementation of data sharing requiring payment based on APIs, e.g. like provided by Cadaster.

Further experience will be gained by implementing one of these platforms to support the proposed technical framework. Semantics is not yet supported according to the four rules for publishing linked open data [4]. URIs to name spaces and definitions used by APIs are part of these APIs, but are not linked to concept definitions and

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762014000300005

associations between those concepts. As APIs based on REST are an implementation of web services, research in that particular area might be applied to add this functionality, e.g. approaches like Uniform Service Description Language (USDL, [23]) or Semantic Annotations for Web Services (SA-WSDL).

A second conclusion is that the market for these type of platforms and its required functionality is not yet mature. First of all, a data sharing architecture covering distributed registries and data sharing with semantic models, complemented with security functionality, is yet to be developed and agreed upon. Furthermore, one can argue that semantic models are required for data sharing and linking different data providers, but development and relating these models to existing APIs is yet for further research. Lack of technology to implement APIs from a semantic model also makes it costly to develop an infrastructure providing this functionality. As there is no business case for semantic or common models for data sharing, it is also difficult to implement real time query formulation on heterogeneous data providers [5]. Many software developers are able to implement APIs as is. The technical framework proposed in this contribution identifies data sharing functionality, but the distribution of this functionality is yet to be defined. A data provider can implement and deploy the functionality, a data user or data provider can do the same with limited central governance, e.g. governance of semantic models. The analyzed software solutions currently support individual organizations publishing their APIs or a central data store for open government data implementing an open data portal. These open data portals provide themes, faceted search and other types of metadata for data discovery; some of them also support API discovery. It implies that open data providers provide (bulk) data and yet have to transform to API support. In terms of the stage model of Berners-Lee [4], this bulk data is structured, but has implicit links to other data, for instance geodata.

Whereas the focus of this paper is a technical framework of open data between any data provider and – user, platform analysis learns that also restrictions to data access can be supported based on supported security and identity management solutions. Furthermore, platforms can also support billing and invoicing, which enables a data provider to charge for data access. Thus, platforms are available to support any data driven applications, either open, with access restrictions, with billing and invoicing, and organization internally.

Whilst current solutions do not yet solve all barriers for open data formulated in literature (section 2.1), one can conclude that large scale implementations of open government data are not yet feasible or require additional investments. Jetzek et.al [17] have found a contribution of open data to innovation and participation, under the assumption that all participants are able to collaborate. Additional investments for large scale implementation is a potential barrier to this contribution of open data to innovation and participation.

A final conclusion relates to potential use cases of (open) data. Literature identifies a relation between innovation by open data and a positive economic and social business case [17], but actual use cases supporting improved decision making with situational awareness are not yet considered. It is expected that data sharing will have a large impact on economy and society by improving decision support based on situational awareness [27].

Acknowledgments

The authors would like to gratefully acknowledge the support of the Next Generation Infrastructure program, a collaboration of the Rotterdam Port Authority and Technical University of Delft, for this research. This paper results from research done in this program within the AOHA project on apps development for the Port of Rotterdam.

Websites List

Site 1: ePSIplatform: Europe's One-Stop Shop on Public Sector Information (PSI) Re-Use
www.epsiplatform.eu

Site 2: The Open Data Portal of the Dutch Government
www.data.overheid.nl

Site 3: Dublin Core Metadata Initiative
www.dublincore.org

Site 4: Kademlia, a distributed hash table for peer-to-peer networks
www.en.wikipedia.org/wiki/Kademlia

References

[1] M. Al Boni, D.T. Andersen and R.L. King, Constraint preserving genetic algorithm for learning fuzzy measures with an application to ontology matching, in Advance Trends in Soft Computing (M. Jamshidi, V. Kreinovich and J. Kacprzyk, Eds.). San Antonio, Texas: Springer, 2014, pp. 93-104.
[2] C. Batini, M. Scannapieco, Data Quality: Concepts, Methodologies, and Techniques. Heidelberg: Springer-Verlag, 2006.

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762014000300005

[3] T. Berners-Lee, J. Hendler and O. Lassila, The semantic web, Scientific American, vol. 284, no. 5, pp. 28-37, 2001.
[4] T. Berners-Lee. (2009, June) Linked data – four rules. W3 Org. [Online]. Available: http://www.w3.org/DesignIssues/LinkedData.html
[5] C. Bizer, T. Heath and T. Berners-Lee, Linked Data - the story so far, International Journal on Semantic Web and Information Systems, vol. 5, no. 3, pp. 1-22, 2009.
[6] F. v. Blommestein, Structured Communication for Dynamic Business - an Architecture for Flexible B2B Communication. Groningen: University of Groningen, 2013.
[7] R. Buyya, C.S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility, Future Generation Computer Systems, vol. 25, no. 6, pp. 599-616, 2009.
[8] C.M. Chituc, A. Azevedo and C. Toscano, A framework proposal for seamless interoperability in a collaborated network environment, Computers in Industry, vol. 60, no. 5, pp. 317-338, 2009.
[9] T. Erl, Service Oriented Architecture - Concepts, Technology and Design. Boston, MA: Prentice-Hall, 2005.
[10] J. Esmeijer, T. Bakker, S.D. Munck. Thriving and surviving in a data-driven society. TNO. Delft: TNO, 2013.
[11] E. Folmer, M. Punter, Management and Organization of Open Standards (BOMOS), NOIV, Enschede, 2010.
[12] E. Gamma, R. Helm, R. Johnson, J. Vlissides, Design Patterns – elements of re-usable object-oriented software, Addison-Wesley, 1995.
[13] C. Gentry, A fully homomorphic encryption scheme, Ph.D. dissertation, Department of Computer Science, Stanford University, Palo Alto, CA, 2009.
[14] E. Goncalves da Silva, Supporting dynamic service composition at runtime based on end-user requirements, user generated services workshop, in Proceedings International Conference on Service Oriented Computing (ICSOC), Stockholm, Sweden, 2009, pp. 23-27.
[15] T. Heath and C. Bizer, Linked Data - Evolving the Web into a Global Data Space, Synthesis Lectures on the Semantic Web: Theory and Technology. San Rafael: Morgan & Claypool Publishers, 2011.
[16] W. Hofman and H. Bastiaansen, A global IT infrastructure improving container security by data completion, in Proceedings European Conference on ICT for Transport Logistics (ECITL), Zaragoza, Spain, 2013.
[17] T. Jetzek, M. Avital and N. Bjørn-Andersen, Generating value from open government data, in Proceedings 34th International Conference on Information Systems, ICIS, Milan, Italy, 2013, p. 11.
[18] S. Kaisler, F. Armour, J. Espinosa, and J. Money, Big data: issues and challenges moving forward, in Proceedings 46th Hawaii Internation Conference, Hawaii: IEEE, 2013, pp. 995-1004.
[19] J. Kim, Supervenience and Mind: Selected Philosophical Essays. New-York: Cambridge University Press, 1993.
[20] J.C. King and J. Stanley, Semantics, pragmatics, and the role of semantic contents, in Semantics vs. Pragmatics (Z. Szabo, Ed.). New York: Oxford University Press, 2004, pp. 111-164.
[21] B. Long and Y. Chang, Relevance Ranking for Vertical Search Engines. Waltham, MA: Morgan Kaufman, 2014.
[22] N. Loutas, E. Kamateri, F. Bosi, and K. Tarabanis, Cloud computing interoperability: the state of play, in Proceedings Third IEEE International Conference on Cloud Computing Technology and Science IEEE Computer Society, Athens, Greece, 2011, pp. 752-757.
[23] D. Oberle, A. Barros, U. Kylau, and S. Heinzl, A unified description language for human to automated services, Information Systems, vol. 38, no. 1, pp. 155-181, 2013.
[24] V. Peristeras, K. Tarabanis and S.K. Goudos, Model-driven eGovernment interoperability: a review of the state of the art, Computer Standards & Interfaces, vol. 31, no. 4, pp. 613-628, 2009.
[25] T. Priebe, W. Dobmeier and N. Kamprath, Supportinga attribute-based access control with ontologies, in Proceedings of The first International Conference on Availability, Reliability and Security (ARES2006), Vienna, 2006, pp. 465-472.
[26] P. Pruksasri, J. v.d. Berg, W. Hofman, and S. Deskapan, Multi-level access control in the data pipeline of the international supply chain system, in Proceedings International Conference on E-business Technology and Strategy (iCETS), Macau, 2013, pp.79-90.
[27] The Economist. (2014, January) The future of jobs: the onrushing wave. The Economist. [Online]. Available: http://www.economist.com/news/ briefing/21594264-previous-technological-innovation-has-always-delivered-more-long-run-employment-not-less
[28] A. Tolk, C.D. Turnitsa, S.Y. Diallo, and L.S. Winters, Composable M&S web services for net centric applications, JDMS, vol. 3, no. 1, pp. 27-44, 2006
[29] Zuiderwijk, N. Helbig, J. Gil-Garcia, and M. Janssen, Guest Editors' Introduction. Innovation through open data: a review of the state-of-the-art and an emerging research agenda, Journal of Theoretical and Applied Electronic Commerce Research, vol. 9, no. 2, pp. I-XIII, 2014.