


An evolvable and transparent data as a service framework for multisource data integration and fusion

Zhipu Xie¹ · Weifeng Lv¹ · Linfang Qin¹ · Bowen Du¹  · Runhe Huang²

Received: 15 December 2016 / Accepted: 22 March 2017 / Published online: 10 April 2017
© Springer Science+Business Media New York 2017

Abstract Combining data from multiple sources is a means of enabling unified and comprehensive description of objects in high-dimensional space and helping unlock the potential value of such data. In recent years, more and more studies have focused on this field of research. However, challenges posed by separately stored data and comprehension barriers about different systems hinder the integration of data from different sources. To overcome these problems, this paper proposes a Transparent Data as a Service framework, a novel approach combining Transparent Computing and Representational State Transfer (REST) Web Services based on Linked Data. This framework is capable of integrating data from different sources and offering data services in a transparent way. That is, consumers use

data services without the need to know details of where or how the data are stored. Our framework is transparent on three levels: transparent data resource integration, transparent data fusion and transparent data service provision. The Data Model Pool and Data Resource Pool are able to evolve as new data models and datasets are generated in the provision of data services. Finally, we demonstrate the feasibility of the framework by implementing a prototype system.

Keywords Data as a service · Transparent computing · Linked data · RESTful Web Service

1 Introduction

Massive amounts of heterogeneous and multisource data are being generated as, increasingly, digital sensors are being deployed, particularly in urban centers [1]. Different companies, organizations or different departments of the government currently collect and separately store these data. However, these datasets collectively offer much greater potential value if they can be meaningfully connected [2, 3] and fused [4].

Currently, there are multiple challenges to achieving this goal. First, storing the data in one place, even in a cloud-based data center, is impracticable when the volume of the data is very large. Nevertheless, data may still be stored separately as long as there is a way to find any particular piece of data. For example, highway management systems typically use distributed storage for massive amounts of surveillance video data. If abnormal vehicle behavior is observed, staff need only look up selected image information to confirm what happened. Second, integrating data from different providers requires first learning about the schemas and specific meanings of data from unfamiliar fields. This effort can

This article is part of the Topical Collection: *Special Issue on Transparent Computing*
Guest Editors: Jiannong Cao, Jingde Cheng, Jianhua Ma, and Ju Ren

✉ Bowen Du
dubowen@buaa.edu.cn
Zhipu Xie
xiezhipu@buaa.edu.cn
Weifeng Lv
lwf@buaa.edu.cn
Linfang Qin
linfangqin@buaa.edu.cn
Runhe Huang
rhuang@hosei.ac.jp

¹ State Key Laboratory of Software Development Environment, Beihang University, Beijing, China

² Faculty of Computer and Information Sciences, Hosei University, Tokyo, Japan

be onerous and time-consuming. Often, several discussions are needed for developers to gain a clear understanding of data from other providers.

Consequently, we aimed to develop a more convenient and practical data sharing paradigm that also facilitates comprehension of data from different sources. Our approach is consistent with the Transparent Computing philosophy [5]. In Transparent Computing, Transparent Clients request services from Transparent Servers without worrying about the implementation details of the services, which are delivered in a networked (wired or wireless) environment. The implementation details of services are invisible to the Transparent Clients [6]. The Transparent Computing paradigm for services inspired our idea of developing a transparent data service paradigm.

Three main challenges in integrating the use of heterogeneous and multisource data were identified. First, data are stored and managed in a distributed fashion. The lack of centralized organization and management means that people who need particular data struggle to know where or how to find that data. Second, there are comprehension barriers about different systems that hinder integration of data from different sources. Third, data services only offer whole datasets. Instead, they should be able to offer more fine-grained access to selected data.

A Transparent Data as a Service (TDaaS) framework is proposed to meet these challenges. First, a data-providing method based on JavaScript Object Notation for Linked Data (JSON-LD) [7, 8] is used to define criteria for those volunteering to share data with others. JSON-LD is designed for presenting Linked Data in the format of JavaScript Object Notation (JSON) [9]. Then, the Transparent Computing philosophy has been applied to design data services that can be easily used without special data management knowledge.

In summary, the contributions of this paper include the following:

- * *Transparent Data as a Service.* We develop a data service framework following the transparent computing paradigm. Our TDaaS framework supports the provision of data services through the transparent integration and fusion of data from different sources. In our framework, transparency is implemented in three layers of activity: data resource integration, data fusion and data service provision.
- * *Evolvable Data Model and Data Resource Pool.* We identify a mechanism for evolving a data resource pool by leveraging data models and datasets generated in the course of data analysis. Data models and datasets generated while providing data services are updated and periodically incorporated into a Data Model and Data Resource Pool. This evolving resource pool supports the efficient provision of additional data services.
- * *A Prototype System.* We implement a prototype system that demonstrates the feasibility of our proposed approach. The prototype shows that data integration and fusion can be simply achieved, while maintaining transparency for both data users and providers. Moreover, this approach is generalizable to other potential applications.

The remaining sections of this paper are organized as follows. In Section 2, we review related research. Section 3 introduces the overall architecture of the TDaaS framework. Sections 4 and 5 describe the main parts of the framework, the Web Services based on Linked Data, and the Data Operating Center, which build on the philosophy of Transparent Computing. Section 6 describes the Evolvable Data Model and Data Resource Pool, while Section 7 documents the prototype system and case studies. We conclude our paper with Section 8.

2 Related works

2.1 Transparent computing

Computing paradigms have shifted over time from centralized mainframe computing towards more distributed computing [10]. Continuing in this trend is ubiquitous or pervasive computing, where computing operations are ever more distributed across a variety of appliances and devices. This approach proposes that users should only need to focus on the services they require, without the distraction of where those services might be running. Traditional computing systems have generated problems, such as complexity, high total cost of ownership, weak security and lack of user friendliness [6]. Fitting within the paradigm of pervasive computing, Transparent Computing approaches decouple the storage and execution of programs. Specifically, the required software and data, which reside on central servers, are transferred to the clients as needed. In this way, users can obtain computing services in a hassle-free way.

2.2 Linked data

Since 2010, when Tim Berners-Lee introduced his five-star rating suggestion for publishing data on the Internet as Linked Data [8], many individuals and groups have participated in contributing to the Linked Open Data Cloud [11, 12]. The success of linked data inspired us with the idea of setting up a similar data sharing mechanism which could replace the traditional practice of setting up data exchange platforms.

The Resource Description Framework (RDF) has been in the engine of the W3C since 1999, although the RDF 1.0

concepts specification was not released until 2004. In 2011, RDF 1.1 was published, along with many welcome updates, including the decoupling of RDF in Extensible Markup Language format (RDF/XML) from RDF [13]. The RDF has significantly contributed to the development of the World Wide Web. However, many developers have resisted the perceived complexity of RDF/XML and the semantic web [14]. Therefore, it is important to continue encouraging people to share data as Linked Open Data, by making Linked Data support as many formats as possible and developing data conversion tools.

Besides RDF/XML, Turtle, RDFa (Resource Description Framework in Attributes) and N3 (Notation3), JSON-LD is designed for presenting Linked Data in the format of JSON. JSON-LD is fully compatible with JSON, and JSON can be upgraded to JSON-LD relatively easily. Although JSON-LD emerged later than other formats of Linked Data, it has value because many systems already use JSON, especially those based on Representational State Transfer (REST) or RESTful Web Services. A shortcut for turning JSON data into Linked Data not only means data already in JSON can be transformed into Linked Data easily, but also creates opportunities for combining the goals of Linked Data and RESTful Web Services [7, 15, 16]. These Web Services can be organized in a transparent way to form the Data as a Service framework.

We can extract knowledge from data resources when transforming them into linked data. A KID (Knowledge-Information-Data) model proposed by Atsushi Sato and Runhe Huang [17] provides details for an implementation. In subsequent research, Sato and Huang showed applications of this theory in the field of smart business [18, 19].

A concept similar to linked data is called the Internet of Data, which has borrowed the expression from the Internet of Things, and is focused on data generation [20].

2.3 Data as a service

The development of advanced knowledge systems involves a multitude of technologies and data processing [21]. However, collecting these heterogeneous data and transforming them into tangible insights still remains a challenge. With the rise of mobile Internet technology and the rapid growth of data worldwide, systems must be able to deal with this Big Data. Recently, various research efforts have concentrated on the development of Data as a Service (DaaS). Among these studies, Xinhua, E et al. have revealed that Big Data as a Service (BDaaS) is a critical approach to realize value from Big Data [22]. BDaaS is the combination of Web Services and data management technology, which supports not only structured data, but also supports an unstructured data model. However, the research of BDaaS is still in its

infancy and many challenges remain to be resolved. As a consequence, an in-depth investigation of BDaaS will be performed. In the transportation context, urban traffic data services still cannot fully analyze and utilize their Big Data. Traffic City Data as a Service (CTDaaS) has been addressed [23].

3 Transparent data as a service framework

The TDaaS framework is proposed to address the challenges associated with massive heterogeneous data drawn from multiple systems. Examples of such data include Points of Interest data in the Geographic Information Systems (GISes), a variety of sensor data produced by Internet of Things systems, and behavioral data generated from citizens' daily activities. The TDaaS framework specifies how to manage data and offer data services within a transparent paradigm.

3.1 Roles of TDaaS

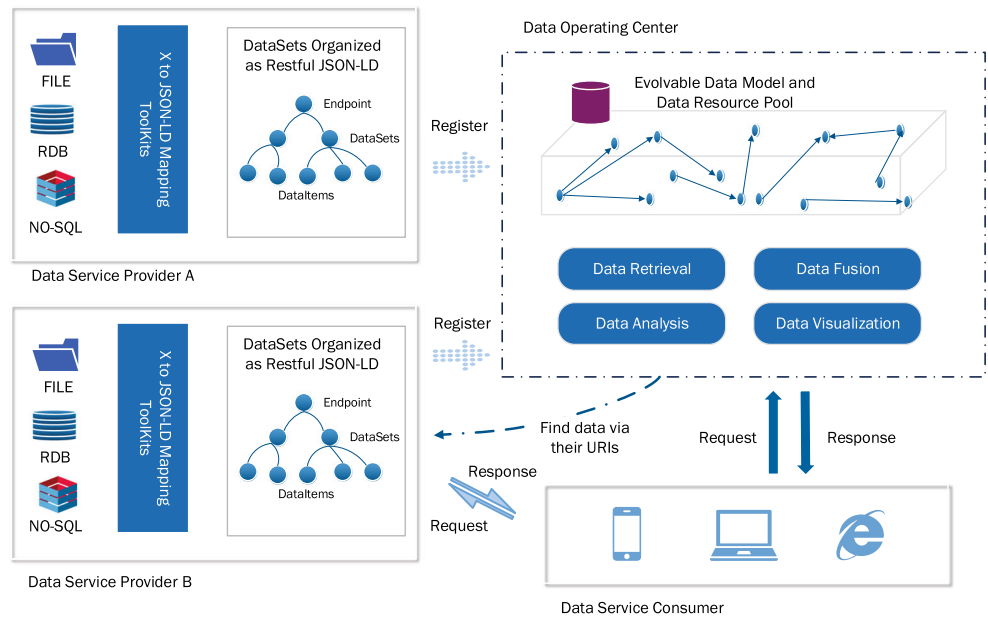
The TDaaS framework defines three roles: Data Service Providers (DSPs), Data Service Consumers (DSCs) and a Data Operating Center (DOC). DSCs are entities that request data services from the system. DSPs are organizations with datasets of interest. DSPs are charged with data preprocessing, including data cleaning and data specifying, to ensure that their data are usable and conveniently accessible by DSCs. The DOC is responsible for maintaining a shared vocabulary, collecting data from different sources, and offering four kinds of data services, namely, data retrieval, data analysis, data fusion and data visualization.

Although a DSP could offer data services directly to DSCs, access through the DOC is more transparent, offering data consumers a more general view of data and the potential to discover additional useful datasets. This approach is similar to the role of search engines on the Internet. Any web site or web application could work alone; however, the information integration ability of search engines helps users get more value from the Internet.

3.2 System structure of TDaaS

In the TDaaS framework, as shown in Fig. 1, data are collected and stored by different companies and organizations. Datasets can be stored in different ways, such as distributed files, relational database tables or No-SQL database documents. However, all these formats can be transformed to JSON very easily because this is just what developers continuously do in implementing web applications. Subsequently, we can transform to JSON-LD just by making the JSON schemed and linked.

Fig. 1 Transparent data as a service framework



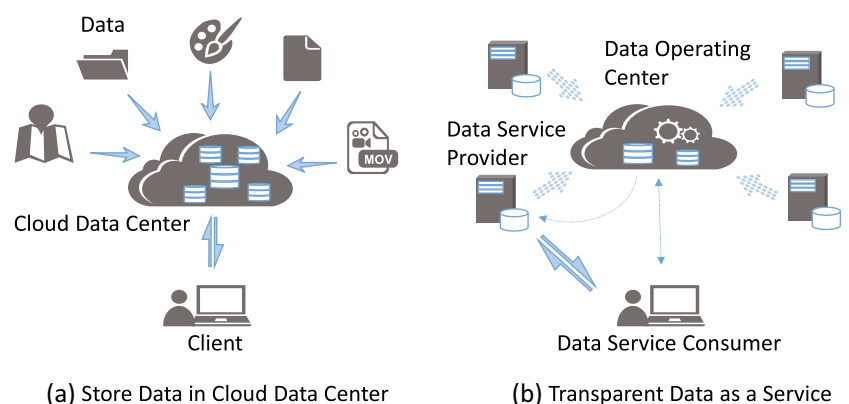
The data providers are asked to transform their datasets to the uniform JSON-LD format, enabling DSCs to access the data services in more convenient ways. To transform original datasets into a unified JSON-LD format, three steps are necessary. First, a unified data model, the extended vocabulary, will be shared through the Internet and kept up-to-date by collecting suggestions from every participant. Next, toolkits will be implemented and used to transform data in other formats into JSON-LD. Finally, each DSP will set up a SparQL endpoint for all its shared datasets.

The DOC accelerates the integration and fusion of multi-source data. It is responsible for describing and organizing all the datasets offered by different companies and organizations. Because all data providers post their data service endpoint, through which all the datasets will be accessible, the DOC knows about each source of data. It is therefore able to form a huge data graph by analyzing the relationships across all the data. Based on the well-organized data

graph, data services can be offered more intelligently via the power of the semantic web. Moreover, the DOC also offers abundant data processing models to provide preprocessing results. The DOC also manages another feature of the TDaaS framework: the Evolvable Data Model and Data Resource Pool, where new data models and data resources will be generated by the data analysis modules.

The TDaaS framework differs from the traditional cloud data center as represented in Fig. 2. Increasingly, researchers recognize the limitations of cloud computing [24]. Network bandwidth constraints and massive amounts of sensor data make it impractical to store everything in a cloud data center. In the cloud data center paradigm, data from different sources are physically gathered, while in the TDaaS framework, data are only generated logically. Data entities remain physically distributed in different locations. The virtualized datasets, which are represented by their uniform resource identifiers (URIs), are integrated in the DOC. When DSCs

Fig. 2 Difference between transparent data as a service and cloud data center



request data from the DOC, the URIs of datasets will be returned and data entities will be transferred from the DSPs.

3.3 Processes of TDaaS

We have illustrated the data service processes in the Transparent Computing paradigm in Fig. 3. In the transparent paradigm, the whole computing environment is divided into three parts: the Transparent Server, the Transparent Network and the Transparent Client. This loosely-coupled architecture offers advantages such as user and application transparency, support for heterogeneous operating systems and devices, and enhanced security. In our TDaaS framework, the DSCs constitute the Transparent Clients, while the DOC and DSPs together constitute the Transparent Servers. The Transparent Network always refers to the Internet or other Internet-like networks.

In summary, transparency is designed into three aspects of our framework. First, data resource integration is transparent. Data resources from multiple sources are generated without actually gathering data in one location, which means that the integrated resources are virtualized. Second, data fusion is transparent. The DOC is designed to support data fusion, which is based on related data analysis and data retrieval modules. When people seek details of a topic,

they do not need to know what datasets have been collected. Instead they just search the database with the URI of the chosen topic and relevant details will be offered. Finally, data service provision is transparent. When requesting DaaS from the DOC, DSCs need only select the data they need without needing to know details of where or how the data are stored.

4 Designing shareable data: a schemed, linked and browsable data service

The Linked Data paradigm is designed for sharing data through the Internet, similar to an Internet of Data. Simply put, we aim to establish a logical part of the Internet, in which datasets, rather than html pages or web applications, are connected. Linked Data principles define a five-star-level dataset as one that is machine-readable, well-formatted and linked. These foundational ideas about Linked Data have been established in recent years. The proposal of JSON-LD not only established a new format for linked data but also highlighted the need for a data service API to be human-readable as well as machine-readable. Moreover, the idea of a browsable API is borrowed from the Django Rest Framework, which allows data service users to navigate

Fig. 3 Processes of transparent data as a service

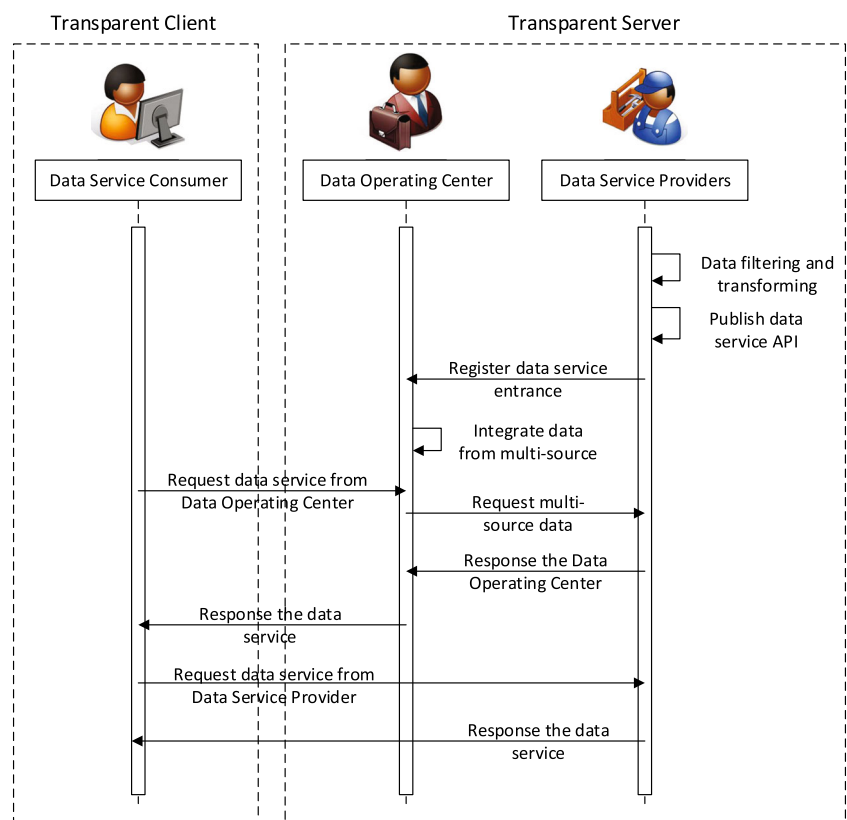
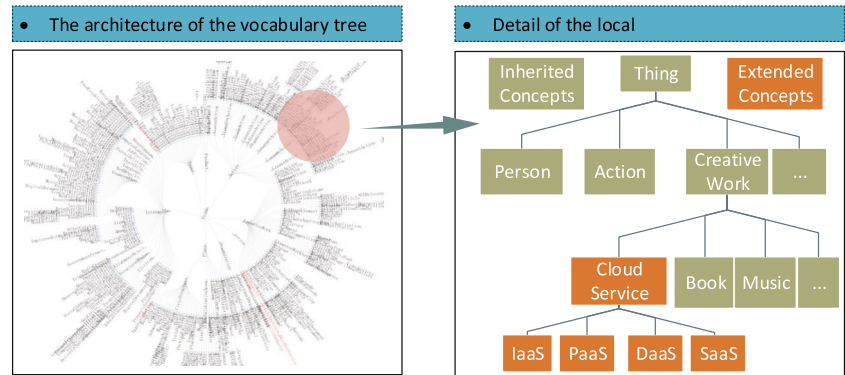


Fig. 5 Whole architecture of the vocabulary illustrated via d3.js



We illustrate the whole architecture of the classes in the vocabulary in Fig. 5. The inherited concepts from schema.org are shown in blue, while the extended concepts are shown in red.

4.2 Data mapping toolkits

After establishing appropriate vocabularies to describe the concepts used to specify shared data, DSPs are able to convert source data in different formats into a unified format. As explained in the above sections, we chose JSON-LD as the output format. JSON-LD is fully compatible with traditional JSON, which means that producing JSON-LD is almost as simple as producing JSON, and all existing tools and libraries can still be used. We can build up toolkits to help transform data in common formats, such as distributed files, relational database tables, and documents in NO-SQL distribute databases, into JSON-LD. These are named, X to JSON-LD Mapping Toolkits.

For any data services, all data should be read-only for DSCs. Accordingly, we need to support query operations in our data service. Specifically, we need to support listing all information about data items, in a paged way, as well as querying a data item via its primary key, the URI location of the resource.

The selection of a suitable form for organizing the data service can significantly improve the efficiency of service management and reduce unnecessary HTTP requests. Therefore, we organize the data resource as a three-layer structure, which includes data items, datasets and endpoints from the bottom up. A data item I describes an entity E in the real scenario, and I is denoted as the following tuple:

$$I = \{URI, Type, Dict\} \quad (1)$$

where URI represents the unique address of a data item and denotes the unique identification of the represented entity.

$Type$ is the type of the represented entity, which will map to a concept in the vocabulary. Finally, $Dict$ is the detailed description of the represented entity, which will be represented as dictionaries in which the key-value pairs are the properties and the values respectively.

A dataset S is a collection of a group of data items, which will be of the same type, and is denoted as:

$$S = \{URI, Type, Timerange, Arearange, Collection\} \quad (2)$$

Here, URI is the unique address and unique identification of a dataset. $Type$ is the type of all the contained data items. $Timerange$, optional, indicates the time range in which the data items are distributed. $Arearange$, also optional, indicates the area range. $Collection$ contains the data items.

A data provider willing to share more than one dataset is recommended to put the URIs for all these datasets at a main entrance. This main entrance is represented by endpoint E , whereas E is represented as:

$$E = \{URI, Author, Createdtime, DSSes\} \quad (3)$$

Again, URI is the unique address and unique identification, while $Author$ is the description of the provider. $Createdtime$ indicates when the endpoint was set up and $DSSes$ represents the included dataset services.

In our recommendation, datasets of the form S should contain every detail of their included data items. In contrast, endpoints of the form E should contain limited details besides the URIs that indicate where the datasets can be accessed. Using this design, it is possible to scan the registered endpoints and know the URIs of the datasets. Then the datasets can be scanned to efficiently discover the details of all data items. This approach eliminates the need for separate HTTP requests for the details of each data item.

4.3 SparQL endpoint for each DSP

A DSP may also share more than one dataset service. To organize these data services in a unified way, a simple uniform architecture is proposed. The dataset services should be organized as a three-level tree. The root node represents the entrance of all of the dataset services, the second layer refers to datasets and the deepest layer accesses individual data items. We highly recommend that the first two layers contain almost all data item information so that the SparQL endpoint can offer a query service using only the first two layers. If too much information is reserved in the third layer, the SparQL endpoint will be forced to traverse through every data item more often, reloading resources inefficiently.

5 Managing shareable data: the role of the data operating center

The presentation of diverse data sources in a unified JSON-LD format, together with a shared vocabulary, enables DSCs to access dataset services in a unified way. However, to

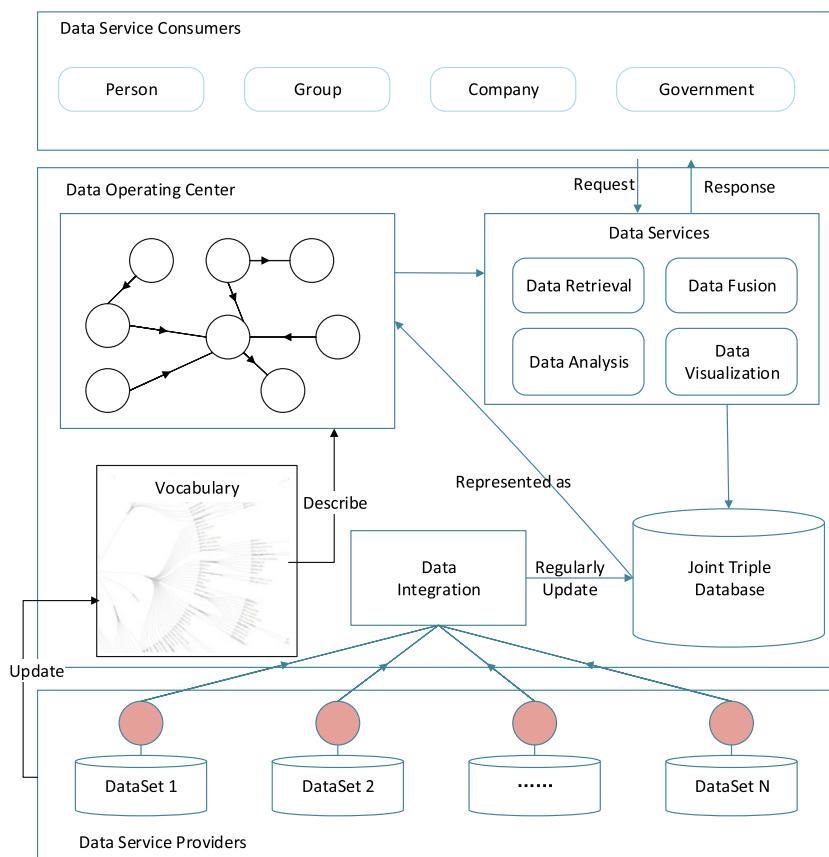
enable more intelligent and transparent data service consumption, a novel data service offering and consuming mode is proposed in which the DOC will play an important role.

As shown in Fig. 6, the three main functions of the DOC include integrating data from different DSPs, managing the unified vocabulary and offering data services.

DSPs are expected to organize their datasets in unified ways and set up main entrances. If all data have been transformed into machine-readable format, when the DOC seeks detailed information about the datasets, it only needs to know the main entrances of the Dataset Services. As the dataset entities are so massive, it is impractical for the DOC to store all the data. Instead, the DOC only organizes the virtualized data, which is presented by unique URIs.

We have mentioned that a unified vocabulary should be shared on the Internet to help describe the datasets among all of the DSPs, DSCs and the DOC. We also recommend that this vocabulary be managed by the DOC. The vocabulary should be organized in a clear hierarchical structure and offer clear definition references as well as a user-friendly search function, to help people quickly familiarize themselves with the concepts in the vocabulary. DSPs should post

Fig. 6 Main functions of the DOC



an update requirement to the DOC if they notice that the current vocabulary lacks concepts to adequately describe their shareable datasets. The DOC should collect and review all change suggestions to determine an updated edition of the vocabulary.

We identify four basic data services of the DOC, while anticipating that new features will be added over time.

Data retrieval Our framework will support data retrieval by selecting data with SparQL through the joint triple database, containing data from multiple DSPs. It will be a giant database so will depend on the use of Big Data technologies, including distributed file storage, parallel computing and even in-memory cluster computing. For example, if we use MySQL for data storage, we could change to MemSQL because it works in memory and can be deployed on clusters.

Data fusion In our structure, data fusion is readily accomplished. For example, if someone wanted to learn as much as possible about an area, but lacked any knowledge about the database contents, the person could just search the database with the unique identification of the area. This approach would return relevant information, which was originally distributed across different data sources, such as Points of Interest distributions, typical transportation and traffic patterns, and housing prices for the area.

Data analysis The data analysis model resembles the cloud computing concept of a Platform as a Service (PaaS). That is, you can analyze data on the DOC, retrieving only the output of the procedure. Procedures are required to be coded within the limited paradigms set up by the DOC, for example, the Hadoop map reduce style or Spark style with Scala. This approach offers DSCs the ability to deal with Big Data without deploying their own computing cluster. Although the metadata may be too massive to transfer across the Internet, the intermediate or final results may be less massive and less costly to transfer to the DSCs. At the same time, these results can also be saved in the database as new data resources for future use.

Data visualization Many data visualization or Office Automation tools include high quality visualization features such as different charting possibilities, support for different data sources, and user-friendly human-computer interfaces. Data visualization will become even more exciting with the availability of a unified data model (vocabulary) and data format. The visualization will become more intelligent. Take the heat map for instance, when we query three columns of data from the triple database via SparQL: strings

in the first column are name values, the second and third columns are latitude and longitude, respectively, and the last column contains float values. The chart will be generated with points represented by the geographic coordinates and with the heat values in the last column, and labeled as their names. User then need only change certain details of the chart, such as the map colors.

6 Generating shareable data: the evolvable data model and data resource pool

As described in the preceding section, the DOC is required to manage the vocabulary, or shared data models, and to organize the virtual data services. We refer to this combination of shared data models and virtual data resources as the Data Model and Data Resource Pool. The data models specify the structure of the data resources, so, together, the data models and data resources comprise a logical whole even though they are stored in different ways. Although we sometimes refer to the Data Resource Pool for short, we note that the value of the data resources is linked to the data models.

New data models and new datasets will be produced through data analysis. In our framework, the generated data models and datasets will return to the Data Model and Data Resource Pool. In fact, the whole process constitutes a cycle, called data evolution, as illustrated in Fig. 7. First, when a data analysis model is invoked, the DOC will request data from DSPs using the URL of the datasets. After data analysis, newly produced data models and data resources will be stored in the Data Model and Data Resource Pool. Consequently, any original data remains stored on the servers of the DSPs, while newly produced data resources will be stored in the DOC.

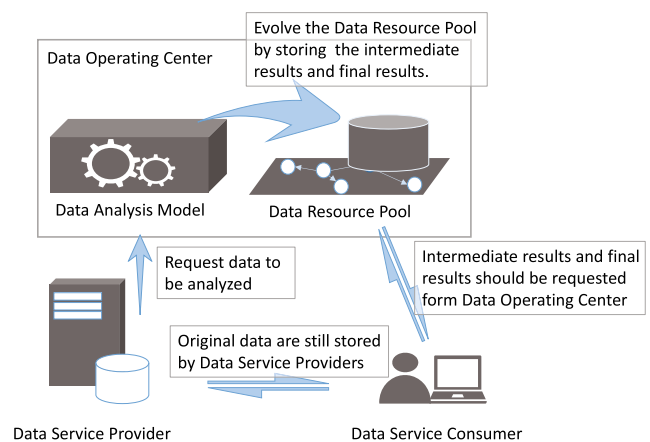


Fig. 7 Evolvable data model and data resource pool

Table 1 Parts of the datasets with brief introduction

Dataset Name	Item number	Introduction
AFC-Subway	51,260	Passengers' riding records produced when traveling by subways with traffic IC cards
AFC-Bus	53,896	Passengers' riding records produced when traveling by buses with traffic IC cards
Buslines	1,755	All the buslines in Beijing, including information of bus stops they pass by
Busstops	6,456	All the busstops in Beijing, including the position information
Houseprice	996	Parts of the house price information in Haidian District, collected from the Internet

7 A prototype system and case study

We implemented a set of prototype systems based on the TDaaS framework. In this section, we describe our experimental environment, the data we gathered and transformed, and the performance of the prototype environment. We also demonstrate how this environment can yield useful information in response to simple, intuitive queries.

We assumed all three roles in the prototype system. Thus, we implemented the data transformation tools as DSPs, we deployed the required functions of a DOC and then we acted as DSCs by requesting the data service that we originally shared.

7.1 The experimental environment

The experimental environment includes a small cluster with eleven DELL Optiplex 9020 computers, each of which is equipped with 32 GB RAM and 2.0 TB disk memory. The eleven computers are divided into one name node, one secondary name node and nine data nodes. These computers are running a 64-bit Linux operating system, version CentOS 6.5. They are recognized as an Apache Hadoop cluster with Spark as the support for distributed in-memory computing. Another personal computer is designated as the DSC, which is not expected to have high

performance hardware. This computer is running Windows 10, on which a virtual machine with Linux 7.1 has been installed.

7.2 Extension of vocabulary and collected data

A shared, practical vocabulary is typically built up through extended group effort. Typically, all participants help to shape the vocabulary, by spending time learning about the vocabulary and then posting their suggestions for improvements to it. In our prototype system, we implemented functions that support easy exploration of vocabulary concepts and properties, and the submission of suggestions.

The prototype system contains eight kinds of datasets, of which a subset is shown in Table 1. We have Automatic Fare Collection (AFC) data for both subways and buses, which capture the travel patterns of passengers.

7.3 Retrieving data from the joint triple database

The joint triple database effectively integrates data from multiple sources. However, the number of records can be very large, especially when the system is deployed to a production environment. As a result, retrieving information from such a large data source can be challenging. Nevertheless, using Big Data technologies, we are able to build up

Table 2 Experiment cases

Case Name	Description
Data loading	Load data from the JSON-LD API provided by the DSPs
Names	Names of all the subjects who own the property name.
Locations of Bus stops	Pairs of all the Longitude and Latitude of all the bus stops. Will be retrieved by multi-conditions
Names and Locations	All the subjects that have a name as well as the location information, even they are from different data source
Data Unloading	Clear the joint triple dataset

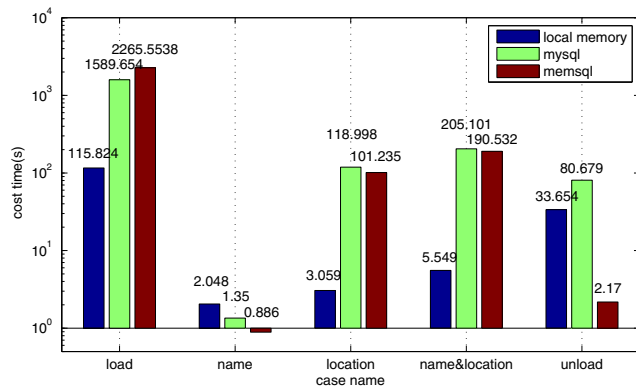


Fig. 8 Performance of different storage strategies

data centers with ample, distributed and flexible storage for dealing with massive amounts of data.

Our first experiment tested the efficiency of different storage strategies. When we had small datasets, we could just store them into local memory. When the data were larger, we tried using relational databases such as MySQL. However, IO limitations inhibit efficient data retrieval. So we adopted a third strategy, replacing MySQL with MemSQL. MemSQL uses similar APIs but which can run in memory, are scalable and can be deployed on a cluster.

The cases are listed in Table 2. Not all cases involved data retrieval, but they reflect the performance of different storage strategies. The results of our experiment are illustrated in Fig. 8. The results show that local memory performs well but, because the volume of storage is very limited,

this storage strategy will seldom be used. Although MemSQL showed only a slight speed advantage over MySQL in processing the given tasks, its scalability and usability will make it the first choice for storing the joint triple datasets.

7.4 Case study for data fusion

Our next experiment tested multisource data fusion based on the TDaaS technology. Data fusion can be classified into many types [26] and the following example belongs to the object refinement technology based on data association. It shows that multisource data integration is the basis of data fusion, especially for tasks such as data refinement.

After hyperlinked references to other resources are built up, connections between two data items can be established. If we want all available information about an object, we must first get its URI, which means its identification in the joint triple database. Then we can simply retrieve all the sentences whose subjects or objects match the URI. The simplicity of this approach depends on the foundational data structures that have already been put in place. For example, a DSP who aims to share high quality data is encouraged to study the relationships among related concepts in the vocabulary and to learn about other datasets to determine potential relationships between his or her datasets and others.

We conducted a demonstration showing passenger flow volume via different modes of transportation, namely, subway, bus and bicycle, from April 23 to May 4 of 2015 in Beijing. We also examine the relationships between different modes of transport. The overall processes are shown

Fig. 9 Processes of the data fusion case study

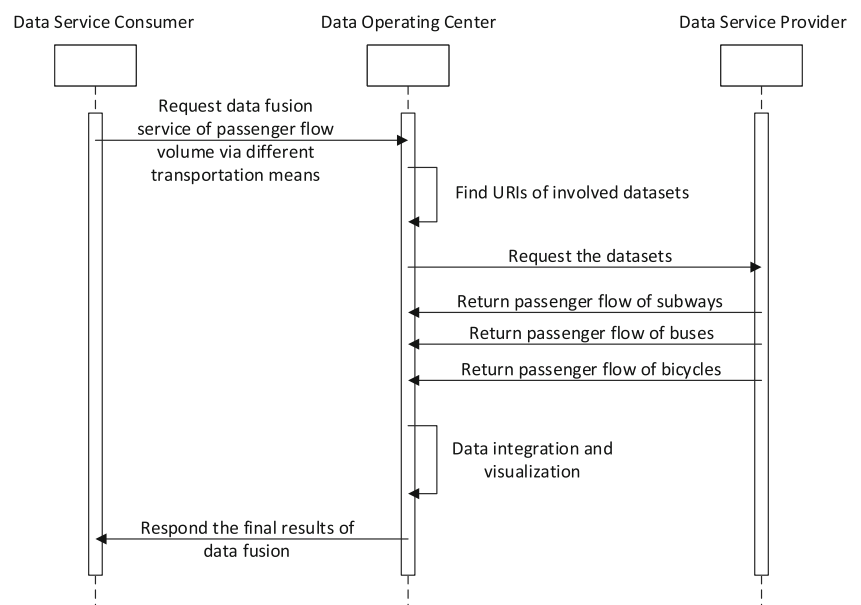
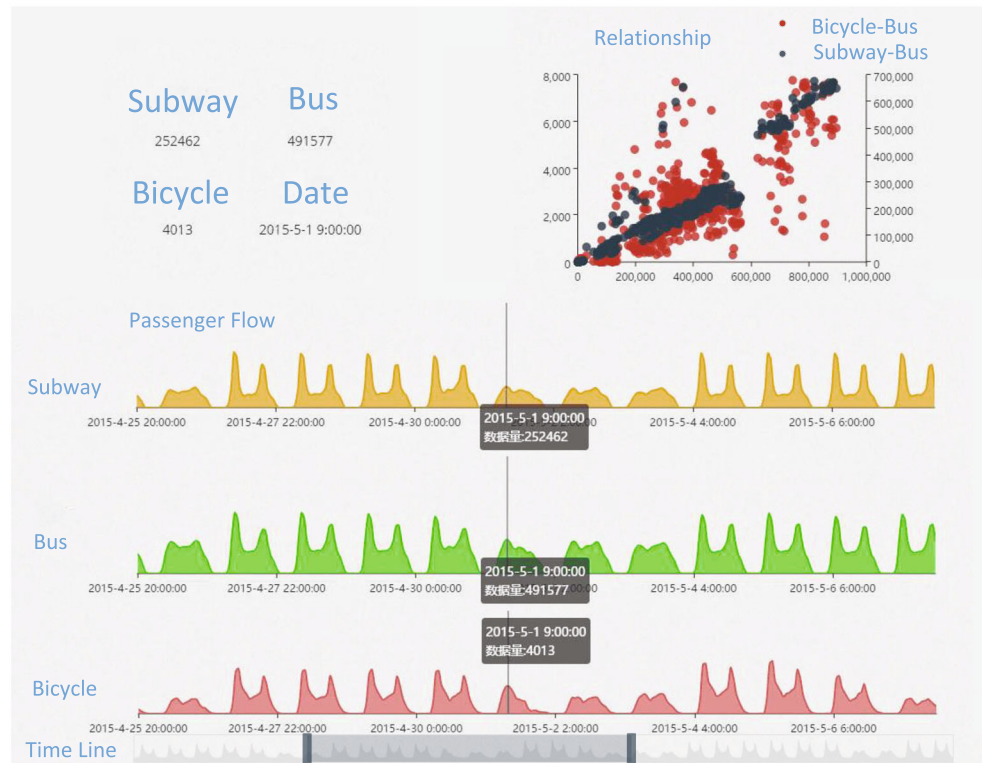


Fig. 10 Data fusion results of passenger flow volume via different transportation means



in Fig. 9. First, a DSC posts the request for a data fusion service to the DOC. Instead of responding with the data directly, the DOC finds the URIs for the relevant datasets and requests that data from the Data Providers in sequence. After integrating the data and creating a visualization, the DOC returns the data to the DSC.

A web page containing the data fusion results is shown in Fig. 10. The data reveal some interesting insights. First, many people still travel to work on May 1, although it is a holiday, because some businesses like sales remain open on holidays. Second, and what particularly draws our attention, people who work on May 1 tend to travel by bicycle. Moreover, we learn that subway passenger volumes are more strongly associated with bus passenger volumes than bicyclist volumes.

8 Conclusion and future works

In this paper, we described an approach to solve the problems associated with gathering data from different sources. The TDaaS framework we proposed comprises two key steps. First, DSPs are encouraged to transform their data into the unified JSON-LD format. Subsequently, providers datasets are recognized by the DOC and data services can be deeply integrated [27]. This approach conforms to the Transparent Computing philosophy, because DSCs do not need to concern themselves with data management

details. Moreover, we have implemented a prototype system based on these theories and demonstrated its effectiveness. In future research, we will detail additional aspects of the TDaaS framework, such as the development of related toolkits and the management of data safety and authority certification.

Acknowledgements The work is partially supported by the Japan Society for the Promotion of Science Grants-in-Aid for Scientific Research (No. 25330270 and No. 26330350), and by National Natural Science Foundation of China (No. 51408018).

References

1. Toppeta D (2010) The smart city vision: How innovation and ict can build smart, livable, sustainable cities. The Innovation Knowledge Foundation Think
2. Zheng Y (2015) Methodologies for cross-domain data fusion: An overview. *IEEE Trans Big Data* 1(1):16–34
3. Wang S, He L, Stenneth L, Philip SY, Li Z, Huang Z (2016) Estimating urban traffic congestions with multi-sourced data. In: 2016 17th IEEE International conference on mobile data management (MDM), vol 1. IEEE, pp 82–91
4. Klein LA (2004) Sensor and data fusion: A tool for information assessment and decision making, vol 324. Spie Press Bellingham
5. Zhang Y, Guo K, Ren J, Zhou Y, Wang J, Chen J (2017) Transparent computing: A promising network computing paradigm. *Comput Sci Eng* 19(1):7–20
6. Zhang Y, Zhou Y (2006) Transparent computing: A new paradigm for pervasive computing. In: International conference on ubiquitous intelligence and computing. Springer, pp 1–11

7. Lanthaler M, Gütl C (2012) On using json-ld to create evolvable restful services. In: Proceedings of the third international workshop on RESTful design. ACM, pp 25–32
8. Janowicz K, Hitzler P, Adams B, Kolas D, Vardeman II et al (2014) Five stars of linked data vocabulary use. *Sem Web* 5(3):173–176
9. Sporny M, Kellogg G, Lanthaler M, W3C RDF Working Group, et al (2014) *Json-ld 1.0: A json-based serialization for linked data*. W3C Recomm:16
10. Weiser M (1991) The computer for the twenty-first century. In: *Scientific American*. IEEE, pp 94–104
11. Bizer C (2009) The emerging web of linked data. *IEEE Intell Syst* 24(5):87–92
12. Bizer C, Heath T, Berners-Lee T (2009) Linked data-the story so far, pp 205–227
13. Daniel V, Lewis J (2011) Computer scientist. An update on RDF concepts and some ontologies. <http://www.ibm.com/developerworks/xml/library/x-rdfconcepts/index.html>
14. Lanthaler M, Gütl C (2013) Hydra: A vocabulary for hypermedia-driven web apis. *LDOW*:996
15. Fielding RT, Taylor RN (2002) Principled design of the modern web architecture, vol 2. ACM, pp 115–150
16. Pautasso C (2014) Restful web services: Principles, patterns, emerging technologies. In: *Web services foundations*. Springer, pp 31–51
17. Sato A, Huang R (2015) From data to knowledge: A cognitive approach to retail business intelligence. In: 2015 IEEE International conference on data science and data intensive systems. IEEE, pp 210–217
18. Sato A, Huang R (2015) A generic formulated kid model for pragmatic processing of data, information, and knowledge. In: 2015 IEEE 12th Intl conf on ubiquitous intelligence and computing and 2015 IEEE 12th intl conf on autonomic and trusted computing and 2015 IEEE 15th intl conf on scalable computing and communications and its associated workshops (UIC-ATC-ScalCom). IEEE, pp 609–616
19. Sato A, Huang R, Yen NY (2015) Design of fusion technique-based mining engine for smart business. *Human-centric Comput Inf Sci* 5(1):1
20. Fan W, Chen Z, Xiong Z, Chen H (2012) The internet of data: A new idea to extend the iot in the digital world, vol 6. Springer, pp 660–667
21. Consoli S, Mongiovì M, Recupero DR, Peroni S, Gangemi A, Nuzzolese AG, Presutti V Producing linked data for smart cities: The case of catania
22. Xinhua E, Han J, Wang Y, Liu L (2013) Big data-as-a-service: Definition and architecture. In: 2013 15th IEEE International conference on communication technology (ICCT), pp 738–742
23. Bowen Du, Huang R, Chen X, Xie Z, Liang Y, Lv W, Ma J (2016) Active ctdaas: A data service framework based on transparent iot in city traffic. *IEEE*
24. Zhang Y, Ren J, Liu J, Xu C, Guo H, Liu Y (2017) A survey on emerging computing paradigms for big data. *Chin J Electron* 26(1)
25. Guha R (2011) Introducing schema. org: Search engines come together for a richer web. *Google Official Blog*
26. Castanedo F (2013) A review of data fusion techniques. *Sci World J*:2013
27. Ren J, Zhang Y, Zhang K, Shen X (2015) Exploiting mobile crowdsourcing for pervasive cloud services: Challenges and solutions. *IEEE Communications Magazine* 53(3):98–105



Zhipu Xie received the B.S. in Information and Computing Science from Centra South University, Changsha, China, in 2013. Ph.D. degree candidate in Computer Science and Engineering from Beihang University, Beijing, China. His research interests include smart city technology, intelligent transportation and city big data mining.



Weifeng Lv received the B.S. degree in Computer Science and Engineering from Shandong University, Jinan, China, and the Ph.D. degree in Computer Science and Engineering from Beihang University, Beijing, China, in 1992 and 1998 respectively. Currently, he is a Professor with the State Key Laboratory of Software Development Environment, Beihang University, Beijing, China. His research interests include smart city technology and mass data processing.



Linfang Qin received the B.S. in Internet of Things from JinLin University, ChangChun, China, in 2011. M.S. degree candidate in Computer Science and Engineering from Beihang University, Beijing, China. Her research interests include smart city technology, city big data mining and machine learning.



Bowen Du received the B.S. degree from Shijiazhuang Tiedao University, Shijiazhuang, China, and the Ph.D. degree in Computer Science and Engineering from Beihang University, Beijing, China, in 2005 and 2013 respectively. Currently, he is an Associate Professor with the State Key Laboratory of Software Development Environment, Beihang University, Beijing, China. His research interests include smart city technology, multi-source data fusion, big data mining and citizen travel pattern mining.



Dr. Runhe Huang is a professor in the Faculty of Computer and Information Sciences at Hosei University, Japan. She received a Ph.D. in Computer Science and Mathematics from University of the West of England in 1993. Before joining Hosei University in 2000, she worked at NUDT for 6 years and University of Aizu for 7 years in the field of Computer Science and Engineering. Her researches include multi-agents systems, computational intelligence, ubiquitous intelligence computing, cloud computing, Hyperworld modeling. She is member of IEEE and ACM.