MARSA: A Marketplace for Realtime Human-Sensing Data

TIEN-DUNG CAO, Tan Tao University
TRAN-VU PHAM, HCMC University of Technology
QUANG-HIEU VU, Khalifa University
HONG-LINH TRUONG, TU Wien
DUC-HUNG LE, TU Wien
SCHAHRAM DUSTDAR, TU Wien

This paper introduces a dynamic cloud-based marketplace of near-realtime human-sensing data (MARSA) for different stakeholders to sell and buy near-realtime data. MARSA is designed for environments where Information Technology (IT) infrastructures are not well-developed but the need to gather and sell near-realtime data is great. To this end, we present techniques for selecting data types and managing data contracts based on different cost models, quality of data, and data rights. We design our MARSA platform by leveraging different data transferring solutions to enable an open and scalable communication mechanism between sellers (data providers) and buyers (data consumers). To evaluate MARSA, we carry out several experiments with the near-realtime transportation data provided by people in Ho Chi Minh City, Vietnam and simulated scenarios in multi-cloud environments.

Categories and Subject Descriptors: Applied computing [**Electronic commerce**]: Data interchange

General Terms: Data Marketplace, Data as a Service, Realtime Data, Human-Sensing Data

Additional Key Words and Phrases: Internet of Things, Platform, Data Contract, Cost model

ACM Reference Format:
Tien-Dung Cao, Tran-Vu Pham, Quang-Hieu Vu, Hong-Linh Truong, Duc-Hung Le, Schahram Dustdar, 201X. MARSA: A Marketplace for Realtime Human-Sensing Data. *ACM Trans. Internet Technol.* X, X, Article X (June 201X), 20 pages.
DOI:http://dx.doi.org/10.1145/0000000.0000000

1. INTRODUCTION

With the advances in Internet of Things (IoT), realtime sensing data is becoming increasingly important to many real-time applications, such as smarter city [Jin et al. 2014] and city traffic management [Chuang et al. 2013; Panichpapiboon 2010]. Realtime data might come from specifically configured sensors in well-designed infrastructures for specialized applications [Jin et al. 2014] or collected from the mass of human participation, such as GPS signals from mobile devices. This type of human sensing data is readily available in public and also crucial for solving many real-life problems. However, it is challenging to gathere or share human sensing data owned by people who have different habits, knowledge/perception, income, benefits from society, and social responsibilities, etc. Apart from technical solutions, such as data storage and delivery, the platform that enables the such gathering and sharing activities needs to

Author's addresses: Tien-Dung Cao, School of Engineering - Tan Tao University, Vietnam; Tran-Vu Pham, Faculty of Computer Science and Engineering, HCMC University of Technology, Vietnam; Quang-Hieu Vu, ETISALAT BT Innovation Center, Khalifa University, UAE; Hong-Linh Truong, Duc-Hung Le, and Schahram Dustdar, Distributed Systems Group, TU Wien, Austria.

DOI:http://dx.doi.org/10.1145/0000000.0000000

have mechanisms to motivate the data owners. Previous studies showed that benefits could be the incentives for data owners to contribute their sensing data [Lee and Hoh 2010; Yang et al. 2012]. Therefore, marketplaces in which data owners are allowed to trade their sensing data for benefits are believed to be the right solution to address the challenge [Lee and Hoh 2010; Yang et al. 2012].

A number of platforms have been introduced for sharing and trading different types of data, but none of them is suitable to motivate owners of realtime human sensing data. Examples are Factual [2015], Amazon Data Set [2015], Gnip [2015], Azure Marketplace [Microsoft 2015], and Xignite [2015]. These platforms either exist in form of data marketplaces or data exchange services. The existing marketplace platforms are commonly designed for trading discrete data packages such as web-based information [Möller and Dodds 2012], graph data for structuring human knowledge [Bollacker et al. 2008], or collected data from the physical world [De et al. 2012]. These available marketplaces, however, do not support realtime data [Munjin and Morin 2012]. On the other hand, while data exchange service platforms (e.g., Compose [2015], EveryAware [2015], Ubicon [2015]) and IoT platforms (e.g., [Etherios 2014; ThingSpeak 2014; Xively 2014]) allow the sharing of realtime or near realtime data, they do not have functions of a marketplace, such as features for selling, marketing, buying and paying for data.

Our work in this paper is motivated by the lack of a right platform that could both enable the sharing and provide the incentives for owners to share realtime human sensing data. We present a new marketplace design that addresses the two challenges mentioned above: (i) be able to handle realtime human sensing data, and (ii) offer mechanisms to motivate data owners of different backgrounds actively contribute their data to the community. In addition, the new marketplace architecture is designed to interact with existing IoT platforms. To develop such a marketplace, we first study closely a real-life scenario in which realtime human sensing data, i.e., GPS data from mobile devices, is needed to address a practical traffic problem. A set of requirements derived from this motivating scenario is incorporated into the design of the marketplace architecture. A proof of concept system is built to demonstrate and evaluate capabilities of our novel marketplace.

The remainder of this paper is organized as follows. Section 2 presents a working scenario of our marketplace. Section 3 analyzes requirements, proposes the architecture and discusses several components in detail. The prototype and its experiments for a real-life case study are shown in Section 4. Related works are discussed in Section 5. Finally, Section 6 concludes the paper and discusses open issues for future work.

2. A WORKING SCENARIO

In this section, we first present an overview of a traffic monitoring system. After that, we analyze features of the system and show how our marketplace can serve as a data exchange platform for the system.

2.1. A Traffic Monitoring System

Traffic congestion is a typical problem of many big cities around the world. Traffic monitoring system is a system that continuously computes the state of traffic (e.g., fast, normal, slow, congested) at any location in a city. Once obtained, the traffic states can be utilized by different stakeholders, from police traffic department to government agencies and public, in realtime, to design solutions for the congestion problem. Generally, the traffic states can be derived by analyzing live traffic information collected from different types of sources such as traffic and surveillance cameras, signals from GPS enabled devices on cars, buses and mobile phones, and through road pressure sensors [Chuang et al. 2013], [Picone et al. 2012], [Panichpapiboon 2010]. In this work, we

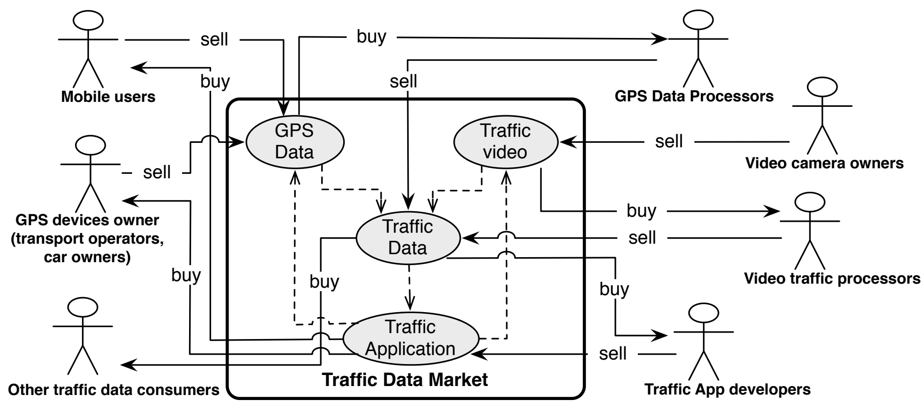


Fig. 1. Data, stakeholders and their interactions in a market oriented view of traffic scenarios in HCM City

chose to study requirements of a traffic monitoring system for Ho Chi Minh (HCM) City, one of the most crowded cities in Vietnam, as an example. This city currently suffers from severe traffic problems and urgently needs innovative solutions to address these problems. In addition to this main reason, we chose this city because of two other reasons:

— Traffic in HCM City is not only crowded but also mainly composed of motorcycles whose drivers' behavior are hard to predict. When an traffic incident happens, it will soon get worse if traffic police cannot take prompt actions. These distinct characteristics make HCM City very different from other cities in developed countries, and hence solutions that are proven to be effective in cities of developed countries cannot be applied in this City.
— Realtime data is essential to solving traffic problems. However, HCM City currently does not have a systematic method to collect traffic data, even though various sources of realtime data that can be utilized for solving traffic problems such as GPS signals from buses, taxis and mobile phones, videos from surveillance cameras, etc. are available. Therefore, it is reasonable to study closely the traffic system in this City to see how a realtime data marketplace can be employed to improve the collection of traffic data.

2.2. A Marketplace for Traffic Data Exchange

Effective solutions to traffic problems require the integration of many sources of data. However, with the current condition of Vietnam economy, it is almost infeasible for the government to buy expensive traffic data collection systems for its major cities. Fortunately, in Ho Chi Minh City, data that can be utilized for solving traffic problems currently exist in different forms and are owned by different parties. For example, the Voice of Vietnam, a broadcasting agency, has many video cameras installed at major intersections to monitor the city traffic, and broadcasts the information through radio. The traffic police department also maintains a number of surveillance cameras on streets to detect traffic violations. Major taxi, truck and bus operators in the HCM City have GPS-enabled devices installed in their cars, as required by the government to keep track of their fleets. Besides, mobile users with GPS-enabled smartphones are also valuable sources of GPS data.

Even though many different valuable sources of data are currently available, the use of these data sources are limited within the scopes of their designated applications. Thus, the challenge is how to motivate the owners of these data sources to share and make them available to those in need to solve traffic issues. The data owners need to have something to compensate for their investments on data collection equipments as well as human resources to keep the equipments operating. A data marketplace where the owners have the incentives to trade their data for benefits could be an appropriate solution in this case [Lee and Hoh 2010],[Yang et al. 2012].

The integration of data sources in the above scenario can be carried out via data exchange in our marketplace, as shown in Fig. 1, in which users can trade their traffic data for some benefits. In the marketplace, the trading is usually achieved via intermediate values, e.g., credits or money (commonly referred to as credits in later discussions). In particular, if a data provider has some data to contribute (e.g. a mobile user with live GPS data), the provider can trade the data for some credits with data consumers. These credits can later be used to exchange integrated data or services from others (e.g. processed information from traffic application providers). With the marketplace, data providers will no longer be limited within the scopes of designated applications, where data has to be directly exchanged for services as in the case of existing traffic applications. Instead, users can freely exchange their data with other participants in the market. As a result, owners of data sources will be motivated to contribute their data. The marketplace also encourages third party data processors (e.g. GPS data processors), who can use their knowledge and tools, to buy raw data (e.g. GPS signals), integrate and convert the data into more valuable data (e.g. traffic data) and sell it back in the market for profits.

To summarize, Table I shows benefits from processed traffic data and costs of maintaining the collection of raw data of various data owners. For each owner, the cost and the benefit are not always balanced. If the cost is greater than the benefit, the owner may hesitate to directly exchange raw data for processed traffic data. However, in the market context, by exchanging via intermediate values, the cost and benefit of each data owner can be balanced. This could motivate the owners to contribute their data to the market.

Table I. Costs and benefits of parties involved in the traffic scenario

Parties	Costs of collecting raw data	Benefits from processed traffic data
Bus, taxi and truck operators	GPS devices, Internet and mobile network subscription fees, acquiring and maintaining data at servers	Able to track status of their buses, knowledge of current traffic conditions to better provide services to commuters
Private car owners	GPS devices, mobile network subscription fees	Knowledge of current traffic conditions to better navigate in cities
Mobile device owners	Mobile devices (e.g. smartphones, tablets), mobile network subscription fees and device battery time	Knowledge of current traffic conditions to better navigate in cities
Video camera owners	Video cameras and network connections to video cameras	Selling of video data and traffic information
Data processors	Cost of raw data, infrastructures for collecting and processing raw data	Selling traffic data
Traffic data users	Buying traffic data	Knowledge of current traffic conditions to better navigate in cities

3. REALTIME HUMAN-SENSING DATA MARKETPLACE

In this section, we will first analyse the characteristics of near-realtime human-sensing data and the needs from various participants in a marketplace in Section 3.1. After that, we present our design for the marketplace in Section 3.2. Finally, we will discuss how data discovery is supported in Section 3.2.2 and how data contract is managed in Section 3.2.3.

3.1. Requirements of Near-Realtime Human-Sensing Data Marketplaces

Generally, there are two primary types of participants involved in a data marketplace: Data Providers and Data Consumers, who sell and buy the data respectively. There could be another type of participants who buy data from the marketplace, process data to add values to it, and re-sell the processed data to others. In the context of a realtime data marketplace, this type of participants is referred to as Intermediate Data Processors. Depending on activities that Intermediate Data Processors perform in the marketplace, they can expose themselves as Data Providers or Data Consumers.

A data marketplace in general must support buying and selling activities of data providers and data consumers such as listing and discovering of data, pricing the data, negotiating and managing data contract, making payment, and transferring data from providers to consumers. However, near-realtime human sensing data and their owners have some distinct characteristics that a data marketplace designed for near-realtime data needs to support.

— *Near-realtime*: the sensing data has its highest value when it is delivered from consumers to buyers immediately. The total time from the creation of data at sources to the point when the data is delivered to the consumers must be reasonable small so that it can be processed and timely decisions to react to the current situations can be made.
— *Streaming*: the data is usually delivered continuously from sellers to buyers in forms of streams of events.
— *Heterogeneity of data providers/consumers*: providers of near-realtime data can be organizations or businesses such as bus or taxi operators (big/commercial providers). They can also be individuals such as people with mobile phones, who have limited computing infrastructures (small/personal providers). For example, in the case of the working scenario for HCM City illustrated in Section 2, while the data from bus and taxi operators only cover main streets, the data from small providers, such as individuals with mobile phones, has much larger coverage. Therefore, the contribution of small data providers plays a significant role to the success of the system. In addition, because of the heterogeneity of the providers, the volumes and the formats of the data streams are also varied. Similarly, data consumers are also diverse
— *Heterogeneity of data quality*: because of the heterogeneity of data providers, the quality of data is also varied. Data quality of the same provider may also be varied through time, as it is depended on other factors, such as network infrastructure and habits of data owners

These characteristics have strong influence in the design of the data marketplace. In particular, the marketplace needs to address the following requirements:

— *Efficient mechanisms for the providers to distribute near-realtime data to the consumers immediately*: different from marketplaces for discrete data packages where the data packages from providers can be stored statically on servers for consumers to download, a marketplace for near-realtime sensing data needs to deliver data immediately from providers to consumers. Near-Realtime data will loss its value if it is stored for a long time in servers.
— *Data stream transferred from providers to consumers in both push and pull modes*: in the *push* mode, the provider actively makes a data connection to the consumer to push the data through this connection as a stream. The *pull* mode happens in an opposite way in which the consumer initiates the data transfer by making connection to the data provider to get the data. For small data providers, e.g., individuals, the *push* mode is more suitable. However, for big data providers, such as taxi operators, who

usually have sufficient computing infrastructure to collect raw GPS data from their fleet and provide the data to the market, the *pull* mode may be more appropriate.

— *Different mechanisms for listing data, pricing the data, negotiating contract and making payment*: Big data providers may make data available through streaming servers, and list information about the streaming servers, together with information about the data, prices, etc. in the marketplace. A consumer, once agreeing on the listed prices, can make a data contract and connect to the server to stream the data. Similarly, big data consumers may also setup servers to receive data, then list the data they want to buy, prices, etc. on the market. If a data provider wants to sell data with the listed prices, he/she can connect to the server of the data provider to transfer the data.
— *Mechanisms for monitoring and assuring data quality*: as near-realtime data is usually delivered in form of data streams over a period of time through the Internet, a selling-buying transaction cannot be completed at the time of buying but has to last for a period of time. Different from other kinds of goods, the quality of near-realtime data (the goods) cannot be verified at the time the provider and the consumer agree on the deal. The quality of data may also fluctuate over time. Hence, there is the need for a mechanism to monitor the quality of data during the data transferring process.
— *API for interactions with data providers and data consumers*: near-realtime data is usually not consumed directly by human beings. Instead, it is fed to applications at the consumer side for further processing. Therefore, there is also a need for an API or a set of services so that applications of end users can be easily integrated with the data marketplace.

3.2. The MARSA Architecture

Based on the requirements identified in Section 3.1, we have developed an architecture for the realtime human sensing data marketplace, named as MARSA, shown in Fig 2. The core of the marketplace is a set of services which deliver the main functions of a market, including *Data Discovery*, *Cost Model Management*, *Payment*, *Data Contract Management* and *Data Quality Analysis*. Besides, *Data Contract Model Management* and the *External Model Management* services are also included in order to support a variety of contract models, due to the homogeneity of data providers and consumers. A Graphical User Interface (GUI) is provided for user interactions with these core services. A set of APIs are also provided so that external programs can interact with the marketplace. In this reference architecture, in order to address the need for streaming realtime data from providers to consumers, we utilize available realtime databuses provided by the current IoT platforms or Data Distribution Service (DDS) middleware [OMG 2007; Al-madani et al. 2013]. By using available databuses, existing applications can easily interact with the marketplace through provided APIs with minimal modifications. Main services and the way that different components of the architecture interact with each other are discussed in detail in the following sub sections.

3.2.1. Marketplace Orchestration. For *pull* model, the data providers first use the *Cost Model Management*, the *Data Contract Model Management* and the *External Model Management* to specify the provided cost and general information of data contract, such as its quality and data rights, and then publish their data description to the *Data Discovery*. Next, they publish their data to the *Databuses* where their data will be delivered immediately to data consumers. The data providers can also access the *Data Quality Analysis* service and the *Payment* service to get a report about quality of data and payment status. On the other side, through the marketplace, data consumers can use the *Data Discovery* service to find their required data by submitting a set

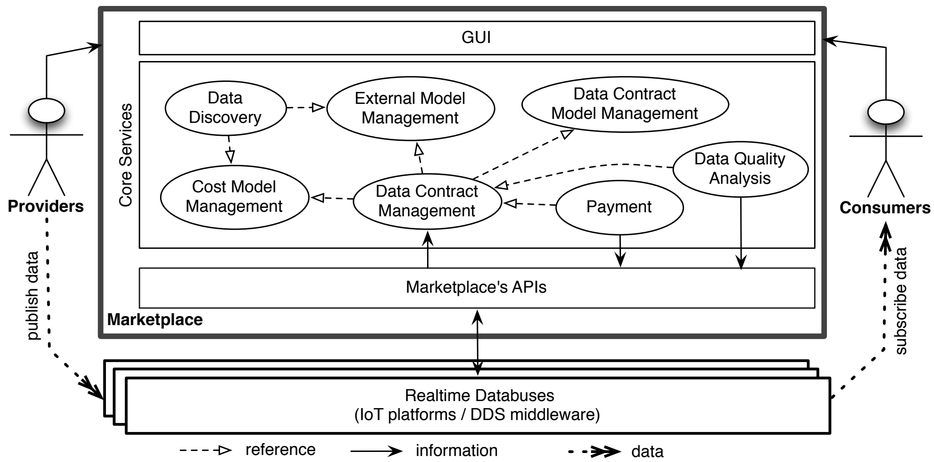


Fig. 2. General design of MARSA

of requirements (e.g., keywords for the data, expecting price of the data, and data quality). If their required data are found, they can subscribe (i.e., choosing the payment model, quality of data model, and accepting the data rights) to use one or several data services. At this step, a data contract/agreement based on the matched properties among service description and customer requirements will be generated and managed by the *Data Contract Management* service. While data transferring process between provider and consumer is executed, through the *APIs*, the *Payment* service and the *Data Quality Analysis* service work as the intermediates to calculate data value to produce the bill and to monitor the quality of data. For *push* model, the role of providers and consumers are inverted.

3.2.2. Data Discovery. The function of Data Discovery service is to allow users (i.e., providers and consumers) to publish and search for desired data streams. Metadata is commonly used to enable automatic data/service lookup [Spillner and Schill 2013; Segura et al. 2014]. However, existing data/service lookup approaches consider data sets and services as discrete and static entities. Therefore, their metadata sets are not appropriate for near-realtime data delivered in streams. Thus, in this work, we leverage DEMODS [Vu et al. 2012] to describe information about data and services for near-realtime human-sensing data.

From the analysis of data owners (Table I), we observe that there exist two main levels of data and service description: (1) *Data Service* level in which the general information of a group of data streams of a provider (or an intermediate provider) is specified; (2) *Data Stream* level in which data information of individual devices is specified. To accommodate these descriptions, we extended DEMODS with: (i) Adding a *device* field which links to an external model to describe the devices; (ii) Adding a *data origin* field to distinguish the raw data of devices and the processing data of re-seller; (iii) Introducing *time* properties, such as data rate, latency and time series; (iv) Redefining *cost models* and *contract models* that are discussed more in the next section; (v) Replacing the *data* field definition with a *data type* model that supports more data types such package-based data and streaming-based data; and (vi) In both level, i.e., Data Service and Data Stream, the databus is also specified to publish/subscribe data. Fig 3 defines structures for necessary items of two main levels, i.e., service and data stream. Some of items (e.g., categories, QoS, Data Types or Devices) refer to external models

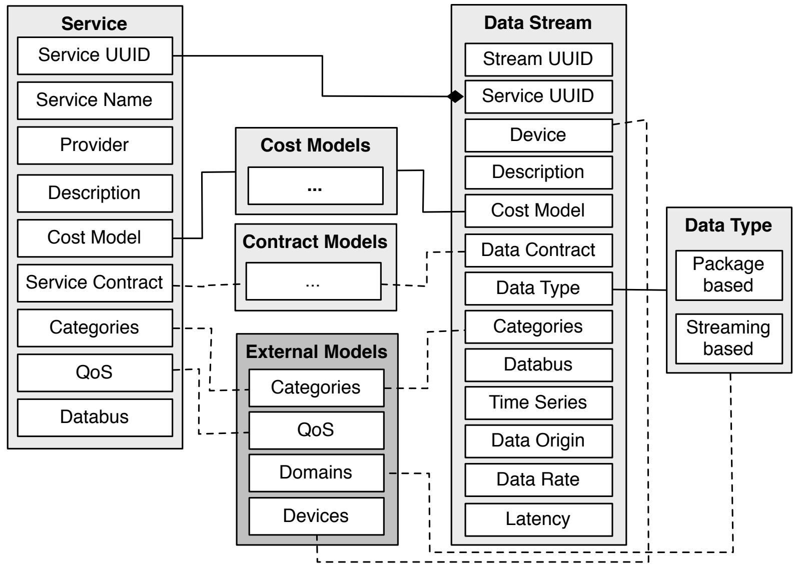


Fig. 3. An overview of general description model for data and services in MARSA

whereas the cost and service contract refer to models defined in Section 3.2.3. Table II describes the detail of these items.

3.2.3. Data Contract Management. Data Contract Management service is designed to handle contracts agreed between providers and consumers. Contracts can be in different forms. Each contract usually consists of terms and conditions that govern the quality of data and the associated costs. As shown in Fig 2, *Data Contract Management* service refers to *Cost Model Management*, *Data Contract Model Management* and *External Model Management* services in order to manage various components of a contract. Section 3.3 discusses the various contract models, quality models and cost models that the data marketplace is designed to support.

3.2.4. Data Quality Analysis. As the quality of realtime streams might vary through time, Data Quality Analysis service is designed to continuously monitor and analyse quality of data streams. Reports produced by this service are the basis for verifying data quality specified in contracts.

3.2.5. Payment. The Payment service continuously monitors the data streams in the databuses. Together with the terms and conditions in data contracts that providers and consumers have previously agreed, the Payment service produces data bills and provides facilities for making transactions between providers and consumers.

3.3. Data Contract and Related Components

A data contract often includes five basic components including *data rights*, *quality of data*, *regulatory compliance*, *cost model* and *control and relationship* [Truong et al. 2012]. The most important component in a data contract for the data marketplace is the cost model which represents a generic way in which the cost, the time, the data size and the number of transactions are specified.

3.3.1. Contract models. The marketplace is designed to support the following four basic contract models:

Table II. Description model for MARSA

Level	Properties	Description
Service Description	Service UUID	+ Service Universal Unique Identifier
	Service Name	+ The data service name
	Description	+ The detail data service description
	Provider	+ The data owner
	Cost model	+ List of cost models where data owners offers to their consumers, it links to the cost model defined in Section 3.3.3.
	Cont. model	+ List of contract types where data owners offers to their consumers, it links to the contract model defined in Section 3.3.1.
	Categories	+ Linking to a category model defined by outside the platform which supports for service/stream discovery
	QoS	+ Description of Quality of Service, it also links to a quality of service model defined by outside the platform
	Databus	+ The place/address that consumers can subscribe all data of service.
Data Stream	Stream UUID	+ Stream Universal Unique Identifier.
	Service UUID	+ The data service owner of stream.
	Description	+ The detail description of stream.
	Device	+ The description of device which generate the data. It links to a device model which supports for Data Service/stream discovery.
	Cost model	+ A stream may offer different cost model from its Data Service.
	Cont. model	+ A stream may offer different contract model from its Data Service.
	Data type	+ Describe the data type of stream, it is classified into object based and non-object/streaming based and links to a model defined by outside the platform.
	Categories	+ A stream may belong to different categories with its Data Service.
	Databus	+ The place/address that consumers can subscribe the data of stream.
	Time series	+ Use if data is object based. The time series that is represented by a pair (x, $\pm u$), in which the first one is a positive integer number measured at successive points in time spaced at uniform time intervals (default in second) and the second one is its uncertainly should be given.
	Data origin	+ Describing the data comes from, for instance, self-generated, collected or location of device.
	Data rate	+ The data rate from data provider side.
	Latency	+ Maximum time delay from data provider to consumer.

- **Obligation-free contract**: is a type of data contract which does not require involving parties to have any obligation to conform to terms and conditions specified in the contract. For example, contracting parties are not required to follow any restriction on data rights. They do not need to provide any guarantee on quality of data, etc.
- **User-centric contract**: is the type of data contract, which focuses on requirements of a service that the service provider has to deliver to users. This contract model contains not only elements to determine the quality of data, such as completeness and accuracy, but also elements to specify the support and indemnification of the service provider in cases of failure. For example, a typical user-centric contract for traffic information would have constraints on the latency of information (e.g., less than 5 minutes), the sample rate (e.g., 10 samples per minute), and the accuracy of sample location (e.g., less than 20 meters in geographical distance error).
- **Provider-centric contract**: is the type of data contract with requirements on data rights and regulatory compliance of the data that users have to follow. In particular, this contract specifies key requirements for using the data (i.e., data rights) and rules on data that are required to obey (i.e., regulatory compliance). For example, constraints about the re-redistribution of traffic information in a provider-centric contract could be no re-distribution for commercial purposes and re-distribution without modification of the data and with a display of the service provider name for non-commercial purposes.
- **Customizable contract**: allows users to modify any of the above contract models. On the one hand, a customizable contract can be developed from any contract model

by adding constraints to it. It can also be developed by mixing above contract models and even modifying the mixed model after that. On the other hand, it is possible for service providers or users to define their own requirements or constraints to add in the contract model. Since a contract is often expected to include constraints from both parties and both users and providers always have some specific constraints to include, customizable contract model is the most popular model used in practice.

3.3.2. Quality of Data models. In a data service, the quality of data (QoD) has a strong relationship with the cost because, in any market, the cost usually goes with quality of data, product or service. Based on QoD models which consider the data in many aspects (e.g., accuracy, completeness, and timeliness) the data provider can define the appropriate cost for their data service. Actually, linking QoD to the data cost is important to prevent providers from increasing the data quantity by adding a lot of noise, e.g., replicated data, out-of-date data or data already on the protocol-layer. Since QoD models depend on many data properties, we suggest the providers to use external models [D'Ambrogio 2006; Dobson et al. 2005] to define and monitor their QoD.

3.3.3. Cost Models. When a data producer sells data to data consumer, the data value will be based on cost. Several cost strategies have been summarized by Muschalle et al [2012] and Schomm et al [2013]. However, in the context of realtime human-sensing data, it is not completely suitable because at different moments and different locations, a data stream may have different values. To adapt to different business models of different consumers/providers for both the *pull* and *push* models, we support the following payment strategies.

— **Payment on package delivering (API handle):** data can be split into separated packages (e.g., messages or images). Consumers are charged every time they successfully receive a number of packages from the marketplace. To describe this cost model, the usage fee of fixed number of packages has to be included in the API description.
— **Payment on data size:** consumers are charged on the size of received data. Similar to the previous case, the basic unit charge fee for a data unit (e.g., 1 MB, 10 MB, 100 MB or 1 GB) should be described.
— **Payment on time of subscription:** providers can split a day or a week into different moments and the corresponding cost for each unit of time (e.g., 1$ for one hour of subscription within from 1 pm to 5 pm) is set up. Consumers are then charged on total time of each moment they subscribed. This model is appropriate for streaming data where the data is generated in a duration.
— **Payment on data unit:** providers split their data in different data units and set up the basic unit charge free for each. Unlike payment on data size, the data unit here can be split not only by data size but also by time and over a group of data streams. Consumers will pay one time and get the data until reaching the limitation of unit.
— **Payment on plan (fixed payment on a period):** consumers subscribe to use data in a subscription period (e.g., a week or a month) and only pay one time for this period with or without maximum limitation of received data. The basic unit charge fee for an unit of time (e.g., 1 hour, 1 day, etc.) is also described.
— **Free users:** in a number of cases, consumers can use these services at no charge from providers. There are some reasons to offer a service for free such as: (i) the data comes from the government and the consumer is a public authority funded by tax money. This case is usually constrained by a data contract. (ii) a person or an organization can also provide the free data as a social responsibility because the generating data fee is supported by the other organization or the government.

To encourage the consumers, for each payment strategy above, we support a mechanism, called **Freemium**, in which the providers offer a limited access (e.g., the limit of package quantity, the data size or a duration) at no cost at the beginning. Besides, the provider can define several requirements that allow their consumers to get a discount (e.g., at specific subscription times on the bill or the data size). Finally, for certain types of data or service, the above cost models can be expanded to include a free payment option with popup advertisements for data consumers (i.e., the data can be provided free-of-charge in exchange for some advertisements).

3.3.4. Contract Governance. A data contract, once established, is governed via two processes: data contract monitor and data contract enforcement. Data contract monitor is used to monitor requirements specified in the contract. While our framework provides a basic monitor, it is possible to allow a third-party to be involved in the monitoring process. For example, the quality of data delivered in a service can be monitored and assessed by a third-party and the result is reported to our framework. On the other hand, data contract enforcement makes sure that the contract is followed strictly by both data providers and data consumers. In cases violations occur, different types of penalty can be applied such as a warning, a temporary suspend of the service, some penalty fees, or even a termination of the service.

4. IMPLEMENTATION AND EXPERIMENTS

4.1. Marketplace Prototype

For demonstration purpose, a prototype[1] of the reference architecture described in Section 3 has been implemented. In this prototype, we fully implemented two basic services: *Data Discovery* and *Cost Model Management*. While the first one was a revision of a previous work [Vu et al. 2012], the second service was a new implementation. A set of APIs have also been built for external databuses such as Mosquitto[2] and Xively IoT platform [Xively 2014] to interact with the marketplace via Web services. These APIs are used to enforce the cost model and support the latency analytics. Mosquitto and Xively IoT platform were chosen to illustrate the interactions between the marketplace and external databuses because they implemented a lightweight broker-based publish/subscribe messaging protocol, i.e., MQTT[3]. A simple version of *Payment* service was also implemented. This service uses data log captured by Databus via APIs and cost managed by Data Contract Management to produce online bills. To prove the realtime capability of the design, we implemented, as a part of *Data Quality Analysis* service, a latency analytics that allows us to measure the near realtime capacity of databuses. An overview of the MARSA prototype is shown in Fig 4. Based on our work, a full version has been also implemented by the TMA Research[4].

Together with core services, cost models used in data contracts as discussed in Section 3.3.3 were also implemented. The class diagram in Fig 5 describes the relationship among payment plans used in cost models in the implementation. Each payment plan was implemented as a class, which was a subclass of *Cost Model*. Even though all payment plans inherited Price property from Cost Model, the use of this property was slightly different, depending on the actual payment plans. For example, with Data Size plan, it was used to represent the price of one data unit (e.g., 1Kb, 10Kb, and 1Mb). In case of Time Plan and Subscribe Plan, it was the price for one time unit (e.g., 1 hour, 1 day or 1 week). With Data Unit plan, it was the cost of using data streams.

In the current prototype, once a consumer subscribes a data service and chooses a particular payment plan, the marketplace enforces the plan by continuously monitor-

[1]The platform is released as an open source under http://dungcao.github.io/marsa.
[2]http://mosquitto.org/
[3]http://public.dhe.ibm.com/software/dw/webservices/ws-mqtt/mqtt-v3r1.html
[4]http://www.tmaresearch.com.

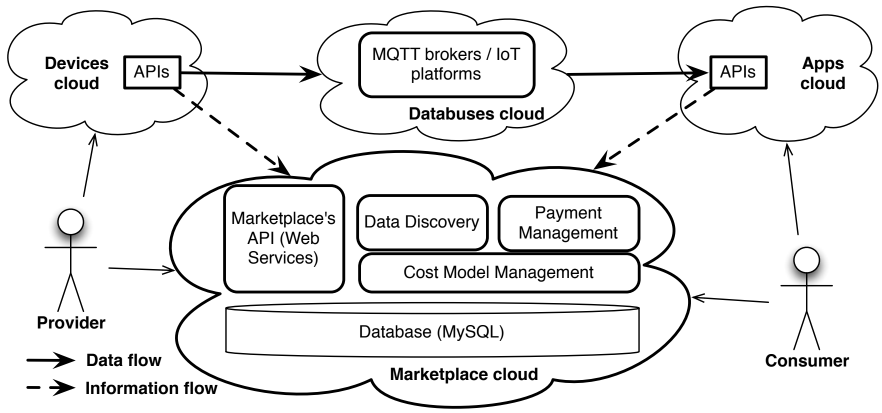


Fig. 4. The MARSA prototype

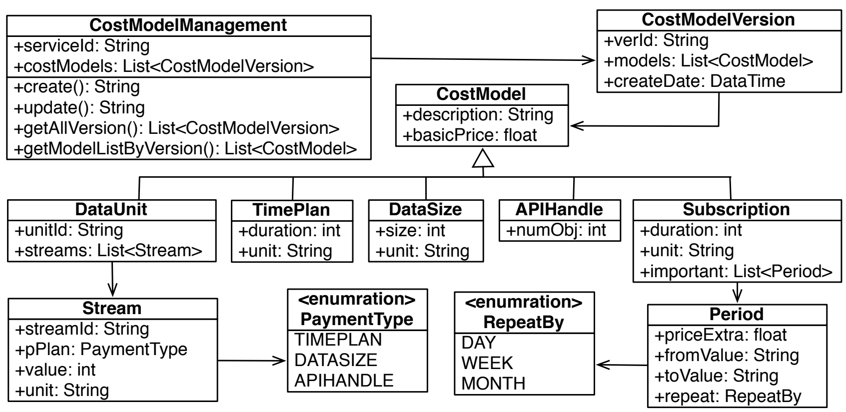


Fig. 5. Class diagram of cost model management

Table III. List of marketplace APIs

Methods	Description	Use for plan
+ subscribeStart()	+ Used when consumer starts to subscribe to a stream.	Subscription
+ subscribeEnd()	+ Used when consumer unsubscribes a stream.	Subscription
+ getTimePlan()	+ Return either the period if consumer chosen the time plan as the payment strategy or null. After calling this function, databus must control the deadline upon the time period it returns	Time Plan/ Data Unit
+ packageCount()	+ Whenever a package is delivered to consumer, the function is called only one time. In case, the current stream touched the limitation (if it's set), this function will return false.	API Handle/ Data Unit
+ setDataSize()	+ Set the transferring data size of objects or data stream within a duration. In case, the current stream touched the limitation (if it's set), this function will return false.	Data Size/ Data Unit

ing and analysing information collected from databuses via the APIs. Depending on the type of the payment plan, the necessary information for enforcing the plan may be different. Table III lists a set of provided APIs and the types of information these APIs can collect for enforcing different payment plans.

4.2. Comparison with other Platforms

There exist a number of data marketplaces. In this section, we compare our proposed marketplace with some popular marketplaces to identify the strengths and weaknesses of each marketplace with respect to their support for realtime human sensing data. Features of a realtime data marketplace, including: supported data types, data sources, data publishing and delivery method, cost model, automatic data lookup, data contract and payment management are used as the main criteria for our comparison. Through this comparison, we want to demonstrate that our solution meets typical requirements of a data marketplace for realtime IoT/human-sensing data. The requirements include reusing the IoT platform for data publishing and delivery, providing flexible contract and cost models for data streams to meet the needs of different data providers and consumers, supporting automatic data look-up services to support different ways of streaming data, and especially the capability to produce online billing in near realtime.

Table IV. Comparison of data marketplaces

Products	**Data type**	**Data source**	**Publishing /delivery**	**Cost model**	**Auto-lookup**	**Data contract**	**Payment**
MARSA	Realtime, streaming	IoT devices	MOM[5], IoT platform	Flexible[6]	yes	yes	online billing
Xignite	Datasets, realtime	Range, finance	Files, API	Asset, delivery	Yes	N/A	N/A
Amazon	Datasets	Range	Files	Free	N/A	N/A	N/A
Azure	Datasets	Range	OData API	Subscription	N/A	Publisher offer terms	N/A
Factual	Datasets	Geography	Files, API	Free/ subscription	Yes	Terms of services	N/A
Trimble insphere	Datasets	Geography	Files	per user/ device/ data	N/A	License Agreement	N/A
Gnip	Realtime, historical	Social network	API	N/A	Yes	N/A	N/A
Sense2Web	Realtime, streaming	IoT devices	MOM, IoT platform	N/A	Yes	N/A	N/A

4.3. Application to a Real-life Case Study

A concrete application, named Traffic Information System[7] (TIS) for HCM City, of urban traffic systems scenario in Section 2 was analysed to show the usefulness of the data marketplace. The main functionality of this system is to provide information about velocity of vehicles on roads of the city in realtime. With this system, the users can make a good plan for their travel in the city, for example, to avoid traffic jams. This system receives raw GPS data (i.e. longitude, latitude, velocity, and timestamp) from GPS devices attached to city buses, taxis and GPS enabled mobile phones in realtime. The system then estimates possible travel speeds that vehicles are moving on the roads. The result is displayed on an online city map, in which the possible travel speeds are represented by colored lines. Generally, the system consists of three main components involved in processing realtime data: *GPS Data Receiver, Traffic Data Generator* and *Traffic Visualizer*. The *GPS Data Receiver* receives realtime raw GPS data from external sources. *Traffic Data Generator* converts raw GPS data into traffic data,

[5]MOM: Message Oriented Middleware.
[6]The providers are very flexible to define their data value by data type, data size or subscription time, etc.
[7]http://traffic.hcmut.edu.vn/

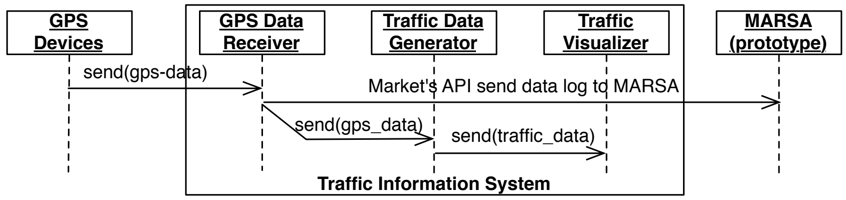


Fig. 6. Interactions between Traffic Information System and the data marketplace (i.e., MARSA)

i.e. possible traveling speeds of city roads, and provides the traffic data to *Traffic Visualizer*. The following description will show how MARSA prototype can be used to facilitate the exchange of data related to the TIS scenario. The overall interactions are described in Fig 6, in which the GPS Data Receiver is considered as a databus of marketplace. We integrated the marketplace APIs into this component to count the amount of data and send it to the marketplace.

4.3.1. Market Interactions. The TIS plays a dominant role in this interaction scenario, as it is the major consumer of the data. Therefore, the price of GPS data is pre-defined by the TIS owner using the *Cost Model Management* service. The information of databus is also specified by the TIS owner. To sell their raw GPS data, GPS data providers such as bus and taxi operators get into the data marketplace and provide a set of information to the data market as described in Fig 3. After registering with the market, the data providers can use the APIs provided by the TIS owner to connect to the GPS Data Receiver to upload their data. It is assumed that the providers have agreed on the terms, conditions, and the cost policies set by the TIS owner. The marketplace APIs integrated in the GPS Data Receiver of the TIS will count the amount of data that the system receives from each provider and regularly updates the information to the data market. The Payment component of the market uses this information for accounting purposes, and decides on the cost the TIS owner has to pay for the data providers. For small data providers such as commuters with mobile devices, a mobile application is built for sending GPS data to the GPS Data Receiver of the TIS. Interactions between the providers and the market as well as the TIS, such as accounting and billing information, registration, etc, are done through this application.

4.3.2. Discussion. The example of the TIS, a practical application of urban traffic scenario, has demonstrated the use of the dynamic data market for an realistic application. Currently, the TIS receives GPS data from around 4000 city buses everyday. On average, each bus sends about 0.25MB of data to TIS GPS Data Receiver per day, equivalent to 7.5MB of data per month. For the whole city bus fleet, the total amount of GPS data received per month is about 30GB. We can now use the data free of charge for research purpose. However, if each MB of GPS data costs 20 US cents in the market, the bus operators will receive around 6000 US dollars for the whole bus fleet. This is an accountable amount, and enough for the bus operators to pay for the cost of 3G/GPRS data connections used to send GPS data to data gathering servers. For mobile users, if each mobile phone sends the same amount of GPS data to the TIS as a bus, the providers will receive approximately 1.5 US dollars per month for each GPS enabled device. Even though this is a relatively small amount, it could be used to pay for half of a 3G data bill for mobiles in the current condition of Vietnam. This illustrated example shows that the data market model can bring some benefits to the users. A number of studies have shown that benefits can be used as incentives to encourage mobile users to participate in the marketplace and contribute their data [Lee and Hoh 2010; Yang et al. 2012].

Technically, as the marketplace is currently built based on a service oriented model, it has certain limitations that affect the flexibility of the market. For example, it is hard for small data owners, such as providers with GPS enabled devices, to sell their data directly in the marketplace. As the realtime data streams must be provided through data services, data providers have to gather their data and then publish the data to some forms of databus so that data consumers can access the data. However, small data owners, such as ordinary mobile users, do not usually have their own computing infrastructure as well as necessary computing skills to do so. The specific GPS data collection application developed for mobile devices used in the TIS application is a walk-around solution to help ordinary mobile users to easily contribute their data.

4.4. Experiments in Simulated Scenarios

4.4.1. Experimental Scenario and Settings. The main purpose of our experiments is to evaluate the practicability of MARSA according to the real world scenarios[8]. We used Mosquitto, an open source MQTT message broker, as the databus.

— Infrastructure setting: As a proof of concept that MARSA works for the scenario in Section 4.3, we deployed one databus in a virtual machine (VM) having 2 CPU Authentic AMD, 2.80GHz, 8GB of RAM and running on Ubuntu Server 12.04.2 LTS 64-bit of our partner cloud (Flexiant[9]). 200 simulated sensors deployed in 4 VMs of the same cloud with the databus and 200 simulated consumers were deployed both inside and outside the cloud of the databus, i.e., three servers in Vietnam (Tan Tao University, HCM City University of Technology and Hanoi University of Technology) and our partner clouds around Europe such as Stratuslab (in France), Distributed Systems Group (DSG lab) at TU Wien (Austria) and also in Flexiant FCO cloud. The simulated sensors publish a text file in which each line is about 100 bytes, to the databus by reading line-by-line after every 5 seconds. On the opposite side, whenever receiving a message, the simulated consumers save it into a text file. We also used 5 personal computers (PC) located in Vietnam as video streaming sensors. These sensors publish a video data which was captured by cameras and whose sound was removed. The simulated camera consumers were deployed on other PCs located in DSG lab. This setting allows us to verify the near-realtime capability on different environments and the capability of the platform to deal with different data types. Fig 7 depicts our setting across distributed sites.
— Cost setting: According to the above setting, we have two types of data (i.e., text/message and video streaming). We assume that 200 simulated sensors and 5 video streaming sensors are owned by a provider, and he/she set the cost of data as follows. For 5 video streams, the value of data is evaluated by subscription time (e.g., 2$/hour/stream). For the text streams, the value of data is evaluated by data size (e.g., 5$/1GB).

Based on the above-mentioned settings, we analyzed how well the platform measures near-realtime capability and evaluates data values. To support this, the marketplace APIs were integrated into all simulated sensors and simulated consumers to capture data log and notify the marketplace whenever a data package is sent/received.

4.4.2. Near-realtime Capability Measurement. In this experiment, we followed the first setting to test the prototype with two types of realtime data related to the traffic scenario, including structured data (i.e., raw GPS) and unstructured data (i.e., video data).

[8]In our experiments, MARSA and its APIs (i.e., Web services) were deployed at http://109.231.124.57:8080/marketplace/default and http://109.231.124.57:8080/ws
[9]http://www.flexiant.com/

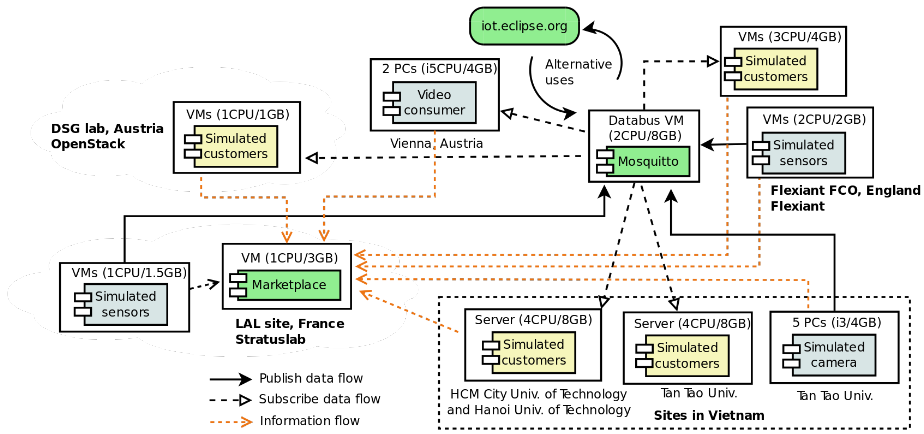
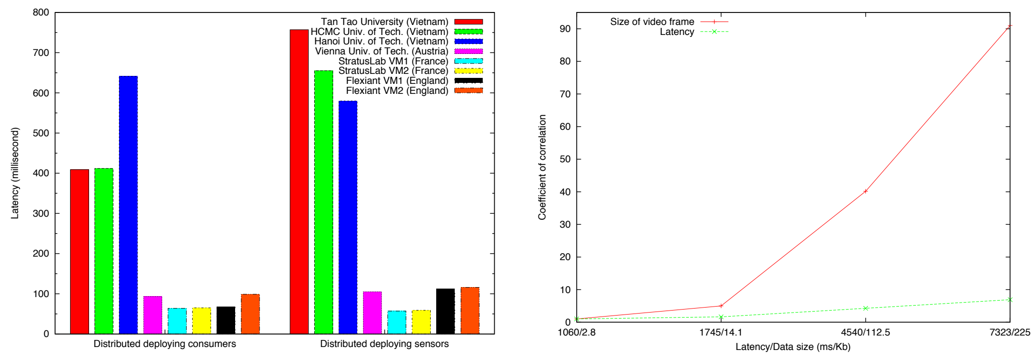


Fig. 7. General resource settings for experiments



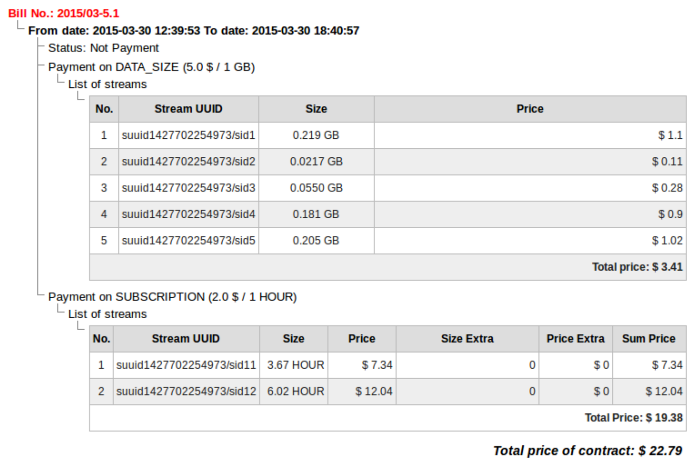
(a) Latency comparison from different locations around the world

(b) Coefficient of correlation between the latency and the size of video frames

Fig. 8. Realtime capability measurement

Then, we measured the average latency of each type of data caused by the databus for each consumer. The latency is calculated by the difference between two timestamps of a data package recorded by the platform (i.e., MARSA): the time when the source device publishes notification and the time when the consumer receives notification. This avoids the problem of time difference between internal clocks since we have too many source devices and consumers. For raw GPS data, on the left of Fig 8(a), we show the realtime capability of some consumers deploying in different clouds to subscribe for the data published by the simulated sensors deployed in the StratusLab cloud. On the right, we inverted the location of the simulated sensors and the simulated consumers, i.e., 200 simulated consumers were deployed in two VMs of the StratusLab cloud and their simulated sensors were deployed in the different clouds and the servers. For the video data, we are interested in the coefficient of correlation between the latency and the size of data package (Fig 8(b)) when we changed the size of video frames. These results show that our marketplace is acceptable for realtime applications.

4.4.3. Data Value Evaluation. In this section, we show how the platform evaluates the value of data in near-realtime. There are many consumers who bought the 200 text streams and 5 video streams above, however, we assume that there is one consumer



Bill No.: 2015/03-5.1

From date: 2015-03-30 12:39:53 To date: 2015-03-30 18:40:57

Status: Not Payment

Payment on DATA_SIZE (5.0 $ / 1 GB)

List of streams

No.	Stream UUID	Size	Price
1	suuid1427702254973/sid1	0.219 GB	$ 1.1
2	suuid1427702254973/sid2	0.0217 GB	$ 0.11
3	suuid1427702254973/sid3	0.0550 GB	$ 0.28
4	suuid1427702254973/sid4	0.181 GB	$ 0.9
5	suuid1427702254973/sid5	0.205 GB	$ 1.02
			Total price: $ 3.41

Payment on SUBSCRIPTION (2.0 $ / 1 HOUR)

List of streams

No.	Stream UUID	Size	Price	Size Extra	Price Extra	Sum Price
1	suuid1427702254973/sid11	3.67 HOUR	$ 7.34	0	$ 0	$ 7.34
2	suuid1427702254973/sid12	6.02 HOUR	$ 12.04	0	$ 0	$ 12.04
						Total Price: $ 19.38

Total price of contract: $ 22.79

Fig. 9. A Screenshot of online bill

who bought 5 text streams and 2 video streams in a contract. Based on the data log notified by the marketplace APIs and data cost defined by provider, i.e., 5$/1GB for text and 2$/hour/stream for video streaming, the marketplace produced a online bill whenever the consumer wants to check his/her used data (Fig 9). Based on this bill, customers may update their plan, for instance, subscribe more data or unsubscribe some data streams.

4.4.4. Discussions. Through the above experiment of the simulated scenario, the realtime capability and the data heterogeneity are supported by our platform although the realtime capability of video streaming data is unexpected. However, this is a problem of databus and we can solve it by using a specific databus for video streaming such as the one in [Al-madani et al. 2013]. Our platform has also the capacity to measure the value of data in realtime. Even though we do not focus on quality of data analysis in this experiment, by integrating our APIs into sensors and applications of consumer, the APIs can analyze the quality of data whenever they send/receive a data package, and then submit the report to the marketplace.

5. RELATED WORK

5.1. The IoT Platforms

Studying the features of existing IoT platforms and comparing them to a set of requirements for marketplace are important to develop a data marketplace in the context of IoT applications. Misra and Pal [2013] made a survey on important trends, key requirements, evolving technologies and emerging solutions for such a platform for IoT and M2M services. They also took a look at some commercial and open source IoT platforms for Data services, Device Management and Application Development in the market. Based on this work and a study of existing IoT platforms (e.g., Arrayent [2014], Axeda [2014], Xively [2014], ThingWorx [2013]), we recognized that most of them have three basic processes: upload/publish data, storage with time-series database and download/query data using time points. Moreover, they allow users to analyze or visualize the data. However, except Xively, all of them do not consider the issue of data delivery to users in realtime. Nimbits [2014], Sentilo [2014] and Kaa [2014] consider the alert settings in the point properties whenever a new value is recorded into a data point.

From the research community, several middleware platforms have been proposed/developed. Using the Web protocol to establish the connectivity of the things into the Internet, Guinard et al [2010] defined a resource oriented architecture for Web of Things in which data is stored in distributed devices. The applications/consumers can directly access these devices using HTTP protocol to query data. This architecture is appropriate for the private platform since the number of queries to devices is limited. However, the challenge of publishing data to consumers with high performance realtime communication is an open issue due to the limitation of HTTP protocol. EVRYTHNG [2014] platform has been using this technology. De et al [2012] describe a platform, named Sense2Web, for real-world services and smart objects. Starting from the IoT information models detailing entities, resources and services, the platform makes linkages with tag, location analysis of the existing resources on the Web, and publishes them on the Web within two user interfaces, i.e., Human-to-Machine interaction via Web and M2M interaction via SPARQL endpoints. Valente and Martins [2011] developed a middleware framework in which it receives smart objects from Wireless Sensor Networks and transforms them into Web services. This platform support a notification mechanism to alert the clients whenever a new smart object is available. Tong and Ngai [2012] developed a publish/subscribe system which supports ubiquitous data access from both wireless sensors and mobile phones. This system plays the role of databus in our platform.

Discussion: w.r.t the problems that need to be solved for a data market mentioned in Section 2, we recognize that all these platforms only satisfy a part of technology infrastructure e.g., missing the mechanism to process data in realtime. Whereas, a set of requirements such as business model, service/data broker, quality of service/data is totally missing. Munjin and Morin [2012] reuse these IoT platforms as the data brokerage platforms to define an architecture for IoT application marketplaces. In this work, the application store is suggested as an element of IoT platforms, which was designed as a registry of networked applications. The authors suggest this registry should support the application description management and the authentication. This work, however, focuses on application development instead of selling data.

5.2. Data Services

In recent years, the number of platforms for data marketplace have grown rapidly. Typical examples include Amazon Data Set [2015], Factual [2015], Gnip [2015], Azure Marketplace [Microsoft 2015], Xignite [2015]. Using these platforms, a registered client can upload their data manually or automatically using the supported APIs. However, many types of data are not near-realtime and become obsolete for some consumers. Some consumers may not satisfy with this situation because they need fresh new data or near-realtime data. To have the realtime data for consumers, it is required that the data must be collected by devices and simultaneously uploaded to the platform to promptly deliver to consumers. All these platforms do not address this issue.

In our previous works,by focusing on the automatic service lookup, data composition and utilization for several DaaS on the Cloud, we defined a description model for DaaSes, named DEMODS [Vu et al. 2012], which covers all basic information of a DaaS such the description of service, of data asset, of APIs and several linked models: pricing, contract, Quality of Service, etc. Also focusing on data composition, we developed the data contracts [Truong et al. 2012] to support concern-aware data selection and utilization from cloud-based data marketplaces. By an empirical study, Muschalle et al [2012] have presented several pricing strategies for data markets. They also presented the attractive research opportunities for the business intelligence community. Li and Miklau [2012] also focused on pricing of data market, however they proposed criteria for interactive pricing instead of analysis of pricing strategies. Using *linked* *data principles*, i.e., all datasets are represented internally as RDF graphs and each item is identified with an URI, Moller and Dodds [2012] developed a platform as a web based information marketplace, named Kasabi. A similar work, Bollacker et al [2008] defined a platform for structuring human knowledge using collaboratively created graph database. However, these works do not consider payment of data. In [Carey et al. 2012], starting from a general architecture for data services which can be deployed on top of data store, the authors reviewed data service concepts and examine approaches to service-enabling data sources, to create an integrated data service from multiple sources, and to manage data in the cloud. Moreover, they highlighted technical challenges including updates and transactions, data consistency or security for data services, as well as discuss emerging trends from future research and development.

6. CONCLUSION AND FUTURE WORK

In this paper, we presented the design and implementation of MARSA, a platform for a human-sensing realtime data marketplace. MARSA consists of a set of services interacting with each other to cover *data discovery*, *cost model management*, and *data contract management*. The roles of payment service, data quality analysis service, marketplace APIs were also analyzed. In addition, a prototype has been implemented and experiments have been executed to evaluate the performance of the proposed marketplace platform. By using multiple databuses, with different data types and the flexibility of choosing databuses at providers and consumers to transfer data, the users can use a specific databus for a concrete data type to improve the performance. Our prototype allows the data providers and their consumers to negotiate about their payment for each type of data according to five supported cost models. It helps them to monitor how their choosing models are executed through the marketplace APIs.

In the future, we plan to continue our work on the evaluation and enforcement of contract policies and data quality in the platform. In the current platform, while we allow data providers to specify contracts to data users, the information of contracts are simply stored in the database and no further actions are taken after this step. We also plan to introduce a dynamic cost model for the platform. All the cost models we introduced in this paper are static. In reality, in some cases, it is desirable to have a dynamic one. Our plan towards this issue is to work for a cost model that can change costs according to the supply and demand of the market.

REFERENCES

Basem Al-madani, Mohammed Al-saeedi, and Anas A. Al-Roubaiey. 2013. Scalable Wireless Video Streaming over Real-Time Publish Subscribe Protocol (RTPS). In *17th IEEE/ACM Int. Symposium on Distributed Simulation and Real Time Applications*. 221–230.

Amazon. 2015. Amazon Data Sets. http://aws.amazon.com/publicdatasets/. (2015).

Arrayent. 2014. IoT Platform Overview. http://www.arrayent.com/platform/overview/. (2014).

Axeda. 2014. M2M Connectivity. http://www.axeda.com/products/m2m-connectivity. (2014).

K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *ACM SIGMOD Int. Conf. on Management of data*.

Michael J. Carey, Nicola Onose, and Michalis Petropoulos. 2012. Data Services. *Communications, ACM* 55, 6 (June 2012), 86–97.

Yi-Ta Chuang, Chih-Wei Yi, Yin-Chih Lu, and Pei-Chuan Tsai. 2013. iTraffic: A Smartphone-based Traffic Information System. In *The 42nd International Conference on Parallel Processing*. 917 – 922.

COMPOSE. 2015. http://www.compose-project.eu/. (2015).

CyberVision. 2014. Kaa. http://www.kaaproject.org. (2014).

Andrea D'Ambrogio. 2006. A Model-driven WSDL Extension for Describing the QoS of Web Services. In *IEEE Int. Conf. on Web Services (ICWS'06)*. 789–796.

Suparna De, Tarek Elsaleh, Payam Barnaghi, and Stefan Meissner. 2012. An Internet of Things Platform for Real-World and Digital Objects. *Scalable Computing: Practice and Experience* 13, 1 (2012), 45–57.

Glen Dobson, Russell Lock, and Ian Sommerville. 2005. QoSOnt: a QoS Ontology for Service-Centric Systems. In *EUROMICRO Conference*. Los Alamitos, CA, USA, 80–87.

Etherios. 2014. Device Cloud. http://www.etherios.com/products/devicecloud/. (2014).

EveryAware. 2015. http://www.everyaware.eu/. (2015).

EVRYTHNG. 2014. http://www.evrythng.com/. (2014).

Factual. 2015. http://www.factual.com. (2015).

Gnip. 2015. http://gnip.com/. (2015).

Dominique Guinard, Vlad Trifa, and Erik Wilde. 2010. A Resource Oriented Architecture for the Web of Things. In *IEEE International Conference on the Internet of Things*. Tokyo, Japan, 1–8.

Jiong Jin, Jayavardhana Gubbi, Slaven Marusic, and M. Palaniswami. 2014. An Information Framework of Creating a Smart City through Internet of Things. *IEEE IoT Journal* 1, 2 (2014), 112–121.

Juong-Sik Lee and Baik Hoh. 2010. Sell your experiences: a market mechanism based incentive for participatory sensing. In *IEEE Int. Conf. on Pervasive Computing and Communications*. 60 – 68.

Chao Li and Gerome Miklau. 2012. Pricing Aggregate Queries in a Data Marketplace. In *15th Int. Workshop on the Web and Databases*. Scottsdale, AZ, USA.

Microsoft. 2015. Window Azure Marketplace. http://datamarket.azure.com/. (l 2015).

Balamuralidhar Prateep Misra and Arpan Pal. 2013. Software Platforms for Internet of Things and M2M. *Journal of the Indian Institute of Science* 93, 3 (Jul-Sep 2013), 487–497.

Knud Möller and Leigh Dodds. 2012. The Kasabi Information Marketplace. In *21nd WWW conference*.

Dejan Munjin and Jean-Henry Morin. 2012. Toward Internet of Things Application Markets. In *IEEE Int. Conf. on Green Computing and Communications*. 156 – 162.

Alexander Muschalle, Florian Stahl, Alexander Löser, and Gottfried Vossen. 2012. Pricing Approaches for Data Markets. In *6th Int. Workshop on Business Intelligence for the Real Time Enterprise.*

Nimbits. 2014. http://www.nimbits.com/. (2014).

OMG. 2007. *Data Distribution Service for Real-Time systems* (1.2/07-01-01 ed.). Object Management Group.

Sooksan Panichpapiboon. 2010. The Future of Traffic Information Systems. *Journal of Information Science and Technology* 1, 2 (Jul-Dec 2010), 70–78.

Marco Picone, Michele Amoretti, and Francesco Zanichelli. 2012. A Decentralized Smartphone Based Traffic Information System. In *2012 IEEE Symposium on Intelligent Vehicles*. 523 – 528.

Fabian Schomm, Florian Stahl, and Gottfried Vossen. 2013. Marketplace for Data: An Initial Survey. *SIGMOD Record* 42, 1 (March 2013), 15–26.

A. Segura, J. Cuadrado, and J. Lara. 2014. ODaaS: Towards the model-driven engineering of open data applications as data services. In *IEEE 18th Int Enterprise Distributed Object Computing Conference Workshops and Demonstrations*. 335–339.

Sentilo. 2014. http://www.sentilo.io/wordpress/. (2014).

J. Spillner and A. Schill. 2013. A Versatile and Scalable Everything-as-a-Service Registry and Discovery. In *CLOSER*. 175–183.

ThingSpeak. 2014. https://www.thingspeak.com/. (2014).

ThingWorx. 2013. What is ThingWorx. http://www.thingworx.com/platform. (2013).

Xiaoyu Tong and Edith Ngai. 2012. A Ubiquitous Publish/Subscribe Platform for Wireless Sensor Networks with Mobile Mules. In *IEEE 8th Int. Conf. on Distributed Computing in Sensor Systems*. 99 – 108.

Hong-Linh Truong, Marco Comerio, Flavio De Paoli, G.R Gangadharan, and Schahram Dustdar. 2012. Data Contracts for cloud-based data marketplaces. *Int. J. Comput. Sci. Eng.* 7, 4 (2012), 280–295.

Ubicon. 2015. http://ubicon.eu/about/. (2015).

Bruno Valente and Francisco Martins. 2011. A Middleware Framework for the Internet of Things. In *The 3rd Int. Conf. on Advances in Future Internet*. 139 – 144.

Quang-Hieu Vu, Tran-Vu Pham, Hong-Linh Truong, Schahram Dustdar, and Rasool Asal. 2012. DEMODS: A Description Model for Data-as-a-Service. In *IEEE Int. Conf. on Advanced Information Networking and Applications*. 605–612.

Xignite. 2015. Market Data Cloud. http://www.xignite.com/market-data/market-data-cloud/. (2015).

Xively. 2014. Cloud Services. https://xively.com/whats_xively/. (2014).

Dejun Yang, Guoliang Xue, Xi Fang, and Jian Tang. 2012. Crowdsourcing to Smartphones: Incentive Mechanism Design for Mobile Phone Sensing. In *ACM Annual International Conference on Mobile Computing and Networking*. 173–184.