Keeping You in the Loop: Enabling Web-based Things Management in the Internet of Things

Lina Yao and Quan Z. Sheng
School of Computer Science
The University of Adelaide
Adelaide, SA 5005, Australia
{lina, qsheng}@cs.adelaide.edu.au

Anne H.H. Ngu and Byron Gao
Department of Computer Science
Texas State University
601 University Drive, San Marcos, USA
{angu, bgao}@txstate.edu

ABSTRACT

Internet of Things (IoT) is an emerging paradigm where physical objects are connected and communicated over the Web. Its capability in assimilating the virtual world and the physical one offers many exciting opportunities. However, how to realize a smooth, seamless integration of the two worlds remains an interesting and challenging topic. In this paper, we showcase an IoT prototype system that enables seamless integration of the virtual and the physical worlds and efficient management of things of interest (TOIs), where services and resources offered by things can be easily monitored, visualized, and aggregated for value-added services by users. This paper presents the motivation, system design, implementation, and demonstration scenario of the system.

Categories and Subject Descriptors

H.3.5 [**Information Storage and Retrieval**]: Online Information Services.; H.4.0 [**Information Systems Applications**]: General

Keywords

Internet of Things, RFID, RESTful Web Services

1. INTRODUCTION

As pointed out by Tim Berners-Lee, the inventor of the World Wide Web, it isn't the documents which are actually interesting, it is the things they are about[1]. The last level of abstraction for the Web is to connect physical things. With the recent advances in radio-frequency identification (RFID), wireless sensors network, and Web services, Internet of Things (IoT) offers the capability of integrating the information from both the physical world and the virtual one. With IoT, it becomes possible to infer the status of real-world entities with minimal delay using a standard Web browser [2].

[1]http://ercim-news.ercim.eu/en72/keynote/the-web-of-things

CIKM'14, November 3–7, 2014, Shanghai, China.

http://dx.doi.org/10.1145/2661829.2661838.

Motivation. IoT describes the evolution from systems linking digital documents to systems relating digital information to real world physical items. While it is well understood that IoT offers numerous opportunities and benefits, it also presents significant technical challenges. One of the main challenges is to provide a smooth and seamless integration of the virtual world and the physical one for effectively managing *things of interest* (TOIs) in IoT [4]. A crucial prerequisite is acknowledging the seamless information access, exchange and manipulation between the two worlds. However, in general, the communication and changes among things in IoT are implicit and invisible to humans, which leads to difficulties in representing and capturing them in formal models.

Our Approach. We explore a new direction in realizing information integration between the physical and the virtual worlds. We design and develop a prototype system that offers an integrated Web-based interface to manage (i.e., connect, monitor, control, mashup, and visualize) things in an IoT environment, which helps people to be aware of their surrounding world and make better decisions. The system provides a Web-based framework, where the information produced by physical things can be managed and socialized.

A central component of our system is the context-aware composition of things. In our system, contextual information includes environmental information (e.g., motion, temperature, light condition), activities (e.g., a person approaching the door of a house) and social events (e.g., tweet updates from people or socialized things). Since things are socialized, they can talk, respond to the context, and behave like humans. We particularly adopt a rule-based approach to aggregate individual things and build context-aware, personalized new value-added services. The context of a real, inhabituated home environment demonstrated in this paper by both virtual resources and physical things provides the basis for various other research in specific domains, such as independent living of the elderly, healthcare and smart homes. Our system can be considered as a step further to realizing the IoT vision. In the following sections, we will overview the design and implementation of the system, and sketch the proposed demonstration.

2. SYSTEM ARCHITECTURE

Figure 1 shows the architecture of our prototype system. The system has a layered architecture, and is developed using the Microsoft .NET framework and SQL Server 2012. Physical things and their related data and events are mapped to the corresponding virtual resources, which can be aggregated and visualized via a range of software components.

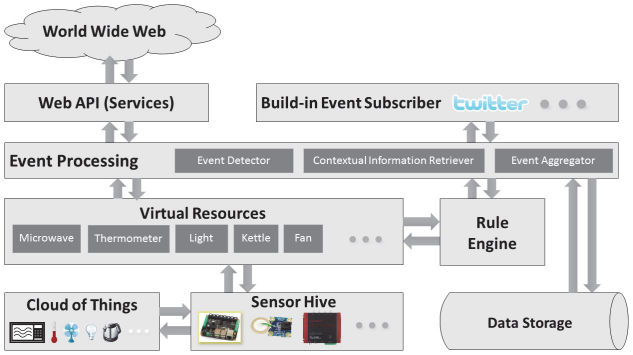

Figure 1: System architecture

The system has two ways to identify physical objects and connect them to the Web. The first one is to use RFID technology, where the physical objects are attached with RFID tags and interrogated by RFID readers. The second one is to attach sensors to objects for transferring the raw data to the network. The raw data captured by readers and sensors will be further processed. In the following, we describe the key modules of the system and their implementation details.

Sensor Hive. This module manages sensors and RFID tags, establishes connections, and queries sensor/RFID status, thus allowing physical things to be mapped to their corresponding virtual resources. It provides a universal API so that higher level programs can retrieve status of sensors with specified address without knowing where and how to find the physical sensors. This module works in a scalable, plug-and-play fashion, where new sensors can be added and old sensors can be easily removed.

Virtual Resources. This module maps a collection of classes to their corresponding physical devices. Each virtual device collects related sensor readings from Sensor Hive and uses the collected information to check the current status of the corresponding physical device. For example, the virtual device of a microwave oven can query its sensor values from the Sensor Hive, then use these readings to decide the current status (idle or busy) of the physical microwave oven. Virtual devices generate events based on the readings, which can be subscribed by high-level programs. Since each kind of device has a virtual resource, systems built with this architecture can be easily extended and maintained.

Event Processing. This module automatically extracts and aggregates things usage events based on data feeds from Virtual Resources in a pipelined fashion. The pipeline consists of three main phases: *event detection*, *contextual information retrieval*, and *event aggregation*.

The Event Detector captures and decides whether a physical thing is in-use. In our implementation, there are two ways to detect usage events of things: *sensor-based* for detecting state changes and *RFID-based* for detecting mobility [4]. In the sensor-based detection, the usage of an object is reflected by changes of the object's status, e.g., a microwave oven moved from an idle to an in-use state. In the RFID-based detection, the movement of an object indicates that the object is being used. For example, if a coffee mug is moving, it is likely that the mug is being used. In this situation, we adopt a generic method based on comparing descriptive statistics of the Received Signal Strength Indication (RSSI) values in consecutive sliding windows [1]. The statistics obtained from two consecutive windows are expected to differ significantly when an object is moved.

The Contextual Information Retriever fetches contextual information contained in things usage events. In our current implementation, we focus on three types of contextual information: *identity* (user), *temporal* (timestamp) and *spatial* (location) information [3, 4]. For the spatial information, we consider two situations. For *static* objects (e.g., refrigerator), the spatial information is a prior knowledge. For *mobile* objects (e.g., RFID-tagged coffee mug), we provide coarse-grain and fine-grain methods for localization. The coarse-grain method uses the RSSI signal received from a tagged object to approximate its proximity to an RFID antenna. Each zone is covered by a mutually exclusive set of RFID antennas. The zone scanned by an antenna with the maximum RSSI signal is regarded as the object's location. The fine-grain method compares the signal descriptors from an object at an unknown location to a previously constructed radio map or fingerprint. We use the Weighted k Nearest Neighbors algorithm (w-kNN) to find the most similar fingerprints and compute a weighted average of their 2D positions to estimate the unknown tag location [4].

The Event Aggregator indexes and stores all the events and services, together with their related information. The indexed events and services, as well as their corresponding contextual information are stored in a database, which can be mined for various purposes (e.g., finding hidden correlations among things, recommendation) [3, 4, 5]. A list of elements are constructed, storing the identifiers of objects, their types and values, as well as the calculated contextual information. In this way, applications can focus on the functionalities without worrying about operations such as connecting to the database, opening connections, querying with specified languages and handling the results.

Rule Engine. This module allows users to control a device automatically by setting up a series of basic rules. The Rule Engine includes three main components: the *Rule Builder*, the *Rule Interpreter*, and the *Rule Parser*. The Rule Builder is a Web-based application implementing a user-friendly GUI for rule creation and action setup, where a user only needs to drag the icon of a device to the editing area, then setup the details by several simple clicks. A string expression is generated for the rule. The Rule Interpreter analyzes and annotates the string statement based on a state machine. The string expression is then translated to a list of annotated objects. The outcome of the Rule Interpreter is a sequence of annotated inputs, which will be passed to the Rule Parser. The Rule Interpreter can also convert the string statement into a simple structure so that the Rule Builder can generate a corresponding GUI for rules. This is important for users who need to modify their existing rules. The Rule Parser is a compiler based on the *shunting-yard* algorithm. It first compiles each part of the input sequence into a .NET Expression object. Then, it combines all such objects together into a complex Expression Tree, which will be compiled into a Lambda Expression, which will be stored in memory when the system is running. It can be invoked when a device status changes or time elapses. If the Lambda expression returns true, a corresponding action will be called.

Web API/Services. This module converts events and data into corresponding services. By providing RESTful APIs for things, applications can easily access data associ-

Figure 2: 3D Web-based interface

ated with a particular thing stored in the database, as well as manipulate the sensors (e.g., turning on or off a light). The APIs are represented using JavaScript Object Notation (JSON), which is developed from the JavaScript for representing simple data structures and associated objects. Our system also offers a 3D Web-based interface (see Figure 2). We particularly adopt WebGL in HTML 5 to enable 3D scene recreation. The 3D models are stored as Digital Asset Exchange (DAE) files, and imported and rendered by using `three.js`[2] with plug-ins.

3. DEMONSTRATION PLAN

Our system offers an integrated Web-based interface where a user can monitor and visualize status changes of things of interest in real-time. Users can monitor, track, and control the current status of physical things by directly observing the status of their corresponding icons (also called avatars) from a Web browser. In addition, our system augments the physical things with key social network functionality. We have developed a real-time notification of things' status by exploiting the Twitter public API. Real-time status changes are sent to the subscribers. Figure 2 shows the integrated Web-based user interface, and we will focus on demonstrating the following main functionalities.

- *Real-time visualizer and monitor:* This function offers access to, and control of, the physical things, allowing the real status of physical things to be visualized in real time. We render the icons with different effects to be consistent with real status of things. For example, if the microwave oven is being used, the microwave oven icon will change to a highlighted status (yellow), otherwise gray (Part 1 in Figure 2). The learned position information from the Event Processing module is used to visualize the locations and traces of things of interest at a coarse-grain level. For example, as shown in Part 2 of Figure 2, the trace of a tagged coffee mug held by a subject is displayed in the green lines.
- *Rules composer:* This function provides a complex graphical interface for users to control the devices by setting up a series of basic rules via Web browser without any programming efforts (Part 3 in Figure 2, please see the video clips for more details). It can create a composite service using the rule-based composition component provided by the system, which consists of a *Widget* panel and a *Rules Editor* panel. The Widget panel shows the virtualized objects (things and people) and each object has a set of actions (e.g., the light is associated with two actions: turn on and turn off). Users just drag any thing's widget to the Rules Editor panel and start to create a new rule or change the old rules by clicking the `Edit` button next to each rule.

Our demonstrative video clips can be accessed from the Youtube[3].

4. REFERENCES

[1] S. Parlak, I. Marsic, and R. S. Burd. Activity Recognition for Emergency Care Using RFID. In *Proc. of the 6th Intl. Conf. on Body Area Networks*, 2011.

[2] A. Pintus, D. Carboni, and A. Piras. Paraimpu: A Platform for a Social Web of Things. In *Proc. of the 21st Intl. World Wide Web Conf. (WWW)*, 2012.

[3] L. Yao and Q. Z. Sheng. Exploiting Latent Relevance for Relational Learning of Ubiquitous Things. In *Proc. of the 21st ACM Intl. Conf. on Information and Knowledge Management (CIKM)*, 2012.

[4] L. Yao, Q. Z. Sheng, B. Gao, A. Ngu, and X. Li. A Model for Discovering Correlations of Ubiquitous Things. In *Proc. of the 13th IEEE Intl. Conf. on Data Mining (ICDM)*, 2013.

[5] L. Yao, Q. Z. Sheng, A. H. Ngu, H. Ashman, and X. Li. Exploring Recommendations in Internet of Things. In *Proc. of the 37th ACM SIGIR Conf.*, 2014.

[2] `http://threejs.org`

[3] `https://www.youtube.com/playlist?list=PL8nHiAwRrq8I4aYQBPdelwGXIaqMVtYni`