A Generic Platform for Ubiquitous and Subjective Data

Martin Becker
L3S Research Center
University of Würzburg
Würzburg, Germany
becker@informatik.uni-wuerzburg.de

Juergen Mueller
L3S Research Center
University of Kassel
Kassel, Germany
mueller@cs.uni-kassel.de

Andreas Hotho
L3S Research Center
University of Würzburg
Würzburg, Germany
hotho@informatik.uni-wuerzburg.de

Gerd Stumme
L3S Research Center
University of Kassel
Kassel, Germany
stumme@cs.uni-kassel.de

UbiComp'13 Adjunct, September 8–12, 2013, Zurich, Switzerland.

ACM 978-1-4503-2215-7/13/09...$15.00.

http://dx.doi.org/10.1145/2494091.249977

Abstract

In the context of the *Internet of Things*, an increasing number of platforms like Xively or ThingSpeak are available to manage ubiquitous sensor data. Strict data formats allow interoperability and informative visualizations, supporting the development of custom user applications. Yet, these strict data formats as well as the common device-centric approach limit the flexibility of these platforms: there are no means to incorporate people and their subjective impressions about the collected data. In order to build the Internet of Things *and People* and ultimately the *Internet of Everything*, we aim at providing an *extendable* concept of data which allows to enrich existing data points with any kind of additional information. This enables us to gain semantic and user specific context by attaching subjective data to objective values. For this end we support data ranging from text-based formats like JSON to images and video footage. This paper provides an overview of our architecture including concept, implementation details and present applications. We distinguish our approach from several other systems and describe two sensing applications namely AirProbe and WideNoise that were implemented for our platform.

Author Keywords

System Architecture; Citizen Science; Ubiquitous Data; Data Mining; Internet of Things

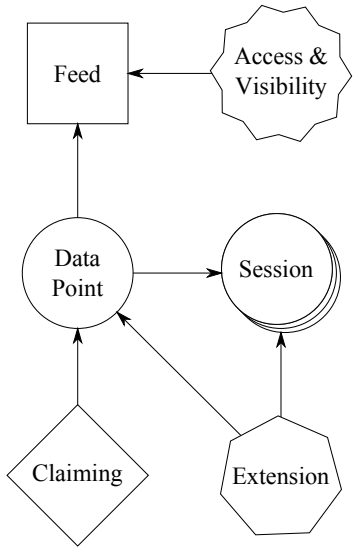


Figure 1: Conceptual design of the EveryAware system.

ACM Classification Keywords

C.0 [General]: System architectures; D.2.11 [Software Architectures]: Data abstraction; H.2.8 [Database Applications]: Data mining

Introduction

In the context of the *Internet of Things*, many new applications have been designed for mobile devices enabling people to record environmental data by making use of embedded sensors, such as a microphone, camera, accelerometer, gyroscope and GPS receiver. Collecting such data at a central place turns these applications into a tremendous amount of ubiquitous sensors. Cuff et al. suggest that there is a wide range of applications in which people can be engaged in mobile sensing and they expected a rapidly growing field and applications for urban sensing [2]. This is in particular true for citizen science where volunteers contribute for the benefit of human knowledge and science [4]. Methods and techniques of flexibly acquiring and handling this data play a central role in paving the way towards behavioral shifts within large citizen populations.

Thus, the citizen science movement is especially supported by the emerging web based platforms making it easy to collectively upload content and share data. The data in this context can be divided into two classes. (1) Objective data, which stems mainly from sensors and includes measurements like sound intensity or gas concentration. (2) Subjective data, which comprises reactions and perceptions of humans faced with particular environmental conditions.

Traditional *Internet of Things* approaches cover just the class of objective data. Yet objective data can change its interpretation entirely in different semantic contexts. For example high noise levels at a rock concert are perceived as enjoyable while a dropping water-tap can be considered as *noise pollution*. Therefore, on the way to the *Internet of Everything*, the next step is an Internet of Things *and People* [8] not only working on objective data but incorporating people to add impressions, interpretations and other subjective context.

Even without considering the synergy of objective and subjective data, the main advantage of a social information technologies is the sheer amount and diversity of the data being collected. It allows data alignment and aggregation methods to create representative statistics and visualizations and enables advanced knowledge discovery algorithms to discover hidden patterns and relations as mentioned in [5]. At the same time this centralized collection of data is also subject to security and privacy issues which need to be handled with care.

The EveryAware platform aims at providing a highly efficient, generic data collection and processing framework featuring a powerful extension mechanism to allow for semantic data augmentation and at the same time supporting a flexible data processor architecture to incorporate advanced knowledge discovery tasks in order to fully exploit the synergies of a central data storage and the wide variety of objective and subjective information. Privacy issues are addressed by progressive access control, which can be customized to the individual needs.

The rest of the paper is structured as follows: The next section gives a brief overview on the area of data collections and exchange. Providers of similar services will be presented and we will distinguish our platform from these. The following section presents the two parts of the EveryAware system, the conceptual and implementation layer, and describes them in detail. To show the applicability of our approach, we will present two reference

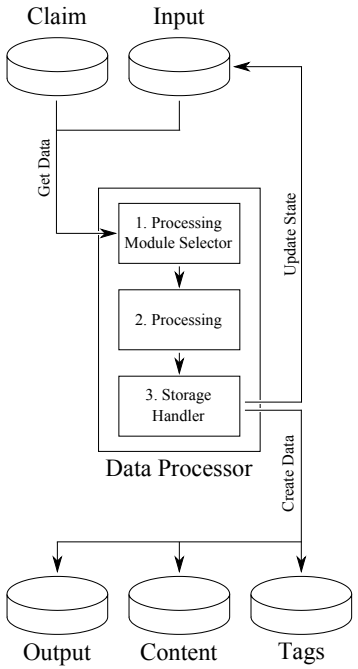


Figure 2: Architecture of the data processor engine.

implementations for our system in the following section. Finally, the last section concludes with a summarization our approach and several directions of further work.

Related Work

The Internet of Things lives from the connectedness of devices. Thus, data exchange must be as simple as possible. For this end several data storage and distribution platforms have emerged.

This section summarizes some related work for data collections and exchange. There are several well-known projects which provide corresponding services: Xively[1] (formerly Cosm and before that Pachube), Open.Sen.se[2], Device Cloud[3] (formerly iDigi), Eye on Earth[4] and ThingSpeak[5].

In general the Internet of Things is not restricted to technical sensors like heart rate monitors and air pollution sensors but incorporates a much larger variety of data to be shared including images, video footage or even perceptions. All reviewed platform restrict their API's to accept only pre-defined and hardly flexible types of data. In most cases, time-series are the main concept and collected data must be numeric or alpha-numeric as is true for Xively and Open.Sen.se. Eye on Earth restricts its input to geo-spatial data. None of the mentioned providers seems to be able to accept sequences of image or video footage. The EveryAware[6] system, on the other hand, tries to be as flexible as possible on this regard not restricting the type of data it accepts at all.

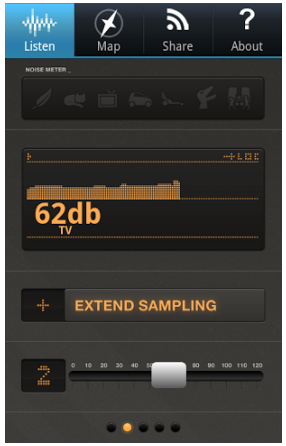
[1] https://xively.com/
[2] http://open.sen.se/
[3] http://etherios.com/devicecloud/
[4] http://eyeonearth.org/
[5] https://thingspeak.com/
[6] http://cs.everyaware.eu/

The Internet of Things has focused on actual *things* and the corresponding data but not on how people interact with these things [8] or what the data actually means to them. This mindset is mirrored by the providers mentioned above, thus, no functionality to share information about the collected data or to annotate subjective impressions is present. With the EveryAware system we try to work towards a future *Internet of Things and People* by allowing data points to be extended with any kind of data. This helps to put data into a meaningful context including tags and other subjective data people may think of. To the best of our knowledge, this approach is new with the EveryAware platform.

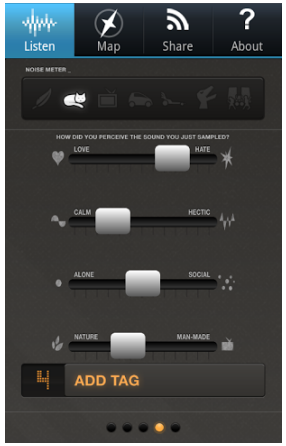
Introducing a flexible data processor engine, our platform also allows the application of various data mining and knowledge discovery tools. This aspect addresses challenges as mentioned for example by Hotho et al. [5] and further distinguishes our platform from the mentioned providers.

Architecture

The platform has been designed to facilitate the combination of sensing technologies, networking applications and data processing tools. This enables users to collect and visualize environmental information and at the same time augment the collected data with arbitrary information explicitly supporting subjective context. Additionally the data processing engine allows for the application of dedicated data mining and knowledge discovery algorithms in order to fully exploit the synergies of a central data storage and the wide variety of objective and subjective information. Our platform is based on the Ubicon framework [1] and extends it to provide the functionality needed for our architecture.



(a) Recording



(b) Perceptions

Figure 3: Screenshots of the WideNoise Android application.

The platform comprises two main layers. The conceptual layer defines the basic entities and features the EveryAware system supports. It introduces an innovative framework to collect, process and retrieve data and features straight forward usability, flexible access control as well as a powerful data extension mechanism. The implementation layer introduces the data processor which ensures high availability as well as generic data handling and processing. The conceptual and the implementation layer are described in the following.

Conceptual Layer

The conceptual layer defines the basic entities and features the EveryAware system supports. The core concepts are data points with descriptions, sessions and feeds. Data points and sessions can be extended by other data points (see Figure 1). Single data points can be claimed by users, i.e., the user can claim ownership. Feeds are access restricted and can define visibility levels.

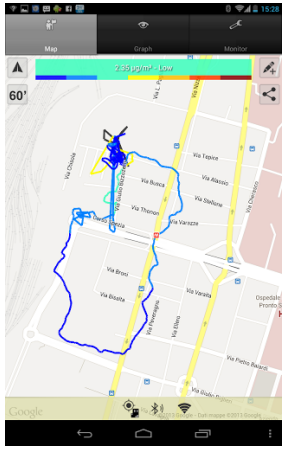
The basic entities of the EveryAware system are data points. Each data point consists of a set of fixed description attributes in addition to the actual data. These attributes ensure the processability as well as dynamic querying of arbitrary content including binary data like photos or videos. The description attributes are divided into three categories: (1) Meta attributes are attributes which allow to keep track of data independent information like received time, recording time, device ID, session ID, etc. (2) Geo attributes make it possible to record the location of the sample being taken including longitude and latitude as well as accuracy and the provider of the location fix. (3) Content attributes describe the content and its format. They help the system to further process the data. These attributes include the data type (e.g., air, noise, image) and format (e.g., JSON, XML, PNG).

Sessions are collections of data points limited to a fixed timespan. Sessions allow to introduce semantic entities such as "my way to work" or "a stroll in the park".

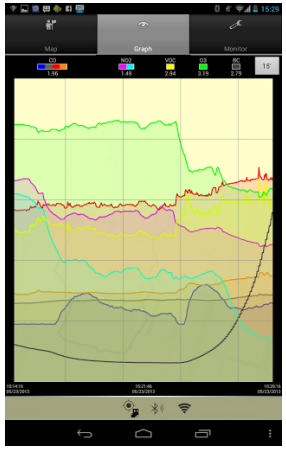
Data points as well as sessions can be extended with additional information using other data points. This makes the data representation very flexible and inherently support the augmentation of objective data with a semantic context. One application is tagging. Sessions and data points can be tagged by extending them using tag data points referring to the respective data point or session IDs to be tagged. Tagging is not only restricted to actual text-tags but can be any kind of data including videos, sound files or air quality measurement. Using this scheme, it is also possible to update data points as well as sessions after they have been sent without losing the original data. Since no raw data is deleted, this also allows to always access the version history of a data point.

Data points can be organized in feeds. A data point is always part of the global feed, but can also be pushed into several other feeds. Users can contribute to existing feeds or create their own feeds. While useful for organizing data points, feeds also allow to attach data points to real world entities such as major events like music festivals, places like the Eiffel Tower or portable things like a smartphone. Feeds can be access restricted and a visibility level can be specified for each data point in a feed.

Feeds can be *open* or *closed* concerning read and write access, where *write access* refers to the possibility of adding new data points to a feed. Open feeds are accessible by everyone including anonymous users. Closed feeds are only accessible by a limited set of users (*members*). The global feed is always open for read and write access. It contains all data points without exception. The access restriction allows users to create feeds and



(a) Map



(b) Graph

Figure 4: Screenshots from the Google Play Store of the AirProbe Android application.

share them with friends or other interested users without making their data publicly available.

Since privacy is an issue and users may want to contribute in different ways to the collected data and corresponding statistics which might be derived from it, the EveryAware system introduces visibility levels for each data point in a feed. This is particularly important since all data points are part of the global feed. There are four visibility levels: (1) *details* allows everyone who has access to the feed to see the raw content as well as the description attributes of the data point. (2) *statistics* restricts the data point to be considered in user statistics derived from the data points in the feed, e.g., average values for the corresponding user. (3) *anonymous* restricts the data point to only be considered in overall statistics derived from the data points in the feed, e.g., average values for an area or timespan. No association with the user is possible. (4) *none* allows only the owner of the data point to access the data point and its description attributes.

Claiming is a concept which allows anonymous contribution to the EveryAware system while giving the user possibility to claim data points as soon as she decides to register a user account. This makes it convenient to contribute data to the EveryAware system and provides some level of anonymity since no previous registration process is required. At the same time the collected data is not lost for the user but can be accessed at a later time. Claiming works by exploiting the device ID, which is usually sent as part of a store request. If a registered user sends a data point with a device ID, she can claim all data points she has sent before with the same device ID.

Implementation Layer

The core data handling is implemented in the second layer comprising two steps: data receptions by a REST endpoint and data processing by the so called *data processing engine*. The data processing engine is responsible for parsing the received data, resolving extensions, apply knowledge discover processing steps and augmenting them with additional context information from various sources.

The most basic priority of our data collection platform is to reliably store the received data points. To ensure this, computational overhead such as syntactic checks is kept at a minimum. The received data is directly written to the *input table*. Only afterwards will the data processor start parsing, extending and verifying the stored data using a modularized approach. This gives us the following advantages: (1) High performance and availability: Any computation in the endpoint would slow down the data reception. This is particular true for applications with large content and high transmission frequencies. (2) Flexibility: We can accept literally any data, since the endpoint does not restrict the type of data sent via the REST endpoint. Different data types are handled by a dynamically manageable set of processing modules. (3) Robustness: Storing the data in its raw form enables us to recreate the processed content at any point in time.

In the second step, the data processor engine post-processes the data. Figure 2 shows the general architecture. The data processor reads data from the *input table* and combines it with information about ownership from the *claim* table. Data is then parsed, interpreted and augmented using a chain of processing modules. This includes applying knowledge discovery steps in order to gain more insights about the data. The resulting information is stored in the so called *content tables* which are query-able from data access endpoints. Additionally, tags are anchored directly into our system in order to define a generic way to express semantic context. Those are written

Figure 5: The sensor box used with AirProbe.

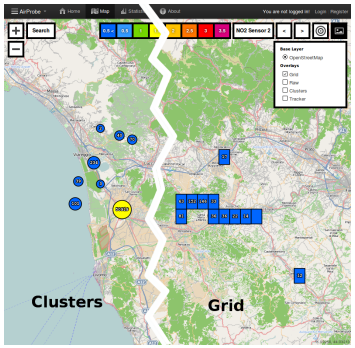


Figure 6: A screenshot of the *map page* of AirProbe. The left side shows the cluster view, the right side shows the grid view.

into the *tags table*. In order to ensure generic data access, to take the load from the *input table*, all data is written to the *output table* as is including the description attributes like geo-coordinates or ownership information. Furthermore, this division of input and output storage allows for replication of the REST endpoint for data reception on different servers and enables distributed processing of large amounts of incoming data. Also, the data processor sets the processing state for each processed data point based on the processing result (e.g., sucess, error, etc.).

The data processer constantly checks for data which needs processing: new data and data that should be parsed a second time. After retrieving the data to process the data processor runs through its different components: the *module selector*, the *selected processing module* and the *storage handler*. The *module selector* selects a *processing module* from a priority-chain. The matching module is selected based on its data type defined by the data points descriptions attributes or by deriving the data type from the raw content. The second component is the *selected processing module*. It extracts the actual data from the raw content (e.g., a JSON file) and possibly augments the data with additional information, calculates statistics or handles missing information. Then, the *storage handler* stores the resulting data in dedicated *content tables*, the *output table* and possibly the *tags table*. After the storage engine has finished writing the data to the corresponding tables, it adjusts the processing state of the data in the *input table*.

This architecture has several advantages: The priority-chain-approach in the *module selector* allows for flexibility in extending the data processor engine with additional processing modules on demand. The modularized approach in general makes it easy to deploy updates without risking to break the existing data. The *processing state* set by *storage handler* for each processed data point is the key to flexible module extensions and ensures robustness against processing failures. Assuming a processing module changes, the module can simply be exchanged. The processing state for affected input data is reset. The data processor engine will then process only the marked data and replace the data in corresponding output tables.

Applications

There are two major applications which have been implemented specifically for our platform: namely WideNoise and AirProbe. Both have been developed as part of the EveryAware research project and have been mentioned before in [1].

WideNoise, originally developed by WideTag[7] and enhanced by the EveryAware team, is an application for monitoring noise pollution (see Figure 3(a)). AirProbe, developed by CSP[8] as part of the EveryAware project, monitors air quality (see Figure 4). Both applications have a smartphone interface as well as a web interface. The smartphone gathers data and transmits the data to our server where it is augmented and aggregated by the data processor to be visualized on the web frontend. WideNoise gathers the data using the build-in microphone of the smartphone, while AirProbe retrieves the data (such as NO_2, CO, O_3, VOC, temperature and humidity) from an external senso box[3] as shown in Figure 5. The following paragraphs will focus on WideNoise and AirProbe and their common as well as distiguishing features.

Noise pollution and air quality are both interpreted in highly subjective contexts. High noise levels for example

[7] http://widetag.com/
[8] http://csp.it/

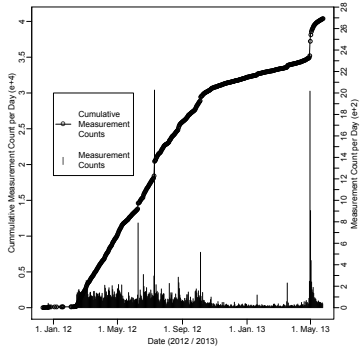


Figure 7: Development of the WideNoise sample count.

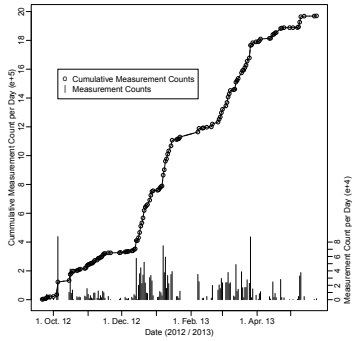


Figure 8: Development of the AirProbe sample count over time.

are not perceived as pollution when users are attending a rock festival. It is also interesting to see if users perceive high ozone levels as bad when they are sunbathing. Thus, both, WideNoise and AirProbe support extending the objective sensor readings with subjective data as explicitly supported by the EveryAware system. In both cases, this subjective data is expressed as tags. Also, WideNoise allows to add noise estimates as well as user perceptions to the noise samples as depicted in Figure 3(b).

After the data is received by the EveryAware system, the data processor parses and aggregates the data by applying several dedicated processing modules. The results are statistics and corresponding visualizations on a global scale as well as on the user level. One major visualization is the world map as shown in Figure 6. It displays, for example, a clustered view of the recorded data (providing corresponding detail information on demand [7]) as well as a tag cloud characterizing the summarized data by its semantic context.

WideNoise has been running for more than a year now and is used for example in the Heathrow airport area to monitor noise pollution caused by air traffic. Until now we collected more than 40,000 noise samples recorded by over 13,500 devices from all over the world. The AirProbe system is not yet used on a large scale, but during some preliminary case studies to test the system, we have already collected close to two million air quality samples from only 39 devices. This large difference in numbers is due to the discrete nature of WideNoise, while AirProbe applies a continuous sampling scheme. Also see Figures 7 and 8 for sample growth rates. The peaks are mostly due to events, media announcements or dedicated case studies. The large number of samples makes the AirProbe application our main benchmark system. The goal is to provide a user experience which feels as live as possible: when new data is sent to the server, updates should optimally be visible in at less than eight seconds [6].

Conclusion and Outlook

The EveryAware platform has been explicitly designed to support subjective impressions in conjunction with sensor data acquisition by introducing a extendable data concept. A central server efficiently collects, analyses and visualizes data sent from the arbitrary sources.

The platform offers a highly flexible way to store and exchange data for Internet of Things and People applications. To the best of the authors' knowledge, this is the first system allowing for dynamical data extension and at the same time giving the possibility to accept any kind of data while providing such a generic approach to data exchange and processing. A wide variety of meta, geo and content information which can be attached to any data point as well as a flexible data processor engine are the keys to this task. This mechanism provides the unique ability to enrich data with contextual information explicitly including subjective impressions. Different collection concepts like sessions to represent time-interval-based entities and feeds to organize data points in a continuous way allow to further introduce semantic relations. The extension mechanism together with the collection concepts and the resulting various possibilities to collectively gather, share and analyze data represents an important step towards the Internet of Everything by building the foundation of an Internet of Things *and People*. The functionality of the EveryAware platform is demonstrated by the WideNoise and AirProbe application. They illustrate the flexibility of our platform to handle different data types with one underlying structures.

The EveryAware system provides a powerful framework for data collection, processing and exchange. Nevertheless, there are still some challenges to be addressed including, for example, more advanced entity models and corresponding authentication schemes, in depth evaluation of distribution techniques inherently provided by the data processor architecture, user customizations of processing modules or a generic data definition with allowing for flexible visualization tools.

While there are still many ways to improve our current work, the EveryAware platform as it is now is already a powerful data collection and exchange platform. We will keep on refining the architecture and add new features in order to help build the *Internet of Everything*.

Acknowledgments

Part of this research was funded by the European Union in the 7th Framework Prog. project EveryAware (FET Open).

References

[1] Atzmueller, M., Becker, M., Doerfel, S., Kibanov, M., Hotho, A., Macek, B.-E., Mitzlaff, F., Mueller, J., Scholz, C., and Stumme, G. Ubicon: Observing social and physical activities. In *IEEE International Conference on Cyber, Physical and Social Computing, CPSCom 2012, Besanon, France, 20-23 November, 2012*, IEEE (2012), 317–324.

[2] Cuff, D., Hansen, M., and Kang, J. Urban sensing: Out of the woods. *Communications of the ACM 51*, 3 (Mar. 2008), 24–33.

[3] Elen, B., Theunis, J., Ingarra, S., Molino, A., Van, J., den Bossche, Reggente, M., and Loreto, V. The everyaware sensorbox: a tool for community-based air quality monitoring. In *Sensing a Changing World Workshop, Wageningen, The Netherlands, 9 May, 2012. Proceedings* (2012).

[4] Haklay, M. Citizen science and volunteered geographic information: Overview and typology of participation. In *Crowdsourcing Geographic Knowledge*. Springer, 2013, 105–122.

[5] Hotho, A., Ulslev Pedersen, R., and Wurst, M. Ubiquitous data. In *Ubiquitous Knowledge Discovery*, no. 6202 in Lecture Notes in Computer Science. Springer, 2010, 61–74.

[6] King, A. B. *Speed Up Your Site: Web Site Optimization*, 1 ed. New Riders, 2003.

[7] Shneiderman, B. The eyes have it: A task by data type taxonomy for information visualizations. In *Symposium on Visual Languages, Boulder, CO, USA – 3-6 September, 1996. Proceedings*, IEEE (1996), 336–343.

[8] Uckelmann, D., Harrison, M., and Michahelles, F. An architectural approach towards the future internet of things. In *Architecting the Internet of Things*. Springer, 2011, 1–24.