Poster Abstract: Dyser – Towards a Real-Time Search Engine for the Web of Things*

Benedikt Ostermaier, B. Maryam Elahi, Kay Römer
Institute for Pervasive Computing, ETH Zurich
Zurich, Switzerland
ostermaier@inf.ethz.ch, elahib@ethz.ch, roemer@inf.ethz.ch

Michael Fahrmair, Wolfgang Kellerer
Ubiquitous Networking Research, DOCOMO Communications Laboratories Europe GmbH
Munich, Germany
fahrmair@docomolab-euro.com, kellerer@docomolab-euro.com

ABSTRACT

The increasing penetration of the real world with embedded and globally networked sensors enables the formation of a *Web of Things* (WoT), where high-level state information derived from sensors is embedded into Web representations of real-world entities (e.g. places, objects, creatures). A key service for the WoT is searching for entities which exhibit a certain dynamic state at the time of the query, which is a challenging problem due to the dynamic nature of the sought state information and due to the potentially huge scale of the WoT. Below we report on our initial efforts to construct such a search engine and the underlying WoT.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Search process*; H.3.5 [**Information Storage and Retrieval**]: Online Information Services—*Web-based services*

General Terms

Design, Algorithms

1. INTRODUCTION

The increasing penetration of the real world with embedded and globally networked sensors (e.g., built into mobile phones), which are getting connected to the Internet, enables novel applications based on the current state of the real world. This trend has led to the development of middleware (e.g., GSN [1], SenseWeb [4]) that offers unified interfaces to access such sensors. Given these developments, we envision a *Web of Things*, where Web representations of real-world entities are augmented with real-time state information derived from sensors. For example, a Web page that represents a meeting room, containing static information on its size and location, could be augmented with real-time state information on the presence of people in the room. Another example are social networking platforms, where users may be interested in sharing their personal state (e.g., where they are, what they are doing, whom they are with) with their friends by publishing this dynamic state (e.g., derived from mobile phone sensors) on their personal Web pages. In contrast to existing middleware such as GSN or SenseWeb, the WoT is entity-centric rather than sensor-centric: We believe that users are primarily interested in *entities* of the real-world (e.g., places, objects, creatures) and their high-level *states* (e.g., room is occupied) rather than in *sensors* and their *raw output*.

As in the traditional Web, *search* is also a key service in the WoT, enabling users to find real-world entities that exhibit a certain *current* state (e.g., currently empty rooms of a certain size, locations where many people sharing my interests are currently hanging out). Searching the WoT is a challenging problem as the sought information (i.e., real-time state of entities) is highly dynamic, inherently distributed, and the number of entities in the WoT is potentially huge. Below we present our initial efforts to devise a WoT and a WoT real-time search engine.

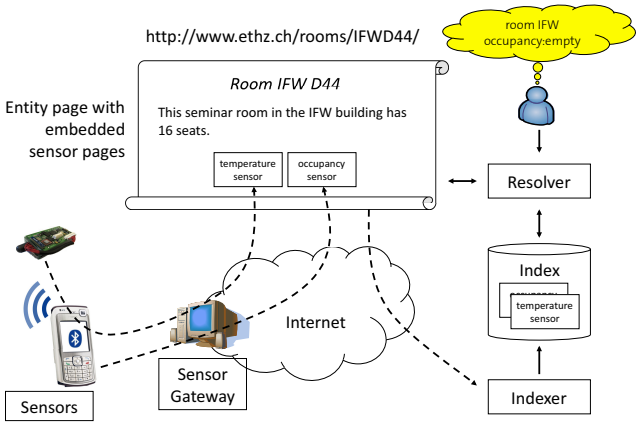

Figure 1: Overview of the system architecture.

2. TOWARDS A WEB OF THINGS

As our primary interest is the search engine, we devise simple and easy to use core abstractions for the WoT that capture the key concepts rather than aiming at a complete and comprehensive architecture. Fig. 1 depicts the key elements of our design. *Sensors* are connected to the Internet by *sensor gateways* which also transform and merge raw sensor data to obtain high-level state information. This high-level state information can be interpreted as the output of a virtual sensor [1]. For example, a virtual *room occupancy sensor* can output either *occupied* or *empty* and could be implemented by a PIR motion sensor, a humidity sensor (human breath raises air humidity), or by fusing the output of both (e.g., using middleware such as GSN or SenseWeb).

*This work was funded by DOCOMO Euro-Labs.

SenSys'08 November 5–7, 2008, Raleigh, North Carolina, USA.
ACM 978-1-59593-990-6/08/11.

Sensor pages are Web representations of virtual sensors, i.e., HTML pages which contain the type of the sensor (e.g., *room occupancy*), its current value (e.g., *occupied*), and additional meta information such as the location of the sensor. This information is semantically annotated using so-called microformats[1]. For example, the HTML code `<div class="sensortype">occupancy </div>` indicates that *occupancy* is the type of a sensor. The advantage of this approach is that microformats can be easily interpreted by applications relying on structured data (e.g., the search engine) and at the same time can be flexibly formatted for the presentation in a Web browser using Cascading Style Sheets (CSS).

Entity pages are Web representations of real-world entities, i.e., HTML pages which contain static descriptions of the represented real-world entity and data embedded from sensor pages, much like images can be embedded into Web pages using the `<img>` tag. In our current implementation, embedding is performed on download by the Web server using PHP.

3. SEARCH ENGINE

Dyser allows to search for entity pages that contain certain static information (e.g., *large* rooms) and exhibit a certain dynamic state (e.g., *empty* rooms). The syntax of the search language is based on that of major web-search engines. For example, the term `eth occupancy:empty` will find all entity pages which contain both the static keyword "eth" and an occupancy sensor which is currently sensing the state "empty". The dynamic nature of the state information constitutes the key difference and key challenge when compared to traditional search engines. It renders traditional indexing approaches useless as the state of entities typically changes far more frequently than this information can be re-indexed. Searching Web pages without some kind of index, as suggested in [3], does not scale to large sets of elements.

To deal with this challenge, our approach is based on the observation that a large class of state information tends to show periodic behavior, in particular people-centric states as humans are creatures of habit [5]. Exploiting this observation, we can construct prediction models from past states that allow us to estimate the probability that a given entity will be in a certain state at the time of a query. The idea is that gateways compute these models which are then included as meta information in sensor pages. Dyser crawls the WoT to find sensor pages, extracting and indexing these prediction models as depicted in Fig. 1. To answer a query, Dyser uses these indexed prediction models in a process called *entity ranking* to identify entity pages that are very likely to match the dynamic state requested in the query. After looking up candidate entity pages that match the static information given in the query (using traditional indexing techniques), Dyser uses the indexed prediction models to compute the probability of a candidate entity page matching the dynamic state requested in the query. The candidate entity pages are then ranked by descending probability and checked for actual compliance with the query in the order of their rank until enough matching entities have been found. For this check, the current values of embedded sensor pages are downloaded and compared to the values requested in the query. Dyser also assesses the accuracy of the individual prediction models to prioritize re-indexing of models with poor accuracy.

We are investigating different types of prediction models. One such model is based on the assumption that the output of a virtual sensor has a single dominant period, i.e., the output repeats (more or less) every period (e.g., day, week, month). Here, we compute the dominant period and the frequency distribution of the sensor values over the period duration using past states. Another, more elaborate model does not make the assumption of a single dominant period and is based on a technique to mine obscure periodic patterns [2]. Each such pattern describes the probability of a periodic behavior with a certain period length.

[1] see `http://microformats.org`

4. IMPLEMENTATION AND RESULTS

We implemented a prototype of the WoT and Dyser. The system supports both real sensors (we have deployed motion sensors in our offices to measure room occupancy) and recorded traces (we use the MERL [6] data set and logs from our university's online room reservation system). Instead of crawling the Web for sensor and entity pages, Dyser makes use of Google. For this, sensor and entity pages contain a special marker string that can be included in a Google query to find all entity and sensor pages.

While we are currently evaluating and optimizing our implementation, we have obtained initial results to assess the quality of the computed rankings as this is a key element of our approach. Each entity page in a ranked list may or may not actually match the query. An ideal ranking would contain all matching entity pages at the top of the list. To assess the quality of a ranking, we use the following metric. We lookup the lowest-ranked entity that matches the query, count all non-matching entity pages that are ranked higher, and divide this number by the rank of the lowest-ranked matching entity. As the search engine checks the entity pages for compliance with the query in the order of their rank, this metric measures the fraction of checked entity pages that do not match the query and hence represent overhead.

Using the single-period prediction model described earlier, we ran a query for occupied rooms on a room occupancy trace extracted from the MERL data set [6], which contains the output of more than 200 PIR motion sensors deployed in an office building. The average value of the above-defined metric over all the query instances was 0.033, meaning that only 3.3% of the checked entities did not match the query on average. This result is encouraging, as already a simple prediction model resulted in an accurate ranking.

5. CONCLUSIONS

Triggered by the availability of globally networked sensors, we have made an initial effort towards a Web of Things. To support searching for entities that exhibit a certain dynamic state, we are utilizing prediction models to rank entities according to their probability of matching a query. Initial experiments indicate that even simple prediction models result in accurate rankings. Using our approach, one might also be able to "search the future", by evaluating the prediction models with a future time. Although the actual future states are then unknown, the results may might still be useful.

6. REFERENCES

[1] K. Aberer, M. Hauswirth, and A. Salehi. Infrastructure for data processing in large-scale interconnected sensor networks. In *MDM*, 2007.

[2] M. G. Elfeky, W. G. Aref, and A. K. Elmagarmid. Using Convolution to Mine Obscure Periodic Patterns in One Pass. In *EDBT*, 2004.

[3] A. C. Ikeji and F. Fotouhi. An adaptive real-time web search engine. In *WIDM '99: Proceedings of the 2nd international workshop on Web information and data management*, pages 12–16, New York, NY, USA, 1999. ACM.

[4] A. Kansal, S. Nath, J. Liu, and F. Zhao. SenseWeb: An Infrastructure for Shared Sensing. *IEEE MultiMedia*, 14(4):8–13, 2007.

[5] J. Reades, F. Calabrese, A. Sevtsuk, and C. Ratti. Cellular Census: Explorations in Urban Data Collection. *IEEE Pervasive Computing*, 6(3):30–38, 2007.

[6] C. R. Wren, Y. A. Ivanov, D. Leigh, and J. Westhues. The MERL Motion Detector Datasets. In *MD '07: Proceedings of the 2007 Workshop on Massive Datasets*. ACM, 2007.