FuseViz: A Framework for Web-based Data Fusion and Visualization in Smart Environments

Giacomo Ghidini and Sajal K. Das
Center for Research in Wireless Mobility and Networking
The University of Texas at Arlington
E-mail: {giacomo,das}@uta.edu

Vipul Gupta
Oracle Labs, Oracle Corporation
Redwood Shores, CA
E-mail: vipul.x.gupta@oracle.com

***Abstract*—Recent advances in technology and algorithms for smart environments have made it possible to collect and store large amounts of data about many aspects of human life and the surrounding environment with limited effort and cost. However, such data become useful to lay users with no background in data analysis only if they are presented in a fashion that supports intuitive interaction to spot the patterns and trends, thus transforming the data into valuable information.**

In this paper, we introduce FuseViz, a framework for Web-based fusion and visualization of data in smart environments. FuseViz addresses the challenges posed by large, live, heterogeneous, and dynamic data streams from autonomous data sources, and lay users, with two basic features: fusion and visualization. CouchDB, a schemaless database with a ReSTful API and MapReduce support, is used to fuse data streams from multiple sources, while Web-based visualization is implemented on top of D3, a JavaScript library for manipulation of data-driven documents. We demonstrate the capabilities of FuseViz with E^{2}Home, a case study application for energy-efficient smart home environments. We show how the precise information provided by E^{2}Home can help the user easily improve the home energy efficiency by more than 10%.

***Index Terms*—data visualization; data fusion; smart environments; MapReduce; CouchDB; D3.**

I. Introduction

In recent years, pervasive computing and communications technology has enabled the development of applications across a wide spectrum of domains: from traffic and transportation [1] to smart environments [2], from health and well being [3] to fitness and sports [4], and from games and entertainment [5] to social networking [6], just to name a few. As depicted in Fig. 1, these domains have two important common features: *(i)* diverse data sources, and *(ii)* lay (application) users.

Data. Thanks to advances in pervasive computing, wireless mobile communications, and micro-electronic mechanical systems (MEMS) technology, data can be collected from a large array of sources spanning from wireless sensor nodes (*e.g.,* [8], [9]) and smartphones (*e.g.,* [10], [11]) to online databases [12]. To present a complete view of a domain (*e.g.,* smart home), an application often needs to rely on various data sources (*e.g.,* appliances and user's smartphone). Furthermore, the same data source (*e.g.,* user's smartphone) may contribute to applications in different domains (*e.g.,* smart home and road navigation). Therefore, it is important to develop a systematic solution for the *fusion* of multiple data sources in pervasive computing applications. Features of data streams and sources [13], such as size, format, liveness, heterogeneity, dynamicity, and autonomy, pose several challenges that the fusion mechanism should address.

Fig. 1. **Data sources and users.** In smart environments, valuable information for users is scattered in various kinds of data originating from diverse independent sources. A smart environment application should fuse the relevant data from all sources in a meaningful way, and present them to lay users in a format such that it is easy for them to explore the data and obtain the necessary information. (Redesign of Fig. 1 in [7].)

Users. Due to their pervasive nature, applications in the above-mentioned domains often feature an audience of lay users with no data analysis background. Therefore, it is important that the data be presented in a fashion that makes it easy for these users to analyze them intuitively, and extract the relevant information. *Visualization* is an effective solution for intuitive data analysis as it leverages the human vision capability to spot patterns and trends.

Case study. To better present our arguments about data and users, let us introduce an example which we use as the case study throughout this paper, namely the *Energy-Efficient Home* (E^{2}Home) application. In a smart home scenario, lay users are interested in the electricity usage at their home, and would like to understand the patterns and trends in order to adjust their behavior, thus making the home more energy-efficient.

First of all, data from the electricity meter are needed. However, these data alone are likely to be insufficient to understand how to change one's behavior to lower energy consumption. In fact, energy consumption data could be joined with the user location to understand where the user is, when high energy consumption occurs. The user could then try to reduce high energy consumption occurring when he/she is not home. For instance, the heating, ventilation, and air conditioning (HVAC) system could be better programmed to match the user's daily routine.

978-1-4673-2433-5/12/$31.00 ©2012 IEEE

Notwithstanding the final set of data sources, we observe that they are likely to be diverse and autonomous from each other, while all being relevant to the problem, and thus instrumental to improve the energy efficiency of a smart home. In our case study, electricity usage data are programmatically retrieved via the Smart Meter Texas (SMT) Web site [14], and the user's location is collected using an Android [11] app on a Google Nexus One [11] smartphone.

Our contributions. In this paper, we present FuseViz, a framework for the development of Web-based data fusion and visualization applications for smart environments. The objective of FuseViz is to exploit the full potential of available data sources in the application domain to the user's advantage by easing the ascent of the data-information-knowledge-wisdom (DIKW) pyramid [15] as depicted in Fig. 2. To achieve this goal, FuseViz relies on two basic features: *(i) fusion*, and *(ii) visualization*.

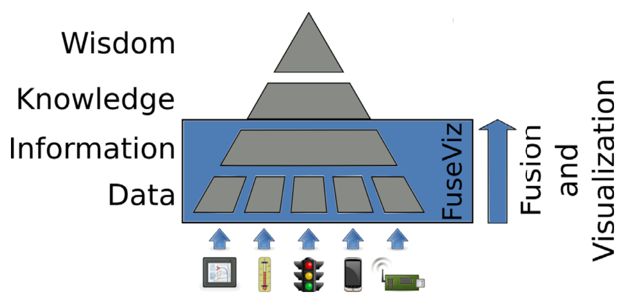

Fig. 2. **DIKW pyramid and the FuseViz framework.** In the DIKW pyramid model, data are transformed into information, then information into knowledge, and finally knowledge into wisdom. The FuseViz framework uses fusion and visualization to transform data from multiple sources in a smart environment into information for lay users.

In FuseViz-based applications, data from multiple sources are stored in CouchDB [16], a schemaless database, and then fused into a joint data stream using MapReduce [17] functions. To the best of our knowledge, the proposed fusion mechanism is the first solution to join multiple autonomous data streams in smart environments. Joint data streams are visualized on scalable vector graphics (SVG) [18] charts and maps in a Web page using D3 [19], [20], a JavaScript library to manipulate data-driven documents. The resulting SVG-based interactive visualizations are very responsive owing to element-specific re-rendering enabled by D3.

Due to fusion and visualization in FuseViz, data from multiple autonomous sources can be transformed into valuable information for lay users in any application scenario. To assess the proposed FuseViz framework, we developed several visualization applications, including the E^{2}Home app for the case study in this paper.

Outline. The rest of the paper is organized as follows. Section II summarizes the related work. Section III formalizes the requirements of the FuseViz framework, and discusses the related challenges. Section IV offers a detailed description of FuseViz, and Section V presents the case study of the E^{2}Home application. Section VI discusses the performance of FuseViz based on the E^{2}Home application, and compares it with that of other data storage and/or visualization frameworks. Finally, conclusions are drawn in Section VII.

II. Related work

Over the last few years, there has been a growing interest on visualization in the context of pervasive computing and communications. We observe a shift from customized solutions to more generic ones, and from static visualizations to interactive ones. Nevertheless, none of the existing systems for visualization of data from smart environments provides a solution for the fusion of multiple autonomous data sources.

Biketastic [21], [22] and the Copenhagen Wheel [23] are domain-specific systems that offer several advanced visualizations, including map overlays of data (*e.g.,* pollution levels, traffic congestion, and road conditions) collected by users.

Sensor.Network [7], [24] is a data storage and exchange platform for the Internet of Things (IoT) that offers a ReSTful (representational state transfer) [25] API for users, or devices/services acting on their behalf, to manage data streams as well as insert or retrieve data. As such, Sensor.Network is successful in creating a common place for storing data streams from different domains.

Earlier we developed SNViz [13] as a visualization framework for data streams in Sensor.Network [7]. SNViz caters to lay users by making it easy to visualize different dimensions of a data stream on charts and maps and providing support for panning, zooming, and brush-and-linking. However, interactivity suffers from the intrinsic shortcomings of using Protovis [26]. Furthermore, SNViz does not address issues arising from autonomous data sources as it is tightly coupled with Sensor.Network, nor data streams in Sensor.Network can be fused into a joint data stream.

FuseViz addresses the issues related to existing data visualization solutions in pervasive computing and communications, including diverse data streams and static visualizations. The outcome is a novel framework for Web-based fusion and visualization of data from multiple autonomous sources.

In the next section, we formalize the requirements that guide the development of FuseViz, and discuss the challenges.

III. Requirements and challenges

The FuseViz framework for analysis-oriented visualization by lay users in smart environments has the following requirements: *(i)* support for Web access; *(ii)* modular and extensible data sources and visualizations; and *(iii)* embeddable, interactive, and responsive visualizations.

First of all, FuseViz-based applications should be accessible from any device, including smartphones and tablets, because in a pervasive computing scenario users likely want to access the relevant information independent of the devices currently at their hands.

Second, FuseViz should be modular and extensible both in terms of data sources and visualizations, because application domains are very diverse.

Finally, FuseViz-based applications should be interactive, responsive, and embeddable in existing Web pages, in order to be more appealing for users interacting with them to explore the patterns and trends in the data.

In order to develop an effective solution for analysis-oriented visualization, several challenges posed by the nature of data and users need to be addressed. In this study, we argue that major challenges are due to the following features of the data streams and sources in a smart environment: *(i)* size, *(ii)* liveness, *(iii)* heterogeneity and dynamicity, and *(iv)* autonomy.

In the next section, we describe the architecture of the FuseViz framework, and highlight how it fulfills the requirements and addresses the challenges that we outlined.

IV. Proposed FuseViz framework

A. Preliminaries

Before describing the FuseViz framework, let us introduce two important building blocks: CouchDB [16] and D3 [19]. CouchDB is a distributed schemaless database with a ReSTful JSON (JavaScript Object Notation) API to manage databases and documents over HTTP. D3 is a JavaScript library for manipulating documents based on data.

As discussed in the rest of the paper, CouchDB and D3 are instrumental in providing fusion and visualization in FuseViz as they help us address the challenges due to data and users in smart environment applications.

B. Architecture

Figure 3 portrays the architecture of the FuseViz framework that consists of the following components: *collection, storage, fusion,* and *visualization.* In the following, these are described in details.

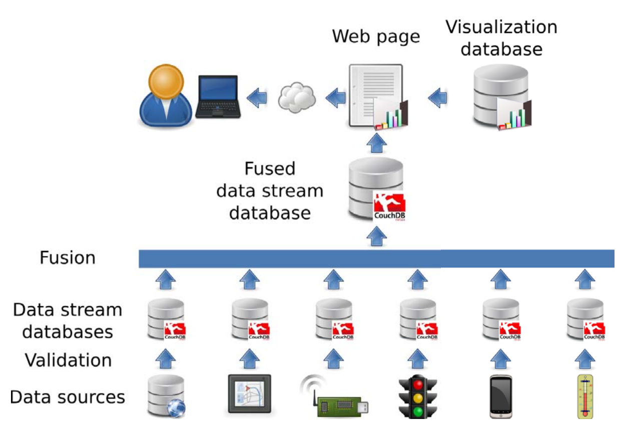

Fig. 3. **Architecture of FuseViz-based application.** Data from heterogeneous sources are *(i)* validated and pushed into CouchDB databases; *(ii)* fused into a single joint data stream using MapReduce functions; and *(iii)* presented to the user in a Web page using interactive visualizations.

1) Data collection: Data from all relevant sources for the visualization application are imported into a CouchDB database. Challenges due to data and sources are easily addressed using CouchDB. In fact, CouchDB's Erlang-based HTTP engine easily scales up to handle heavy loads of requests [27], thus enabling data collection for multiple autonomous, large, and live data streams. Furthermore, CouchDB's ReSTful JSON API provides a straightforward solution to import data from heterogeneous and autonomous sources.

2) Data storage: Using CouchDB, large, heterogeneous, and dynamic data streams from autonomous sources can be easily handled by FuseViz. As long as a data point passes validation, it can be stored in the CouchDB database for its data stream, thus avoiding problems incurred into if using relational databases with a fixed schema.

3) Data fusion: Once data streams used in the FuseViz-based application are stored into CouchDB databases, they need to be fused into a joint data stream to be visualized on a Web page. Fusion is needed because autonomous data sources are likely to generate data streams with different frequencies and offsets, or unnecessary dimensions.

The data fusion component addresses several challenges due to the features of data streams. First of all, the MapReduce paradigm is well-suited to process heterogeneous and dynamic data streams. Not only can specific operations be designed for different data streams being fused, but also they can be easily updated. Second, the result of MapReduce functions is stored in the database, so that they do not have to be recomputed for each request. Therefore, the computational load on CouchDB and the latency of HTTP responses are minimized even in case of large data streams. Finally, the result of a MapReduce function is recomputed on-the-fly for a HTTP request, if there exist new data points in the database due to live data streams.

4) Data visualization: Once data from multiple sources have been fused into a joint data stream, they are ready to be visualized in a Web page. Since the objective of a Web-based data visualization application is to be accessible by the users anytime anywhere, it is important that the technologies used in its implementation are supported not only on laptops and desktops, but also smartphones and tablets. We decided to use HTML, JavaScript, SVG and cascading style sheets (CSS) as these technologies are standards and are already available on a wide array of devices, or are likely to become available in the near future.

Use of D3 implies large, live, and heterogeneous data streams can be rendered on interactive visualizations for lay users. First of all, D3 allows for editing of subsets of DOM (*e.g.,* SVG) elements, so that only those elements affected by user-generated events (*e.g.,* brush-and-linking) are updated. Similarly, elements can be added to or removed from visualizations of live data streams without having to re-render other elements. Furthermore, D3's agnostic approach to standards does not restrict visualizations to one technology (*e.g.,* SVG). As a result, visualizations including not only scalar data, but also multimedia ones (*e.g.,* images) can be incorporated, thus broadening the applicability of FuseViz to more domains in pervasive computing. Finally, D3's selection mechanism greatly simplifies the implementation of panning, zooming, and brush-and-linking: three features instrumental to transform raw data into valuable information for lay users.

C. Application

A Web-based analysis-oriented data visualization application can be quickly developed on top of the visualization and fusion components in FuseViz.

The joint data stream is retrieved from the CouchDB database via a HTTP GET request to the list or view URI. The visualizations are loaded as scripts in the Web page along with the visualization manager. Visualizations are laid out on the Web page and registered with the visualization manager responsible for passing the data points to them, and relaying events, such as brush-and-linking, among them.

New data points in live data streams are retrieved using AJAX (asynchronous JavaScript and XML), and are seamlessly visualized using D3's support for dynamic data sets via transformations. Since CouchDB uses Etags [28], bandwidth consumption due to periodic polling of data stream lists or views by client Web pages is minimized. Due to CouchDB's support of JSON padding (JSONP), visualization applications can also be embedded in other Web pages.

D. Implementation

FuseViz is implemented as a set of CouchApp [29] applications. CouchApp is a toolkit on top of CouchDB that enables the development of Web-based applications. In FuseViz, we have implemented sample CouchApps for source and joint data streams, visualization, and application. The sample CouchApps feature also default validation, view, and list functions, when applicable. As we demonstrate in the next section, developers can easily implement and deploy a data visualization application using these CouchApps as building blocks.

V. Case study: E^{2}Home

To showcase the strengths of FuseViz, we developed a few data visualization applications, including the E^{2}Home application for the case study described in Section I. To recap, the objective of the E^{2}Home application is to provide lay users with valuable information about their home electricity consumption, so that they can take actions to make the home more energy-efficient. We argue that electricity consumption data alone are not sufficient to achieve this goal, and other data should be also incorporated, such as the user's location.

The architecture of the E^{2}Home application follows the template set by FuseViz, and thus consists of data collection, storage, fusion, and visualization on a Web page. The electricity usage data are retrieved from the SMT Web site [14], and are imported to the CouchDB data stream database using a Python script. The user location is retrieved using an Android app developed for E^{2}Home, and then posted to the CouchDB data stream database.

The E^{2}Home application is accessed via a Web page currently featuring two visualizations: *(i)* a focus and context time line, and *(ii)* a map on which the data points retrieved from the remote CouchDB live joint data stream database are plotted. For each data point, the average power consumption is plotted on the time line, while the average location is plotted on the map. A screenshot of the visualizations is portrayed in Fig. 4.

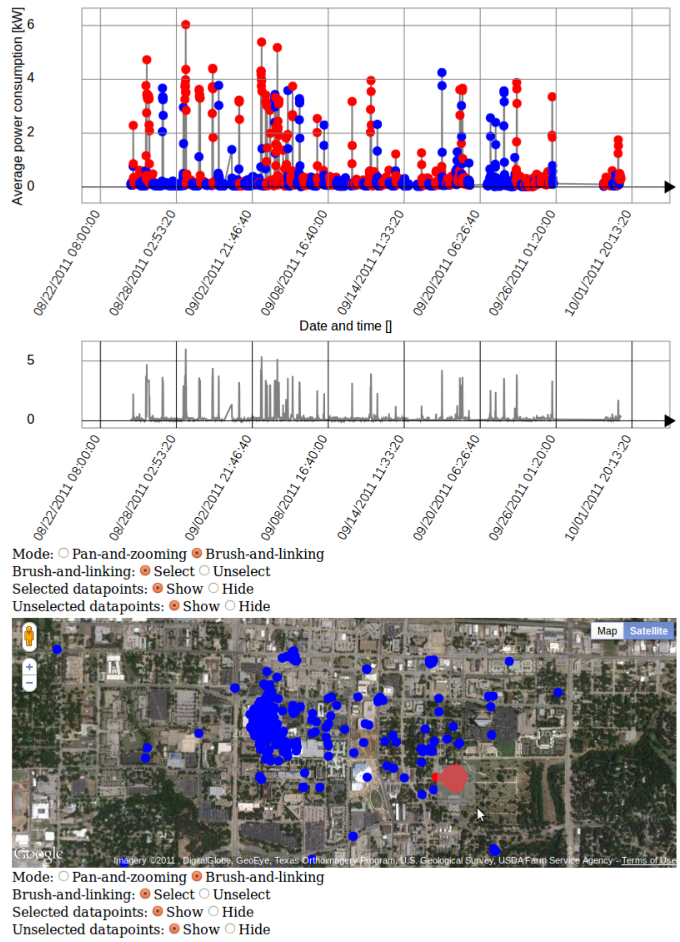

Fig. 4. **Brush-and-linking on E^{2}Home data.** The user can select (brush) data points on the map corresponding to time slots when he/she was home (red data points inside gray rectangle in bottom chart), and then observe the average power consumption during the selected time slots (linking). Significant levels of average power consumption are observed also when the user is not home (blue data points > 2 kW in top chart). This could be due to wasteful consumption (*e.g.,* A/C on when nobody is home), or not (*e.g.,* washer on). Incorporating additional data sources such as all tenants' locations and energy consumption of individual appliances would provide even more valuable information.

VI. Discussion

We perform the following analysis of the experimental data to measure what impact the FuseViz-based E^{2}Home application can have on making a home more energy efficient. The electricity usage for the 38-day period between 08/24/2011 and 09/30/2011 is reported in Table I. The 32 data points with high average power consumption (> 2 kW) have a disproportionate impact on the overall energy consumption. Although existing

TABLE I

Electricity usage 08/24/2011 – 09/30/2011. The joint data stream covers 30 days 13 hours instead of 38 days because the user location was not available for short time periods when the smartphone was turned off. The (Not home, > 2 kW) and (Not home, ≤ 2 kW) sets consist of unselected data points (blue) in Fig. 4 with average power consumption above 2 kW, and below 2 kW, respectively; the (Home) set contains selected data points (red).

Data point set	No. of data points	Time [dd:hh:mm]	Total energy [kWh]	Avg. energy $\left[\frac{\text{kWh}}{\text{data point}}\right]$
Not home, > 2 kW	32	00:08:00	24.71	0.772
Not home, ≤ 2 kW	$1,530$	15:22:30	49.438	0.032
Home	$1,370$	14:06:30	143.817	0.105
All	$2,932$	30:13:00	217.965	0.074

applications such as Google PowerMeter [30] and SMT [14] provide a time chart for the electricity consumption, they do not include the contextual information (*i.e.,* user location) needed to help the user easily identify the exact times when high average power consumption occurred while he/she was not home.

Now, using the information provided by the E^{2}Home application, the user knows exactly at what times the high energy consumption occurred when he/she was not home. Therefore, he/she can take precise actions to remove this wasteful consumption in the following days. If this is the case, the 32 data points accounting for wasteful high energy consumption will be affected in the future, thus bringing about a projected overall energy efficiency improvement of 10.86%.

FuseViz visualizations are much more responsive since data rendering and re-rendering are much faster than in SNViz. FuseViz supports open development of data visualization applications, since any set of data sources can be used and fused into a joint data stream. Finally, other features of FuseViz, such as JSONP-support and use of well-established standards, such as SVG and HTTP, make it very appealing for the embedding of FuseViz-based data visualization applications on social networking Web sites and blogs.

VII. Conclusions

In this paper, we presented a novel framework, called FuseViz, for the development of Web-based data fusion and visualization applications in smart environments. We analyzed the requirements and challenges of such a system, and provided a thorough description of the proposed solution. To demonstrate the capability of FuseViz, we developed E^{2}Home, a data visualization application for smart home energy consumption, and showed how lay users can change their behavior to improve the home energy efficiency by more than 10% based on the valuable information presented by the FuseViz-based application.

We are currently working on the incorporation of multimedia data streams, such as images and audio/video, in the FuseViz framework.

Acknowledgments

The authors would like to thank David Levine at UT Arlington for pointing them to the Smart Meter Texas Web site; and Michael Bostock at Stanford University for answering questions about D3.

References

[1] INRIX Inc., "INRIX Traffic," 2011. [Online]. Available: http://www.inrixtraffic.com

[2] Mobile Integrated Solutions LLC, "MobiLinc App," 2011. [Online]. Available: http://www.mobileintegratedsolutions.com

[3] Demand Media Inc., "LiveStrong iPhone iPad App Calorie Tracker," 2011. [Online]. Available: http://www.livestrong.com/thedailyplate/iphone-calorie-tracker

[4] Nike, "Nike+," 2010. [Online]. Available: http://nikerunning.nike.com/

[5] Groundspeak Inc., "Geocaching Live," 2011. [Online]. Available: http://www.geocaching.com/live/default.aspx

[6] Foursquare Labs Inc., "foursquare," 2011. [Online]. Available: https://foursquare.com

[7] V. Gupta, P. Udupi, and A. Poursohi, "Early Lessons from Building Sensor.Network: an Open Data Exchange for the Web of Things," in *8th IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, 2010, pp. 738–744.

[8] Sun Labs, "Sun SPOT World," 2010. [Online]. Available: http://sunspotworld.com/

[9] MEMSIC Corp., "Wireless Modules," 2010. [Online]. Available: http://www.memsic.com/products/wireless-sensor-networks/wireless-modules.html

[10] Apple, "Apple iPhone," 2010. [Online]. Available: http://www.apple.com/iphone/

[11] Google Inc., "Android," 2011. [Online]. Available: www.android.com

[12] U.S. Government, "Data.gov," 2011. [Online]. Available: www.data.gov

[13] G. Ghidini, V. Gupta, and S. K. Das, "SNViz: Analysis-oriented Visualization for the Internet of Things," in *Urban Internet of Things Workshop*, 2010.

[14] Smart Meter Texas, "Smart Meter Texas," 2011. [Online]. Available: http://www.smartmetertexas.com

[15] R. L. Ackoff, "From Data to Wisdom," *Journal of Applied Systems Analysis*, vol. 19, pp. 3–9, 1989.

[16] Apache Software Foundation, "CouchDB," 2011. [Online]. Available: http://couchdb.apache.org

[17] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," in *Proc. of the 6th Symposium on Operating System Design and Implementation (OSDI)*, 2004.

[18] W3C, "Scalable Vector Graphics," 2011. [Online]. Available: http://www.w3.org/Graphics/SVG

[19] M. Bostock, "D3.js: Data-Driven Documents," 2011. [Online]. Available: http://mbostock.github.com/d3

[20] M. Bostock, V. Ogievetsky, and J. Heer, "D3: Data-driven documents," *IEEE Transactions on Visualization and Computer Graphics*, in press.

[21] S. Reddy, K. Shilton, G. Denisov, C. Cenizal, D. Estrin, and M. Srivastava, "Biketastic: Sensing and Mapping for Better Biking," in *Proc. of the 28th International Conference on Human Factors in Computing Systems (CHI)*, 2010.

[22] UCLA's Center for Embedded Networked Sensing, "Biketastic," 2010. [Online]. Available: http://biketastic.com/

[23] MIT SENSEable City Lab, "The Copenhagen Wheel Project," 2010. [Online]. Available: http://senseable.mit.edu/copenhagenwheel

[24] Oracle Sun Labs, "Sensor.Network," 2010. [Online]. Available: http://sensor.network.com

[25] E. Wilde, "Putting Things to REST," UC Berkeley School of Information, Tech. Rep., 2007.

[26] M. Bostock and J. Heer, "Protovis: a graphical toolkit for visualization." *IEEE Transactions on Visualization and Computer Graphics*, vol. 15, no. 6, pp. 1121–8, 2009.

[27] J. C. Anderson, J. Lenhardt, and N. Slater, *CouchDB: The Definitive Guide*, 1st ed. O'Reilly, 2010.

[28] W3C, "HTTP/1.1: Header Field Definitions," 1999. [Online]. Available: http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html

[29] CouchApp.org, "Simple JavaScript Applications with CouchDB," 2011. [Online]. Available: http://couchapp.org

[30] Google Inc., "Google PowerMeter," 2011. [Online]. Available: www.google.com/powermeter