# SensorMap for Wide-Area Sensor Webs

**Suman Nath, Jie Liu, and Feng Zhao**
Microsoft Research

**The SensorMap portal and its accompanying tools will allow for more online live data.**

Geocentric Web interfaces such as Microsoft Virtual Earth and Google Maps are useful for visualizing spatially and geographically related data such as driving directions, directory entries, and weather and traffic conditions, to name a few. The desire to add useful information to these interfaces has led developers to create custom applications that overlay housing prices, crime rates, bus locations, and other data on top of browsable maps. These applications are possible due to useful APIs that Google Maps and Microsoft Virtual Earth publish to overlay location data on maps.

We envision a new class of applications that relies on real-time sensor data and its mash-up with the geocentric Web to provide instantaneous environmental visibility and timely decision support. As an example, imagine wanting to find current waiting times at all Thai restaurants in a university district. This would require an infrastructure to support publishing and querying real-time data over interfaces such as the geocentric Web.

## EXISTING SOLUTIONS

Existing solutions are useful for writing simple applications, but there are impediments to using them for more sophisticated purposes.

First, publishing even a single data stream is not a trivial task. Much useful data isn't being published because the data owners don't have enough programming expertise, and publishing the data requires too much effort.

Second, existing applications are mutually incompatible. A user can't bring up a single map that shows both restaurant information and crime rates in an area.

Third, existing solutions don't provide useful primitives, such as querying live sensors on demand based on keywords or location and aggregating the results in useful ways.

The SenseWeb Project (http://research.microsoft.com/nec/senseWeb) at Microsoft Research addresses these challenges with a Web portal called SensorMap (http://atom.research.microsoft.com/sensormap) and a set of tools that data owners can use to easily publish their data. Users can take advantage of the portal and tools to make queries over live data sources.

SensorMap transparently provides mechanisms to archive and index data, process queries, and aggregate and present results on geocentric Web interfaces such as Microsoft Virtual Earth. We believe that such a platform will encourage the community to publish more live data on the Web and enable people to build useful services on top of it.

## SENSORMAP ARCHITECTURE

As Figure 1 shows, the current SensorMap prototype is a centralized Web portal consisting of four components: the GeoDB database, the DataHub Web service, the Aggregator for creating icons, and the SensorMap (GUI).

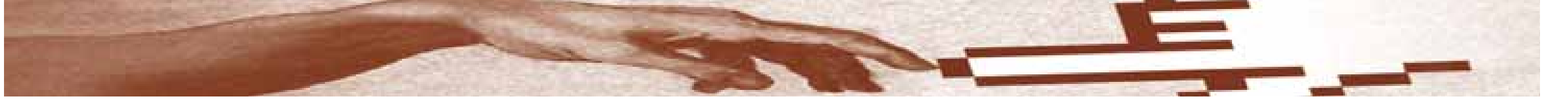
### GeoDB

GeoDB is a database housing sensor metadata including the publisher's name; the sensor's location, name, and type; the data type; and freetext descriptions. We envision users basing their queries on sensor types, descriptive keywords, and geographic locations, such as the list of cameras along a route or the average temperature that thermometers report inside a geographic region. GeoDB uses geo-indexing techniques to efficiently support these types of queries.

### DataHub

DataHub provides two ways to make real-time data available on SensorMap. Sensors with public Web interfaces can register their URL directly to GeoDB. The SensorMap client uses these URLs to fetch real-time data. For sensors with an Internet connection but no URL (such as those in mobile phones or behind firewalls), DataHub provides a simple interface to cache sensor data.

The sensors are clients for DataHub and can send data in real time using standard Web service calls. The Aggre-



gator or the SensorMap GUI directly retrieves these cached data from DataHub rather than try to contact the sensors.

### Aggregator

The Aggregator creates icons representing sensor data that users can mash up with maps. Depending on the sensor type, an aggregator can reside on either the client or server side. It accepts queries from the client and redirects the geographic components of the queries to GeoDB.

After obtaining the metadata of a set of sensors that satisfies a client query, the Aggregator contacts the sensors and DataHub for their real-time data. It then aggregates the data accordingly (depending on sensor type and the underlying map's zoom level). For example, for data collected from thermometers, the Aggregator displays average and standard temperature deviations reported in a neighborhood. By doing so, SensorMap usefully summarizes data to the client without clogging the map with overlapping icons.

### SensorMap GUI

The SensorMap GUI is based on Windows Live Local and therefore provides features such as zooming, panning, street maps, satellite images, and 3D views. In addition, it lets end users pose queries on available sensors. SensorMap currently supports three types of queries:

- geographic queries that drawing geometric shapes directly on the map specifies (for example, within a region, near a route);
- type queries that sensor types specify within the viewport; and
- freetext queries that keywords describing sensors specify.

The GUI overlays the results that the Aggregator returns on Windows Live Local maps. GeoDB and the Aggregator are transparent to both the data publishers and users. The GUI lets users save views (geographical region or sensor-type filters) on the client machine as cookies for quick retrieval. Figure 2 depicts an interface showing street-parking data that StreetlineNetworks published for a section of San Francisco.

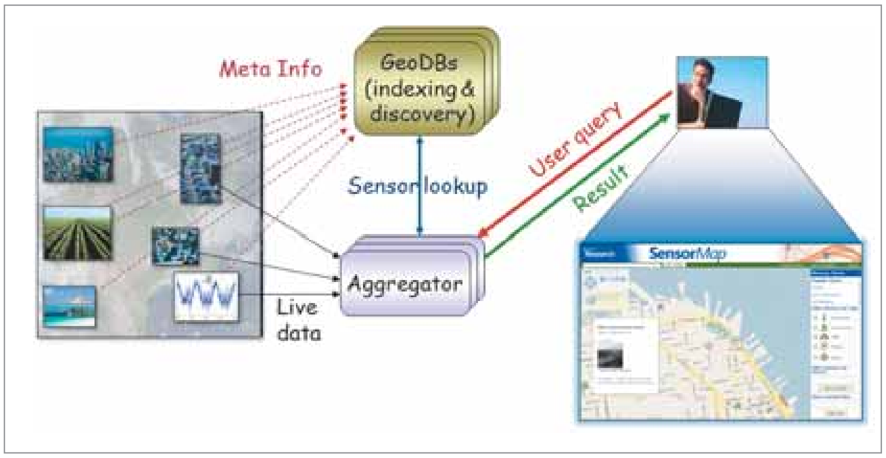


*Figure 1. SensorMap architecture. The prototype calls for a centralized Web portal consisting of four components.*

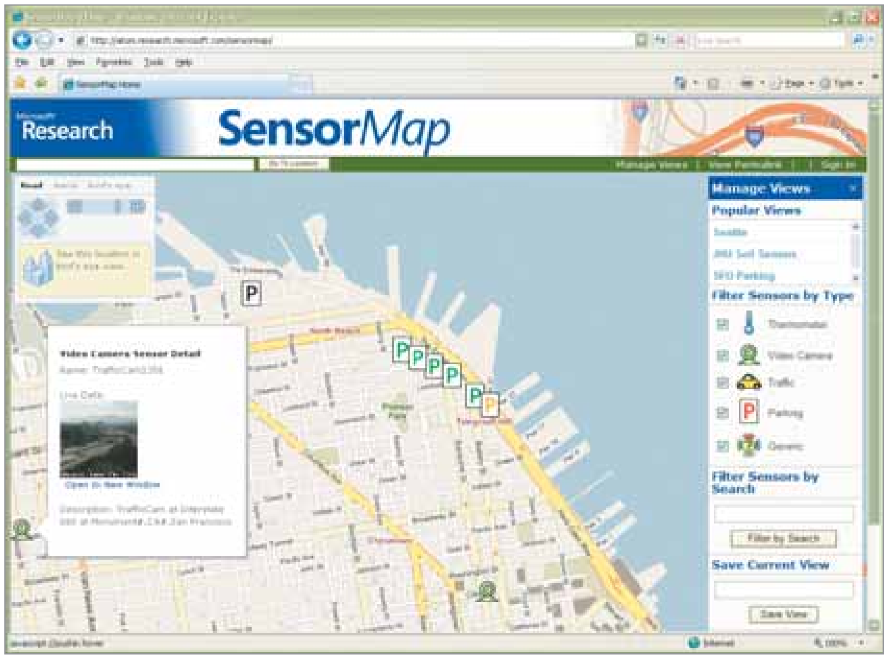


*Figure 2. SensorMap GUI. Based on Windows Live Local, the GUI provides zooming, panning, street maps, satellite images, and 3D views.*

## ADDRESSING KEY CHALLENGES

The challenges in building a Web portal like SensorMap primarily stem from the goal of collecting and presenting continuously changing and diverse types of data, which pushes the limits of current Web technologies.

### Data publishing

Aggregating data from vastly different sensors and services on a shared Web portal poses a few fundamental challenges. First, data sources might have very different interfaces such as intermittent links, proprietary communication protocols, and accessibility policies. Moreover, networked sensors, even the stationary and Internet-ready ones, are typically behind firewalls due to management boundaries and security concerns.

SensorMap uses Web services to tunnel through firewalls with HTTP ports and XML encoding. Before publishing data to SensorMap, a data publisher must first register the sensor by providing its static description using the sensor description markup language. Sensors behind firewalls can push data to the DataHub Web service to act as an external cache of data. All these capabilities are supported by the MSRSense toolkit, which data publishers can use to easily incorporate their sensors into SensorMap.

Another data-publishing challenge is interoperability and extensibility of different sensor types. For example, public Web sites like weather.com, as well as weather stations from hobbyists' backyards, can publish weather data. To meaningfully aggregate data from multiple sources, we need common representations of sensor types and units.

Semantic Web technologies can address many of these problems. For example, using a standard ontology to publish data will enable automation of processing tasks within the portal.

## Scalable data management

The large amount of data a portal provides poses new data-management problems. Consider a centralized solution where the portal itself collects sensor data on demand, computes clusters at required granularity, and aggregates data within each cluster. This offers a different model from the traditional data warehouse where underlying data changes infrequently, or a traditional data stream where data is continuously pushed and queries are long running.

A more appropriate model is to maintain an approximate view of the database and to materialize the portion of interest on demand. A user can perform materialization by using unexpired cached data and collecting additional data from a carefully chosen subset of sensors in the area of interest.

SensorMap addresses these requirements with COLR-tree, a novel data structure that provides an R-tree-like interface and hides the materialization of sensor data from users. It incorporates the cost of collecting data from the sensors into query planning, optimizations, and cache management.

Moreover, to answer aggregate queries over a large geographic region, COLR-tree collects data from a carefully chosen subset of sensors. Finally, it performs spatiotemporal caching of aggregate and raw sensor data to allow queries to reuse data even if queries overlap partially in space and time and different sensors have different expiry times.

**The large amount of data a portal provides poses new data-management problems.**

## Data visualization

The display must depict the variety of sensor data on the portal in meaningful ways. SensorMap's current version shows sensor data as points. An icon whose shape and color encode sensor type and current sensor value represents each sensor or sensor type.

For much sensor data, such a simple display method is insufficient. For example, the system could better display data from a dense deployment of temperature sensors as contour maps showing temperature gradients. The system could display a traffic sensor's archived data in a way that highlights temporal congestion patterns. SensorMap aims to identify and provide a small set of simple abstractions useful for composing a variety of visualizations.

## Sensor discovery

Many useful sensors already exist on the Web. For example, many transportation departments put traffic cameras online, the US Geological Survey puts real-time stream-gauge information on the Web, and so on. Just as existing search engines automatically crawl the Web to discover new pages, a sensor portal needs similar crawlers to automatically discover and index live data sources online.

Although people can access most existing sensors through their Web pages, automatically discovering them is challenging for a few reasons. First, these pages aren't easily identifiable as sensor pages. Second, unlike typical Web pages, sensors don't link to each other. Finally, even if you find a page representing a sensor, it's not easy to extract the necessary metadata that describes the sensor.

SensorMap's current version includes a crawler that addresses the problems in a narrow domain. It can automatically discover traffic cameras available on the Web and annotate them with their latitude and longitude.

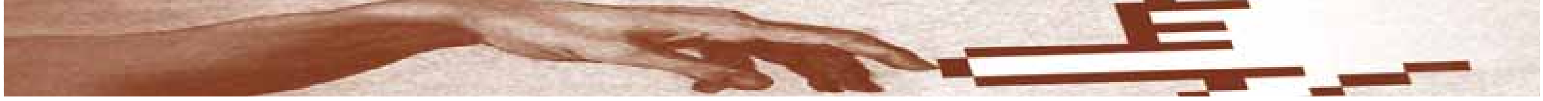
## Mash-up APIs

To realize the full potential of a portal like SensorMap, it should be easily extensible and mashed up with other applications and service. As example scenarios, a user should be able to visualize real-time traffic data from SensorMap and driving directions from MapPoint (www.microsoft.com/mappoint) together or set a trigger that sends a message when traffic conditions are optimum.

Providing a general mash-up framework is tricky since it might need to deal with the semantics of the underlying data. In sending a trigger to advise a user of optimum traffic conditions, the mash-up code must parse and understand the traffic data from SensorMap. We are currently working on a set of modular and composable APIs to facilitate mashing up SensorMap with other services.

## Other challenges

Privacy and data ownership are big concerns for sharing physical, real-time data. Sensor data might reveal other information about publishers and their surroundings. A publisher might want to control how the data is being used.



We've resolved these issues by using an authentication framework and allowing a publisher to decide his sensors' privacy levels. For example, just the publisher, a designated group he creates, or everyone might view the sensor.

However, this isn't sufficient. A much deeper social issue is data ownership. Just because someone can set up a Web camera from his apartment window to view a restaurant across the street doesn't give him the right to publish that restaurant's waiting time on the Web. These social concerns are beyond our current technical focus, but they require further investigation because they might have profound implications for SensorMap's success.

We released the first version of SensorMap in July 2006. In addition to the data sources we incorporated, a few projects (Streetline Networks' San Francisco parking-spot availability and Johns Hopkins University's soil data) have volunteered to publish data on SensorMap. Currently, we're working toward a new version that addresses many of the challenges to encourage publishing more data on SensorMap.

*Suman Nath and Jie Liu are researchers and Feng Zhao is a principal researcher in the Networked Embedded Computing Group of Microsoft Research. Contact Nath at sumann@microsoft.com, Liu at liuj@microsoft.com, and Zhao at zhao@microsoft.com.*

**Editor: Wayne Wolf, School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta; wayne.wolf@ece.gatech.edu**