Data Management in the Worldwide Sensor Web

Harvesting the benefits of a sensor-rich world presents many data management challenges. Recent advances in research and industry aim to address these challenges.

Magdalena Balazinska
University of Washington

Amol Deshpande
University of Maryland

Michael J. Franklin
University of California, Berkeley

Phillip B. Gibbons
Intel Research

Jim Gray and Suman Nath
Microsoft Research

Mark Hansen
University of California, Los Angeles

Michael Liebhold
Institute for the Future

Alexander Szalay
Johns Hopkins University

Vincent Tao
Microsoft

With the rapidly increasing number of large-scale sensor network deployments, the vision of a worldwide sensor web is close to becoming a reality. Ranging from camera networks that monitor large wildlife reserves to biological sensors implanted in the body to monitor vital signs, these deployments generate tremendous volumes of priceless data. Simply put, data is the raison d'être of any sensing exercise. Most sensor network researchers would probably agree that we have placed too much attention on the networking of distributed sensing and too little on tools to manage, analyze, and understand the data. However, with standards in place for many of the networking and other infrastructure-related issues, or at least an emerging consensus on how to solve them, the demands of a sensor web are largely shaping up to be questions of information management. What, then, are the challenges in working with data in distributed sensing systems; what services might promote good data sharing and analysis practices? Can we design sensor networks with data quality in mind?

Motivating application: Public healthcare

Consider, for instance, how we might build a public-health early warning system on top of the sensor web. First, we would deploy sensor sources in large ecosystems, such as habitats and watersheds, and in human bodies. In fact, several such deployments are already being put in place. Researchers in public-health management of zooanotic diseases (diseases that can be transmitted between species) such as *E. coli*, West Nile virus, and avian flu are beginning to deploy such global data-collection networks, for detecting the pathogen in the environment and for tracking infected animals. Hospitals and other healthcare providers are beginning to report to the US Centers for Disease Control and Prevention in real time (for example, through the CDC's PulseNet database). Medical patients, including those infected with zooanotic diseases, are increasingly being monitored in real time (using RFID tags and/or wearable or implanted sensors that can monitor cardiac, respiratory, and other vital signs).

Second, we would build a system that combines these reports into a data warehouse that would serve to provide early warning and track and manage an outbreak's evolution. But to monitor and manage an outbreak such as avian flu, many domains must be able to gather and exchange data. We must also be able to gather and process the data at high speed and serve it to higher-level processes such as large-scale data mining, analysis, visualization, and modeling.

Published by the IEEE Computer Society ■ 1536-1268/07/$25.00 © 2007 IEEE

Managing sensor web data

The worldwide sensor web will generate too much data to visualize or analyze manually. The popular tools of the trade, such as Matlab, R, Mathematica, and Excel, cannot scale up to the worldwide sensor web's needs—and they are primitive by most standards. On-the-fly and highly distributed aggregation, fusion, and summarization of the data are a must to handle the data volumes and the distributed nature of the data generation. Furthermore, to simplify interaction with the data, the sensor web must incorporate logical data abstractions and visualizations that can shield users from the complexities of the underlying sensing infrastructures but still propagate measures of uncertainty associated with calibration or sampling effects.

Database management systems provide a good starting point for managing, storing, and processing sensor data. They can scale up to very large volumes of data and can exploit the significant processing power of parallel and distributed systems. They provide logical data independence, using high-level abstractions to shield the user from the underlying hardware and software platforms. Databases support declarative querying interfaces that are significantly more intuitive than writing custom programs for processing the data. They also support several intuitive concepts for managing and analyzing large-scale data.[1] (For example, one can use the data cube operator to quickly browse through enormous multidimensional data sets.) Finally, databases provide facilities both to protect data by keeping many replicas and to protect privacy with record-granularity access control mechanisms.

Existing database systems, however, have several critical shortcomings that prevent using them directly to process live sensor data. First, they are too heavyweight and slow, devoting much complexity to handling tasks that might be irrelevant to the sensor web (for example, transactions). Several recently developed lightweight data management systems somewhat alleviate this concern. Furthermore, the aforementioned advantages will start outweighing this concern as the scale of the data being managed increases. More important, however, the sensor web's global distributed scale and the sensor data's inherently noisy and ephemeral nature raise challenges that are vastly different from those of managing the enterprise data for which databases are typically built.

We now look at some of the most important data management challenges that must be solved to enable the worldwide sensor web vision. Simultaneously, we present several recent advances in data management research that address these challenges, with the hope that the scientific, ubiquitous, and sensor network communities will embrace these solutions.

Data ingest

Sensor networks generate multidimensional data streams. Each stream has some common metadata, such as the organization responsible for the deployment (and basic facts about the deployment design), sensor type, location and physical context, calibration parameters, precision, accuracy, and maintenance history. However, the bulk of the information is often a time series of measurements (granted, we must take a liberal view of the term "time series"; in the case of camera networks, the observations taken in time are images). In one common operational model, a sensor reports a measurement averaged over a given time period; the measurement area (or measurement point) is often fixed and promoted to the metadata. So when asking who-what-when-where-why, the data stream is a where-when-what array or, for fixed instruments, a when-what array. In actuated networks, sensors report data only when they have detected an event. An event might consist of threshold crossings of measurements from other sensors on a given node. Or, it could be a trigger sent by a device at a higher level in the network (with a corresponding shift in scale spatially and possibly temporally). For actuation, the "why" becomes important.

A sensor network's data streams present, almost by definition, complex issues related to data quality. Data is often missing, and when not missing is subject to potentially significant noise and calibration effects. For example, temperature and moisture sensors report voltages that must be converted to temperature (Celsius) and moisture (partial pressure and dew point) units. Also, because sensing relies on some form of physical coupling, the potential for faulty data is tremendous. Depending on where a fault occurs in the data reporting, observations might be subject to unacceptable noise levels (for example, due to poor coupling or analog-to-digital conversion) or transmission errors (packet corruption or loss).

A sensor network's data streams present, almost by definition, complex issues related to data quality. Data is often missing or subject to noise and calibration effects.

Applications that draw on this data, or end users hoping to perform an analysis, will need to contend with observations that do not fit nicely on a grid in either space or time. Spatially, complexities in

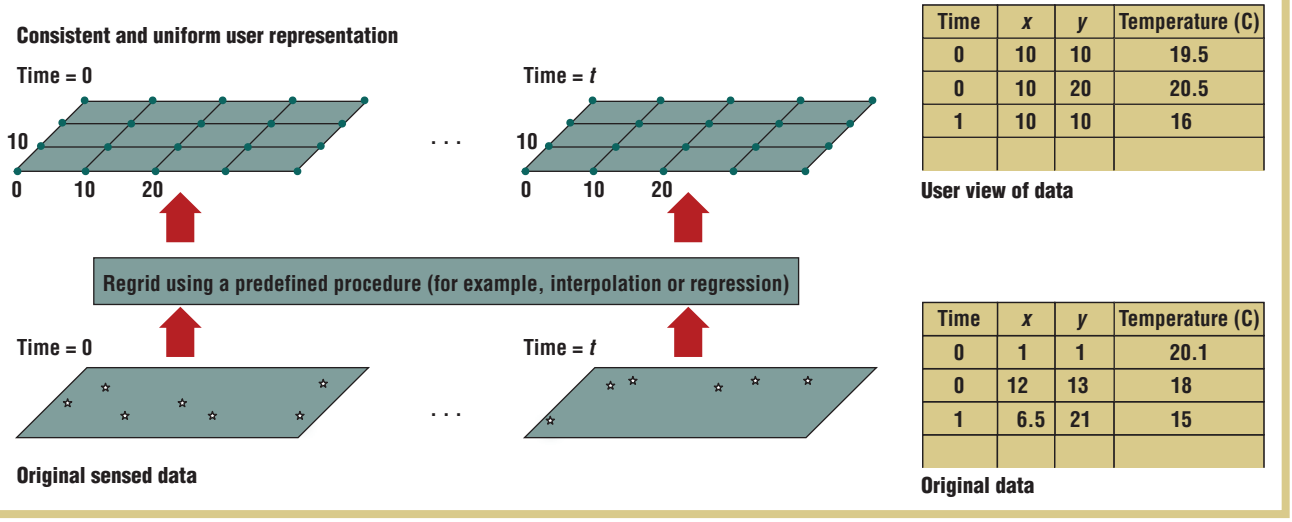


Figure 1. Sensor networks typically generate highly nonuniform observations, and regridding is necessary before the data can be further processed or analyzed. We can use abstractions such as table functions, user-defined functions,[2] or model-based views[3] to push this process into databases.

the local terrain or intentional experimental design choices might have dictated an irregular arrangement of nodes (keep in mind that the dominant early design paradigm for sensor networks was random placement). Temporally, missing data or actuated observations yield complex reporting patterns.

So, these data streams are often "gap filled" by interpolating (estimating) nearby and historical data to fill in missing values. They are then regridded to convolve the measured data into a uniform time-space data set that is easier to analyze, visualize, and compare with other data sets in a uniform framework (see figure 1). These operations can be quite complex, depending on the kinds of auxiliary observations available and their correlations with the signal of interest, and on the underlying physical phenomena's spatial and temporal variability. These gap-filling observations must provide some estimate of uncertainty for the gridded data. Indeed, points that were determined from a handful of neighboring (original) observations will be noisier than those involving hundreds of nearby data points. In all cases, we would prefer to preserve the original data set, but most analysis would use the regridded data.

Data ingest is the generic name for this calibrate, gap-fill, regrid process. Building and operating the data-ingest pipeline is a major part of building and operating a sensor information system. Unfortunately, there are few generic tools for data ingest, so each system is largely custom built. Database systems have extract-transform-load tools to build data warehouses, but ETL tools typically lack the calibration, gap-fill, and regrid software. Instrument vendors often provide calibration software, but that is just one component of data ingest—and the software from different instruments and sensor networks is quite diverse. Generic data-ingest tools that you can easily customize for specific sensor networks would be a real boon, saving experimenters considerable design and deployment effort.

Managing temporal and spatial data

Time and space play central roles in representing and analyzing sensor data. They often provide the linking dimensions that let us combine and compare data sets. Continuing the healthcare example, geolocating patients allows comparison of disease rates with detection of diseased animals in various regions and comparison of climate variables (temperature and pollution levels) with epidemiological data. Understanding trends in time and space (historical analysis) and responding to emerging trends or localized events (real-time detection and response) are important functions that we need to support.

Spatial point data is easily represented by geographical coordinates (latitude and longitude). The OpenGIS standard (www.opengeospatial.org/standards) defines a representation for regions and gives an algebra for point-point, point-region, and region-region comparisons. The representation of spatially gridded data has received less standardization. Gridding is implicit in the HDF (Hierarchical Data Format) and NetCDF (Network Common Data Form) standards that represent data as a (time) sequence of *N*-dimensional arrays. However, gridding algorithms are so diverse and idiosyncratic that no standard has yet emerged.

Temporal representations might seem obvious, but again there are no standard tools for temporal regridding. Also, representing spatiotemporal objects such as a disease's spread over a geographic area is not well understood and has no off-the-shelf solution. In many such cases, a

model for the data (at a fundamental level, regridding is a kind of modeling), or rather the sufficient statistics of a model, might be a useful abstraction for representing such objects.

Combining two or more gridded data sets to produce derived data is common (for example, to compare human disease rates to animal disease rates). Although conceptually simple, performing the calculations and visualizing the results require some programming today (as would performing more formal statistical analyses to make quantitative comparisons or conducting hypothesis tests of various kinds). Once configured and loaded with data, geospatial toolkits and workbenches such as ESRI's ArcInfo make certain kinds of analysis straightforward—but setting up such systems requires considerable skill.

Data exploration, analysis, and visualization

Data analysis begins with, well, data. In the context of a sensor web, that implies the analyst either knows the data sources or can identify them through some search or discovery process. Search in this context can be carried out at the metadata level (the sensor locations and types, the governing authority, and so on) or through links from previous analyses. Both of these frameworks are close in spirit to how we identify resources on the World Wide Web, and we can imagine crawlers or other kinds of automated processes that build an index of available data.

Sensing applications, however, might also want to consider characteristics of the sensed signal when forming a search query: "Provide me with a list of sites experiencing a drop of 20 percent in the last hour," "Find me regions of the country with measurements in a certain interval," and so on. Sensor database projects such as sensorbase.org are already investigating this kind of signal search, but there is much work to do. In addition, the underlying framework for constructing data streams, publishing them, and republishing after aggregation and computation is still in its infancy.

Once the analyst has identified data sources, he or she must deal with a host of data quality issues. As we already discussed, faults and gaps plague current sensing deployments, leading various groups of researchers to label data quality as *the* problem facing embedded sensing.

Even if we could assemble a perfect kind of time-space matrix of data, complete with uncertainty estimates, the analysis of multimodal, multiscale, spatiotemporal data would still be a challenge.

But even if we could assemble a perfect kind of time-space matrix of data, complete with uncertainty estimates, the analysis of multimodal, multiscale, spatiotemporal data would still be a challenge. Models are likely to be a mix of mathematical specifications (partial differential equations) and stochastic components and must be easily updatable given the stream nature of sensor data. Once an analysis is performed (perhaps through modules in analysis packages such as R, Matlab, SAS, or ArcGIS), there is an inevitable desire to share the analytic products and computations and to republish the data for others' use. Toward this end, projects such as Kepler (http://kepler-project.org) are attempting to create workflows from data streams, linking analysis and data and achieving a kind of prepublication mechanism.

Finally, the analysis itself might take place in various contexts, in the field or in an office. The kinds of visualizations that would be effective differ greatly according to the context (for instance, in the field, the researcher might carry only a handheld device).

Statistical modeling of sensor data

Statistical analysis and modeling are perhaps the most ubiquitous processing tasks performed on sensor data. This has always been true of scientific data management, where sensor data collection usually aims to study, understand, and build models of real-world phenomena. Increasingly, however, the need to use statistical-modeling tools arises in nonscientific application domains as well. Many of the most common sensor-data-processing tasks can be viewed as applications of statistical models. Examples include

- forming a stochastic description or representation of the data,
- identifying temporal or spatial trends and patterns in the data,
- online filtering and smoothing (for example, Kalman filters),
- predictive modeling and extrapolation,
- detecting failures and anomalies, and
- probabilistically modeling higher-level events from low-level sensor readings.

Statistical models for spatial data arise in the context of function estimation (smoothing) and spatial process fitting (Kriging and Gaussian processes). In particular, hierarchical Bayesian models of spatial processes have received considerable attention in statistics research in the last decade or so. Owing primarily to advances in Bayesian computation, rich families of

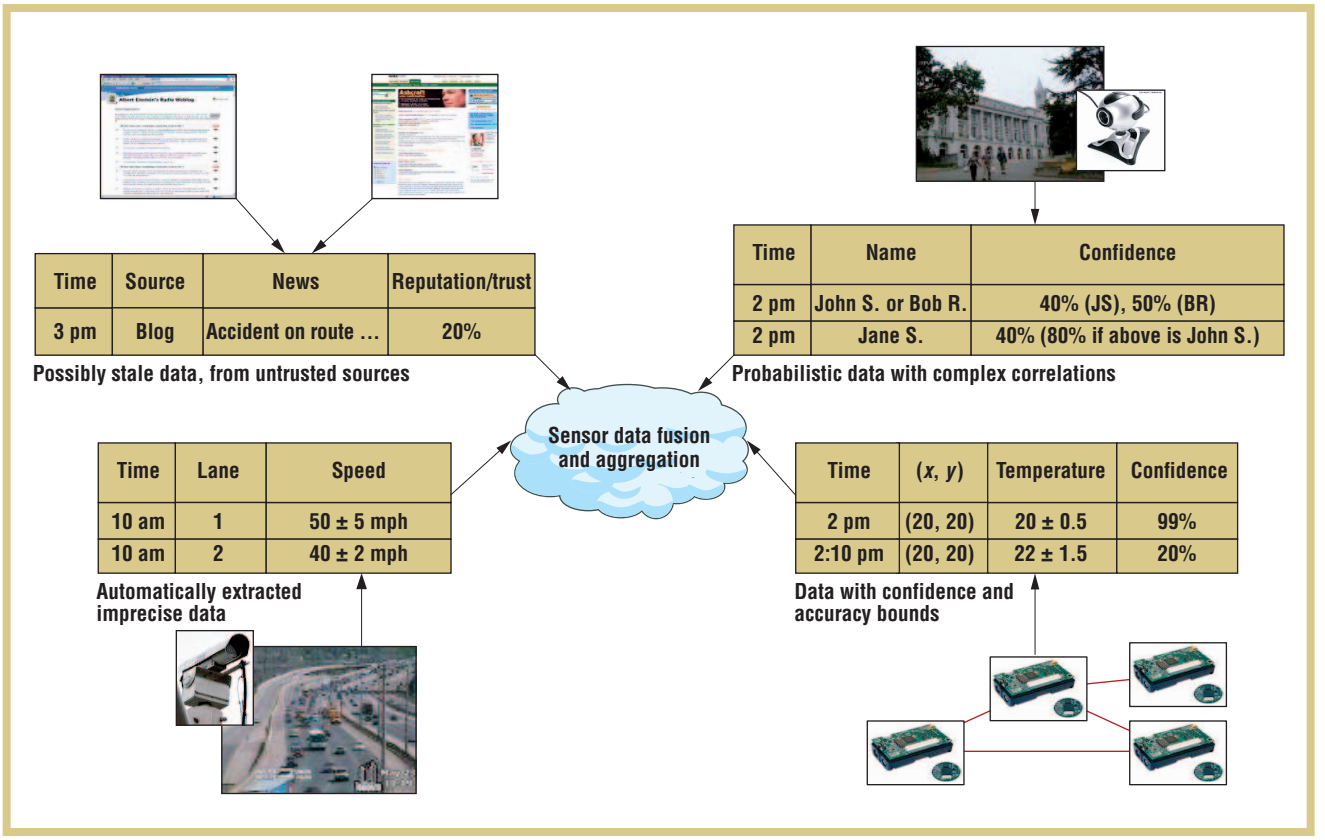


Figure 2. Even if the sensor web data sources used intuitive, well-defined interfaces to publish their data, the complex and semantically disparate measures of data quality and uncertainty typically associated with sensor webs make data fusion a challenge.

models are available to describe many environmental processes. Spatiotemporal modeling has received similar attention and is an area of continued growth.

Unfortunately, today's data management systems do not support statistical-modeling tasks natively, forcing users to employ external tools for this purpose. Scientists, for instance, typically import raw data into an analysis package such as R, Matlab, SAS, or ArcGIS, where they apply various models to it. Once they have filtered the data, they typically process it further using customized programs that are often quite similar to database queries (for example, to find peaks in the cleaned data, extract particular subsets, or compute aggregates over different regions).

Some traditional database systems do support querying of statistical models. For example, DB2's Intelligent Miner supports models defined in PMML (Predictive Model Markup Language). These approaches tend to abstract models as user-defined functions that one can apply to raw data tables. Several commercial data-mining tools (for example, the SAS Analytics Tool) can similarly interact with databases. This level of integration is, however, insufficient for many applications because no support exists for efficient parameter updates or for maintaining the models when new data is available.

On a more promising note, several researchers have recently considered integrating specific statistical models into database systems, for both ease of use and more efficient execution. For example, researchers have considered integrating tasks such as association rule mining,[2] interpolation,[4] and classification models[5] inside database systems. Similarly, the MauveDB system aims to integrate arbitrary statistical models into relational databases through the *model-based views* abstraction[3] (see figure 1).

Far from suggesting a single modeling paradigm, we would hope that a sophisticated data management platform for sensor network data would let us audition different tools via some form of workflow tool (such as in the Kepler project). A common format for models and the products of modeling could provide the required flexibility. Recent progress toward unifying the treatment of probabilistic models (for example, through abstractions such as dynamic Bayesian networks) could also pave the way for building a rich toolbox of statistical models.

Managing data uncertainty

Sensor web data inherently has some notion of quality attached to it—confidence, trust, accuracy, a probability distribution over possible values, and so on, or even a mix of these (see figure 2). Traditional database systems, built with

exact data in mind, are not well suited to manage such data. In recent years, researchers have tried to extend relational databases to handle uncertain data. This research has spanned issues ranging from theoretical development of data models and data languages to aspects of practical implementation such as indexing techniques.

The overall problem's complexity has resulted in numerous approaches to handling uncertain data. Much of this research has used probability theory as the basis for representing uncertainty. Under this model, the data uncertainty is encoded in the form of probabilities, and the operations on the uncertainty itself are in accordance with the laws of probability theory.

Probabilistic databases

Approaches to integrating probabilities into a relational-database framework fall into two broad categories. *Tuple-level uncertainty approaches* attach existence probabilities to the tuples of a database table. Consider, for instance, an RFID reader that detects a tag but does not have sufficient confidence in the detection. An exact approach would require the reader to either report the tag or not report the tag, both of which are unsatisfactory. Instead, if the reader attaches a confidence value to the tag, the applications at the higher layer can make more informed decisions.

Attribute-level uncertainty models associate (possibly continuous) probability distributions with tuple attributes. Such models are especially attractive for sensor data because the measurement errors and noise typically do not allow exact determination of the sensor attributes.

Several recent systems for managing probabilistic data support tuple-level or attribute-level uncertainty or, in some cases, a mix of the two. For a survey of recent research in this area, see the March 2006 *IEEE Bulletin of the Technical Committee on Data Engineering*.

Most sensor network deployments implement ad hoc and case-by-case solutions to deal with data uncertainty. Probabilistic databases promise a systematic, intuitive alternative to handle such uncertainty. However, several significant hurdles still must be crossed before those become a reality. Even if such systems were in place today, their application for sensor network data would still be challenging. Sensor data contain numerous sources of noise and uncertainty, some of which are assessable only after spatial or temporal aggregation. So, while simply attaching a notion of the individual sensor's accuracy to each observation would seem straightforward, we might also need to consider other sources of uncertainty that this number does not capture. Examples of such sources include those related to the sensor's physical coupling, its calibration, and actuation logic.

Data provenance

A related issue to managing data uncertainty is data provenance. In a worldwide sensor web, monitoring, data collection, and data analysis will be carried out in a distributed, autonomous fashion. In such a domain, the ability to determine a particular piece of information's origins, including the sequence of processing tasks that generated it, becomes critical. This is required for both tracing back incorrect information to its origins and deciding how much to trust it. Data provenance refers to recording such information with the data itself by making the data self-describing.

Data provenance is especially hard in a sensor web scenario for two reasons. First, most sensor data is typically thrown out after an initial fusion or aggregation step. Second, the data typically also goes through many layers of summarization, aggregation, and modeling, making reasoning about the final result's origins nearly impossible. Although researchers in the databases and sciences communities have been investigating data provenance, more work is necessary.

Data interoperability

A useful query on the worldwide sensor web might need to compare or combine data from many heterogeneous data sources maintained by independent entities. For example, while treating a patient, healthcare professionals might query hospitals for the patient's health profile and airports for his or her recent travels. They might then correlate this information with similar information from other patients suffering from similar diseases. Because much of the query-processing task in the worldwide sensor web will be automated, data must have a well-defined syntax and semantics.[6] Several such standards are emerging for low-level representations—for example, the IEEE 1451 series of transducer standards, and the DICOM (Digital Imaging and Communications in Medicine) standards.

Similarly, the Semantic Web can address many of the technical challenges of enabling interoperability among data from different sources. This technology enables information exchange by putting data with computer-processable meaning (semantics) on the World Wide Web.

The Semantic Web has three key aspects. First, data is encoded with self-describing XML identifiers, enabling a standard XML parser to parse the data. Second, the identifiers' meanings (properties) are ex-

In a worldwide sensor web, monitoring, data collection, and data analysis will be carried out in a distributed, autonomous fashion.

pressed using the Resource Description Framework. RDF encodes the meaning in sets of triples, each triple being like an elementary sentence's subject, verb, and object, with each element defined by a URI (uniform resource identifier) on the Web.

Finally, ontologies express the relationships between identifiers. For example, two data sources can publish data in XML as "<Temperature><Celsius>20</Celsius></Temperature>" and "<Temperature><Fahrenheit>68</Fahrenheit></Temperature>." An associated RDF document can describe that Celsius and Fahrenheit are temperature units, and an ontology can define the relationship between Celsius and Fahrenheit. So, a data-processing system can automatically infer that these two data points represent the same temperature value. Such encoding typically introduces overhead both in the effort to define and agree on the XML tags and ontologies and in the extra bytes that must be carried around with data. Although some cases (for example, in a closed enterprise) might employ more succinct representations, seamless interoperability across organizations might not be achievable without incurring such overhead.

Major industries are working to establish their own ontological standards for the Semantic Web. Noteworthy examples are RosettaNet (electronics), the Open Travel Alliance (travel), STAR (Standards for Technology in Automotive Retail), and UMLS (Unified Medical Language System). The Open Geospatial Consortium's Sensor Web Enablement initiative has made solid progress in building a framework for describing geospatial information and tools to support interoperable processing.[7] The initiative's goals include discovery of sensor systems and their capabilities, access to sensor parameters and live data, and tasking sensors. These goals align closely with the vision of a worldwide sensor web.

Much of the framework for a semantic sensor network is in place, but the remaining work will require contributions from many communities. First, data publishers need to start publishing data using Semantic Web technologies. Unfortunately, there are not many useful tools to aid them, and so far, data publishers typically have not found enough incentives to write their own tools (let alone have sufficient expertise to do so). To address this problem, more software tools must be developed, on different platforms, to ease publishing data through XML, RDF, and ontologies and to discover, compare, query, analyze, combine, or integrate diverse data sources. Finally, domain experts must develop domain-specific structured vocabularies and ontologies that data publishers can use. Several disciplines have already started the process, and Vincent Tao, Steven Liang, and Arie Croitoru provide one early example of sensor-interoperable processing based on the Open Geospatial Consortium's sensor web framework.[8] But no one has undertaken such efforts for a large fraction of the available data (for example, community sensor data).

Because sensors are geographically distributed and produce data at high rates, the worldwide sensor web will require a distributed data management infrastructure.

Distributed, large-scale data processing

Because sensors are geographically distributed and produce data at high rates, the worldwide sensor web will require a distributed data management infrastructure (see figure 3). In this infrastructure, sensor data is stored near its source, and data processing and filtering are pushed to the edges. Such an architecture reduces bandwidth requirements and enables parallel processing of sensor feeds. Queries can be posed from anywhere in the world, but often the querier and the data of interest will be in the same geographical region (for example, healthcare professionals will query data about patients in their vicinity). Building such a distributed, large-scale data-processing infrastructure presents four key challenges.

Query processing

For a number of reasons, a worldwide sensor web will be a distributed-systems nightmare. While many distributed systems are geared toward workloads that are read-intensive, low volume, or not time critical, the sensor web will be write-intensive, high volume, and often time critical. Additionally, the sensors, storage nodes, and users might be mobile. The data will often be multimodal (from scalar temperature readings to rich multimedia data), and the system itself will be heterogeneous in its mix of sensor platforms, storage platforms, communication capabilities, and administrative domains. Sensor nodes will be subject to harsh conditions, yet critical data should always be available. The worldwide sensor web must also support both sensing and actuation. A key goal is to make this distributed, heterogeneous system both easy to manage and a convenient application platform. It should thus be queriable as a unit, with standard query-processing languages that hide the underlying infrastructure's complexity.

The long history of distributed databases has not adequately addressed these

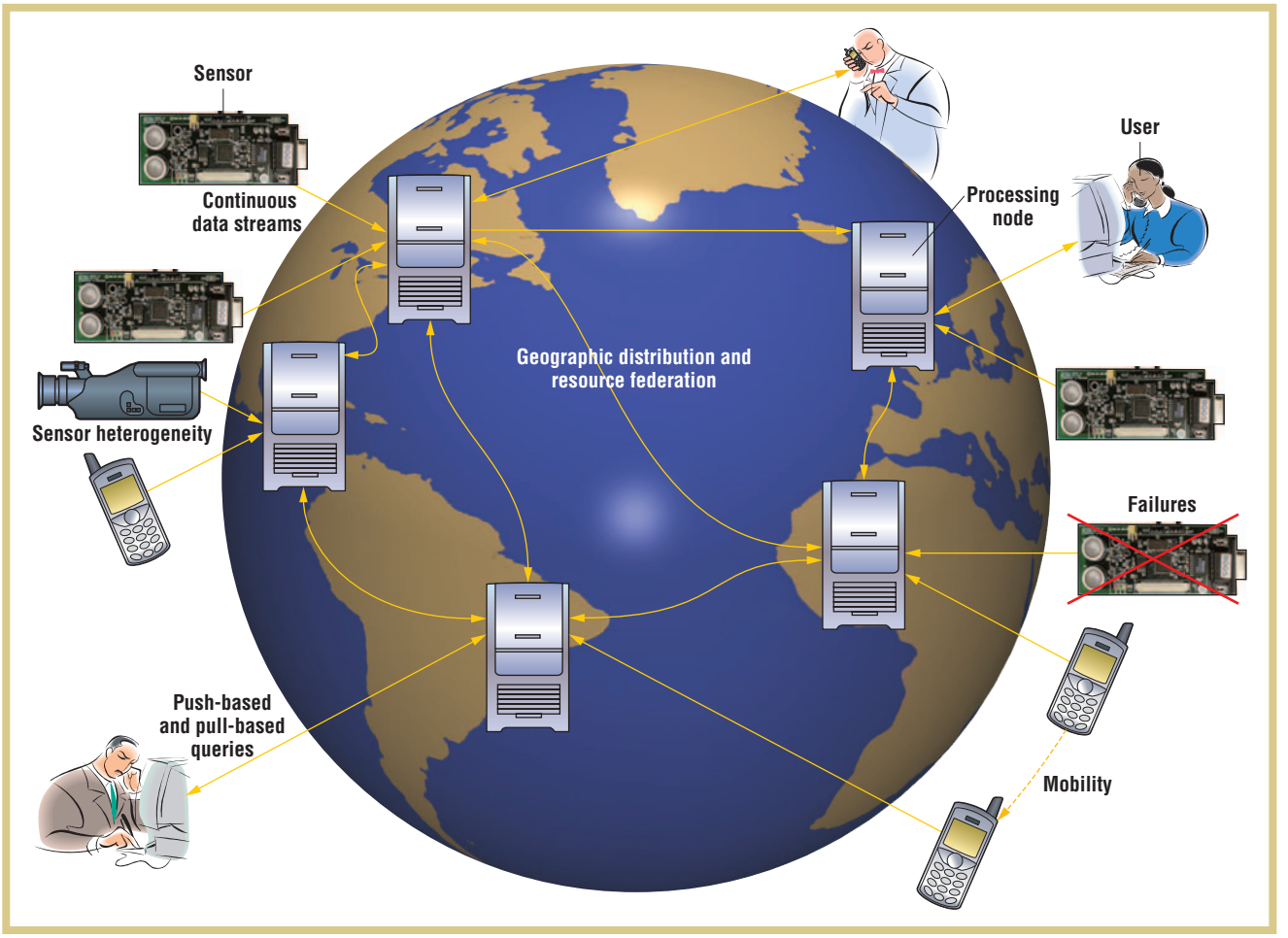


Figure 3. The worldwide sensor web distributed infrastructure.

challenges, focusing instead on traditional data sources and settings, and limited scale. Recent work on peer-to-peer data-processing systems[9,10] has begun addressing some of the scaling and robustness challenges in shared wide-area distributed systems, using distributed hash tables (DHTs) that support a simple put-get interface. An early system designed to provide a worldwide sensor web is IrisNet.[11] It supports distributed XML processing over a worldwide collection of multimedia sensor nodes, and addresses a number of fault-tolerance and resource-sharing issues. The Aurora*/Medusa,[12] Borealis,[13] and Telegraph-CQ[14] distributed-stream-processing engines enable low-latency continuous processing of push-based data streams from geographically distributed sources. These engines provide sophisticated fault-tolerance, load-management, revision-processing, and federated-operation features for distributed data streams. HiFi also supports integrated push-based and pull-based queries over a hierarchy where the leaves are the sensor feeds and the internal nodes are arbitrary fusion, aggregator, or cleaning operators.[15]

While these systems provide steps in the right direction, none addresses all the challenges of the worldwide sensor web. The latter requires rethinking design decisions at a whole new scale, for a whole new class of applications.

Continuous, integrated push-based and pull-based processing

Traditional databases support one-time queries over stored data, while stream-processing engines[12,13] focus on processing push-based data continuously. Some systems have started to recognize the need to support integrated queries over live and historical data.[14,15] Such queries are critical to help users understand new events in the context of past observations.

To enable queries over both live and historical data, the worldwide sensor web infrastructure must make important decisions regarding where to store data, how much data to store, what data to summarize, and when to discard data. Combining continuous and historical queries also raises new challenges regarding choosing locations for different operators (that is, sending queries to the data or data to the queries). This problem is especially hard because of the data archive size and the high aggregate data rate.

Reliability

With its large scale and the brittleness of its input data sources, the worldwide sensor web will never be fully functional: a subset of the nodes will always be encountering component and communication failures. Moreover, as in any web-accessible system, flash crowds (a sudden increase in the number of users on a Web site) and other load spikes will require the system to quickly react to large-scale load variations. The sensor web thus needs to define quality-of-service goals that are sufficient in this new environment yet do not require excessive resources.

As we mentioned before, IrisNet, Aurora*/Medusa, Borealis, TelegraphCQ, and DHT-based systems provide mechanisms for increasing the availability and durability of data in large-scale, distributed push-based or pull-based systems and for load managing in those systems. What makes the sensor web particularly challenging is its unprecedented scale, its mix of pull-based and push-based processing, and its infrastructure and administrative heterogeneity.

Federated and shared infrastructure

As with any other expensive scientific apparatus (for example, large-scale telescopes), economics dictate that the sensor web be a shared resource. Because building the worldwide sensor web requires a huge investment of money, time, effort, and maintenance, different parties must provide and maintain subsets of all sensors and servers. Additionally, many users worldwide will want to run queries not only on nearby sensors but also on sensors in various remote locations. Because processing is distributed and occurs close to sensing devices, the resulting infrastructure must be federated.

A rich literature exists on federated distributed systems. However, supporting many different types of applications requires carefully planning the most suitable abstractions that the system should expose. These abstractions should support a large class of applications, providing them flexible features while hiding system complexity. Sharing resources also requires models and algorithms to define and achieve fairness between applications and to address problems of competing actuation, where different applications want to actuate the system in conflicting ways (for example, pan a camera in opposite directions). Finally, data sharing raises important privacy and security concerns.

As with any other expensive scientific apparatus (for example, large-scale telescopes), economics dictate that the sensor web be a shared resource.

Data privacy and security

Much of the data that the sensor web collects will be highly private (for example, location information). Developing mechanisms to control access to this data, keep it secure, and regulate its use are among the sensor web's most important challenges. This problem is in part one of public policy, but we must still provide technologies that support the desired policies.

Security and privacy issues arise at several places in the sensor web.[16] First, the data might be compromised on the way between the sensors and the system because of compromised sensor nodes or eavesdropping. Fortunately, resilient hardware and encryption can largely solve this problem. Once the data reaches the system, keeping the data secure and ensuring privacy remain nontrivial tasks.

Database systems have long supported authorization mechanisms that grant or restrict data access at different granularities (tables versus attributes), in many cases using multiple security levels (secret, confidential, and so on). Recently, Rakesh Agrawal and his colleagues have proposed *Hippocratic databases*, which have privacy as the central tenet, and have identified the technical challenges in building such systems.[17] Hippocratic databases explicitly manage privacy by letting users control the storage and release of their personal data individually. There has also been much research on building data storage solutions on top of untrusted servers (for example, Oceanstore[18]). These solutions provide more limited authorization mechanisms but allow distributed storage. Researchers have also proposed algorithms for performing complex computations (for example, database joins) on untrusted servers. These techniques are critical for the sensor web, where different parties will own and administer subsets of all resources.

These approaches can help control access to data while allowing limited sharing or publishing of data. In general, we would like to share as much data as possible without compromising individual privacy. However, reasoning about and characterizing privacy loss as a function of released data have proven challenging. For example, an adversary that gains access to temperature sensors in a home might be able to extract details about the inhabitants' private activities.[16] Reasoning about such inferences requires modeling complex real-world processes and knowledge.

To address this challenge, many promising approaches have emerged in recent years, typically under the topic of *privacy-preserving algorithms*.[19,20] These

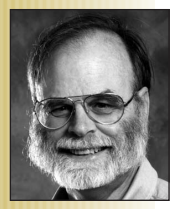
the AUTHORS

Magdalena Balazinska is an assistant professor in the University of Washington's Department of Computer Science and Engineering. Her research interests are broadly in the fields of databases and systems. Her current work focuses on building monitoring systems, including systems for computer network monitoring and RFID data management. She received her PhD in computer science from the Massachusetts Institute of Technology. She's a member of the IEEE, the ACM, and ACM SIGMOD. Contact her at the Univ. of Washington, Dept. of Computer Science and Eng., Box 352350, Seattle, WA 98195-2350; magda@cs.washington.edu.

Amol Deshpande is an assistant professor in the Department of Computer Science at the University of Maryland, College Park. His research focuses on data stream processing, data management for sensor networks, statistical modeling of data, and probabilistic and uncertain databases. He received his PhD in computer science from the University of California, Berkeley. He's a recipient of the US National Science Foundation CAREER award. Contact him at the Univ. of Maryland, Computer Science Dept., 3221 A.V. Williams Bldg., College Park, MD 20742; amol@cs.umd.edu.

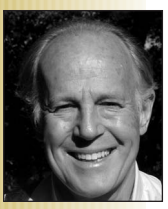
Michael J. Franklin is a professor of computer science at the University of California, Berkeley and CTO of Amalgamated Insight. At Berkeley, his research focuses on the architecture and performance of distributed data management and information systems. He received his PhD in computer science from the University of Wisconsin, Madison. He's a fellow of the ACM and a recipient of the US National Science Foundation CAREER award and the ACM SIGMOD Test of Time award. Contact him at the Computer Science Division, 687 Soda Hall, #1776, Univ. of California, Berkeley, CA 94720-1776; franklin@cs.berkeley.edu.

Phillip B. Gibbons is a principal research scientist at Intel Research Pittsburgh. His research interests are sensor systems, parallel and distributed processing, and querying massive databases and data streams. He received his PhD in computer science from the University of California, Berkeley. He's a fellow of the ACM, cited for contributions to parallel computing, databases, and sensor networks. Contact him at Intel Research Pittsburgh, 4720 Forbes Ave., Ste. 410, Pittsburgh, PA 15213; phillip.b.gibbons@intel.com.

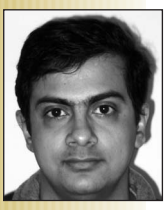


Jim Gray is part of Microsoft's research group. His research focuses on e-science—using computers to analyze scientific data—and on the related topics of databases and transaction processing. He received his PhD in computer science from the University of California, Berkeley. He's a fellow of the ACM, the US National Academy of Engineering, the US National Academy of Sciences, and the American Association for the Advancement of Science, and he received the ACM Turing Award for his work on transaction processing.

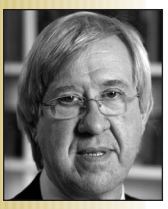
Mark Hansen is an associate professor and the vice chair for graduate studies in the Department of Statistics at the University of California, Los Angeles. Hansen is a co-principle investigator for the Center for Embedded Networked Sensing, a US National Science Foundation Science and Technology Center (Cooperative Agreement CCR-0120778). He received his PhD in statistics from the University of California, Berkeley. Contact him at the Dept. of Statistics, 8125 Mathematical Sciences, UCLA, Los Angeles, CA 90095; cocteau@stat.ucla.edu.



Michael Liebhold is a senior researcher for the Institute for the Future, focusing on the geospatial web foundations for context-aware and ubiquitous computing and on the future of abundant computation. He previously was a visiting researcher at Intel Labs, working on a pattern language based on Semantic Web frameworks for ubiquitous computing. Contact him at the Institute for the Future, 124 University Ave., 2nd Floor, Palo Alto, CA 94301; mliebhold@iftf.org.



Suman Nath is a researcher in Microsoft Research's Networked Embedded Computing Group. His research interests lie in the intersection of sensor networks, databases, and distributed systems. He received his PhD in computer science from Carnegie Mellon University. He's a member of the ACM. Contact him at Microsoft Research, One Microsoft Way, Redmond, WA, 98052; sumann@microsoft.com.



Alexander Szalay is the Alumni Centennial Professor of Astronomy and a professor in the Department of Computer Science at Johns Hopkins University. He's a cosmologist, working on the statistical measures of the spatial distribution of galaxies and galaxy formation. He received his PhD in astrophysics from Eötvös Loránd University. He's a corresponding member of the Hungarian Academy of Sciences and a fellow of the American Academy of Arts and Sciences. Contact him at the Dept. of Physics and Astronomy, Johns Hopkins Univ., 3701 San Martin Dr., Baltimore, MD 21218; szalay@jhu.edu.

Vincent Tao is the director of Microsoft Virtual Earth, responsible for the technology and business development of the Microsoft Virtual Earth program. He's on leave from York University, Canada, where he held the Canada Research Chair and a full professorship in geomatics. He received his PhD in geomatics engineering from the University of Calgary. He's the chair of the International Society of Photogrammetry and Remote Sensing Working Group on Multiple Platform Mapping and Sensor Networks. Contact him at One Microsoft Way, Redmond, WA, 98052; vincent.tao@microsoft.com.

approaches aim to control the released data, either through random perturbations or by hiding identifying attributes, so that individual privacy is not compromised but useful data mining can still be performed. However, these approaches are still in their infancy and will not become mainstream for some time.

Harvesting the sensor-rich world's benefits requires us to address many hard data management challenges, and

needs concerted multidisciplinary effort encompassing databases, machine learning, networking, and distributed systems, among other fields. We have discussed some of the most important challenges, but this list is hardly complete. As we make progress toward building the sensor web, additional issues such as data preservation and archiving loom large. Several challenges we raised already form active research areas in various communities. Useful data management tools are beginning to emerge, but much more work remains to make the sensor web a reality. P

REFERENCES

1. A. Szalay and J. Gray, "Science in an Exponential World," *Nature*, vol. 440, no. 7083, 2006, pp. 413–414.
2. S. Sarawagi, S. Thomas, and R. Agrawal, "Integrating Association Rule Mining with Relational Database Systems: Alternatives and Implications," *Proc. 1998 ACM SIGMOD Int'l Conf. Management of Data*, ACM Press, 1998, pp. 343–354.
3. A. Deshpande and S. Madden, "MauveDB: Supporting Model-Based User Views in Database Systems," *Proc. 2006 ACM SIGMOD Int'l Conf. Management of Data*, ACM Press, 2006, pp. 73–84.
4. S. Grumbach, P. Rigaux, and L. Segoufin, "Manipulating Interpolated Data Is Easier than You Thought," *Proc. 26th Int'l Conf. Very Large Data Bases* (VLDB 00), Morgan Kaufmann, 2000, pp. 156–165.
5. S. Chaudhuri, V. Narasayya, and S. Sarawagi, "Efficient Evaluation of Queries with Mining Predicates," *Proc. 18th Int'l Conf. Data Eng.* (ICDE 02), IEEE CS Press, 2002, pp. 529–542.
6. M.J. Egenhofer, "Toward the Semantic Geospatial Web," *Proc. 10th ACM Int'l Symp. Advances in Geographic Information Systems* (GIS 02), ACM Press, 2002, pp. 1–4.
7. C.V. Tao, "The Smart Sensor Web," *GeoWorld*, Sept. 2003, pp. 28–32.
8. C.V. Tao, S. Liang, and A. Croitoru, "GeoServNet SensorWeb: A Tool for Open Geospatial Sensing Services," *GeoSensor Networks*, A. Stefanidis and S. Nittel, eds., CRC Press, 2004, pp. 267–274.
9. I. Stoica et al., "Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications," *Proc. ACM SIGCOMM 2001*, ACM Press, 2001, pp. 149–160; www.sigcomm.org/sigcomm2001/p12-stoica.pdf.
10. R. Huebsch et al., "The Architecture of PIER: An Internet-Scale Query Processor," *Proc. 2nd Biennial Conf. Innovative Data Systems Research* (CIDR 05), 2005, pp. 28–43; http://www-db.cs.wisc.edu/cidr.
11. A. Deshpande et al., "Cache-and-Query for Wide Area Sensor Databases," *Proc. 2003 ACM SIGMOD Int'l Conf. Management of Data*, ACM Press, 2003, pp. 503–514.
12. M. Cherniack et al., "Scalable Distributed Stream Processing," *Proc. 1st Biennial Conf. Innovative Data Systems Research* (CIDR 03), 2003, pp. 257–268; http://www-db.cs.wisc.edu/cidr.
13. D.J. Abadi et al., "The Design of the Borealis Stream Processing Engine," *Proc. 2nd Biennial Conf. Innovative Data Systems Research* (CIDR 05), 2005; http://www-db.cs.wisc.edu/cidr.
14. S. Chandrasekaran et al., "TelegraphCQ: Continuous Dataflow Processing for an Uncertain World," *Proc. 1st Biennial Conf. Innovative Data Systems Research* (CIDR 03), 2003, pp. 269–280; http://www-db.cs.wisc.edu/cidr.
15. M.J. Franklin et al., "Design Considerations for High Fan-In Systems: The HiFi Approach," *Proc. 2nd Biennial Conf. Innovative Data Systems Research* (CIDR 05), 2005; http://www-db.cs.wisc.edu/cidr.
16. H. Chan and A. Perrig, "Security and Privacy in Sensor Networks," *Computer*, vol. 36, no. 10, 2003, pp. 103–105.
17. R. Agrawal et al., "Hippocratic Databases," *Proc. 28th Int'l Conf. Very Large Data Bases* (VLDB 02), Morgan Kaufmann, 2002, pp. 143–154.
18. J. Kubiatowicz et al., "OceanStore: An Architecture for Global-Scale Persistent Storage," *Proc. 9th Int'l Conf. Architectural Support for Programming Languages and Operating Systems* (ASPLOS IX), ACM Press, 2000, pp. 190–201.
19. R. Agrawal and R. Srikant, "Privacy-Preserving Data Mining," *ACM SIGMOD Record*, vol. 29, no. 2, 2000, pp. 439–450; http://doi.acm.org/10.1145/335191.335438.
20. P. Samarati and L. Sweeney, *Protecting Privacy When Disclosing Information: k-Anonymity and Its Enforcement through Generalization and Suppression*, tech. report SRI-CSL-98-04, Computer Science Laboratory, SRI Int'l, 1998; www.csl.sri.com/papers/sritr-98-04.