Programming Support for Sensor-based Scientific Applications *

Nanyan Jiang and Manish Parashar
The Applied Software Systems Laboratory
Department of Electrical and Computer Engineering
Rutgers University, Piscataway NJ 08855, USA
{*nanyanj, parashar*}@*caip.rutgers.edu*

Abstract

Technical advances are enabling a pervasive computational ecosystem that integrates computing infrastructures with embedded sensors and actuators, and are giving rise to a new paradigm for monitoring, understanding, and managing natural and engineered systems – one that is information/data-driven. This research investigates programming systems for sensor-driven applications. It addresses abstractions and runtime mechanisms for integrating sensor systems with computational models for scientific processes, as well as for in-network data processing, e.g., aggregation, adaptive interpolation and assimilation. The current status of this research, as well as initial results are presented.

1 Introduction

Technical advances are rapidly enabling a revolution in the type and level of instrumentation of natural and engineered systems, and is resulting in a pervasive computation ecosystem that integrates computers, networks, data archives, instruments, observatories, experiments, and embedded sensors and actuators. This in turn is enabling a new paradigm for monitoring, understanding, and managing natural and engineered systems – one that is information/data-driven and that opportunistically combines computations, experiments, observations, and real-time data/information to model, manage, control, adapt, and optimize.

Several application domains, such as waste management [20], volcano monitoring [25], city-wide structural monitoring [12], habitat and environmental monitoring [17], and end-to-end soil monitoring [23], are already experiencing this revolution in instrumentation, and can enable new levels of monitoring, understanding and near real-time control of this systems. However, (1) the data volume and rates, (2) the uncertainty in this data and the need to characterize and manage this uncertainty, and (3) the need to assimilate and transport required data (often from remote sites over low bandwidth wide area networks) in near real-time so that it can be effectively integrated with computational models and analysis systems, present significant challenges. As a result, data in most existing instrumented systems is used in a post-processing manner where data acquisition is a separate offline process.

The overall goal of this research is to develop sensor system middleware and programming support that will enable distributed networks of sensors to function, not only as passive measurement devices, but as intelligent data processing instruments, capable of data quality assurance, statistical synthesis and hypotheses testing as they stream data from the physical environment to the computational world. Specifically, this research develops a programming system to support in-network data processing mechanisms, and enable scientific/engineering applications to discover, query, interact with, and control instrumented physical systems using semantically meaningful abstractions. The programming system enables sensor-driven applications at two levels. First, it provides programming abstractions for integrating sensor systems with the computational models for scientific processes (e.g. biophysical, geophysical processes) and with other application components in an end-to-end experiment. Second, it supports programming models and systems for developing in-network data processing mechanisms. The former should support complex querying of the sensor system, while the latter should enable development of in-network data processing mechanisms such as aggregation, adaptive interpolation and assimilation, both via semantically meaningful abstractions. A key requirement here is being able to specify and enforce dynamic data requirements and quality of data and service constraints, as well as investigate tradeoffs between data quality, resource consumption, and performance.

The rest of the paper is organized as follows. Section 2 identifies the requirements of sensor-driven applications. Section 3 describes the programming system including its overall architecture, the two levels of programming abstractions, and the in-network data es-

*The research presented in this paper is supported in part by National Science Foundation via grants numbers CNS 0723594, IIP 0758566, IIP 0733988, CNS 0305495, CNS 0426354, IIS 0430826 and ANI 0335244, and by Department of Energy via the grant number DE-FG02-06ER54857, and was conducted as part of the NSF Center for Autonomic Computing at Rutgers University.

978-1-4244-1694-3/08/$25.00 ©2008 IEEE

timation mechanisms. Initial results are also presented. Section 4 concludes this paper.

2 Requirements of Sensor-driven Applications

In many current scientific and engineering applications, modeling to predict system behavior is largely done using static historical data. This approach makes it impossible for such models to accurately predict temporal and spatial variations in the real-world. With advances in sensor technology, it now becomes possible to feed in near real time, measured data from diverse and complex, distributed sensor networks, enabling more accurate modeling, prediction and control. This research investigates conceptual as well as systems software issues for enabling the integration of sensor systems with computational applications. Specifically, the research is driven by the management and control of subsurface geosystems, such as managing subsurface contaminants at the Ruby Gulch waste repository [20] and management and optimization of oil reservoirs [11]. Crosscutting requirements of these applications include:

Multi-scale, multi-resolution data access: Spatial and temporal variations in the phenomenon being understood and managed by the application requires data at multiple scales and resolutions at different locations of the monitored field. This requires online (near real-time) spatial and temporal interpolation of the data from sensors before it can be integrated with executing simulations.

Data quality and uncertainty estimation. Scientific investigation present strict requirements on data quality and uncertainty estimation and management. Further, the data assimilation and uncertainty estimation mechanisms should be near real-time and managed within the sensor system to be able to respond adequately and opportunistically to anomalous events.

Predictable temporal response to varying application characteristics. Ensuring robust sense-evaluate-actuate cycles of scientific/engineering processes requires low and predictable information (not just data) generation latencies. These requirements may differ from application to application and for different phases of the same application. For example, applications involving aerodynamic stabilization and neural control may require millisecond level response while geosystem management (such as the applications described above) may require responses in seconds, hours, days or weeks, depending on the nature of the control task.

In addition to the above application requirements, there are a number of challenging system level requirements including scheduling and adaptive runtime management of in-network data processing, load balancing, quality of service management, resource management and computation/communication/energy tradeoffs. Furthermore, integrating sensors system with applications requires sufficiently high-level and easy-to-use programming abstractions and systems.

3 A Programming System for Sensor-based, Dynamic Data-Driven Applications

A conceptual overview of the overall approach is illustrated in Figure 1. The goal of the programming system being developed as part of this research is to provide abstractions and mechanisms to seamlessly access and integrate remote and distributed sensor data into computational models and support scalable in-network data processing. The underlying approach is to virtualize the physical sensor grid to match the representation of the physical domain used by the models, and dynamically discover and access sensor data independent of any change to the sensor network itself. The sensor network periodically estimates data at the grid points specified by the application using available physical data values. The estimation mechanisms are specified by the applications and are implemented within the sensors network in a decentralized and scalable manner.

The abstractions enable applications to query the sensor system for data/information using flexible content descriptors that are semantically meaningful.

3.1 Related Work

There has been a significant body of research focused on programming support for sensor networks, for both, interfacing with them and developing applications that run on them. Recent efforts are broadly classified below. Several systems [15, 14, 24, 26] provide abstractions for specifying the local behaviors of sensors. EnviroSuite [15], state programming [14] and CLB [8] are based on emergence, where the programmer specifies local sensor behaviors and global behaviors emerge from local behaviors and interactions. Abstract region [24] and Hood [26] provide neighborhood-based programming models for sensor networks, which are similar in concept to the approach presented in this paper.

In the *macroprogramming* approach, global behaviors are specified and the programming system generates the local behaviors and interactions necessary, e.g., Kairos [7] and ATaG [2]. *Database-oriented* approaches provide abstractions that view the sensor network as a virtual database system and provide SQL-like interfaces for querying the networks, e.g., Cougar [27] and TinyDB [16]. The related *data streaming* approach, supported by TelegraphCQ [4], views sensor data as information streams, and applications monitor and react to them as it passes through the network.

3.2 The GridMap/iZone Approach

The programming systems discussed above have to be extended to support end-to-end sensor-driven ap-

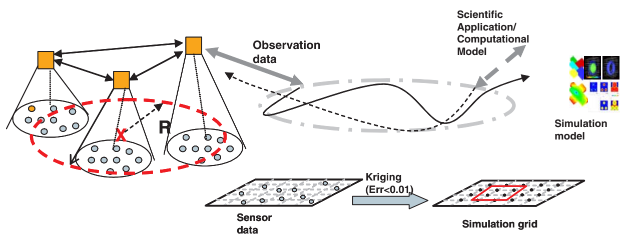

Figure 1. Overview of the programming system for sensor-driven applications.

plications and the interactions between computational models and the sensor system. They must address the mismatch between the locations of the sensors and the data requirements of the computational models, the dynamic nature of application requirements and of the sensor system, data uncertainty and application constraints on quality of data and service. Furthermore, the system software should be able to actively support intelligent processing, such as adaptive interpolations, assimilations, which are needed by the models. Ideally, a scientist should only have to specify application data requirements using high-level abstractions, and the system should transform these requirements into appropriate operations, interactions and coordination within the sensor system to transform the sensed data to match these requirements.

While systems such as Kiaros [7] and TelegraphCQ [4] address some aspects of these requirements, the key difference is in the type of querying and processing supported. A key aspect of the GridMap approach presented in this paper, is the indexing of data in the sensor system in a domain and locality aware manner using meaningful content descriptors. The descriptors are derived from the application domain and are used to define a semantically specified information space [21], which is the basis for all interactions with and within the sensor system. Note that many such information spaces can co-exist corresponding to different applications. This enables applications to query the sensor system for data/information using flexible content descriptors that are meaningful. For example, an oil reservoir simulation looking for well data in a particular region may specify this query using the coordinates that define the region, the type of well (production or injection) and details of the data required (i.e., type, resolution, etc.). Note that the content descriptors may also include ranges and wildcards. This abstraction can also specify data processing operations such as aggregations and interpolations and these operations can also be localized to specific regions in the sensor field. Our preliminary work has explored the feasibility and effectiveness of this approach [9].

There also exist recent research efforts that are similar to the presented approach in their use of virtual sensors [3, 10, 18, 19, 1] to support sensor-based applications. However, the underlying concepts and implementation of the system described in this paper is quite different from these approaches in that it uses virtualization to address the mismatch between the instrumentation of the physical domain and its discretization in the computational model, rather than to create, for example, a virtual sensor for a derived data type. Other related efforts include spatial interpolation and aggregation [5, 6, 22], and redundant sensor sampling [13]. The focus of this work is different in that it addresses programming abstractions to facilitate implementations of in-network data estimation algorithms (e.g. various interpolation algorithms), as well as runtime mechanisms to investigate tradeoffs between dynamic coordination costs and data quality.

3.3 System Architecture

A schematic overview of the overall programming system architecture is presented in Figure 2. It consists of a two-level programming abstraction, the end-to-end *GridMap* programming abstraction that enables computational applications to access and integrate sensor data into their models, and the in-network *iZone* programming abstraction to enable the development of scalable in-network data processing mechanisms. The underlying middleware provides an decentralized data processing engine, which supports efficient data dissemination, aggregation and processing in dynamic, resource-constraint heterogeneous sensor networks.

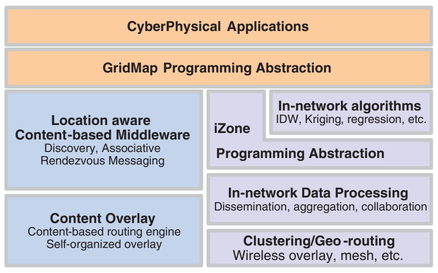
CyberPhysical Applications
GridMap Programming Abstraction
Location aware Content-based Middleware
Discovery, Associative Rendezvous Messaging
iZone
In-network algorithms
IDW, Kriging, regression, etc.
Programming Abstraction
Content Overlay
Content-based routing engine
Self-organized overlay
In-network Data Processing
Dissemination, aggregation, collaboration
Clustering/Geo-routing
Wireless overlay, mesh, etc.

Figure 2. A schematic overview of the programming system.

3.4 GridMap & iZone Programming Abstractions

Scientific applications often require measurements at pre-defined grid points, which are often different from the locations of the raw data provided directly by the sensor network. As a result, the sensor-driven scientific and engineering applications require a virtual layer, where the logical representation of the state of the environment provided to the applications may be different from the physical representation of raw measurements from sensor network. The abstractions described in this section enable applications to specify such a virtual layer and the models (e.g. regression models, interpolation functions, etc.) that should be used to estimate data on the virtual layer from sensor readings, as well to develop in-network implementations of the data estimation mechanisms.

The *GridMap* abstraction consists of two operators. The first operator allows the application to construct a virtual grid, corresponding to the computational grid used by the application, on the instrumented domain. Once this virtual grid has been overlayed on the sensor system, the application can use the second operator to query data corresponding to a region of interest on this virtual grid. The interface of this second operator includes a specification of the method that should be used to estimate data at a grid point in the region of interest using physical data from sensors that are in the neighborhood of the point. The operator also includes parameters such as the size of neighborhood that should be used in the estimation, and what are the constraints on the accuracy and cost of the estimation.

The *iZone* abstraction in turn, enables the implementation of the estimations functions. The *iZone* itself is a representation of the neighborhood that is used to compute a grid point, and may be specified using a range of coordinates, a function, etc. The *iZone* abstraction also provides operators for obtaining sensors data corresponding to the region of interest as well as for defining in-network processing operators to compute a desired grid point from sensor values from this region.

The *GridMap* and *iZone* abstractions thus abstract away the details of the underlying measurement infrastructure and hide the irregularities in the data by using a virtualization of the sensor field and estimation methods, to present a consistent representation, over time and space, to applications using the data. Once an *iZone* is defined, computing the data value at a grid point consists of (1) identifying a coordinator, which could be the sensor node that is closest to the grid point and has the required capabilities, (2) discovering the sensors in the *iZone* that will be used in the estimation, (3) planning the in-network estimation strategy based on desired cost/accuracy/energy tradeoffs, and (4) executing the estimation and returned the computed data value at the grid point.

The goal of the in-network data processing mechanisms is to minimize the number of sensors queried and the overall coordination costs, while guaranteeing that the estimation quality and error thresholds specified by the application are satisfied. The current prototype implements two estimation approaches. In the baseline centralized approach, the coordinating sensor collects raw measurements from selected *iZone* sensors and does the estimation itself. While this approach is resilient to the dynamics and uncertainties of the sensor system, it is also expensive in terms of communication costs and energy consumed. In the distributed approach, parameters corresponding to the estimation model are first computed at the coordinating sensor and are distributed to the selected *iZone* sensors. Decentralized energy-efficient aggregation schemes are then used to estimate the data. This approach is more energy efficient when the sensor system is relatively static.

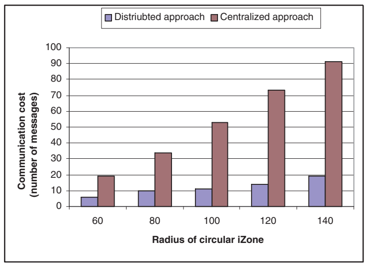

Figure 3. Cost of in-network estimation: centralized vs. distributed approach.

The two in-network data estimation mechanisms were evaluated using a simulator. The evaluation experiments were driven by a real-world sensor dataset obtained from an instrumented oil field with 2000 sensors, and consisted of pressure measurements sampled 96 time per day. The experiments simulated a sensor network with 500 to 2000 sensors, and measured the cost of in-network interpolation of pressures using the centralized and distributed approaches. Each experiment was repeated 1000 times and the average cost of each approach, in terms of the number of messages, was measured. The performance of the two approaches are plotted in Figure 3. The horizontal axis in this plot is the radius of the circular *iZone* used and the vertical axis is the cost, i.e., the number of messages required. The plot clearly shows the advantage of using a distributed approach. We are currently conducting additional experiments, and are also working on optimizing the distributed approach by exploring temporal and spatial correlation in the sensor measurements and using these to tradeoff complex coordination behaviors for energy savings.

4 Summary

The research presented in this paper investigates programming systems for sensor-driven applications. The programming system includes semantically meaningful abstractions and runtime mechanisms for integrating sensor systems with computational models for scientific processes, as well as for in-network data processing such as aggregation, adaptive interpolation and assimilation. The paper described the overall architecture of the programming system as well as the design of its key components. The current status and an initial evaluation was also presented.

References

[1] K. Aberer, M. Hauswirth, and A. Saleh. Zero-programming sensor network deployment. *Next Generation Service Platforms for Future Mobile Systems (SPMS)*, 2007.

[2] A. Bakshi, V. K. Prasanna, J. Reich, and D. Larner. The abstract task graph: A methodology for architecture-independent programming of networked sensor systems. *Workshop on End-to-End, Sense-and-Respond Systems, Applications, and Services, EESR 05*, 2005.

[3] M. Brown, S. Gilbert, N. Lynch, C. Newport, T. Nolte, and M. Spindel. The virtual node layer: A programming abstraction for wireless sensor networks. *ACM SIGBED Review*, 4(3):7–12, 2007.

[4] S. Chandrasekaran, O. Cooper, A. Deshpande, M. J. Fanklin, J. M. Hellerstein, W. Hong, S. Krishnamurthy, S. Madden, V. Raman, F. Reiss, and M. Shah. TelegraphCQ: Continuous dataflow processing for an uncertain world. *In Proceedings of CIDR Conference*, 2003.

[5] S. Ganeriwal, C.-C. Han, and M. Srivastava. Poster abstract: Spatial average of a continuous physical process in sensor networks. *Proceedings of the First ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2003.

[6] D. Ganesan, S. Ratnasamy, H. Wang, and D. Estrin. Coping with irregular spatio-temporal sampling in sensor networks. *2nd Workshop on Hot Topics in Networks (HotNets-II)*, 2003.

[7] R. Gummadi, O. Gnawali, and R. Govindan. Macro-programmingwireless sensor networks using *Kairos*. *Proceedings of the International Conference on Distributed Computing in Sensor Systems (DCOSS)*, pages 126–140, June 2005.

[8] N. Jiang, C. Schmidt, V. Matossian, and M. Parashar. Enabling applications in sensor-based pervasive environments. In *Proceedings of the 1st Workshop on Broadband Advanced Sensor Networks (BaseNets), San Jose, CA, USA*, October 2004.

[9] N. Jiang, C. Schmidt, and M. Parashar. A decentralized content-based aggregation service for pervasive environments. In *International Conference of Pervasive Services (ICPS)*, pages 203 – 212, 2006.

[10] S. Kabadayi, A. Pridgen, and C. Julien. Virtual sensors: abstracting data from physical sensors. *Proceedings of the 2006 International Symposium on World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pages 587 – 592, 2006.

[11] H. Klie, W. Bangerth, X. Gai, M. F. Wheeler, P. Stoffa, M. Sen, M. Parashar, U. Catalyurek, J. Saltz, and T. Kurc. Models, methods and middleware for grid-enable multiphysics oil reservoir management. *Engineering with Computers*, 22:349–370, 2006.

[12] V. A. Kottapalli, A. S. Kiremidjiana, J. P. Lyncha, E. Carryerb, T. W. Kennyb, K. H. Lawa, and Y. Lei. Two-tiered wireless sensor network architecture for structural health monitoring. *SPIEs 10th Annual International Symposium on Smart Structures and Materials*, 2003.

[13] P. Liaskovits1 and C. Schurgers. Leveraging redundancy in sampling-interpolation applications for sensor networks. *Proceedings of the third International Conference on Distributed Computing on Sensor Systems (DCOSS)*, pages 324–337, 2007.

[14] J. Liu, M. Chu, J. Liu, J. Reich, and F. Zhao. State-centric programming for sensor-actuator network systems. *IEEE Pervasive Computing*, 2(4):50–62, 2003.

[15] L. Luo, T. F. Abdelzaher, T. He, and J. A. Stankovic. EnviroSuite: An environmentally immersive programming framework for sensor networks. *ACM Transactions on Computational Logic*, 5(3):543 – 576, 2005.

[16] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. TinyDB: an acquisitional query processing system for sensor networks. *ACM Transactions on Database System*, 30(1):122–173, 2005.

[17] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson. Wireless sensor networks for habitat monitoring. *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, pages 88–97, 2002.

[18] L. Mottola and G. P. Picco. Logical neighborhoods: A programming abstraction for wireless sensor networks. *Proceedings of the second International Conference on Distributed Computing on Sensor Systems (DCOSS)*, pages 150–168, 2006.

[19] P. M. Paolo Corsini and A. Vecchio. Virtus: a configurable layer for post-deployment adaptation of sensor networks. *International Conference on Wireless and Mobile Communications, ICWMC '06*, 2006.

[20] M. Parashar, V. Matossian, H. Klie, S. G. Thomas, M. F. Wheeler, T. Kurc, J. Saltz, and R. Versteeg. Towards dynamic data-driven management of the ruby golch waste repository. *ICCS*, 2006.

[21] C. Schmidt and M. Parashar. Flexible information discovery in decentralized distributed systems. In *Proceedings of the 12th High Performance Distributed Computing (HPDC)*, pages 226–235. IEEE Press, June 2003.

[22] M. Sharifzadeh and C. Shahabi. Utilizing voronoi cells of location data streams for accurate computation of aggregate functions in sensor networks. *GeoInformatica*, 10(1):9–36, March 2006.

[23] K. Szlavecz, A. Terzis, R. Musaloiu-E., J. Cogan, S. Small, S. Ozer, R. Burns, J. Gray, and A. S. Szalay. Life under your feet: An end-to-end soil ecology sensor network, database, web server, and analysis service. *MSR-TR-2006-90*, 2006.

[24] M. Welsh and G. Mainland. Programming sensor networks using abstract regions. *in Proceedings of the First USENIX/ACM Symposium on Networked Systems Design and Implementation (NSDI '04)*, 2004.

[25] G. Werner-Allen, K. Lorincz, M. Ruiz, O. Marcillo, J. Johnson, J. Lees, and M. Welsh. Monitoring volcanic eruptions with a wireless sensor network. *Second European Workshop on Wireless Sensor Networks*, 2005.

[26] K. Whitehouse, C. Sharp, E. Brewer, and D. Culler. Hood: A neighborhood abstraction for sensor networks. *Proceedings of the International Conference on Mobile Systems, Applications, and Services (MOBISYS'04)*, pages 99–110, June 2004.

[27] Y. Yao and J. E. Gehrke. The cougar approach to in-network query processing in sensor networks. *Sigmod Record*, 31(3), September 2002.