H.B. Lim, Y.M. Teo, P. Mukherjee1, V.T Lam, W.F. Wong and S. See, **Sensor Grid: Integration of Wireless Sensor Networks and the Grid,** Proceedings of 30th IEEE Conference on Local Computer Networks, IEEE Computer Society Press, pp. 91-98, Sydney, Australia, November 2005.

Sensor Grid: Integration of Wireless Sensor Networks and the Grid *

Hock Beng Lim[1], Yong Meng Teo[1,2], Protik Mukherjee[1], Vinh The Lam[1], Weng Fai Wong[1,2], Simon See[3]

[1] Singapore-MIT Alliance, National University of Singapore.
[2] Department of Computer Science, National University of Singapore.
[3] Sun Microsystems, Inc.
E-mail: [limhb, teoym]@comp.nus.edu.sg

Abstract

Wireless sensor networks have emerged as an exciting technology for a wide range of important applications that acquire and process information from the physical world. Grid computing has evolved as a standards-based approach for coordinated resource sharing. Sensor grids combine these two promising technologies by extending the grid computing paradigm to the sharing of sensor resources in wireless sensor networks.

There are several issues and challenges in the design of sensor grids. In this paper, we propose a sensor grid architecture, called the Scalable Proxy-based aRchItecture for seNsor Grid (SPRING), to address these design issues. We also developed a sensor grid testbed to study the design issues of sensor grids and to improve our sensor grid architecture design.

Keywords: *Wireless Sensor Networks, Grid Computing, Sensor Grid*

1. Introduction

With the convergence of technologies such as MEMS sensor devices, wireless networking, and low-power embedded processing, wireless sensor networks [1, 2] have emerged as an exciting new computing platform with the potential to seamlessly couple the digital world and the physical environment. At the heart of wireless sensor networks is a new class of sensor nodes which contain small, low-cost, low-power and self-contained sensor devices or instruments with sensing, data processing, and wireless communication capabilities.

Wireless sensor networks are increasingly being deployed in many important applications requiring the interaction between users and the physical world. They allow the physical environment to be measured at high resolutions, and greatly increase the quantity and quality of real-world data and information for applications. Important applications of wireless sensor networks include environmental and habitat monitoring, healthcare monitoring of patients, weather monitoring and forecasting, military and homeland security surveillance, tracking of goods and manufacturing processes, safety monitoring of physical structures and construction sites, smart homes and offices, and many other uses that we do not yet imagine.

Sensor devices in a wireless sensor network are resource-constrained since they have limited sensing capability, processing power, and communication bandwidth. However, with a large number of such devices being deployed and aggregated over a wide area, a wireless sensor network has substantial data acquisition and processing capability. Thus, wireless sensor networks are important distributed computing resources that can be shared by different users and applications.

In recent years, grid computing has evolved as a standards-based approach for the coordinated sharing of distributed and heterogeneous resources to solve large-scale problems in dynamic virtual organizations [3]. Much of the existing developments in grid computing have focused on *compute grids* and *data grids*. A compute grid provides distributed computational resources to meet the computational requirements of applications, while a data grid provides seamless access to large amounts of distributed data and storage resources.

The emerging domain of *sensor grids* extends the grid computing paradigm to the sharing of sensor resources in

* This work is sponsored in part by Sun Microsystems, Inc.

wireless sensor networks. A sensor grid is the result of the integration of wireless sensor networks with the conventional wired grid fabric. Note that the term sensor grid has been used in the literature to describe a sensor network with a grid-like deployment topology, but such a definition does not apply to this work.

There are several rationale for sensor grids. First, the vast amount of data collected by the sensors can be processed, analyzed, and stored using the computational and data storage resources of the grid. Second, the sensors can be efficiently shared by different users and applications under flexible usage scenarios. Each user can access a subset of the sensors during a particular time period to run a specific application, and to collect the desired type of sensor data. Third, as sensor devices with embedded processors become more computationally powerful, it is more efficient to offload specialized tasks such as image and signal processing to the sensor devices. Finally, a sensor grid provides seamless access to a wide variety of resources in a pervasive manner. Advanced techniques in artificial intelligence, data fusion, data mining, and distributed database processing can be applied to make sense of the sensor data and generate new knowledge of the environment. The results can in turn be used to optimize the operation of the sensors, or influence the operation of actuators to change the environment. Thus, sensor grids are well suited for adaptive and pervasive computing applications.

Sensor grid is a relatively new area of research. Thus, the design of sensor grids is not well understood yet, unlike that of compute and data grids. Wireless sensor networks are usually based on proprietary designs and protocols, and so it is challenging to integrate them with the standard grid architecture and protocols. In this paper, we discuss the issues and challenges in the integration of wireless sensor networks with the grid.

We propose a sensor grid architecture, called the Scalable Proxy-based aRchItecture for seNsor Grid (SPRING), to address these design issues. The key idea is to use proxy systems as interfaces between the wireless sensor networks and the grid fabric. To study the design issues of sensor grids and improve our SPRING design, we have developed a sensor grid testbed. This testbed consists of a set of sensor nodes (or ”motes”), a 54-node Sun cluster based on AMD Opteron processors, and several Linux-based proxy and user systems.

The rest of this paper is organized as follows. In Section 2, we discuss the related work and our contributions. Section 3 presents the important issues and challenges in the design of sensor grids. Then, we discuss the SPRING framework in Section 4. In Section 5, we describe the design and implementation of the sensor grid testbed. Finally, we conclude this paper in Section 6.

2. Related Work

There has been much effort in the development of software platforms for grid computing. The Globus Toolkit [4] is becoming the de facto standard for grid middleware. It provides tools and libraries for communications, resource management, data management, security, and information services. The Global Grid Forum developed the specifications for the Open Grid Services Architecture (OGSA) [5] based on the concept of *grid services*. The grid services architecture enables resources to be dynamically discovered and shared. The Open Grid Services Infrastructure (OGSI) [6] is the first specification to implement the OGSA framework, and the Globus Toolkit 3 is the first implementation of the OGSI specifications. Recently, the Web Services Resource Framework (WSRF) [7] was developed to address some limitations of OGSI. The Globus Toolkit 4 implements the WSRF specifications. Our sensor grid architecture leverages these existing and evolving grid middleware standards and tools.

Software tools for the management of wireless sensor networks are necessary for the efficient and effective utilization of wireless sensor networks. MoteLab [8] is a web-based sensor network testbed developed at Harvard University. Its hardware setup consists of a set of permanently-powered and Ethernet-connected MICAz motes from Crossbow Technology, Inc (http://www.xbow.com). MoteLab provides a web-based interface that makes it easier for users to program the motes, create sensor jobs, reserve time slots to run sensor jobs on the motes, collect the sensor data, and perform simple administrative functions. Other sensor network management software include EmStar [9] and Kansei [10]. However, such systems can only manage a standalone wireless sensor network testbed, and they are not integrated with the grid fabric.

Recently, research efforts are beginning to study the integration of wireless sensor networks and grid computing. Researchers in the UK are studying how sensors can be integrated into e-Science grid computing applications. The Discovery Net project (http://www.discovery-on-the.net) is building a grid-based framework for developing and deploying knowledge discovery services to analyze data collected from distributed high throughput sensors. The applications include life sciences, environmental monitoring, and geo-hazard modelling. However, these application-driven projects tend to use sensor grid architectures that are custom built for specific applications. Although these architectures are efficient and can deliver good performance for the targeted applications, they are not flexible and not scalable.

There are also efforts to define the middleware architecture for sensor devices to facilitate the integration with the grid. The Common Instrument Middleware Architecture

(CIMA) project [11] aims to ”grid enable” instruments and sensors as real-time data sources to facilitate their integration with the grid. The CIMA middleware is based on current grid standards such as OGSA. Such a middleware architecture uses a standard instrument representation format and software stack. A problem with this approach is that the middleware architecture might be too complex to be implemented on simple sensor devices with low computational and processing capability.

Our SPRING framework integrates wireless sensor networks with the grid. It is a flexible architecture that is not constrained by the characteristics and requirements of specific target applications. By using proxy systems as interfaces between the wireless sensor networks and the grid fabric, the SPRING architecture can support a wide range of sensor devices, even the less computationally powerful ones. Furthermore, the SPRING architecture is scalable, and it can integrate multiple heterogeneous wireless sensor networks with the grid.

3. Design Issues and Challenges

In this section, we discuss the important issues and challenges in the design of sensor grids. Most of these design issues and challenges arise due to the inherent limitations of sensor devices, such as limited processor performance, small storage capacity, limited battery power, and unreliable low-bandwidth wireless communication.

3.1. Grid APIs for Sensors

A natural approach to integrate sensor nodes into the grid is to adopt the grid standards and APIs. The OGSA is based on established web services standards and technologies like XML, SOAP, and WSDL. If sensor data were available in the OGSA framework, it would be easier to exchange and process the data on the grid. However, since sensor nodes have limited computational and processing capability, it may not be feasible for sensor data to be encoded in XML format within SOAP envelopes, and then transported using Internet protocols to applications [12]. Grid services are also too complex to be implemented directly on most simple sensor nodes.

3.2. Network Connectivity and Protocols

In conventional grids, the network connections are usually fast and reasonably reliable. Many grid deployments leverage the Internet infrastructure. On the other hand, the sensor nodes in sensor grids are connected via wireless ad hoc networks which are low-bandwidth, high-latency, and unreliable. The network connectivity of sensor nodes is dynamic in nature, and it might be intermittent and susceptible to faults due to noise and signal degradation caused by environmental factors. The sensor grid has to gracefully handle unexpected network disconnections or prolonged periods of disconnection.

Grid networking protocols are based on standard Internet protocols like TCP/IP, HTTP, FTP, etc. On the other hand, wireless sensor networks are often based on proprietary protocols, especially for the MAC protocol and routing protocol [1]. It is not practical for sensor nodes to have multiple network interface capabilities. Thus, efficient techniques to interface sensor network protocols with grid networking protocols are necessary.

3.3. Scalability

Scalability is the ability to add sensor resources to a sensor grid to increase the capacity of sensor data collection, without substantial changes to its software architecture. The sensor grid architecture should allow multiple wireless sensor networks, possibly owned by different virtual organizations, to be easily integrated with compute and data grid resources. This would enable an application to access sensor resources across increasing number of heterogeneous wireless sensor networks.

3.4. Power Management

Power management is a major concern as sensor nodes do not have fixed power sources and rely on limited battery power. Sensor applications executing on these devices have to make tradeoffs between sensor operation and conserving battery life. The sensor nodes should provide adaptive power management facilities that can be accessed by the applications. From the sensor grid perspective, the availability of sensor nodes is not only dependent on their load, but also on their power consumption. Thus, the sensor grid’s resource management component has to account for power consumption.

3.5. Scheduling

In wireless sensor networks, scheduling of sensor nodes is often performed to facilitate power management and sensor resource management. Researchers have developed algorithms to schedule the radio communication of active sensor nodes, and to turn off the radio links of idle nodes to conserve power. Similarly, for applications like target tracking, sensor management algorithms selectively turn off sensor nodes that are located far away from the target, while maximizing the coverage area of the sensors.

Sensor grids are data-centric in nature. A scheduler is needed for the efficient scheduling of applications to use the sensor resources for collecting sensor data. A sensor

grid has a set of sensor nodes spread across multiple wireless sensor networks. These sensor nodes may provide sensors that collect different types of data such as temperature, light, sound, humidity, vibration, etc. Also, the sensor nodes may be shared by multiple applications with differing requirements.

The design of a sensor grid scheduler is influenced by some important differences between sensor jobs and computational jobs. Unlike computational jobs, sensor jobs are not multitasking in nature. A sensor node can execute only one sensor job at a time, and it cannot execute multiple sensor jobs via multitasking. While computational jobs automatically terminate upon completion, the durations of sensor jobs have to be explicitly specified. Sensor jobs are also more likely to require specific time slots for execution compared to general computational jobs.

3.6. Security

Organizations are reluctant to share their resources on a grid unless there is guarantee for security. Grid security is an active research area, in particular the security of grid services [13]. Several grid security standards and technologies have been proposed, such as the Grid Security Infrastructure (GSI) of the Globus Toolkit, WS-Security [14], the Shibboleth system (http://shibboleth.internet2.edu), the Security Assertion Markup Language (SAML), and the Extensible Access Control Markup Language (XACML).

Wireless sensor networks are prone to security problems such as the compromising and tampering of sensor nodes, eavesdropping of sensor data and communication, and denial of service attacks. Techniques to address these problems include sensor node authentication, efficient encryption of sensor data and communication, secure MAC and routing protocol, etc.

For sensor grids, it is necessary to ensure that the grid security techniques and the wireless sensor network security techniques are integrated seamlessly and efficiently.

3.7. Availability

Due to the power issue and the unpredictable wireless network characteristics, it is possible that applications running on the sensor nodes might fail. Thus, techniques to improve the availability of sensor nodes are necessary.

Sensor grids should support job and service migration, so that a job can be migrated from a sensor node that is running out of power or has failing hardware to another node. If sufficient resources are available, services can be replicated so that the loss of a node will not result in service disruption. Finally, if unexpected interruptions occur, the system should be able to recover and restart the interrupted jobs.

3.8. Quality of Service

Quality of Service (QoS) is a key issue that determines whether a sensor grid can provide sensor resources on-demand efficiently. Enforcing QoS in sensor grids is made complicated by the unpredictable wireless network characteristics and sensor power consumption.

The specification of the QoS requirements of sensor applications should be described in a high-level manner. A good mechanism is needed to map the high-level requirements into low-level QoS parameters. These parameters specify the amount of resources to be allocated, such as amount of sensors, memory, and network bandwidth. Similarly, service descriptions are necessary to express what a sensor service does, how to access it, and the QoS parameters of the service.

A service request might require several sensor resources. Thus, it might be necessary to make reservations of these resources to achieve the required QoS. Resource reservation is closely tied to the scheduling of sensor resources. Due to the highly dynamic sensor grid environment, any attempt at QoS provisioning should be adaptive in nature. It is necessary to consider the changes in resource availability, network topology, and network bandwidth and latency, so that the sensor grid can provide the best possible QoS to the application. Finally, mechanisms to enforce QoS in wireless sensor networks and grids have been developed separately. For sensor grids, the QoS should be enforced in a coordinated manner by integrating the wireless sensor network and the grid QoS mechanisms.

4. Sensor Grid Architecture and Design

4.1. Sensor Grid Organization

Figure 1 illustrates the organization of a sensor grid in terms of its resource components. A sensor grid consists of wireless sensor networks (WSNs) and conventional grid resources like computers, servers, and disk arrays for the processing and storage of sensor data.

The resources in the sensor grid are shared by several Virtual Organizations (VOs). In fact, certain resources might belong to more than one VO. Users from various VOs may access the resources in the sensor grid, even if the resources are not owned by their VO.

4.2. The SPRING Framework

We propose a proxy-based approach for our sensor grid architecture. With this approach, sensor devices can be made available on the grid like conventional grid services although they are resource-constrained. Also, the

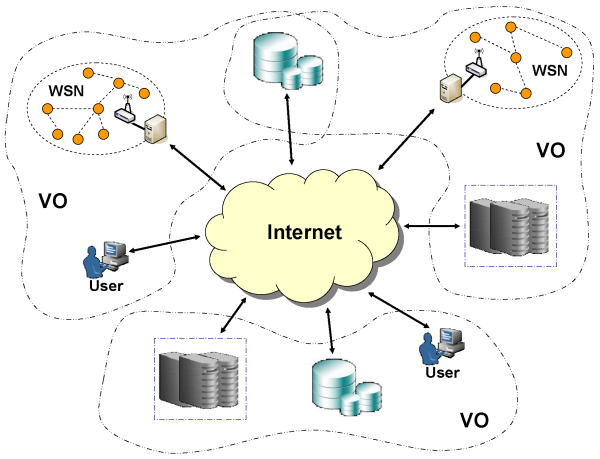

Figure 1. Organization of a sensor grid

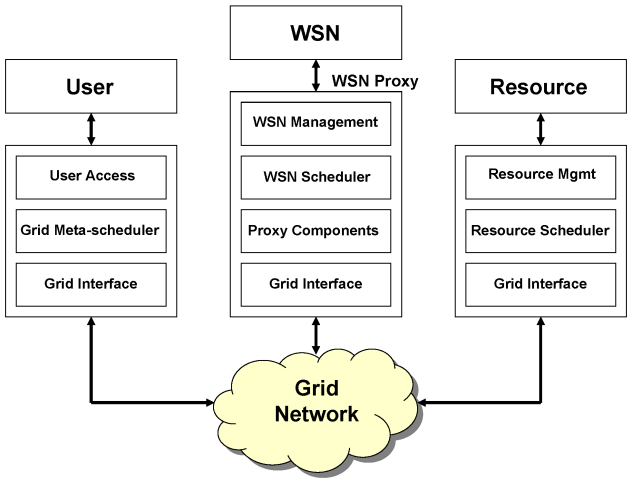

Figure 2. The SPRING framework

proxy can support a wide variety of wireless sensor network implementations, and thus providing interoperability. The SPRING framework is shown in Figure 2.

4.2.1. The WSN Proxy In the SPRING framework, the WSN Proxy acts as the interface between a wireless sensor network and the grid. The proxy serves several important functions, and addresses the design issues of sensor grids that we have discussed. First, the proxy exposes the sensor resources as grid services that can be discovered and accessed by any sensor grid application. It also translates the sensor data from its native format to a suitable OGSA format such as XML.

Second, the proxy coordinates the network connectivity between the wireless sensor network and the grid network. It provides the interface between the sensor network protocols and the Internet protocols. By using techniques like buffering, caching, and link management, the proxy can mitigate the effects of unexpected sensor network disconnection or long periods of disconnection.

Third, the scalability of the sensor grid is enhanced by the use of the WSN Proxy. New wireless sensor networks can be integrated with an existing sensor grid by adding proxy systems. From the grid standpoint, the proxy exposes sensor resources that are accessible in a similar manner as other compute or data resources. Finally, the proxy provides various services such as power management, scheduling, security, availability, and QoS for the underlying wireless sensor network.

4.2.2. SPRING Features SPRING is based on a layered-architecture approach. The layers represent the main software components that are used to build a sensor grid. Each layer defines services that are accessible via Application Programming Interfaces (APIs) for the application or other layers. The *Grid Interface* layer supports a standard grid middleware, such as the Globus Toolkit, that enables different types of resources to communicate over the grid network.

On the user side, the *User Access* layer provides an interface such as a grid portal or a workflow management tool that enables users to submit applications for execution on the sensor grid. The application might consists of sensor jobs to be executed on the wireless sensor network to collect sensor data, and also computational jobs to process the sensor data. A *Grid Meta-scheduler*, such as the Community Scheduler Framework (CSF) [15], is used to schedule and route the jobs according to their required resources.

On the wireless sensor network side, the *WSN Management* layer provides an abstraction of the specific APIs and protocols to access and manage the underlying heterogeneous sensor resources. It manages the configuration of the sensor nodes and provides status information of the sensor nodes. It also accepts sensor job requests from the grid and invokes the specific commands to execute the jobs on the sensor nodes.

The *WSN Scheduler* is the local resource scheduler for the wireless sensor network. It implements the low-level scheduling algorithms for sensor power management and resource management mentioned in Section 3.5. Furthermore, it controls the scheduling of sensor job requests from the users. The parameters of a sensor job include the amount of sensor resources required, desired start time, duration, priority, etc. The scheduler considers these job parameters, checks whether the required sensor resources are available, and reserves the resources for the job. It also works in conjunction with other Proxy Components to provide services for availability and QoS.

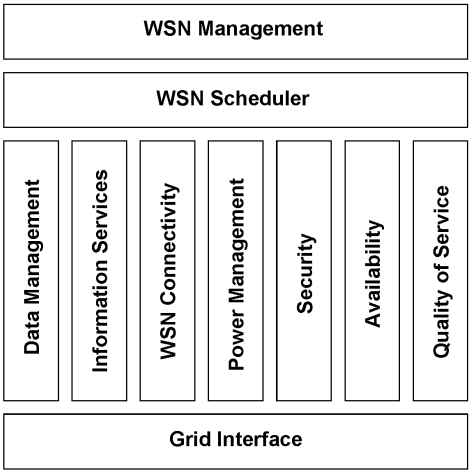

Figure 3. Proxy software architecture

The *Proxy Components* work closely with the WSN Management layer to provide important services for sensor data management, information services, network connectivity, power management, security, availability, and QoS. We will elaborate on these components in Section 4.2.3.

On the resource side, the *Resource Management* layer provides the APIs to access and manage the computational and storage resources for the execution of grid jobs. The *Resource Scheduler* performs the scheduling of the grid jobs based on local usage policies.

4.2.3. Proxy Software Architecture The software architecture of the WSN Proxy is shown in Figure 3. We have already explained the *Grid Interface*, *WSN Management*, and *WSN Scheduler* layers in Section 4.2.2. Now we will discuss the Proxy Components and the important services they provide for a sensor grid.

The *Data Management* component handles the conversion of sensor data from its native format to a grid-friendly format like XML or other formats desired by the user. This component also performs data fusion and other optimizations to improve the quality of data collected from multiple sensors. It supports several methods for the transfer of the sensor data to the user application, such as using GridFTP [4] or by streaming the data.

The *Information Services* component manages the discovery and monitoring of sensor resources. This component advertises the available sensor resources as grid services via mechanisms such as the Indexing Service (IS) of OGSA. Users can query the availability and status of sensor resources via mechanisms such as the Monitoring and Discovery Service (MDS) of OGSA. The static and dynamic information on sensor resources are directly relevant to the WSN Scheduler.

The *WSN Connectivity* component provides services to interface the sensor network protocols with the grid networking protocols. It buffers the transmission of sensor data, caches the routing information of sensor nodes, and manages the ad hoc sensor network links.

The *Power Management* component keeps track of the power consumption of the sensor nodes. Together with the WSN Scheduler, it performs actions to conserve power for the sensor nodes. The *Security* component implements OGSA-compliant grid security technologies. For example, it can use technologies such as the Generic Security Service (GSS) API to perform authentication between the proxy and the sensor nodes.

The *Availability* component provides services to improve the availability of the wireless sensor network. It monitors the sensor nodes to find those with possible failing hardware or weak power level, and migrate the jobs in such nodes to the reliable nodes. To do so, it works closely with the WSN Scheduler. The Availability component can also provide fault tolerance features such as the replication of services and the recovery of interrupted jobs.

Finally, the *Quality of Service (QoS)* component supports the provisioning of QoS in the sensor grid. In conjunction with the WSN Scheduler, the QoS component performs the reservation and allocation of sensor resources based on QoS requirements of sensor jobs. It works with the proxy's WSN Connectivity component to adapt to the varying network conditions in order to provide the desired QoS.

5. Sensor Grid Testbed

We have developed a prototype sensor grid testbed under a joint project between the Singapore-MIT Alliance and Sun Microsystems, Inc. The testbed enables us to study the design issues of sensor grids using real hardware. This will help us to improve our SPRING framework.

5.1. Hardware setup

The wireless sensor network in our testbed consists of 12 Crossbow MICA2 and 8 MICAz motes. The MICA2 uses a 7.3MHz Atmel ATmega128L microcontroller, 128KB of flash memory for code, 4KB of EEPROM for data, and a Chipcon CC1000 radio operating at 433MHz and 38.4Kbps data rate. The MICAz is an upgraded version of the MICA2, with the IEEE 802.15.4 compliant Chipcon CC2420 radio operating at 2.4 GHz and 250Kbps data rate.

We use 5 MIB600 Ethernet interface boards for hosting the "base station" motes. To collect sensor data, we use 12 MTS310CA sensor boards that are plugged onto the motes. Each sensor board contains a variety of sensors. The MICA2 and MICAz both run the TinyOS [16], a small open-source operating system designed for embedded wireless sensor networks. The sensor applications are developed

using nesC [17], an extension of the C programming language.

The testbed uses a Sun cluster with 4 server nodes and 50 compute nodes. Each server node contains dual AMD Opteron 2.2 GHz processors and 4 GB RAM, while each compute node contains dual AMD Opteron 2.2 GHz processors and 2 GB RAM. The cluster uses a Sun StorEdge 3510 FC Array with 12 TB of disk storage. We also use Linux-based PCs as the proxy systems and user systems.

5.2. Testbed Implementation

We use the Globus Toolkit (GT) 3.2 to implement the Grid Interface layer of the SPRING framework. GT 3.2 is installed on the WSN proxies, the Sun cluster, and the user systems. Currently, a testbed user submits a sensor job or a computational job from the command-line interface. We implemented a Grid Meta-scheduler using the Community Scheduler Framework (CSF) [15] to route sensor and computational jobs to their intended destinations.

The Sun Grid Engine (SGE) 6.0 [18] plays an important role in the implementation of the WSN Scheduler, as well as the Resource Scheduler for the compute cluster. For the WSN Scheduler, we set up a SGE queue for sensor jobs. Similarly, the compute Resource Scheduler has a SGE queue for computational jobs. To pass jobs from GT 3.2 to the SGE queues, we use a toolkit for integrating GT 3.2 with SGE called EPIC [19].

Another important component of our testbed is the WSN Management layer, which is based on MoteLab [8]. We reuse some software components of MoteLab; namely, the database backend, the job daemon, and the data logger. Our WSN Scheduler controls the operation of these components, and bypasses MoteLab's web frontend. We also implemented additional functionalities for this layer. For example, we added more functions to monitor the status of the motes. In MoteLab, all the motes are permanently connected to the network and are programmed via MIB600 interface boards. For our testbed, five "base station" motes are connected to the network via MIB600 boards. The rest of the motes are programmed wirelessly over-the-air in an automated manner using XnP [20].

We are currently working on the design and implementation of the Proxy Components. Most of these Proxy Components have not been fully implemented yet.

5.3. Testbed Status and Future Work

At present, a prototype sensor grid testbed has been successfully deployed. We can execute sensor jobs to collect sensor data and process them. We plan to enhance the testbed in several ways. First, we will continue to work on the design and implementation of the Proxy Components. Second, we will develop a workflow tool to help users develop sensor applications, and automate the submission of jobs. Third, we plan to enhance the wireless programming capability of the WSN Management layer by using a better tool called Deluge [21].

After the key enhancements to our sensor grid testbed are completed, we intend to conduct extensive experimental studies to obtain performance measurements from the testbed. This would enable us to evaluate the effectiveness of the SPRING design.

6. Conclusion

Wireless sensor networks and grid computing are promising technologies that are being adopted in the industry. By integrating wireless sensor networks and grid computing, sensor grids greatly enhance the potential of these technologies for new and powerful applications. Thus, we believe that sensor grids will attract growing attention from the research community and the industry.

In this paper, we have examined the important design issues and challenges for sensor grids. To address these design issues, we proposed a novel sensor grid architecture called the Scalable Proxy-based aRchItecture for seNsor Grid (SPRING). We have developed a sensor grid testbed to study the design issues of sensor grids. From our experience, the sensor grid testbed is a very useful research tool to study sensor grid issues and to improve our sensor grid architecture design.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: A survey. *Computer Networks*, 38(4):393–422, March 2002.

[2] D. Culler, D. Estrin, and M. Srivastava. Overview of sensor networks. *IEEE Computer*, pages 41–49, August 2004.

[3] I. Foster, C. Kesselman, and S. Tuecke. The anatomy of the grid: Enabling scalable virtual organizations. *Intl Journal of Supercomputer Applications*, 15(3):200–222, 2001.

[4] I. Foster and C. Kesselman. Globus: A metacomputing infrastructure toolkit. *Intl Journal of Supercomputer Applications*, 11(2):115–128, 1997.

[5] I. Foster, C. Kesselman, J. Nick, and S. Tuecke. The physiology of the grid: An Open Grid Services Architecture for distributed systems integration. *Open Grid Service Infrastructure WG, Global Grid Forum*, June 2002.

[6] S. Tuecke, et. al. Open Grid Services Infrastructure (OGSI) version 1.0. *Global Grid Forum Draft Recommendation*, June 2003.

[7] K. Czajkowski, et. al. From Open Grid Services Infrastructure to WS-Resource Framework: Refactoring & evolution, http://globus.org/wsrf/specs/ogsi_to_wsrf_1.0.pdf.

[8] G. Werner-Allen, P. Swieskowski, and M. Welsh. Motelab: A wireless sensor network testbed. In *Proc. of the 4th Intl. Conf. on Information Processing in Sensor Networks (ISPN '05), Special Track on Platform Tools and Design Methods for Network Embedded Sensors (SPOTS)*, pages 73–78, April 2005.

[9] L. Girod, et. al. Em*: A software environment for developing and deploying wireless sensor networks. In *Proc. of the 2004 USENIX Technical Conference*, April 2004.

[10] V. Naik, et. al. Kansei: Sensor testbed for at-scale experiments. In *2nd Intl TinyOS Technology Exchange*, February 2005.

[11] R. Bramley, et. al. Instruments and sensors as network services: Making instruments first class members of the grid. Technical Report 588, Indiana University CS Department, December 2003.

[12] M. Gaynor, et. al. Integrating wireless sensor networks with the grid. *IEEE Internet Computing*, pages 32–39, Jul/Aug 2004.

[13] V. Welch, et. al. Security for grid services. In *Proc. of the 12th IEEE Intl Symp on High Performance Distributed Computing (HPDC '03)*, pages 48–57, June 2003.

[14] G. Della-Libera, et. al. Security in a web services world: A proposed architecture and roadmap. White paper, IBM Corporation and Microsoft Corporation, April 2002.

[15] C. Smith. Open source metascheduling for virtual organizations with the Community Scheduler Framework (CSF). White paper, Platform Computing, Inc, 2004.

[16] J. Hill, et. al. System architecture directions for networked sensors. In *Proc. of the 9th Intl Conf on Architectural Support for Programming Languages and Operating Systems (ASPLOS 2000)*, pages 93–104, November 2000.

[17] D. Gay, et. al. The nesC language: A holistic approach to networked embedded systems. In *Proc. of the 2003 ACM SIGPLAN Conf on Programming Language Design and Implementation (PLDI 2003)*, pages 1–11, June 2003.

[18] P. Bulhoes, et. al. N1 Grid Engine 6 features and capabilities. White paper, Sun Microsystems, Inc.

[19] Epic - Sun Grid Engine integration with Globus toolkit, http://www.lesc.ic.ac.uk/projects/epic-gt-sge.html.

[20] Mote in-network programming user reference. Reference document, Crossbow Technology, Inc, 2003.

[21] J. Hui and D. Culler. The dynamic behavior of a data dissemination protocol for network programming at scale. In *Proc. of the 2nd ACM Conf on Embedded Networked Sensor Systems (SenSys '04)*, pages 81–94, November 2004.