A Middleware for OSCAR and Wireless Sensor Network Environments

D. J. Ferreira, M.A.R. Dantas
Departament of Informatics and Statistics (INE)
University of Santa Catarina (UFSC)
Florianopolis, Brazil 88040-900
ded,mario@inf.ufsc.br

A. R. Pinto, C. Montez
Graduation Program in
Electrical Engineering
University of Santa Catarina (UFSC)
Florianopolis, Brazil 88040-900
arpinto,montez@das.ufsc.br

Martius Rodriguez
Fluminense Federal University (UFF)
Niteroi - Rio de Janeiro
martius@cm.uff.br

Abstract

There has been a steady increase in the use of mobile computing, especially with appliances such as sensors. The main challenge in this scenario is to increase network sensor lifetime. Cluster computing integration with wireless sensor networks can represent an interesting answer for high-performance computing to monitor several environments. Computational resources available from cluster configurations are cost effective components to improve several classes of applications in many organizations. Scientific, industrial and commercial applications are more relying on cluster performance. OSCAR is a useful open software approach to manage cluster of workstations. In this article, we present a middleware design and implementation that integrates an OSCAR cluster configuration with a wireless sensor network environment.

1. Introduction

Distributed computing applications are developed and implemented to solve a large variety of different problems. This approach can offer some advantages such as: resources sharing, transparency of complexities to gather equipment and fault tolerance. These characteristics are attractive to developers of applications that require high capacity to process tasks and memory space for large amount of data.

The use of wireless networks brought a wide range of possibilities for distributed applications. New technologies for mobile devices have appeared and are providing an incremental number of benefits to users. On the other hand, this new paradigm brought new challenges, especially because mobile devices have limited facilities. These devices present challenges like such as low power processing capability, battery lifetime, limited amount of memory, and issues related to communication link [14].

The mobility paradigm provides remote access of information anytime, from anywhere, thus providing an enlargement of the view of distributed computing concepts. Several researches has focus on how to improve data access in a fast and secure fashion in wireless environments. The main purpose of these efforts is to provide to users new facilities comprising a large number of services, tools and applications in many areas.

Wireless sensor network differ from ordinary pervasive devices in many aspects. These tiny nodes usually have a battery, as the energy source, and offer radical changes on the way people interact with the environment. It is a paradigm improves the interaction between applications and users, considerably increasing system's flexibility [9]. However, there are many challenges to be explored. In several cases, it is necessary to have a large number of sensor nodes to monitor a specific area. As a result, these sensors gather a large amount of data that has to be processed and stored. Some statistic and conversion operations over all data received could be hard to calculate, requiring a high level of processor power. An ordinary computer generally does not have space to store all data results and enough processing capability. The use of a cluster helps to process the required data, utilizing techniques such as parallel computing and resources sharing. The OSCAR (Open Source Cluster Application Resources)[8] is an interesting solution to applications in a distributed system configuration. It has many tools that help developers to manage clusters and do

21st International Symposium on High Performance Computing Systems and Applications(HPCS'07)
0-7695-2813-9/07 $20.00 © 2007 IEEE

not require a more specialized knowledge related to distributed computing.

In this paper, we present a middleware prototype that integrates a wireless sensor network with a OSCAR cluster configuration. The goal is to allow processing and storing of a large amount of data transparently distributed over all cluster nodes from the sensor network.

This paper is organized as follows. Related research works are shown in Section 2. Section 3 presents some aspects of OSCAR cluster computing. Wireless network and wireless sensor networks are presented in Section 4. Section 5 presents the main concepts of the proposed middleware. Finally, in Section 6 we draw some conclusions about the present research and our contribution, indicating some future directions for this work.

2. Related Work

High performance computing and wireless networks are two technologies closely related to the present work. There are several researches on clusters environments [15, 13, 2] and sensor networks [5, 14]. However, in the literature there are few efforts to integrate these two paradigms. In this section we briefly describe some efforts on pervasive computing and cluster integration.

Dyo [7] proposed the design of a middleware to connect sensor network and mobile devices. The idea behind this work is to provide to mobile users spacial queries for locations of specific data from the sensor network field. The work of Dyo explores data aggregation and data collection algorithms. The main difference between the present contribution and the proposal from Dyo is that it worked in an approach that has decentralized characteristics. In this case, there is no computer interconnecting with the wireless sensor network to mobile devices. This can be a problem, because sensor nodes have limited process capacity , similar to others mobile devices. On the other hand, a cluster infrastructure between this equipment can operate heavy processes over sensed data from sensor network and store related results. In our project the decision was to design an approach to utilize a cluster configuration, which avoids mobile devices to process sensed data and consume energy.

Lin presented another research associated with this article [11]. The proposal is a binary sensor network based on a parallel annealing algorithm that executes a search strategy with majority of computation time devoted to data fusion. It simulates the sensor network. The main difference of our proposal is that we employed real sensor motes (motes are well known sensors nodes from Crossbow Technology Inc. [1]).

In [6] is presented a prototype implementation project to utilize a wireless LAN together with a mobile computing approach to enhance the monitor function of a COW (cluster of workstation) configuration. The present proposal contribution is an improvement, integrating wireless sensor networks with cluster environments. In other words, our goal is to provide a middleware between wireless sensor network and cluster system executing the OSCAR package. Figure 1 shows the architecture prototype.

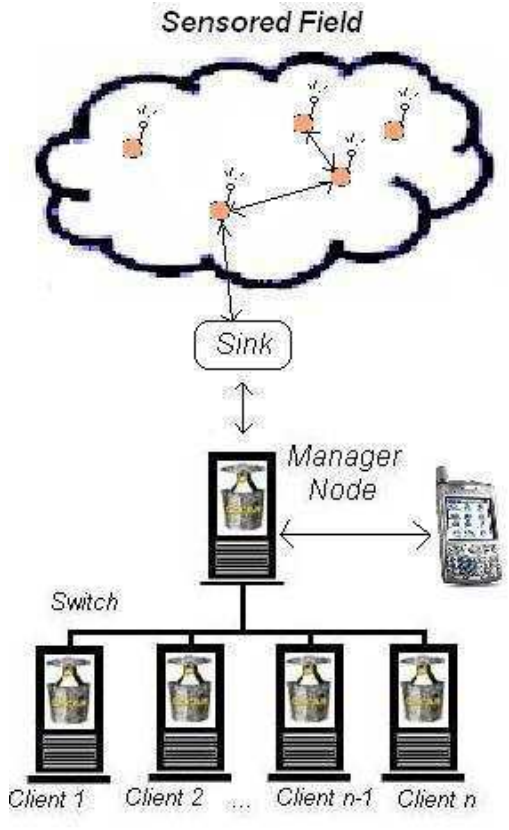

Figure 1. Architecture Proposal.

3. OSCAR Cluster Computing

In a COW the single system image (SSI) is an important part of the environment since it manages globally and transparently cluster resources (e.g Project SSI-OSCAR) [15]. Users from these configurations believe that they are using a single powerful machine.

Nowadays, clusters of workstations have become an essential computational resource to enhance scientific, industrial and commercial applications. Therefore, a cluster configuration should provide some level of security, availability and tools to easy management of the environment. These facilities can be achieved reachable with OSCAR.

OSCAR consists of a fully integrated and easy to install software bundle designed for high performance computing

cluster. Everything required to install, build, maintain and use a Linux cluster is included in the suite [8]. There are several packages in the core program that help a user to monitor parallel processes and share resources of the system. These features can improve performance of the cluster, by considering aspects of fault tolerance. Libraries and tools such as MPI (Message Passing Interface) [2] and PVM (Parallel Virtual Machine) [13], widely used for distributed parallel computing, are part of the default OSCAR suite.

An OSCAR cluster environment generally consists of three main components [8]:

1. Server: machines that provide services to the cluster, responsible for submission of tasks to client nodes. It includes PBS [3] server, NFS file server and others.

2. Gateway: it must have at least two network connectors, a internal network of the cluster and another public external network. This schema helps to protect the cluster. Moreover, the level of security can be fitted to the user needs. Usually a server node has both functionalities, as gateway and server.

3. Client nodes: machines that execute tasks submitted to the cluster. OSCAR allows homogeneous or heterogeneous nodes in the cluster.

4. Wireless Network

Mobile computation environments are frequently seen everywhere on people's life and work. However, there are many limitations. Issues related to the communication bandwidth, mobility of equipment and limited resources and frequent disconnection of the network are some typical examples.

The main feature of these systems is the use of wireless infrastructure[11], which is an electronic data communications system providing an extension, or alternative, to a wired LAN. Wireless LANs use a variety of communication mechanisms to replace the traditional cables and wires of a LAN. In a traditional LAN, data is transmitted as electronic pulses (or signals) along a physical wire (or carrier). Similarly, in a wireless LAN, there are transmitters, receivers, and a carrier on which data is modulated [12].

The mobile technology provides flexibility to applications and a large number of functionalities. One of the benefits is productivity. There is the possibility to access remote information. One of the challenges is to allow workers to move without interruption of processing on devices. However, the main challenge on a wireless network is conserve energy, in order to prolong network lifetime. Factors like data latency and bandwidth traffic directly influence battery consumption.

4.1. Wireless Sensor Network

Wireless sensor networks can be considered distributed computing platforms with many severe constraints including limited CPU speed, memory size, power, and bandwidth. Individual nodes in sensor networks are typically unreliable and the network topology dynamically changes [14].

A sensor network is composed of a large number of sensor nodes that are densely deployed either inside the phenomenon or very close to it [5]. These sensors have some level of intelligence, making answers to the environment, or not. Due to the sensor memory and processor limitations, usually the collected data are sent to a main server. These data are still in raw digital signal format, or in hexadecimal. After processing these data, the server sends back results by Internet or satellite to the final machine [9].

Sensors nodes are tiny devices with low process, communication and battery capacity. Each node typically consists of the five components: sensor unit, analog digital convertor (ADC), central processing unit (CPU), power unit, and communication unit [10]. Figure 2 shows a conventional sensor node schema.

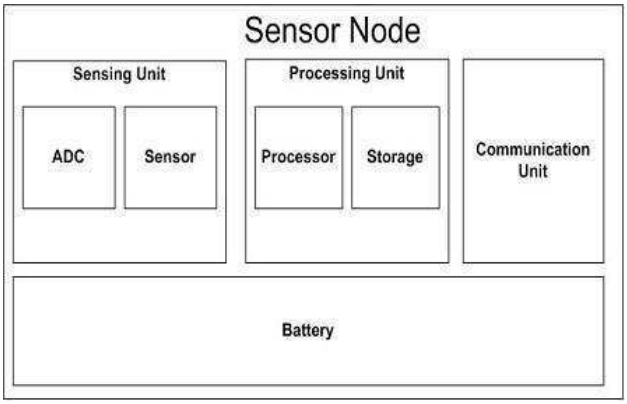

Figure 2. Sensor Node Schema [5].

Each node can have several sensors. These sensors can measure any type of physical phenomena like temperature, sound, pressure or acceleration. These nodes operate as sensors, wireless routers or both, depending on application needs and network configurations. The node position in the network does not need to be set or even known. Therefore, they can be randomly scattered on the monitored field, being able to access disasters and places unreachable by humans [14].

The grand challenge of wireless sensor networks is to increase the network lifetime. Routing policies to retain energy should be taken, since wireless communication spends much more battery than processing. Sensor acquisition can

be achieved at 1 nJ per sample, and modern processors can perform computations as low as 1 nJ per instruction. Current radio frequency techniques consume about 100 nJ per bit for a distance of 10 to 100 m, making communication very expensive compared to acquisition and processing [9].

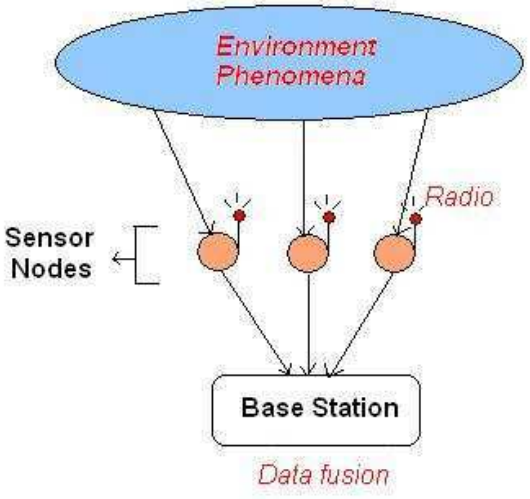

Figure 3. Sensor Network Environment Example.

Some features of sensor networks like fast deployment, self-organization, fault tolerance make them useful for a wide range of applications. Tracking contamination in hazardous environments, habitat monitoring in the nature preserves, enemy tracking in battlefield environments, traffic monitoring, surveillance of buildings, are examples of systems that use wireless sensor networks facilities [10]. Figure 3 shows a typical sensor network environment.

5. The Middleware Architecture Proposal

The main contribution of the proposed middleware is the integration between wireless sensor network, mobile devices and OSCAR cluster computing. Therefore, it can be considered a high performance solution to monitor wireless sensor networks with large amounts of sensed data to be processed and stored in an OSCAR cluster using mobile devices to show results.

In this section, we first present aspects related to the experimental configuration and sensor network schema. After characterizing the sensor network environment and schema we present the architecture of the proposed middleware.

5.1. Experimental Environment

Wireless sensor networks generally work with a lot of information since each node can sense and transmit data. Sensor nodes have lots of resource constraints, each one can execut just some operations over sensed data. Therefore, information arrives at user computer partially processed. The sensed values read from the sensor node base are still in raw analog or digital format and they have to be converted to engineering units. Moreover, depending on application function, statistics operations and many other processes could be made over collected data. Some of this computations demand too much processor time. If a network with thousands of nodes is considered, there could be a large amount of data to be processed at any time. These operations consume a great amount of memory and it could lead to a processor overload. An ordinary machine would not be able to operate over so much information. Furthermore, it would not have enough disk space to store the results.

Cluster systems can solve that problem, since it runs parallel application and distribute memory among cluster nodes. It is necessary considerable knowledge about computer system and programming, demanding time. This cost could be avoided with OSCAR tool, which provides facilities to programmers in sharing cluster resources and load processes control. Figure 4 shows the system schema.

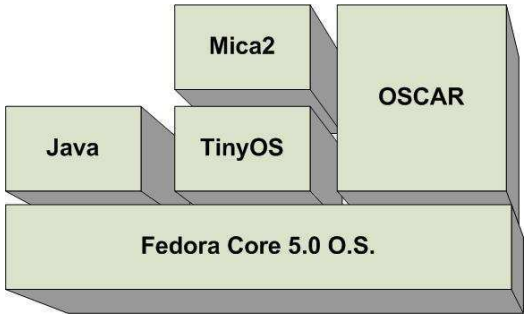

Figure 4. System Schema.

The core parts of the experimental environment infrastructure are shown in tables 1 and 2:

Table 1. OSCAR Cluster Characteristics

Master	Slave	Palm Tungsten
1.8 Ghz	1 Ghz	400 MHz
Pentium IV	Via Nehemiah	Intel XScale
512 Mb	1 Gb	64 Mb
Fedora Core 5.0	Fedora Core 5.0	Palm O.S.

Table 2. Wireless Sensor Network Components

Component	Model	Radio Frequency
Sensor Node	Mica2	915 Mhz
Sensor Board	MTS300	-
Programming Board	MIB510	-

5.2. Sensor Network Schema

The TinyOS was chosen as the operating system for our motes because it is a open software designed to minimize power consumption. This is an important factor since the main challenge on sensor networks is to extend battery life of the nodes. TinyOS is an event driven operating system that handles the power consumption and radio communication. TinyOS commands are only executed if an event occurs. This feature avoids processor use when it is not necessary [4].

TinyOS uses NesC as language to program motes. NesC is an extension of the C language that wires application components with "configurations" and than they can be reused [4]. The program downloaded in our sensor nodes is a temperature reader, that periodically collect temperature sensor readings, and send them to the base station. It uses some components like a timer, to calculate the regularity of readings. When the timer expires, a read of environment temperature is done. At this point, timer restarts. The data read is 16 bit unsigned integer. Therefore, the DemoSensor component, which in our case is represented by mica2 architecture, is responsible to decide which sensor is going to be used as a sample. When this operation is finished, a message with data sensed is broadcast over the radio [4]. The base station receives sensed information. While sending the readings, the program uses a led component to turn on the green led. It helps the user to see when the radio is being used, since communication is the main factor on spending energy for sensor networks.

We used a centralized topology, where information from the sensors flows directly to a single fusion center, the node on base station interface. This node is programmed to receive packets over the radio network and forward them to the master node from the cluster, using a serial port. When the base station program receives a packet, it toggles a led, and when it sends this packets to the serial port it toggles another led.

Accessing the serial port, the base station sends the packets to one computer. The OSCAR cluster master receives this sensed data. Packets that arrive in master node can be monitored by the Oscilloscope program, a Java based graphics tool of TinyOS that permits a user to visualize sensor readings on the computer.

5.3. OSCAR Middleware

The proposed OSCAR middleware was designed and implemented using Java, which interconnects the data received by the sensor network to the OSCAR cluster. Data transmitted by sensors are raw digital readings. The base station forwards this raw data to the middleware, that has to convert them to engineering units. It calculates in degrees of Celsius the raw temperature information received.

OSCAR is used to save the readings in a distributed form in the cluster nodes. The system reads environment temperature periodically, and in few hours there is a huge amount of data to be processed and stored. Using OSCAR tools helps to compute a number of processes that utilizes the the large amount of sensed data. C3 is a tool that allows parallel execution of commands and file distribution across all cluster nodes. The proposed middleware interacts with the cluster machines and makes them cooperate to compute and store the files containing collected data.

Furthermore, this data can be seen by the user from their desktop PC or by a mobile device, with a program that make queries for the cluster asking for monitored items from the last few ours.

The middleware is divided in modules that work together with the cluster system involking tasks to help the performance of processes. Figure 5, shows environment integration schema. The Cluster Module interacts with the system. A Data Receiver module is responsible to get data from base station and convert them to engineering units. Operations Controller module can make operations over sensed data, obtaining results to be distributed stored in the cluster.

Mobile devices can make queries connecting to the cluster by Server module. Persistence calls from the cluster requested data and Server returns the answer to the mobile device.

6. Conclusions and future work

In this article we have presented a middleware proposal that represents an important tool to efficiently integrate wireless sensor networks, mobile devices and OSCAR clusters environments.

The use of mobile devices to monitor environment provides mobility to users obtain information of the system anytime, anywhere.

Empirical experiments were executed considering a field monitored by a mobile device. These tests were characterized by a cluster with three client nodes connected by the proposed middleware to a mica2 wireless sensor network and to the mobile device. The mobile device requests the last readings. The middleware is already under development

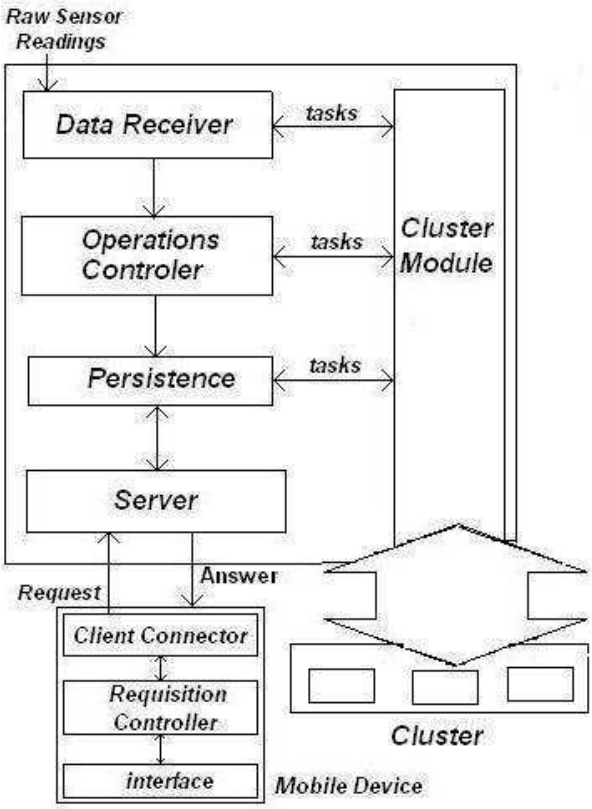

Figure 5. Middleware Architecture.

and our early results indicates that it reaches a successfully level of our objectives for the wireless monitoring approach.

There are many challenges to be solved as a future work. Some examples are introducing routing topologies to the sensor network and make to enhance the prototype providing alerts to the mobile device, when sensored values are out of a pre-determinated range. Another aspect is to make available the prototype for the OSCAR Group.

7. Acknowledgments

This work was supported in part by CAPES(Coordenação de Aperfeiçoamento de Pessoal de Nível Superior) and BITEC 2006.

References

[1] Crossbow technology inc. URL: http://www.xbow.com/, 2007.

[2] The message passing interface (mpi)standard. URL: http://www-unix.mcs.anl.gov/mpi/, 2007.

[3] Portable batch system (openpbs). URL: http://www.openpbs.org, 2005/, 2007.

[4] Tinyos 2.0 tutorials. URL: http://www.tinyos.net/tinyos-2.x/doc/html/tutorial/, 2007.

[5] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor networks. *Communications Magazine, IEEE*, 40(8):102 – 114, Aug. 2002.

[6] M. A. R. Dantas and C. Rista. A wireless monitoring approach for a ha-oscar cluster environment. *HPCS 2005. 19th International Symposium on High Performance Computing Systems and Applications*, (15-18):302 – 306, May 2005.

[7] V. Dyo. Middleware design for integration of sensor network and mobile devices. *Proceedings of the 2nd international doctoral symposium on Middleware, ACM International Conference*, 114:1 – 5, July-Aug. 2005.

[8] T. O. C. Group. Oscar (open source cluster application resources). URL: http://oscar.openclustergroup.org/, 2007.

[9] M. Ilyas and I. Mahgoub. *Handbook of Sensor Networks: Compact Wireless and Wired Sensing Systems*. CRC Press, 2005.

[10] Q. Jiang and D. Manivannan. Routing protocols for sensor networks. *CCNC 2004. First IEEE Consumer Communications and Networking Conference*, (5-8):93 – 98, January 2004.

[11] H. Lin, J. Rushing, S. J. Graves, S. Tanner, and E. Criswell. Real time target tracking with binary sensor networks and parallel computing. *2006 IEEE International Conference on Granular Computing*, (10-12):112 – 117, May 2006.

[12] J. Nikolaidis. Wireless security and privacy, best practices and design techniques. *Network, IEEE*, 17(4):6 – 7, July-Aug. 2003.

[13] M. Pernice. Pvm: Parallel virtual machine: A users guide and tutorial for networked parallel computing. *Parallel & Distributed Technology: Systems & Applications, IEEE*, 4(1):84, Spring 1996.

[14] J. A. Stankovic, T. E. Abdelzaher, C. Lu, L. Sha, and J. C. Hou. Real-time communication and coordination in embedded sensor networks. *Proceedings of the IEEE*, 91(7):1002 – 1022, July 2003.

[15] G. Vallee, S. L. Scott, C. Morin, J.-Y. Berthou, and H. Prisker. Ssi-oscar: a cluster distribution for high performance computing using a single system image. *HPCS 2005. 19th International Symposium on High Performance Computing Systems and Applications*, (15-18):319 – 325, May 2005.