Building the Senceive System

Waltenegus Dargie
Technical University of Dresden
Chair of Computer Networks
Faculty of Computer Science
01062 Dresden
waltenegus.dargie@tu-dresden.de

Alexander Schill
Technical University of Dresden
Chair of Computer Networks
Faculty of Computer Science
01062 Dresden
alexander.schill@tu-dresden.de

ABSTRACT

The conception and development of pervasive systems, i.e, the systems which will be used in pervasive computing environments, involve interdisciplinary team work. Apparently, the team consists of people with a diverse research background and expertise. While such a composition is an essential prerequisite to solve real world problems, it brings with it also challenges that should be dealt with. To begin with, team members should establish a shared understanding of what should be done. This understanding includes the terminologies that are used as well as the expected project goals. Secondly, there has to be a division of task and a clear plan as to how different components or building blocks should come together to make up a unified, consistent, side-effect free and wholesome system. In this paper we discuss the development of the Senceive System within a graduate project course work. The project work involves students from computer science, computer engineering and electrical engineering. Technically, the Senceive System offers a stepwise abstraction of low-level concerns (sensing) from higher-level use of meaningful features.

Categories and Subject Descriptors

C.2.4 [**Computer-Communication Networks**]: Distributed Systems—*distributed applications*

General Terms

building a sensing system, wireless sensor networks

Keywords

Activity recognition, project work, Complex system design, Wireless sensor networks, Network configuration

1. INTRODUCTION

Recent advances in wireless sensing and processing devices have made the promise of using wireless sensor networks of diversity of applications. Unlike many applications that are proposed or developed for pervasive computing, the development of wireless sensor networks for a specific applications involves several people from different research fields. For example, in habitat monitoring, an expert knowledge of wild birds and animals is required [11]; for structural health monitoring, knowledge of building and bridges as well as the static and dynamic properties of structures is essential [9]; for healthcare applications, one is required to know about Parkinson's disease (PD) or body segments and activity of daily life (ADL) etc. [1], and [10]; and toxic gas detection in oil refineries requires knowledge about pipelines, toxic gases and safety regulations [5], [2].

Most existing projects on wireless sensor networks are graduate or post graduate projects and require a significant conception and development time. Some of the members that make up the project teams may not have system engineering or programming experience before. Sometimes this results in frustrations and a significant portion of the project period is spent as members learn how to organise themselves; how to share tasks; and how to integrate individual contributions.

The graduate level study of computer science and engineering at the Technical University of Dresden includes compulsory practical project works before students begin their thesis. The aim of these projects is to prepare students to develop complex systems by working in teams and by applying both theoretical and technical backgrounds. The project requires interdisciplinary interactions and a variety of hardware and software technologies.

The team size as a rule does not exceed five people, and in one project assignment, there can be as many as five groups, each group having one supervisor. Students are expected to invest four hours a week to come together and work as a group. On average a total of 80 hours are required to successfully complete the task. Additional eight hours are reserved for students to present progress report and get feedback from their fellow students and their supervisors.

1.1 Challenges

One of the typical challenges of these project courses is ensuring that all members of a team contribute equitably - neither dominant nor passive members make up an ideal team. Another challenge for students is exercising self-discipline and keeping deadlines - there is an associated penalty with each deadline violated. There are altogether six deadlines, five of them for reporting progress and one of them for demonstration and the final presentation. An additional challenge is related to the background experience students bring to the task, since they have to divide among themselves the project tasks according to their experience and preference. During this division of labor, some students may not get a task for which they are well equipped; and they need to quickly update themselves on particular technologies or programming languages. Their learning pace determines the pace of the whole team.

1.2 A model Topic

One of these projects is the Senceive system [1], a system that integrates wireless sensor networks and probabilistic-based reasoning systems to recognize various everyday human activities in different settings, such as in lecture and seminar rooms at the university and in different carriages of passenger trains and street trams. In the following sections of this paper, the development of the system as a project work will be reported.

The remaining part of this article is organized as follows: in section 2, a higher-level requirement to the system is defined; in section 3, the detailed execution plan of the project will be introduced; in section 4 and 5, a higher-level conceptual architecture and its refined version are discussed; in section 6, the implementation and evaluation of the Senceive system will be discussed in detail; and finally, in section 7, the lessons and experienced learned during the implementation of the project will be reported.

2. REQUIREMENTS

At the beginning of the project, students were given a general description of the system along with a few basic requirements.

2.1 Platform

Senceive would be used by users who would like to determine the activities taking place in physical places without actually being there - lectures, parties, meetings, casual discussions, etc., in lecture and seminar rooms; fighting, conversations, shouts, etc., in train carriages. The users would be mostly mobile and would own mobile devices - for example, students and professors would like to locate lecture sessions or meetings with the help of Senceive. Therefore, the system should be able to support mobility and mobile devices.

2.2 System Integration

During the project work, team work is very much expected, but individual members should also be able to work independently. The balance can be made by requiring students develop loosely-coupled components that can be brought together by well defined interfaces. To ensure the smooth integration, students are required to develop the conceptual architecture of the systems and to define all the interfaces of the main building blocks of the architecture together.

2.3 Sensing

Students were required to use existing wireless sensor nodes (Micaz and Mica2 nodes [14]) to establish the sensing subsystem. A wireless sensor network was preferred because it was easier to deploy the nodes without disrupting the normal functions of rooms and carriages. Moreover, wireless sensor networks were one of the main research focus of the Chair, and the expected results could be exploited by some of the actively running research projects.

[1]The acronym stands for Sensing and Perceiving. A comprehensive treatment of the technical aspect of the Senceive system is given in [7].

Furthermore, the sensing subsystem should support multiple applications. Since a wireless sensor network was chosen for the sensing task, this seemed to stand in contradiction with the basic assumption in wireless sensor networks. In the literature, it is argued that a wireless sensor network is application specific; the sensing task should be known at the time the network is deployed; and the task does not change over time [8]. This premises is the basis for developing energy efficient communication protocols and data aggregation algorithms. Because wireless sensor nodes operate with exhaustible batteries and charging or replacing these batteries is not a simple task, energy efficiency is a serious concern. At the same time, however, wireless sensor networks are emerging technologies. While designing energy efficient networks is important, identifying suitable applications for them and rapid prototyping and testing is vital as well. Subsequently, for this project, the energy consumption issue was not a priority concern.

The sensing subsystem should therefore offer three different interfaces to applications and users. The first interface should enable applications to access sensed data; the second interface should enable an administration to have a complete knowledge of the deployed nodes and to configure the network. The administration may not have detail knowledge of network programming and his task should be limited to adjusting some known parameters, manage access rights, and monitor if all existing nodes are functioning properly. The third interface should enable an experienced programmer to perform fundamental reconfigurations, not only by changing parameters, but also by entirely changing a part of the modules that are deployed on individual nodes. This type of reconfiguration is essential because (1) during deployment, complete knowledge of the deployment setting may not be known; (2) both applications' requirement and the properties of the environment can change over time; and (3) once a network is deployed, it is necessary to detect and fix bugs while the network is still performing the sensing task. This is particularly important since manual reconfiguration of individual nodes is not desirable, as it means collecting a large number of nodes that are deployed in an extensive field.

3. MILESTONES AND PROCEDURES

One way of ensuring the timely completion of the project work is by defining specific milestones and by requiring students keep deadlines. Given the complexity of the system they develop and the relatively short duration of the project, the challenge is not so easy. In order to help students better organize themselves, the important milestones along with the associated duration to reach the milestones are defined for them. The milestones are the same for all teams, regardless of the topic they choose. This way, it is possible to ensure that all groups complete their task uniformly.

Table 1 shows the detailed time plan, the milestones and the expected results of the project.

To further assist students keep deadlines, each supervisor broke down the system development task into subtasks. The breaking down of tasks into subtasks would enable the students define short and long term goals and to better organize themselves. For the Senceive system, table 2 displays

Week	Date	Meeting	Assignment	Milestones	Expected Results
1.	13.10.2006	All	Introduction		Group formation
2.	20.10.2006	Team meeting	First team meeting; Starting requirement analysis		
3.	27.10.2006	Team meeting	Requirement analysis		Division of tasks
4.	03.11.2006	Presentation	Assessment of related work; a conceptual architecture	M1	Specification of the architecture
6.	17.11.2006	Presentation	Refined conceptual architecture	M2	Refinement
9.	08.12.2006	Implementation	Prototype	M3	In part
12.	12.01.2007	Presentation	Prototype	M4	Complete
14.	26.01.2007	Presentation	Testing and evaluation	M5	Refinement
15.	02.02.2007	Team meeting	Finalizing the documentation	M6	Submission

Table 1: The Project's schedule and milestones

Step	Task	Remark
1.	Refining the Architecture	Detail understanding the components of the architecture
2.	Platform Specification	Identification of the devices and technologies required
3.	Protocols and algorithms	definition of the Network's topology and data processing algorithms
4.	First phase prototyping	Implementation of the sensing subsystem
5.	Modeling	Extracting higher-level features and feature modeling
6.	Reasoning	Implementing and training the recognition scheme
7.	Testing	Testing the system as a whole
8.	Final Presentation	Submission of the final document

Table 2: A guideline for a step-by-step implementation of the Senceive system

the procedure that was prepared and made available to the students by their supervisor.

4. CONCEPTUAL ARCHITECTURE

Three weeks after the project was formally started and two weeks after the teams met their supervisor, the first milestone would be reached. By this time, students have closely studied the typical features of the system they would develop; made requirements analysis; and drafted a higher-level conceptual architecture which would serve as a basis for division of labor. Accordingly, the Senceive group met twice and individuals shared their experience. It is worth to note that the students who made up the team had quite a diversity of cultural and educational background, even though they had been studying similar subjects. This was a challenge at the outset, since establishing a shared understanding of the main task was difficult, but later it proved to be very useful. The result of the two meetings was a hierarchical conceptual architecture, which is displayed in figure 1.

The group identified four main layers in the architecture. The bottom layer, the sensing layer, delivers raw sensor data which is the basis for reasoning the activities that take place in various places. The modeling layer extracts higher-level features from the sensed data and establishes relationships between these features. This layer requires signal processing and stochastic analysis. The features are stored in a knowledge base. The reasoning layer receives various features and feature models in order to recognize the higher-level activities that are represented by the features. Finally, applications put query and subscription requestes to the recognition layer to intelligently make decisions.

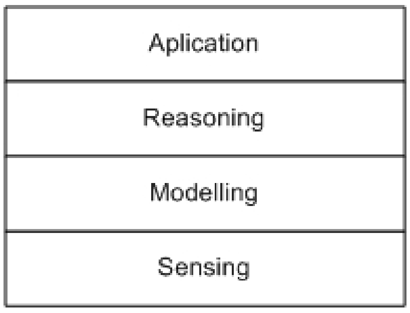


Figure 1: The conceptual architecture of the Senceive system

Once the group agreed on the conceptual architecture, the next step was division of labor[2]. The team members could easily identify where they could best fit. The task that needed more work was at the sensing layer, since besides establishing the network and efficiently collecting data, there was also a management task. Two people volunteered to work together (and the others agreed), one being responsible for establishing the network and defining various commands that would enable dynamic network management, and the

[2]Note the discrepancy between the schedule specified in table 1 and the way things progressed in reality. The schedule specified that division of labor should occur during the second week of the project, before the conception of the system architecture. This could not be done, however, before students acquired a better understanding of the entire system and before they are confident of their own contribution. This was discussed with the supervisor and an agreement was reached to exercise some flexibility as long as the milestones were reached at the appropriate time.

Command	Purpose
Set location	Sets the location label for the selected node
Delet history data	Empties local storage
Auto config (AC) on	Switch on/off global automatic node configuration
AC mic gain	Sets microphone gain (0 to 255)
AC rout update	Sets rout update interval
AC storage mode	Sets storage mode value (8 or 16)
AC LPL cycle	Sets low power listening duty cycle
Status check interval	Sets node status request interval
Node active timeout	sets the sleeping duration of a node
Memory auto-DL	sets downloading interval of stored

Table 3: Control commands for interacting and configuring the wireless sensor network

other being responsible for implementing the commands and designing the management interface that would enable both an experienced programmer and a lay administrator to configure the network. The remaining three students identified their place in the remaining layers.

5. REFINING THE ARCHITECTURE

In the following weeks, students focused mainly on researching related work; acquainting themselves with existing technologies and refining the layer of the conceptual architecture for which they were responsible.

5.1 The sensing layer

The sensing layer supports two specific tasks: establishing the wireless sensor network for collecting raw data from various places and managing the network to accommodate multiple applications and dynamic network (re)configuration. The student who was responsible for establishing the network with Micaz and Mica2 sensor nodes (section 2), identified three essential tasks: interpretation, collection, and configuration. For each of these tasks, he defined a corresponding component and a set of interfaces. Moreover, he defined 10 basic control commands that would enable users and administrators to interact with these components and thereby, with the sensor network. Table 3 displays the commands and their specific purpose.

The student who was responsible for designing the network management subsystem identified three main components that would permit flexible operation and management. These are the query/subscription service, the lower-level configuration service and the higher-level configuration service. The detailed description of the management subsystem is displayed in figure 2.

5.1.1 *The Query/Subscription Processing Service*

The query/subscription processing service enables applications (according to the conceptual architecture, multiple modeling services) to access sensor data declaratively. The premise for the existence of this service is that whether the network is setup with a single application in mind or not, most existing applications extract sensor data from the network and perform data processing elsewhere. For example, structural health monitoring [3] and active volcano monitoring [13] applications collect raw data from the sensor network, but feature extraction takes place with the support of resource rich computers outside of the network. Likewise, existing sensor nodes do not support higher-level digital signal process such as extraction of Mel Frequency Cepstral Coefficients [6] from an audio signal at the local level. Therefore, applications can declaratively express interest by specifying the duration and sampling frequency of data that should be collected by the network.

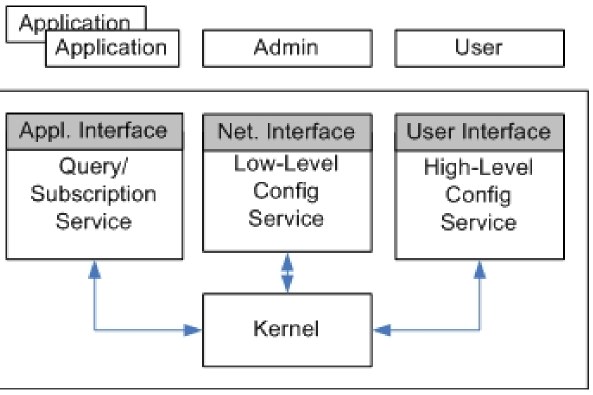


Figure 2: The conceptual architecture of the Senceive Middleware

More specifically, the query/subscription processing service provides the following functionalities:

- Process snapshot queries and historical queries;
- Start long run queries with or without data listener;
- Provide information about available sensors in the network along with their present internal configurations;
- Register listener to process relevant events (changes in network status).

5.1.2 *The Higher-Level Configuration Service*

The higher-level configuration service enables centralized control of the network. A centralized control ensures that the user's policy is enforced and the integrity of the network is maintained, and that only eligible applications are accessing the network. It enables also an administrator to monitor the status of individual nodes. If the network supports multiple applications, there can be some potential conflicts, for example, if three applications put an alert (alert-if-below) subscription request, in which case the applications are interested to be notified when the temperature of a certain region falls below 12, 15 and 17řC, respectively. Because an "alert-if-below" request is a simple request, nodes can process such a subscription locally. However, a simple node may process only a few of these requests because of local memory constraint[3]. In this case, the higher-level configuration service decides which of these thresholds should be evaluated locally and which of them centrally so that the overall data traffic that results due to these requests is minimized.

More specifically, the higher-level configuration services provides the following functionalities:

[3]Note that there are additional sensors and other resources in a single node and associated requests that should be processed locally as well

- Provides detailed network status information
- Provides information about configuration aspects
- Modifies individual node configuration
- Modifies global configuration
- Dynamically integrates and configures new nodes and update corresponding routing and medium access policies.

5.1.3 The Lower-Level Configuration Service

In secion 2, it was mentioned some of the reasons why we need to reconfigure an already deployed network. The higher-level configuration discussed above is usually referred to as soft-configuration, since it does not affect the runtime code of a node. There are, however, conditions in which one needs to replace certain modules (for example, replacing a low-pass filter with a bandpass filter) or even the entire set of modules, as in the case of a considerable fault in the deployed code or a total change in the user's requirement.

Typical tasks of a lower-level (re)configuration task is to make sure that a program code is propagated successfully and the new set of modules are installed in a consistent manner. If some nodes have installed the code and others are still running old modules, there will be a significant conflict. For this reason, the lower-level configuration services should be able to monitor consistency of modules and successful code propagation and installation.

5.1.4 The Kernel

The kernel interfaces the higher-level services with the wireless sensor network. Its basic purpose is to transform the higher-level syntax into lower-level syntax that can be processed and executed by the network elements. Moreover, it plays a vital role when in-network decision is made based on global knowledge. For example, it stores global variables such as the number of cluster heads allowed to form a hierarchical topology.

5.2 Modeling, Reasoning, and Application

The three upper layers are very closely related because the modeling and reasoning processes are application specific. Hence, the first important decision that was made by the three responsible students was to identify the sensors which would deliver meaningful data [4]. The student responsible for the modeling layer identified two important components that were useful to storing raw sensor data (a distributed database) and the higher-level features and their interpretation (a knowledge base). Furthermore, he defined the way to analyze the statistical significance of the data in the database and defined how to save interpretations in the knowledge base.

It was decided by the entire team and their supervisor to use fuzzy logic for classifying the stochastic and time domain features. So task of the student who was responsible for the reasoning layer was to define fuzzy sets and membership functions. Therefore, the task of this student was more of analysis that software design.

The student who was responsible for the application layer was responsible to design a graphical user interface and to interface the application with the reasoning layer. The graphical user interface should enable users to declaratively put requests to the sensing layer or the reasoning layer so that they can observe the change in the measurement data or simply obtain higher-level activities from the reasoning layer.

[4]These were temperature, light, and acoustic sensors.

6. IMPLEMENTATION AND EVALUATION

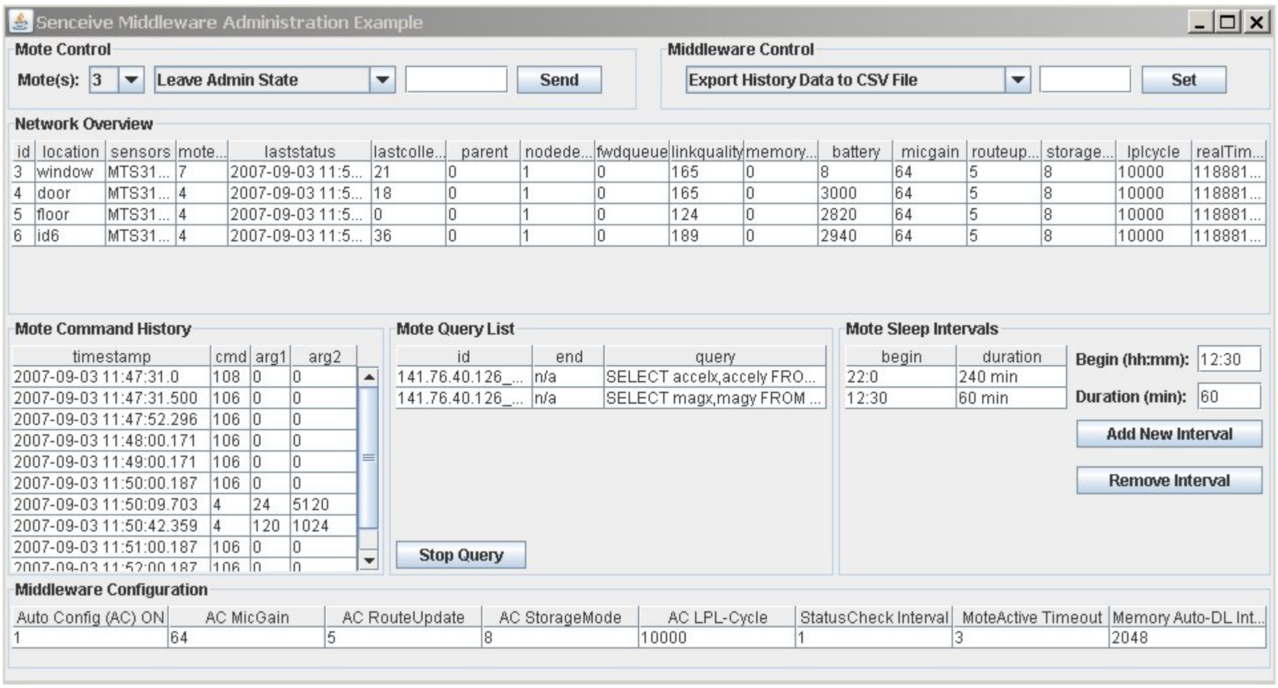
The Senceive team presented the conceptual architecture and the refined version thereof to the whole project groups and received constructive feedback. Afterwards, they proceeded with the implementation plan. The sensor network was setup based on the TinyOS 2.x runtime environment. The network was made up of Crossbow MICAz nodes with MTS300 and MTS310 sensor boards. The sensing subsystem supported single sensor access, stream sampling, resource arbitration and power management. Newly arriving nodes could autonomously register with the network, synchronize to global network time, receive commands to alter configuration or start sensing tasks and reliably deliver data to a sink.

The management subsystem provided external control of and dynamic binding to the network. Remote applications could access the network and collect data using the Java Remote Method Invocation (RMI). The query/subscription processing service made available a complete description of the services that were supported by the sensing subsystem, including the type of query and subscription requests that could be processed. Likewise, the higher-level configuration service provided network administrators with information pertaining to the number of available nodes and their spatial distribution as well as the configuration functionalities. Figure 3 shows the administrator's view of the Senceive system.

The wireless sensor network supported multi-hop communications; the minimum distance for a single-hop communication was determined by the service quality (tolerable end-to-end delay and packet loss) that was defined by the administrator. The received signal strength was used to set a threshold for multi-hop communication. For data gathering and command dissemination (higher-level configuration), the data collection and dissemination protocols introduced by sdlib [4] was adopted.

6.1 Data Collection

The Senceive group performed also an experimented with regard to the impact of adopting a layered architecture of the timeliness (latency) of the recognized activity (context). A time diffusion [12] protocol was used to calculate the time needed to collect data from any node within the network to a remote base station that interfaced the network with a laptop computer. The experiment was conducted thus: students distributed several MICAz nodes in different rooms, at the faculty of computer science in such a way that the node depth increased with every node. Whereas some nodes directly communicated with the base station, other nodes used intermediate nodes, based on the local decision regarding the signal strength. This ensured the establishment of a link with a reliable quality. The average collection time was about 10 ms for nodes within a one-hop range; 14ms for nodes within two-hop range; 20ms for nodes within three-hop range and 46ms for nodes within four-hop range. The data collection time fluctuated from 10ms to 25ms for nodes within one-hop range; 0ms to 24ms for nodes within two-hop range; 3ms to 45ms for nodes within three-hop range; and 33ms to 64ms for nodes within four-hop range. A further


Senceive Middleware Administration Example

Mote Control

Mote(s): 3 · Leave Admin State · Send

Middleware Control

Export History Data to CSV File · Set

Network Overview

id	location	sensors	mote...	laststatus	lastcolle...	parent	nodede...	fwdqueue	linkquality	memory...	battery	micgain	routeup...	storage...	lplcycle	realTim...
3	window	MTS31...	7	2007-09-03 11:5...	21	0	1	0	165	0	8	64	5	8	10000	118881...
4	door	MTS31...	4	2007-09-03 11:5...	18	0	1	0	165	0	3000	64	5	8	10000	118881...
5	floor	MTS31...	4	2007-09-03 11:5...	0	0	1	0	124	0	2820	64	5	8	10000	118881...
6	id6	MTS31...	4	2007-09-03 11:5...	36	0	1	0	189	0	2940	64	5	8	10000	118881...

Mote Command History

timestamp	cmd	arg1	arg2
2007-09-03 11:47:31.0	108	0	0
2007-09-03 11:47:31.500	106	0	0
2007-09-03 11:47:52.296	106	0	0
2007-09-03 11:48:00.171	106	0	0
2007-09-03 11:49:00.171	106	0	0
2007-09-03 11:50:00.187	106	0	0
2007-09-03 11:50:09.703	4	24	5120
2007-09-03 11:50:42.359	4	120	1024
2007-09-03 11:51:00.187	106	0	0
2007-09-03 11:52:00.187	106	0	0

Mote Query List

id	end	query
141.76.40.126_...	n/a	SELECT accelx,accely FRO...
141.76.40.126_...	n/a	SELECT magx,magy FROM ...

Stop Query

Mote Sleep Intervals

begin	duration
22:0	240 min
12:30	60 min

Begin (hh:mm): 12:30

Duration (min): 60

Add New Interval · Remove Interval

Middleware Configuration

Auto Config (AC) ON	AC MicGain	AC RouteUpdate	AC StorageMode	AC LPL-Cycle	StatusCheck Interval	MoteActive Timeout	Memory Auto-DL Int...
1	64	5	8	10000	1	3	2048

Figure 3: An overview of the Senceive configuration service administration interface)

test starting multiple long run queries on the nodes resulted in the same collection times.

The results led to the expected conclusion that collection time is most significantly affected not due to the higher-level services but due to an increase in node depth. Only nodes within two-hop range showed unexpected behavior, usually needing a shorter time to deliver message to the base station. A probable cause for this is the time synchronization. As the implemented time synchronization protocol does only provided synchrony with a resolution of around 20ms, real collection times could be higher. The above results can nevertheless be regarded as proof of reliability. All node messages reach the base station with a relatively short delay.

6.2 Command Dissemination

Dynamic configuration of the network was supported by defining several simple commands that could directly be executed inside a node. These commands related to measurement thresholds, transmission power levels, low-power listening status, buffer size, acoustic gains and so on. By aggregating these commands, control of the behavior as well as performance of the network. The basic challenge was to successfully (and timely) propagating these commands at runtime.

The test environment for the collection performance was also used to test the reliability of command dissemination. Unfortunately the dissemination protocol could not offer any meta information about routing status as the collection protocol did. Thus it was hard to account lost packets and how many times they were retransmitted. As a result, it was not possible to determine whether the delay in command dissemination was due to retransmission of lost packets or due to the system complexity.

With regard to the impact of multi-hop communication on command dissemination, the experiment showed an expected behavior, nevertheless. Nodes with a node depth of 1 always received commands with a negligible delay and responded promptly. With a higher node depth, however, the response time increased. During testing, nodes within one-hop always responded directly, while nodes within two and three hops away usually responded with a 0 to 2 second delay and nodes within four hops away usually needed 1 to 2 seconds to respond. In some cases, the response time was significantly higher, reaching up to 4 seconds. The positive result of the test was that no command was lost.

The results showed that the dissemination protocol reliably delivered commands to the nodes as long as time span between sending two commands was large enough. The protocol did not guarantee, however, a low delivery time if multi-hop command delivery were necessary. As the number hops increased, the command delivery time reached up to the factor of ca. 100 times slower than the data collection time.

The network management subsystem was implemented in Java and so was the reasoning subsystem. mySQL was used to implement the knowledge base and the database.

6.3 Discussion

During the modeling process, frequency domain audio features (which were very useful for the recognition task) could not be extracted as expected. To recognize human speeches or speech related activities, the surrounding acoustic should be sampled at a frequency of at least 8000Hz. This is because, the human speech lies in the frequency range of 0-4KHz. This much sampling could not be supported both by the available memory size of individual nodes and by the TinyOS runtime environment. Therefore the modeling process of acoustic data was limited to time domain and stochastic feature.

Senceive was implemented and tested according to its initial description. The students successfully delivered the prototype and the documentation on time. This same system was later used in two third-party projects with minor modifications: in one of them, the simple fuzzy logic recognition was replaced by a more advanced scheme that was based on the hidden Markov models, and the wireless sensor network was complemented by wired, high sensitive microphones to capture and process acoustic signals. In the other project, additional contexts for a train setting were introduced to determine if seats in a carriage were occupied or free; and if windows and light systems were properly functioning.

The Senceive system proved to be relatively simple to setup and to configure for both settings. Users with little experience in wireless sensor networks could be able to operate and monitor it. Perhaps the simplicity and easiness in supporting multiple applications can be explained by the software complexity in the middle layers and in the management subsystem. Both the query processing service and the configuration services perform a significant computation to transform user friendly, higher-level commands and requests into lower-level system commands that can be processed by the wireless sensor nodes. Apparently, implementation of the two services was labor intensive. Such complexity can only be justified if indeed multiple applications can use the sensing system. It is also worth to note that the system is not energy efficient.

7. LESSONS LEARNED

project works prepare graduate students not only to the real world outside of their universities, but also to their immediate concern - the writing of a meaningful graduating thesis. Because they have to work in teams and keep deadlines, the challenges help them learn to efficiently organize themselves, manage time and set priorities.

Several lessons were learned during the development of the Senceive system. First, the performance or achievement of students depends on the standards set for them by their supervisors and by the clarity of the objectives of their task. This does not mean, however, that students' educational background does not play a role. On the contrary. This withstanding, a project work being a first step towards developing complex systems, most of the students lack confidence and are vague about their contribution. Setting a high standard enables them to discover their potential. Second, defining milestones was the crucial step in supporting students manage their time properly and measure progress objectively. Having said this, project works require flexibility. Everything may not be flowing as expected because individual students approach the same problem in different ways. Here the supervisor need to demonstrate a good judgment and accommodate individual differences. Third, the main objective of a project work should be less of developing an excellent, compact, and efficient system than identifying and understanding the various components of a complex system and building individual subsystems that function in concert. Finding the right balance is very challenging because of the associated time and the implication on keeping deadlines.

8. REFERENCES

[1] A. Benbasat, Paradiso, and A. Joseph. A framework for the automated generation of power-efficient classifiers for embedded sensor nodes. In *SenSys '07: Proceedings of the 5th international conference on Embedded networked sensor systems*, pages 219–232, New York, NY, USA, 2007. ACM.

[2] X. Chao, W. Dargie, and L. Guan. Energy model for h2s monitoring wireless sensor network. In *CSE '08: Proceedings of the 2008 11th IEEE International Conference on Computational Science and Engineering*, pages 402–409, Washington, DC, USA, 2008. IEEE Computer Society.

[3] K. Chintalapudi, T. Fu, J. Paek, N. Kothari, S. Rangwala, J. Caffrey, R. Govindan, E. Johnson, and S. Masri. Monitoring civil structures with a wireless sensor network. *IEEE Internet Computing*, 10(2):26–34, 2006.

[4] D. Chu, K. Lin, A. Linares, G. Nguyen, and J. M. Hellerstein. Sdlib: a sensor network data and communications library for rapid and robust application development. In *IPSN '06: Proceedings of the fifth international conference on Information processing in sensor networks*, pages 432–440, New York, NY, USA, 2006. ACM.

[5] W. Dargie, X. Chao, and M. Denko. Modelling the energy cost of a fully operational wireless sensor network. *Springer Journal of Telecommunication Systems*, 2009.

[6] W. Dargie and T. Tersch. Recognition of complex settings by aggregating atomic scenes. *IEEE Intelligent Systems*, 23(5):58–65, 2008.

[7] C. Hermann and W. Dargie. Senceive: A middleware for a wireless sensor network. In *AINA '08: Proceedings of the 22nd International Conference on Advanced Information Networking and Applications (aina 2008)*, pages 612–619, Washington, DC, USA, 2008. IEEE Computer Society.

[8] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva. Directed diffusion for wireless sensor networking. *IEEE/ACM Trans. Netw.*, 11(1):2–16, 2003.

[9] B. H. Koh and S. J. Dyke. Structural health monitoring for flexible bridge structures using correlation and sensitivity of modal data. *Comput. Struct.*, 85(3-4):117–130, 2007.

[10] K. Lorincz, B. Chen, J. Waterman, G. Werner-Allen, and M. Welsh. Resource aware programming in the pixie os. In *SenSys '08: Proceedings of the 6th ACM conference on Embedded network sensor systems*, pages 211–224, New York, NY, USA, 2008. ACM.

[11] A. Mainwaring, D. Culler, J. Polastre, R. Szewczyk, and J. Anderson. Wireless sensor networks for habitat monitoring. In *ACM International Workshop on Wireless Sensor Networks and Applications (WSNA 2002)*, pages 88–97, 2002.

[12] W. Su and I. F. Akyildiz. Time-diffusion synchronization protocol for wireless sensor networks. *IEEE/ACM Trans. Netw.*, 13(2), 2005.

[13] G. Werner-Allen, K. Lorincz, M. Welsh, O. Marcillo, J. Johnson, M. Ruiz, and J. Lees. Deploying a wireless sensor network on an active volcano. *IEEE Internet Computing*, 10(2):18–25, 2006.

[14] J. Yick, B. Mukherjee, and D. Ghosal. Wireless sensor network survey. *Comput. Netw.*, 52(12), 2008.