WIRELESS SENSOR NETWORK
(AKA WISENET)

TEAM: **D. PATNODE**
J. DUNNE

ADVISERS: **DR. O. MALINOWSKI**
DR. D. SCHERTZ

OCTOBER 30, 2002

Functional Description

Background

The technological drive for smaller devices using low power with greater functionality has created potential in the sensors and data acquisition sectors. Low-power micro-controllers integrated with RF transceivers and various digital and analog sensors allow a wireless, battery-operated network of sensor modules ("motes") to acquire a wide range of environmental data. This data can be downloaded onto a computer and stored in a database for later retrieval and analysis via a web-based application. These results can then be accessed by a standard browser from anywhere on the Internet.

The TinyOS project at UC-Berkeley (http://today.cs.berkeley.edu/tos/) has created an operating environment to address the priorities of such a sensor network (low power, robust communications). However, it is currently only developed for two or three hardware platforms. The first stage of our project involves creating a new hardware platform based on the Intel 8051 architecture. Once the platform is completed and TinyOS has been ported to it, the next stage is to use this platform to create a small-scale system of wireless networked sensors. The purpose of this system would be to monitor environmental conditions in the labs and offices in the ECE department at Bradley University.

System Description

The overall system block diagram is shown in figure 1. There are two primary subsystems Data Analysis and Data Acquisition and three major components Sensor Motes, Server, Client. Each is described below.

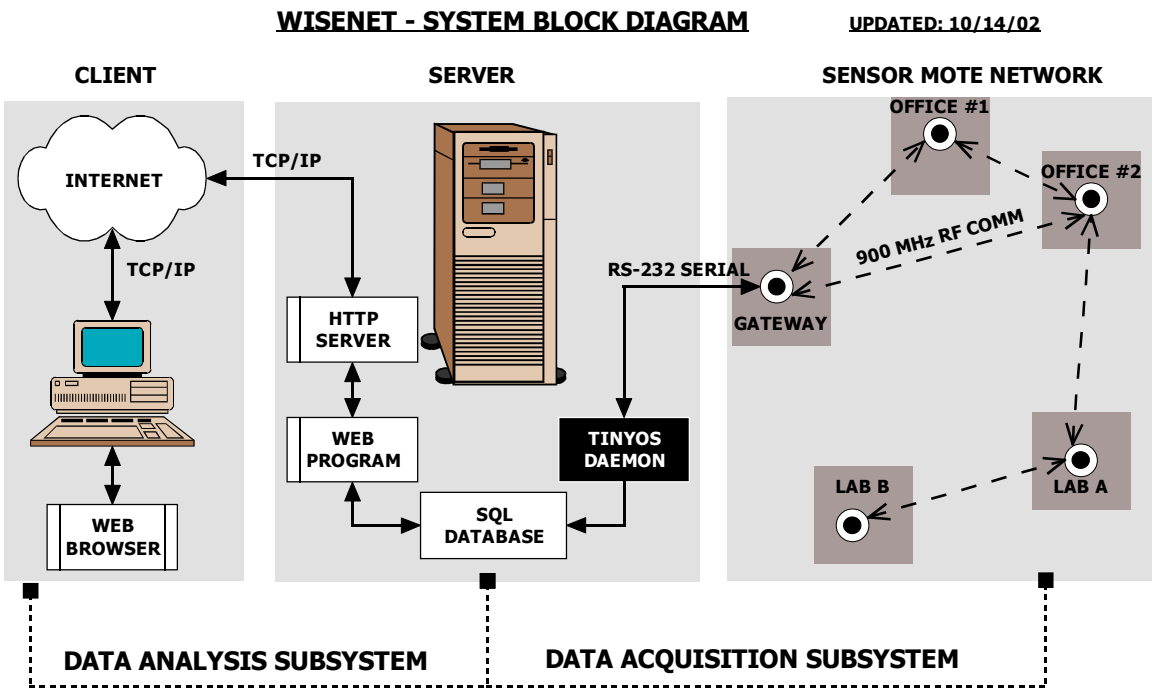

Figure 1: System Block Diagram

Primary Subsystems

There are two top-level subsystems – Data Analysis and Data Acquisition.

Data Analysis: This subsystem is software-only (from our project's perspective). It relies on existing Internet and web infrastructure (HTTP) to provide communications between the Client and Server components. The focus of this subsystem is to selectively present the raw environmental data collected by the Data Acquisition system in a meaningful manner to the end user. This might include historical graphs, averages, highs/lows, etc.

Data Acquisition: The purpose of this subsystem is to collect and store raw environmental data in a database for later processing by the data analysis system. This is a mix of both PC & embedded system software, as well as embedded system hardware. It is composed of both the Server and Sensor Mote components.

System Components

There are three primary system components: Client, Server, and Sensor Mote. Each is described below, complete with an input/output block diagram.

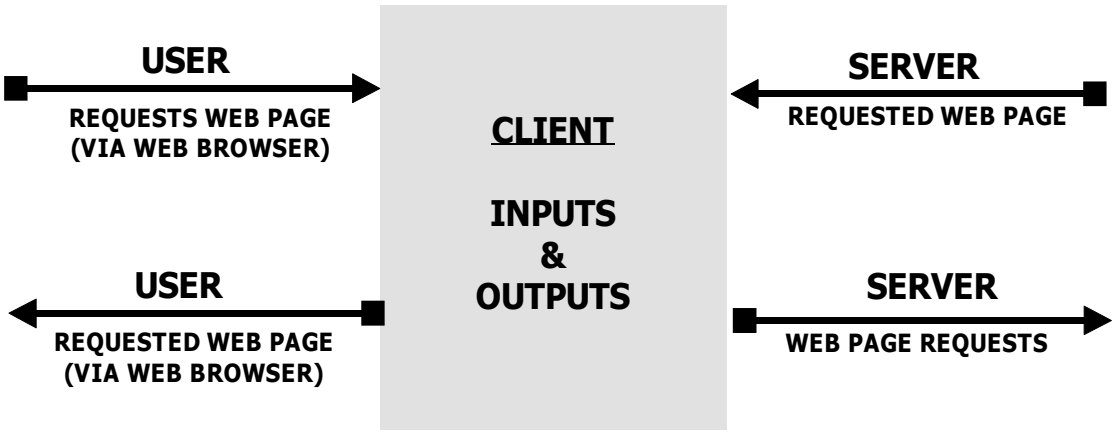

Figure 2: Client Component

Client (figure 2): From our project's perspective, the Client component is necessary but external. That is, so long as a Client (any computer with a web browser and Internet access) is available, no more work needs to be done for this component. It serves only as a user interface to the data analysis subsystem.

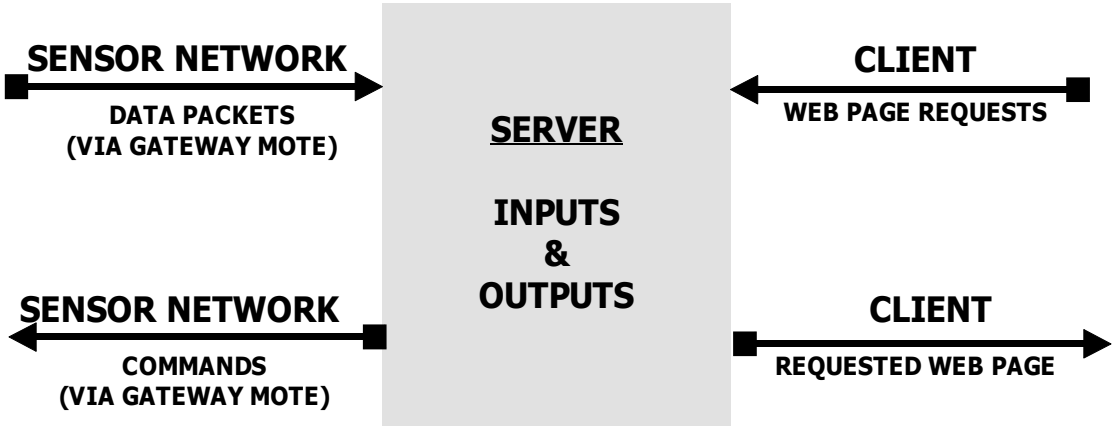

Figure 3: Server Component

Server (figure 3): This is a critical component, as it serves as the link between the Data Acquisition and Data Analysis subsystems. On the Data Analysis side is an HTTP server hosting a web application. When a page request comes in, the HTTP server calls the web application, which retrieves data from the SQL database, processes it, and returns a web-page which the HTTP server serves to the Client. For the Data Acquisition system there is a daemon running to facilitate communication with the sensor network (ie Sensor Motes). This daemon is responsible for sending commands over a RS-232 serial link to the gateway mote for transmission to the sensor network. It is also responsible for collecting data from the sensor network (again via the gateway mote.) The data collected is then deposited into the SQL database with minimal processing. Thus,

the SQL database provides the link between the Data Acquisition and Data Analysis subsystems. It should be noted that since the SQL database communicates via TCP/IP, only the HTTP server and web-program blocks (see fig. 1) need to be located on the same physical machine. The HTTP server, the SQL database, and the TinyOS daemon can all be on different physical machines connected via the Internet. From a project perspective, this would make no difference.

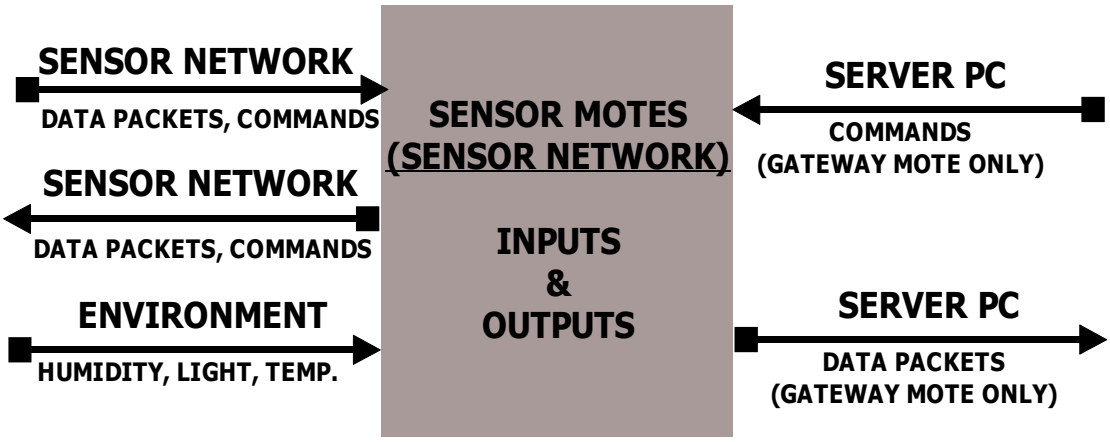

Figure 4: Sensor Mote Component

Sensor Motes (figure 4): The primary focus of our project will be developing the Sensor Mote component. It is the component responsible for collecting raw environmental data and transmitting that data to the Server. It also must receive commands from the Server (possibilities include request for data, reprogram, etc). There will be two physical implementations of this component – the first is the standard mote. The primary purpose of standard motes is to collect and transmit raw environmental data. They communicate over low-power RF links in the 900MHz ISM band and are also responsible for ensuring all data packets are received by the gateway mote. The gateway mote is the second implementation. Its purpose is to serve as the liaison between the Server and the Sensor Mote Network and deliver all data packets to the TinyOS daemon. Both implementations will have the same hardware and software; they will differ only in functionality. See figure 5 for the block diagram of a Sensor Mote. The Sensor Mote Network is simply a collection of Sensor Motes and as such would have the same inputs and outputs (minus the Sensor Network inputs and outputs, which would then become internal).

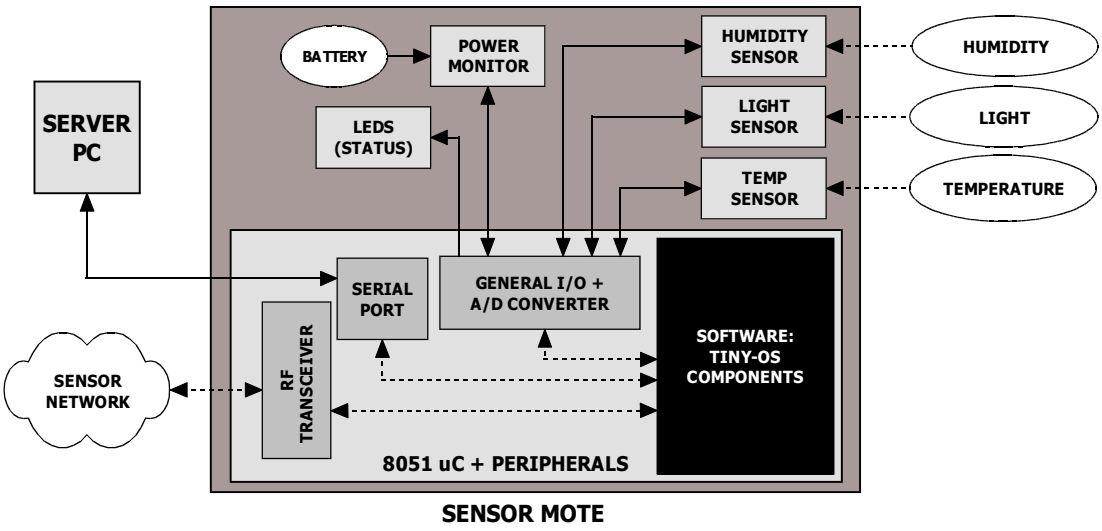

Figure 5: Sensor Mote Block Diagram

Project Goals

Primary (1):

a) Develop a Sensor Mote hardware platform based on the 8051-architecture (with an emphasis on low power consumption and ease of construction)

b) Port relevant TinyOS components to this hardware platform; develop new interfacing components as necessary

c) Port or develop PC software (the TinyOS daemon) to retrieve data from the sensor network and store it in a SQL database

d) Develop a web-application to retrieve (with minimal analysis) sensor data via the Internet

e) Test implementation with a 2-5 mote network

Secondary (2):

a) Implement existing routing component(s) of TinyOS for robust communications

b) Implement aggressive software-based power-saving functionality to maximize battery life

c) Implement power-level monitoring on sensor motes (if not included in 1.a)

Tertiary (3):

a) Analyze and optimize routing component to maximize efficiency and minimize power usage

b) Develop a full-featured web-application to analyze sensor data

c) Implement mote self-reprogramming to facilitate propagation of bug-fixes and feature updates across the sensor network

d) Analyze hardware components for possible additional power-saving