CraneTracker100 Platform Description

David Anthony, William Bennett, Mehmet C. Vuran, Matthew Dwyer, Sebastian Elbaum
Computer Science and Engineering
University of Nebraska–Lincoln
Lincoln, NE 66588-0115
{danthony, wbennett, mcvuran, dwyer, elbaum}@cse.unl.edu
http://cpn.unl.edu

1 Abstract

Monitoring and tracking wildlife over large geographic areas requires novel hardware platforms in order to fulfill the unique sensing and communication challenges posed by this task. Existing platforms fall short in several key areas. To this end, a novel hardware platform, the *CraneTracker100*, is developed using off-the-shelf components. This device uses state of the art communication and sensing techniques. This platform uses commercially developed components, and incorporates multiple sensors and communication mechanisms. The system software utilizes open source components and is based on the popular TinyOS operating system. This combination of hardware and software enables the low-latency collection of a large amount of data. This data is accessible through an HTML5 front-end that is powered by a web-service back-end.

Contents

2 Introduction

The *CraneTracker100* is a hardware platform designed to support migratory bird tracking missions. Construction of the *CraneTracker100* is motivated by a lack of commercially available platforms that are capable of collecting the desired type of data on migratory bird behavior, and the capability to maintain communications over a continental scale.

The two primary goals of the project are to track the migratory paths of cranes and to characterize their behaviors and movements. Characterization of their movements consists of measuring their motion while on the ground and in flight. From these measurements, ecologically interesting questions can be answered, such as how frequently the birds are feeding. Another example of an interesting behavior occurs when the birds are flying. It is highly desirable to determine whether the birds are flying by flapping their wings, or through gliding. Answering these questions will help determine the food needs of the birds, the suitability of different habitats, and where the birds are likely to nest at.

Early in the development of the crane tracking project, an MTS-420 sensorboard from Memsic was used as the base hardware platform in conjunction with an Iris mote [9]. However, early testing revealed that this platform possessed deficiencies that made it unsuitable for the tracking missions. First and foremost, the ZigBee radio on the Iris lacked the transmission range to maintain connectivity while deployed on wild birds. Second, the sensors were incapable of collecting the desired data, such as flight characteristics. The accelerometer on the MTS-420 only operates in two dimensions. This means while the bird is in flight and the device orientation is unknown, the device is ineffective at characterizing a bird's motions. The GPS receiver is an obsolete design that requires a bulky and heavy external antenna that is unsuitable for this task. Finally, the Iris and MTS-420 are powered through AA batteries, which lack the storage capacity to power the device throughout the multi-year missions envisioned for this platform.

To remedy these shortcomings, the *CraneTracker100* is developed. This new sensor platform includes better sensing, communication, and power capabilities. The combination of these enhanced capabilities will make it possible to track migratory birds on a continental scale.

3 Hardware Platform

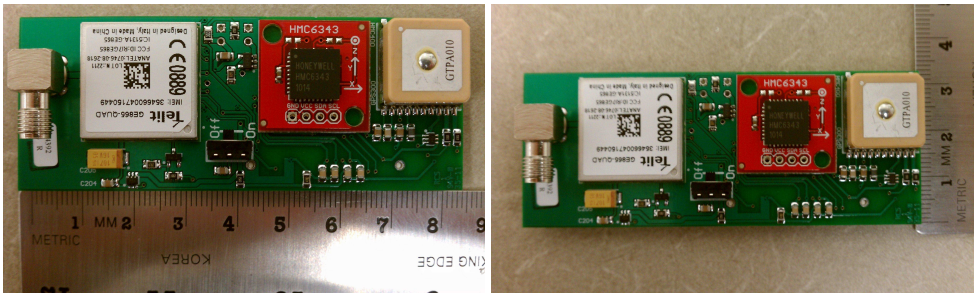
The *CraneTracker100* was designed with the goal of utilizing commercial, off-the-shelf (COTS) components to create a platform capable of tracking migratory birds on a continental scale. COTS components were used for two reasons. The primary reason was to reduce the cost of the platform by avoiding expensive custom-designed hardware. The second reason was to minimize the development time of the platform.

The *CraneTracker100* design can be decomposed into four subsystems. They are the mote, power, sensing, and communication systems. A brief description of these components and their capabilities will now be presented. The final result is shown in Fig. 1.

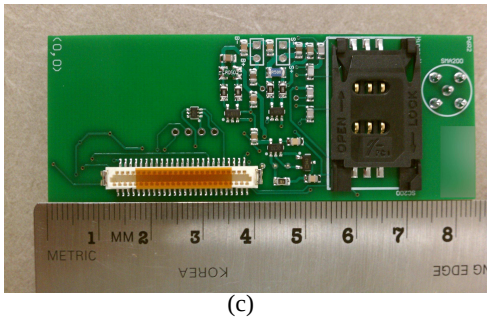
3.1 Mote

A commercial WSN mote provides the basic processing and storage mechanisms for the device. An Iris mote by Memsic [9] was chosen for this task. The Iris is consists of an Atmel 1281 microcontroller, RF230 radio, and AT-45 flash memory. This combination provides low-energy processing and communications, with 512kB of non-volatile storage. This mote can be interfaced to a variety of sensorboards through the 51 pin connector [1].

This mote was originally chosen during development for use with the MTS-420 [9] sensorboard. While the MTS-420 was abandoned because of shortcomings with its communication and sensing capabilities, the Iris was retained. Much of the software developed for the MTS-420 in TinyOS was retained for use with the *CraneTracker100* , which allowed for the rapid development of the *CraneTracker100* .



(a) (b)



(c)

Figure 1: (a), (b) Top of board, (c) Board bottom

3.2 Power

While most most commercial motes and sensorboards by [9] are designed to operate using AA batteries and 3.3V voltage levels, this is insufficient for the *CraneTracker100* . First, the GSM modem requires a nominal 3.6V source to operate. Second, NiCad, NiMH, and alkaline batteries lack the required energy density for this application. Third, these types of batteries have undesirable battery discharge characteristics as the voltage provided by the batteries rapidly decreases as the battery discharges. To achieve the desired voltage level and energy density, lithium polymer batteries are used instead.

Lithium polymer batteries complicate the power supply of the system. The single cell lithium polymer batteries used by the system can supply up to 4.2V at peak charge. This voltage level exceeds the maximum operating level of many of the system's components, including the microcontroller. Even with a lithium polymer battery's energy density, the weight limitations of the mission mean that the battery will not be able to power the device throughout the entire mission duration. Thus, renewable energy in the form of a flexible solar panel [2] is included on the device. This solar panel's specifications state it is capable of providing 50mA of current at 4.8V. The panel's flexibility enable it to be easily incorporated into many mechanical designs, and could even be wrapped around a leg-mount device in the future. Furthermore, lithium polymer batteries require dedicated charge management mechanisms to prevent over- and under-discharging of the battery. This protection is provided through a combination of dedicated integrated circuits [11, 12]. The final system power supply is shown in Fig. 2.

The power supply is enhanced with the capability to monitor the input voltage and current of the solar panel, and the battery voltage and discharge current. These values are monitoring using the ADC ports on the microcontroller. Since the operating voltage range of the battery and solar panel exceed the regulated supply voltage, it is necessary to use voltage dividers to reduce the ranges to values that are measurable by the microcontroller. Op amps designed for current sensing [8] are used to measure the solar panel input current and battery discharge currents. Combined, these capabilities allow for detailed *in-situ* energy profiling of the system. These capabilities are very important to the system, since it is not practical to monitor the system

using external equipment in the field.

Fig. 2 shows the switches [10] used to control the power to the individual components. These switches are operated via general purpose IO lines on the microcontroller. These switches consume very little energy, and help limit the energy consumption of the system by selectively enabling and disabling components. The switches also provide a fail-safe method of turning off a component in case it is malfunctioning.

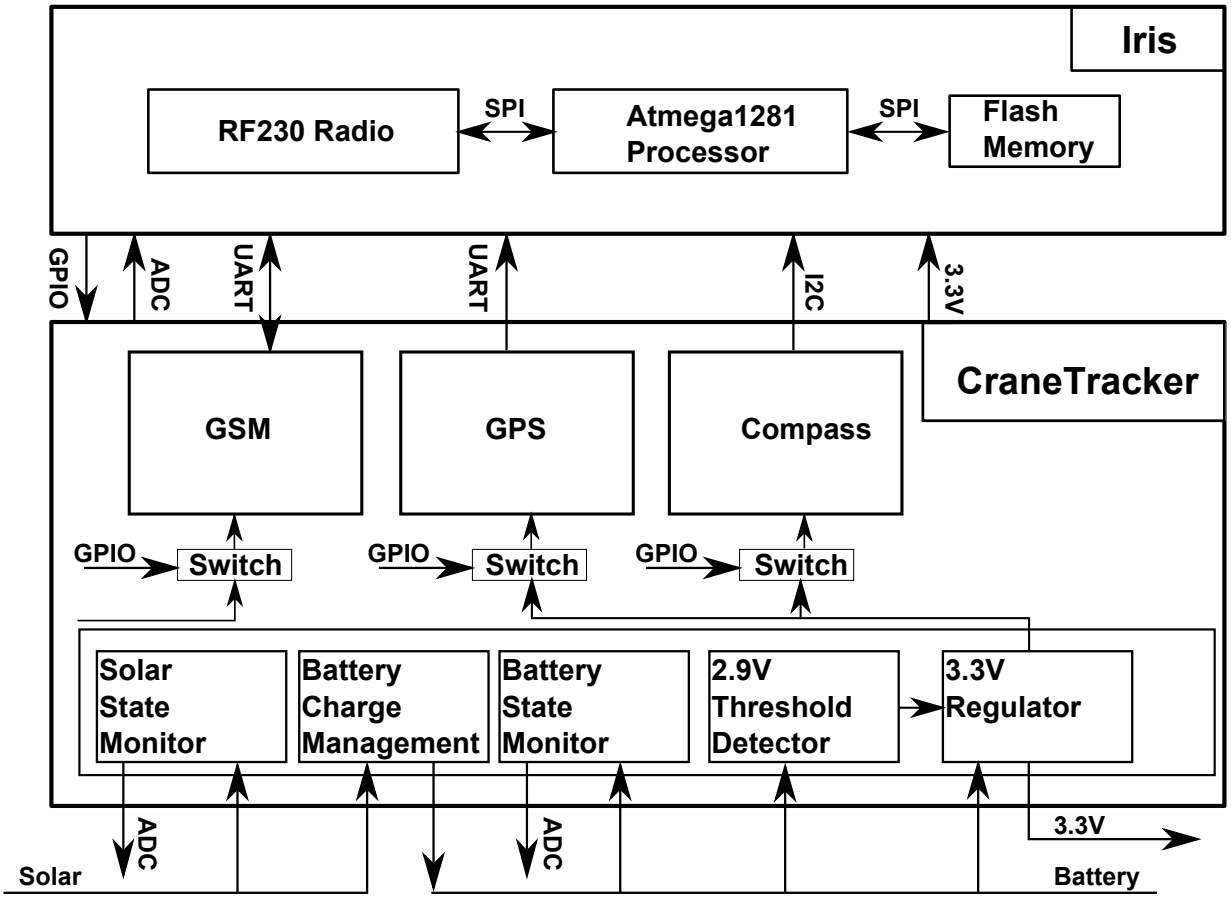


Figure 2: Power Supply and Communication Interfaces

3.3 Sensors

The *CraneTracker100* is designed to accurately monitor and track migratory birds at all stages of their life. In order to do this, a sophisticated set of sensors was added to the *CraneTracker100* . The first of these devices is a GPS receiver [5]. The second is a solid state compass [7]. In addition, the GSM can also measure certain environmental data. A summary of the sensors is given in Table 1.

The GPS sensor communicates to the host microcontroller over UART lines using the NMEA 0183 protocol. This is a widely used protocol that is also employed by the MTS-420. This commonality enabled the platform to re-use much of the code developed for the MTS-420, but enhanced to understand more NMEA 0183 sentences. This particular receiver incorporates a patch antenna into the same physical package as the receiver, which makes it very compact. Empirical tests have also shown that the receiver is very sensitive, and quickly acquires fixes when deployed on wild birds.

The GPS measures several important aspects of crane behavior. Most importantly, it provides the location of a bird. This knowledge can enhance conservation efforts by identifying critical areas of habitat for the birds. The GPS is also capable of measuring the speed of the bird. This information is used to describe the flight characteristics. Further characterization of the flight is given by the altitude provided by the GPS. Finally, the GPS can measure the course over ground of the bird, which can be used to determine where the bird is traveling.

To augment the data collection, the HMC6343 solid state compass is used to collect information on the birds' movement and flight characteristics [7]. The HMC6343 collects three dimensional pitch, roll, and heading information. In addition, the compass incorporates a three dimensional accelerometer that is used for characterizing the birds' movements. Finally, the compass includes a temperature sensor for estimating the ambient air temperature. The HMC6343 is connected to the microcontroller through I2C lines. The commands that can be sent to and received by the compass are defined in its datasheet.

The GE865 GSM module also functions as a sensor in this design [13]. When it sends information, the ID of the cellular tower it is associated with is sent along with the other information. The location of the tower can then be used to identify the area in which the information was transmitted from. While this location information is not nearly as accurate as the data obtained via the GPS, it can still be used as a redundant source of location information in case the GPS receiver is damaged or is otherwise unable to obtain its location.

Device	Interface	Communication Protocol	Data
GPS	UART	NMEA 0183	Location, Altitude, Speed, Course Over Ground
Compass	I2C	Defined in datasheet	Heading, Pitch, Roll, 3D Acceleration, Temperature
GSM	UART	AT Commands	Location

Table 1: Sensor Capabilities

3.4 Communications

The ZigBee compatible radio on the Iris is not capable of meeting the mission requirements by itself. The cranes cover too large of a geographic area, and are too unpredictable in their flight paths, for ZigBee communications to be maintained. Thus, a GE865 GSM module [13] is used to maintain connectivity when the birds have traveled outside of areas covered by ZigBee basestations. The GE865 interfaces to the microcontroller through UART lines. The module can be controlled using the standardized AT command set. The specific commands that the unit supports are defined in the module's datasheet. An SMA connector is used to attach the module to an external antenna.

4 Device Software

4.1 TinyOS

TinyOS is a free and open source operating system designed for wireless sensor networks (WSN). TinyOS and applications that it execute are programmed with nesC programming language. TinyOS is used extensively for an embedded operating system by researchers and industry.

nesC is a dialect of the C programming language. The language was designed specifically for TinyOS and is component based and event driven. Additionally, it is highly optimized for memory constrained

devices. Interaction between components is controlled through interfaces. These interfaces act as a specification that describe the behaviors of the component. With this method implementation details of these components are hidden from external components.

4.1.1 Contributions

To facilitate the development of the tracking platform, some additions to the TinyOS operating system are made. These changes include the I2C hardware implementation for Atmel 128X/246X chips and standard library time implementation.

I^2C, also known as two-wire interface (TWI), is used by integrated circuit manufacturers to communicate with peripheral devices. For the CraneTracker platform, TWI is used to communicate with the HMC-6343 compass. The default TinyOS TWI implementation is implemented in software and utilizes bit banging. This approach was unable to meet the timing requirements of the compass. This problem made the communication between the compass and microcontroller unreliable. To fix this problem, a hardware driven approach was used in a new driver. This driver can be used on both the MicaZ and Iris motes.

TinyOS also lacks many of the basic data and time manipulation functions that are available on other platforms. This system makes heavy use of timing information from the GPS to schedule events at specific intervals, and to support system monitoring. Therefore, a set of functions were created to convert GPS time to the standard Unix representation [4]. In addition to this, the portions of TinyOS that handle the hardware timer firings were changed to maintain a free running clock. This clock is used track how long the system has been operating. This clock is synchronized to the correct world time through the GPS when a fix is acquired.

4.2 Device Drivers

The new sensors and communication devices of this platform required the development of new drivers. In the following section, we will discuss the details of these drivers.

4.2.1 GSM Driver

Support the GSM module posed several challenges. First, the communication between the host processor and GSM is stateful. Communication between these two must be carefully arbitrated, and error conditions must be detected and gracefully handled. To support application development, the details of associating with the cellular network and sending SMS messages are abstracted behind carefully defined interfaces. The details of the GSM control are then hidden from the higher level application layers. This architecture is shown in Fig. 3.

Power to the modem is controlled through a digital switch. This switch is controlled through the general purpose IO lines of the microntroller. After the GSM is powered on, a sequence of commands is sent to assess the state of the GSM and control its operation. These commands are sent to the device over the UART interface supplied by TinyOS. These commands are in the form of standard AT commands [3]. The GsmUartHandler receives the response from the modem, and checks for valid messages. These messages are passed to the GSMDriver module that interprets the responses. Components that use the GSMDriver module are notified of important information, such as network state changes and the status of message transmissions.

The driver also implements fault tolerant features. These features include a watchdog timer that removes power from the GSM module in case it becomes unresponsive, or enters a faulty state the system is unable to recover from. This feature prevents unnecessary power from being wasted when faults occur. GSMDriver also checks the message passed from GsmUartHandler for any error messages. In the case of device or transmission errors, the GSMDriver will attempt to resend any messages.

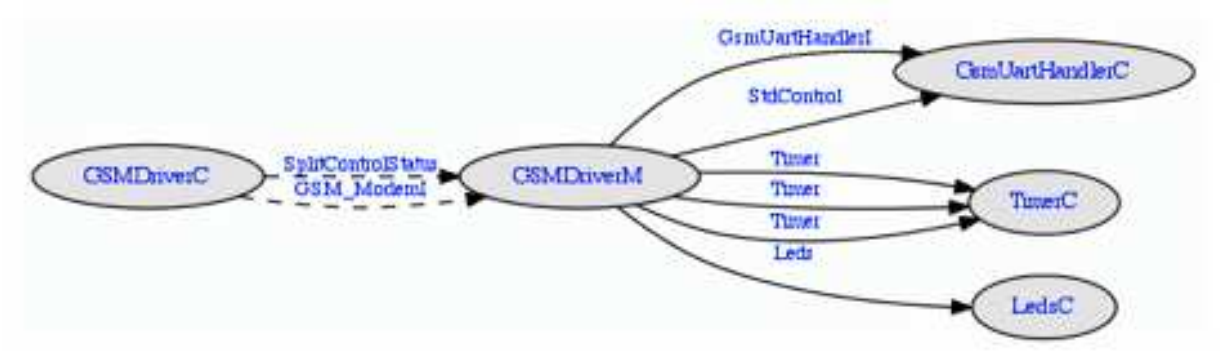


Figure 3: GSM Driver Component Graph

4.2.2 GPS Driver

The GPS driver of the system is based on the driver provided by MoteWorks for the MTS-420 sensorboard. The GPS on the MTS-420 uses the NMEA 0183 communication protocol to communicate with motes over the UART lines. Fortunately, this protocol is widely adopted, and is used by the GPS on the *CraneTracker100* [6]. Although the NMEA 0183 protocol defines many messages, or sentences, that are used by GPSes to send data, the MoteWorks driver only supports the RMC sentence. This sentences contains location, time, fix validity, and other information. For the *CraneTracker100* project, support for the GGA sentence was implemented. This sentences gives the system the altitude. The driver was also modified to only report information when a valid fix was obtained, rather than reporting all data and relying on the higher layers to check for validity. The driver was also extended so that the system time kept by the mote is synchronized with the correct world time provided by the GPS. The finished architecture is shown in Fig. 4.

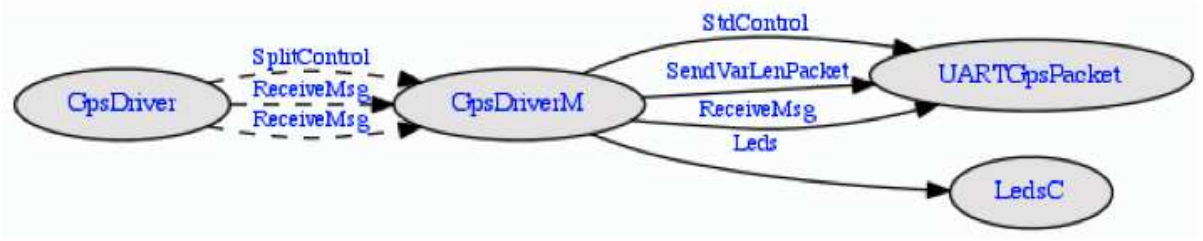


Figure 4: Gps Driver Component Graph

4.2.3 HMC6343 Driver

A driver for the HMC6343 compass was developed for TinyOS. This driver supports reading the heading, pitch, roll, three dimensional acceleration, and temperature from the device. The compass driver utilizes the contributed TWI implementation for TinyOS. The component graph for the compass driver and TWI is shown in Fig. 5 and 6, respectively.

The HMC6343 component controls power to the compass device. The component also contains the process for translating high level requests for data into the I2C commands that the HMC6343 understands. As with the GSM, the driver implements a watchdog timer. This watchdog timer ensures that faulty states or communication errors will not result in the device being left on and draining power unnecessarily.

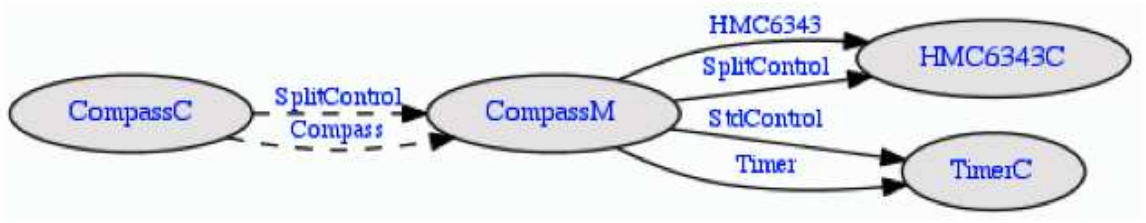


Figure 5: HMC6343 Driver Component Graph

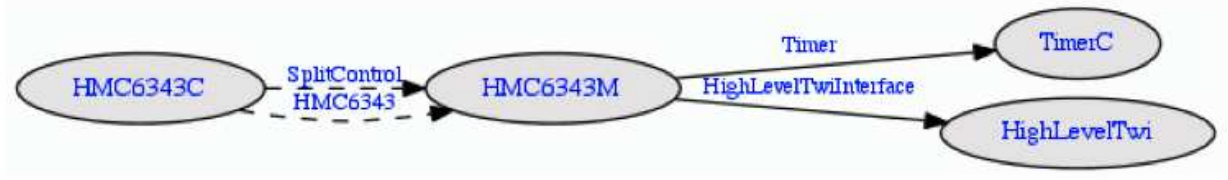


Figure 6: Communication Layer of Compass Driver Component Graph

The CompassC component provides the high level interfaces to turn the compass on and off, enter and exit calibration mode, and fetch data from the compass. These high level interfaces hide the details of the underlying device from higher level software. The component also controls the reporting frequency and power states the compass can utilize.

4.2.4 Power Monitoring Driver

A high level driver was implemented to support sampling the ADC lines that are connected to the power monitoring circuitry. In the case of the voltage, the driver also converts the raw ADC numbers into a standard voltage reading. This conversion makes developing the higher level software simpler, and hides the underlying ADC implementation from the high level software. This driver is shown in Fig. 7.

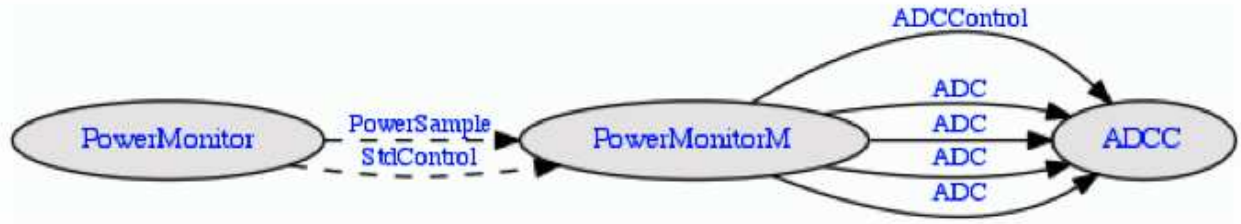


Figure 7: Power Monitoring Driver Component Graph

4.3 Sensing Application

The default program for the sensing platform is the Crane Manager. The crane manager acts as a controller that utilizes all of the features of the platform into a sensing and reporting application. The Crane Manager duty-cycles the sensing and reporting operations. In this operation, sensor readings are stored using the Crane Storage component, and communicated using the crane radio and GSM Manager.

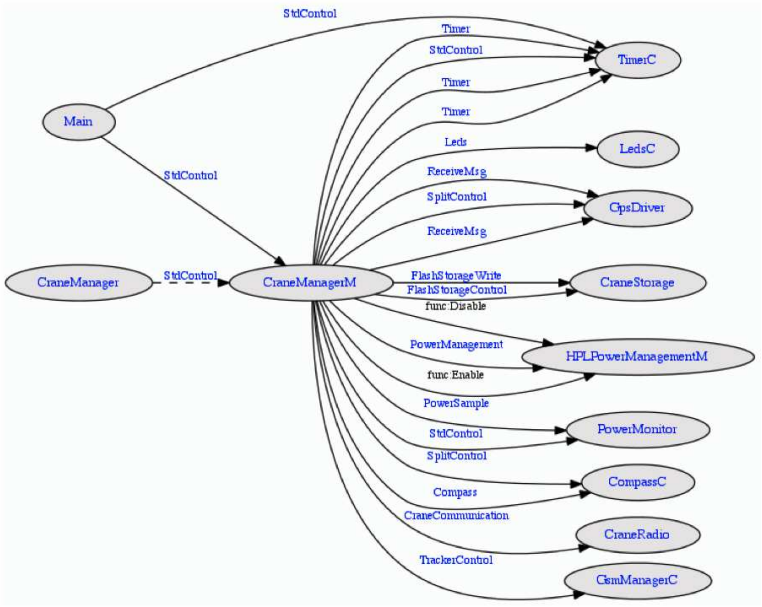


Figure 8: Crane Manager Component Graph

When the manager activates it will check power levels then sample the available sensors, store the samples, listen for a base-station beacon, and if unsuccessful will try to use the long-range GSM radio. Upon completion of these tasks, the Crane Manager will put the device into deep-sleep to conserve energy.

The manager uses power monitoring information to determine the health of the system. For this, the Crane Manager uses compile-time thresholds empirically determined through experimentation. In this process, the software goes through two software checks motivated to extend the life-time of the mission. First, upon startup or deep-sleep wake-up, the manager determines if the voltage is at a critical state. If so, the device will return to a deep-sleep in hopes for recharge from the solar panel. Secondly, upon voltage check, if the device reads above the safe-voltage to communicate, it attempts to operate in the non-critical mode performing the tasks described previously. It was decided to only enable the sensing components at the minimum GSM voltage level, which is 100mV different then the GPS operating voltage.

4.3.1 Crane Storage

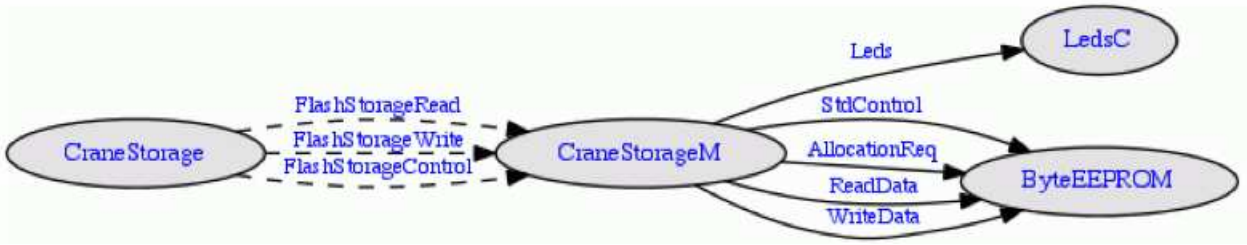


Figure 9: Crane Storage Component Graph

All sensor readings are stored in non-volatile memory. The flash memory components provided by TinyOS, are controlled through the Crane Storage component. The Crane Storage component makes use of a circular-buffer to sustain readings from mission situations where communication is unavailable for

multi-month durations. To facilitate this desired longevity, all accounting and storage states are persisted every commit. Therefore upon energy depletion, storage state can be restored allowing the mote to operate normally when energy resources are available.

4.3.2 Crane Radio and GSM Manager

High level components are developed to control the operation of the radio and GSM. The Crane Radio component is shown in Fig. 10. This component handles the details of searching for, and communicating with, a base station. The component also handles fetching data from the storage component and formatting it for transmission. If the component is unable to find a base station, it attempts to use the GSM functionality to send the data. This is done through the GSM Manager component. This component also interfaces to the storage device to retrieve teh unsent messages. This component is shown in Fig. 11. This component manages the GSM connection to limit the time the GSM is used, to avoid overdepleting the battery.

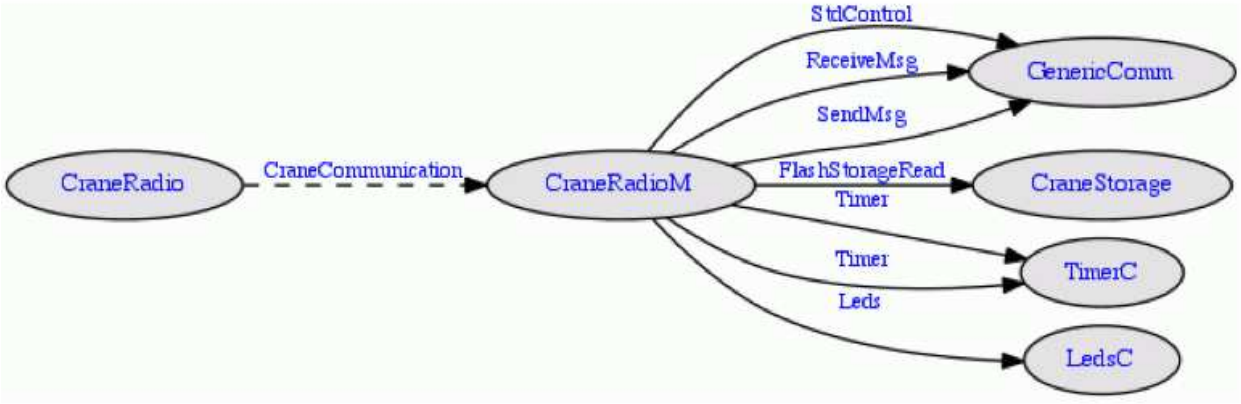


Figure 10: Crane Radio Component Graph

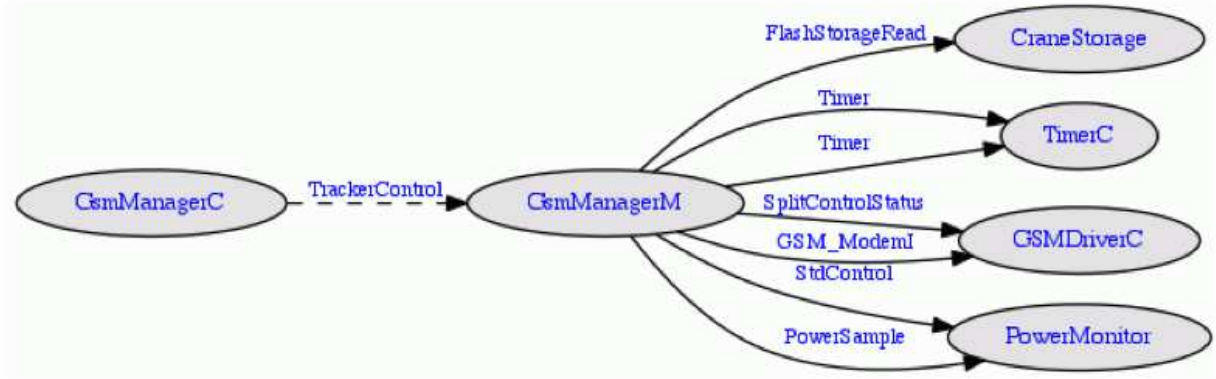


Figure 11: GSM Manager Component Graph

5 Back-End and Front-End Software

The back-end component of the system is required to enable access to sensor readings and mote statuses. A front-end is essential to display the information in a meaningful and easy to understand manner. A depiction of the back-end and front-end of the developed system is shown in Fig. 12. In the system, the back-end is composed of a gateway decoder, parser, and web-service. Lastly, the front-end is a web-enabled GUI to access and view collected data to the user.

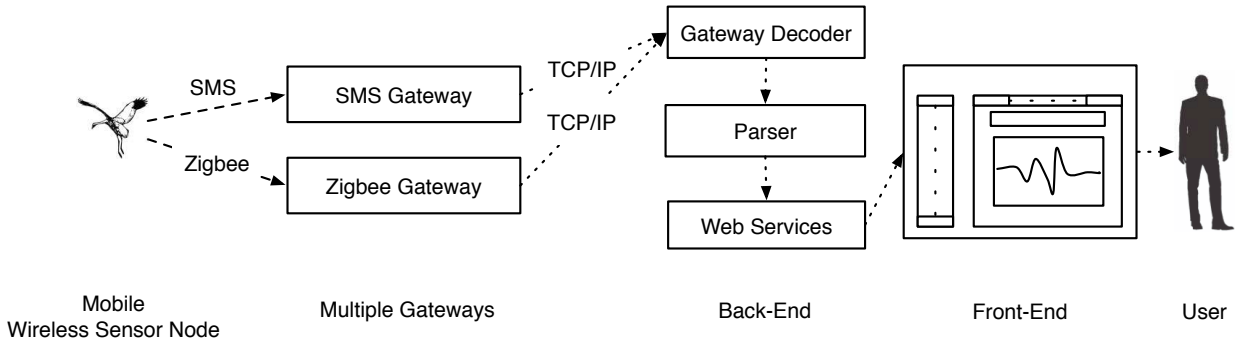


Figure 12: Back-End and Front-End Depiction

5.1 Gateway Decoder and Parser

The gateway decoder manages communication with the SMS gateway service. In addition, it manages communication with base station motes through TCP/IP. Upon receiving messages from an endpoint, the decoder scans data into the appropriate tokens to be handled by the parsing component. The parser analyzes the tokens generated by the decoder, then builds the corresponding data structure that is transmitted to the web service to be permanently stored.

5.2 Web Services

The web services enable syntactically correct data to be committed to storage. The web services are accessible in the form of an REST API. Through these methods, the information can by shared with external parties and consumed by a front-end to visualize the information received.

5.3 Front-End

A visualization client is developed to display the data received from the motes. A screen shot of the client is shown in Fig. 13, which consists of three panes. The node selection pane allows the user to select nodes on the network. The view selection pane enables users to select between different views such as chart view, GPS view, and export view, the results of which are shown in the selected view container. The chart view allows the user to select a sensor and display the collected data via a variety of graphs. The GPS view (selected in Fig. 13 illustrates the location of the mobile nodes. Finally, this information can be exported in other formats (i.e comma-separated valeus) to be analyzed with other software.

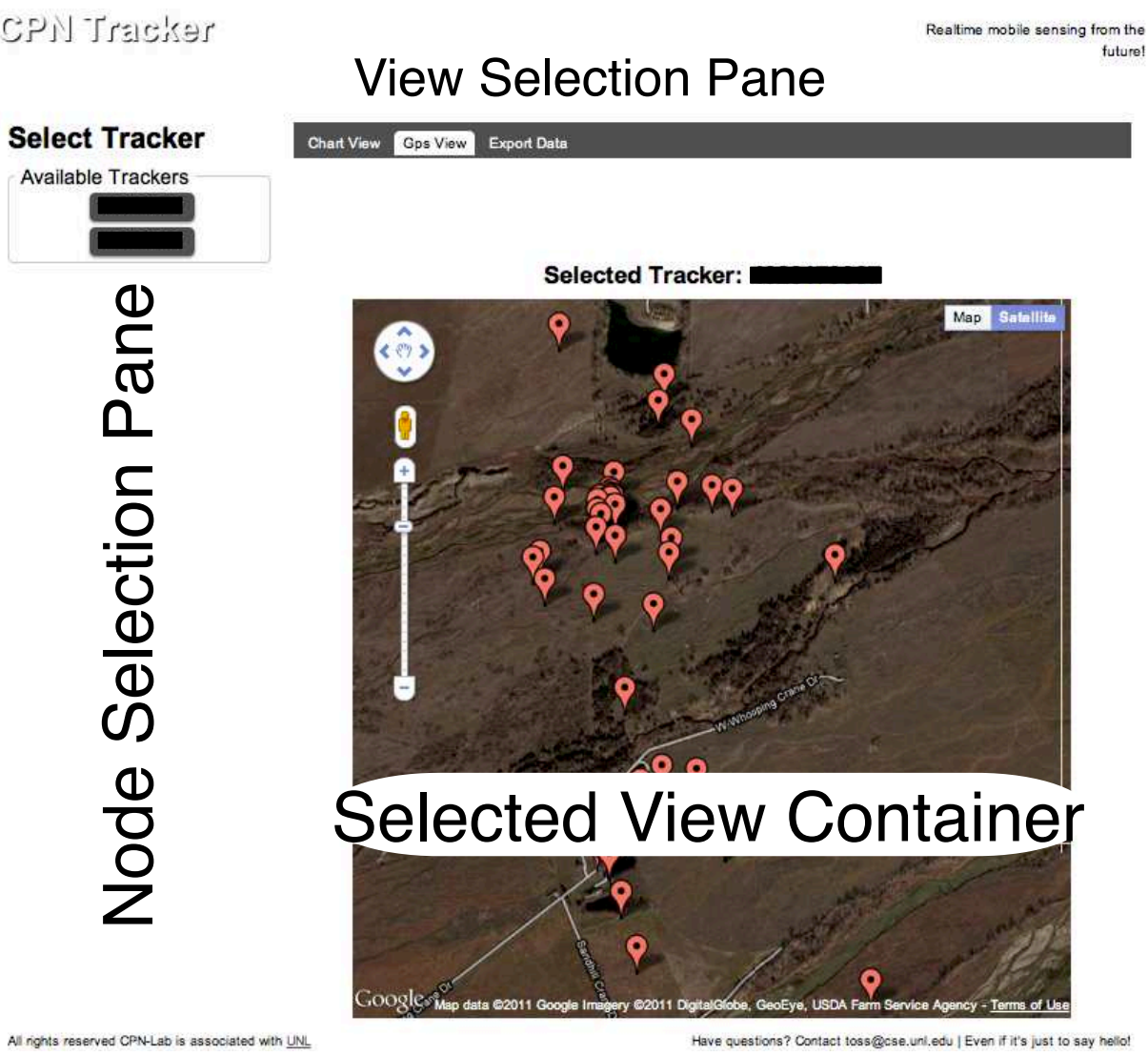


Figure 13: User Interface Depiction

6 Conclusion

The *CraneTracker100* represents a major advancement in tracking migratory birds. Its communication capabilities are able to maintain connectivity where traditional short range radios used in WSNs fail. The integrated sensors are able to characterize bird movements and behaviors, which creates novel research opportunities for wildlife studies. Multiple communication methods and redundant sensing capabilities increase the reliability of the system by providing functionality in the face of component failures. These extensive capabilities come at a reasonable price, as widely available commercial components were used in the design.

CraneTracker100 will enable future wildlife monitoring missions, and will create novel research opportunities. These missions will be undertaken in conjunction with conservationists working with migratory birds, and should yield interesting results.

References

[1] Hirose Electric Group. http://www.hirose.com, feb. 2012.

[2] PowerFilm Solar. http://http://www.powerfilmsolar.com/, feb. 2012.

[3] 3GPP. *AT command set for User Equipment (UE)*, 10.6.0 edition, dec 2011.

[4] Technical Committee. Wgn14/n1124 commitee draft. online.

[5] GlobalTop Tech Inc., http://www.gtop-tech.com. *FGPMMOPA6B*, feb. 2012.

[6] GlobalTop Technology, http://www.gtop-tech.com. *PA6B*, Feb. 2012.

[7] Honeywell. HMC6343 3-Axis Compass with Algorithms. http://www.honeywell.com, feb. 2012.

[8] Maxim Integrated Products, Inc., http://http://www.maxim-ic.com. *MAX9938*, feb. 2012.

[9] Memsic Corporation, http://www.memsic.com/. *Memsic Inc.*, feb. 2012.

[10] Micrel, Inc., http://www.micrel.com. *MIC94060*, feb. 2012.

[11] Inc. Microchip Technology. *MCP73831 Datasheet*.

[12] Inc. Microchip Technology. *TC54 Datasheet*.

[13] Telit Wireless Solutions, http://http://www.telit.com/. *GE865*, feb. 2012.