WIMEA-ICT RC3

EVALUATION REPORT ON BERGEN PROTOTYPE

19/09/2015

METADATA

Document Name	2015-09-19-Evaluation Report on Bergen prototype
Authors	Maximus Byamukama Emmanuel Kondela Mary Nsabagwa
Institution	Makerere University
Number of Pages	22
Version	2.0
Date of Submission	19/09/2015

DOCUMENT APPROVAL

Name	Role	Signature
Mary Nsabagwa	Security, Data integrity and Sensor Shielding	
Emmanuel Kondela	Topology, Needs and Requirements	
Maximus Byamukama	Gateway and Uplinks	

TABLE OF CONTENTS

Abstract

Research Component 3 (RC3) in the WIMEA-ICT project has its primary task as the design of an affordable automatic weather station (AWS), arranging for manufacture and the mass deployment of 70 such units in several areas around East Africa.
A draft version of such an AWS, called the Bergen Prototype, has so far been put together by some of the RC3 advisors as a baseline design and has been operational for some months now. The PhD students on this team, who are the authors of this document, are reviewing this prototype to determine what could be changed to make the design more optimal in several areas such as software, hardware, power consumption and communication.

This document presents the issues found in our analysis and various recommendations that will be discussed and agreed upon to make way for the second generation prototype, which we shall call the East African prototype.

Section 1 begins with an introduction on how each evaluation was done and the general structure of the report. It also gives an architectural and functional overview of the Bergen prototype. Section 2 contains draft needs and requirements specifications according to RC3. Section 3 discusses the evaluation of the topology in use and recommendations. Section 4 discusses the gateway and gives recommendations. Section 5 discusses a few issues that could merit concern in the general design and some matters arising.

1. Introduction

1.1 Structure of the Report

In each section, we will look at the technology in use in the Bergen prototype, compare with alternatives using manufacturer based data, our user experiences and the experiences of our advisors.

In some sections, we recommend benchmarking experiments while others may give straight up conclusions and recommendations, either because the justifications are straightforward from available literatureor simply because the alternatives will not provide much improvement in cost or performance or other parameters in the remaining time.

1.2 Overview of Bergen Prototype

The Bergen prototype consists of a network of wireless sensor nodes (they may be also be called *motes* in this document) each measuring some environmental parameter and transmitting it to a central gateway. This gateway itself has a receiving node, similar to the others, connected to it, called the sink node. The data is stored at the gateway in a file.
The actual components are discussed below in some detail.

1.2.1 Sensor Network

The wireless sensor nodes are small boards with an ATMEGA128RFA1 microcontroller and are classified into the sink node, the 10m node, the 2m node and the ground node. Each node type is really the same device except that it has its own name and address in the network and may have different sensors installed on it than the rest. The parameters measured are temperature, humidity, wind speed, wind direction, solar insolation, due point among others. Some of these parameters, such as temperature, are measured by more than one node. Other non-environmental information sent to the gateway includes power supply voltages and signal strength values.

The microcontroller already contains internal RF circuitry and communication is done using the IEEE 802.15.4 protocol in the 2.4 GHz range.
The frames sent to the gateway are tagged with an acronym of the parameter they represent. A part of a sample frame looks as shown below.

```
TXT=10m E64=fcc23d00000020c8 PS=1 V_MCU=3.60 P0_T=0.000 V_A1=0.41 V_A2=3.31
```

In this example, TXT shows that it is a 10m node, E64 is its MAC address, V_MCU is the microcontroller supply voltage and P0_T, V_A1 and V_A2 are readings from the wind speed and direction sensors. The whole frame contains more information such as location, time, time zone as well as RF specific information such as signal strength and sequence IDs and may be over 200 bytes long [18]. A standard IEEE 802.15.4 frame is 127 bytes.
The sensors communicate with the gateway using a star topology. Relaying of data is also possible.

1.2.2 Gateway

The gateway in use currently is a Raspberry Pi B+(called the *Pi* in this document). The Pi runs the RaspbianOperating system (a derivative ofDebian Linux). The gateway has a sink nodeconnected to it via a serial interface, and a gateway daemon, called sensd, reads incoming frames from the sink node. These frames are saved on the file system as human readable ASCII data in one file.

The data in this file can be accessedover TCP using Ethernet. The data can be visualized in a graphical form on a webpage hosted by a web server on the Pi.

1.2.3 Power Supply

Each node is powered directly by supercapacitor batteries that are charged from solar panels through DC-DC converters. The microcontroller monitors the capacitor voltage and goes into a low power sleep state when this voltage reduces to about 2.4V volts, to prevent breakdown. The maximum capacitor voltage is 3.6 volts and this is set by the DC-DC converter.

Currently, the Pi is powered directly from its USB port usinga DC adapter connected to mains electricity.

2. AWS Needs and Requirements

2.1 Introduction

Before a detailed evaluation of the Bergen prototype, it is essential that we discuss the Needs and Requirements of an automatic weather station in general, from an RC3 perspective. These basically refer to the functions that the device must be able to perform (functional) and the criteria that can be used to judge the performanceof the whole system to determine its acceptability (non-functional).

The important components of this section are the goal of implementation, functional requirements, and the non-functional requirements.

2.2 Goal of Implementation

The system should be configured as a remote stand-alone automatic weather station. Data collection, processing, and transmission should be done according to methods agreed upon by the stakeholders. Because of a future possibility of opening the weather data to the public, it is desirable that all procedures and equipment be as close to WMO specifications as possible. It should however be noted strongly that strict adherence to WMO guidelines is not the objective of the concerned research components in the WIMEA project.

2.3 Functional Requirements

Functional requirements specify exactly what the station will do. The following functional requirements apply

- **Measurement:** The station will capture data from its various sensors at specified time intervals as agreed by the stakeholders. This time interval should be configurable for a given station.
- **Transmission:**The station will transmit data from its location to a specified remote location at time intervals agreed upon by the stakeholders.
- **Storage:** The station will store captured data locally for a specified period of time. The station will have a common interface for retrieving this data manually as a worst case scenario.
- **Diagnostics**: The system will transmit the status of sensors, computing and communication devices. The system should have a facility to reset, re-flash or upgrade the firmware.

2.3 Non-Functional Requirements

Non-functional requirements are those that cannot be expressed as functions and are used to judge the operation of the system. These include Reliability, Scalability, Low power consumption, Accuracy of results, Maintainability, Cost, Security, Data Integrity, Recoverability and others. They are desirable attributes of the system are treated separately for the topology in section 3.1 and for the gateway in section 4.2.

3. Topology

3.1 Introduction

By topology, we refer to the use of a centralized or distributed system in the internal network of the AWS. In a centralized system, the sensors are controlled directly by the gateway processor and the system relies on one clock while in a distributed system each sensor has its own processor unit and clock and independently performs its functions. In each of these systems, the link to the gateway may be wired or wireless.

Centralized wired systems are in use in several AWS across East Africa according to the surveys that were undertaken from November 2014. The Bergen prototype, on the other hand, is a distributed system with a network of sensor nodes transmitting data wirelessly to a sink node connected to the gateway by wire.
To evaluate the topology in use, we first discuss a few things which we believe are characteristics of a good network in the context of automatic weather stations, and how they are affected by centralization or distribution, or by wireless and wired links.

1. **Scalability:** Ideally, the system should allow unrestricted sensor node placement. It should be easy to expand the network of sensor nodes. While mainstream argumentssay that distributed systems are more scalable than centralized systems, the actual bottle-neck would arise if the centralized control of the sensors employed wired connections. In this case, expansion requires physical intrusion from cabling and attaching the sensor node onto the AWS frame. With wireless links however, a mote can just simply be placed near the station without the need to physically access it.

2. **Reliability:**This has to do with the fraction of data that eventually reaches the gateway with integrity, either because the packets are lost along the way or the transmitting de-vice is faulty.Reliability is typically achieved by retransmission and/or redundancy by adding extra sensor nodes [1]. In a wired link, the node-gateway connection is a point of failure, as it is a solid channel that could undergo physical trauma and break. In wireless transmission however, it is rare that the medium itself (air) is faulty. On the other hand, a wired link is more private because it easier to shield from interference than a wireless medium. In both wired and wireless cases, retransmission of data is possible. To achieve redundancy however, the logistical challenges to be met in a wired network are far greater because of the issues presented in item (1).

3. **Cost:** The cost of setting up the link should be as low as possible. Generally, distributed systems are expensive to set up, especially in time and energy. The development effort required to achieve the same goal as a centralized system may be greater. In each scena-rio, using wired links may be more expensive than wireless links if we consider the ex-tra cost of the cabling and the associated labor it takes to lay them. To the contrary,

wireless devices are now very cheap and are scalable as explained in (1). As such, using wireless connections may be much cheaper in terms of time, energy and money.

4. **Range**: A good network should be able to cover a long range. Long range connections using metal wire links suffer voltage drops due to cable length. Optic fibre links also suffer attenuation of the light signal. Wireless signals are also attenuated as they cover longer distances. However, in the latter case, depending on the protocol in use, a wireless signal may be able to travel several tens or hundreds of metres with good reception at the gateway, with considerably lower capital investment than metal wire or optic fibre links.

5. **Power consumption:**The rule of thumb is that the consumption should be as low as possible.Active wired sensors will consume from the same power source as the gateway, or from the gateway itself. In centralized wired connections, power to them can also be turned off when the gateway enters a low power mode. Wireless sensor nodes need their own power supply thus making them points of failure. In an absolute sense, the need for a separate power supply for the wireless sensor node is a disadvantage unless special techniques are used to preserve the power for as long as possible.

6. **Security and Data Integrity:**Data sniffing and alteration should not be possible. While the attention on sniffing can be loosened a bit, alteration of data is a serious concern. As we had mentioned, wired links provide better privacy of the data. Wireless links can be more easily sniffed. Protection against data alteration is not very straightforward and is discussed more in section 3.2.3.

7. **Immunity to the environment:**There are several issues to consider here.

    i. First is the natural environment. Long wires from sensor nodes to the gateway act as radio antennas in a wider radio spectrum picking up electromagnetic interference, from lighting for example, at frequencies lower than the microwave frequencies. This might lead to electrical disturbance that would easily destroy nodes and the gateway [2].In such a case, wireless links are superior to wired links. This superiority persists in cases such as fire, humidity, rain, wind and landslides.

    ii. Second is the human environment. This also includes other animals. Wireless connections are “cut-resistant” connections. Before a human being can cut a connection, they have to know that it is there. Thus, as opposed to wired links, the invisibility of wireless connections enables the creation of networks and links that are extremely hard to disrupt deliberately or by accident.

8. **Mixed Links:**It is fair to assume that items (1) to (7) can best be achieved using some topology in which wired and wireless links are both employed justifiably. Such a network, if well planned, could give optimal performance per link used. This is the case with the Bergen prototype.

3.2 Matters arising

3.2.1 Introduction

The Bergen prototype uses wireless sensor nodes connected to a gateway in a single hop architecture, with a possibility of relaying the data. Any advantages and disadvantages put forward for wireless links in the previous section will apply.A major issue to consider in this decision is that specifications on the sensors or on the number of parameters to measure could change after installation. Making the topology scalable would be a huge advantage in this regard.

It is our opinion that the combined benefits of using distributed wireless links between the sensors and gateway outweigh the disadvantages.

However, some pertinent issues that arise from our user experiences so far are now put forward.

3.2.2 Reliability

Depending on time allowing and complexity of implementation, the following issues on the reliability of the system should be looked into during the design of either the 2ndor 3rd generation prototypes or as possible research issues of the RC3 PhD candidates.

i. The current design has one sink node without back up. As such, if the sink-node fails then the entire system is disrupted. The benefits of using a redundant sink node to increase the reliability should be weighed against the cost and any other parameters.
ii. In the current setup, if the link between a node and the gateway is broken, say from displacement of the node, communication is disrupted. Altering the firmware to implement a multi-hop architecture, with some form of acknowledgment response, could ensure that sensor data is relayed through nearer nodes and delivered to the active sink node.
iii. It is not mentioned anywhere whether retransmission of data is currently being employed.If it is not, it is probably justified because the distances of the nodes from the gateway are very small and the probability of data being received is very high. Moreover, it may not be necessary depending on the sampling time interval of the sensors. If there is sufficient evidence that these distances could increase to levels at which packet loss becomes common, then the use of this technique should be investigated.

3.2.3 Security and Data Integrity

The motes broadcast IEEE 802.15.4 packets formatted with a tag-style encoding,in plain AS-CII, with device addresses and data [4]. This setup can be easily sniffed upon byany similar device developed for this purpose and the possibility of protecting this data by some form of encryption could be investigated.

However, it is our belief that the ability to sniff data at a single AWS is not a security concern in itself, and its investigation is not of paramount importance, since data from only one station is almost useless in the grand scheme of things. More important is the need to protect data from being changed, either between the nodes and the gateway, or between the gateway and the re-mote server that hosts the repository.Collectively, this is known as preserving data integrity.

We now discuss preserving data integrity during both data acquisition and transmission. We have not considered the storage of data.

- **Data Acquisition**

Preserving integrity in the data acquisition stage should be a main concern for RC1, as it is di-rectly related to sensors, their calibrationand other environmental factors. This section is in-cluded however because some of the issues could be fixed by intuitive algorithms developed by RC3. Different sensors may give wrong data readings based on these issues. To that end, the data capture section briefly discusses sources of data integrity degradation in various to sensor types. Sensor non-linearity unlike other factors affects most of the sensors.

1. **Temperature measurements**are mainly affected by: Presence of temperature gradients, temperature drifts, calibration drifts due to hardware degradation, radiant energy and sensor self-heating. The latter can be fixed by appropriate radiation shielding.
2. **Pressure**is affected bytemperature variation and also stability of environmental (air) conditions around the sensor. The latter is because air is not a good conductor of heat. It is therefore important to wait for air conditions to equilibrate hence compromising speed and power since the sensor node must remain active for this.
3. **Insolation**may be affected by presence of a dark current. This is current flows even in the absence of a photons and is highly affected by ambient temperature [14].
4. **Precipitation:**The measured value may depend on the type of precipitation, gauge and location and any errors may be as a result ofevaporation loss, clogging of tipping bucket, trace precipitation (too low to register a tip) and others.
5. **Wind direction:**Loss of accuracy here is dominated by poor alignment of the sensors and incorrect wind direction offset parameters.

- **Data Transmission**

If the sensor measured data correctly, then transmission may affect the integrity of the data. Errors may be introduced from intentional human alteration or media alteration— and more specifically may be as a result of channel fading, electrical attenuation, data collision and others. This can be between the sensor node and the gateway or between the gateway and data repository.

As part of the RC3 PhD research, various detection and correction techniques can be explored to test the data integrity while evaluating their effects on energy, timing of data delivery, space requirements, bandwidth, complexity and accuracy of the output. And while cryptography may be one of the means that can help in detecting the manipulation of the data, the detection of these errors and the effect ofthe complexity of the error detection and correction algorithms on the performance of automatic weather stations should be weighed against other factors in the common good, such as power consumption, before decision is made.

- **Recommendations for Data Integrity**

We recommend that RC1 approve any sensors using rigorous techniques in their selection framework. This ensures that RC3 design work puts minimum effort in programming or electronic design directed to fix sensor-specific issues. The sensors given to RC3 to use should as “plug and play” as possible.

Unlike data capture, transmission calls for an evaluation of the available error detection and correction standards. Unfortunately, since embedded systems are resource-constrained, thorough investigation of the effects of such algorithms must be studied to recommend one that does not compromise the performance of the weather stations in other regards.

3.2.4 Power consumption

The main issue here is that the sink node is always on and that causes a major power concern.Power savings can be achieved by the use of well-timed sleep/wake up modes such that the sink node is asleep when all motes are asleep and awake when any of the motes is awake. This requires some form of clock synchronization across the entire system.
For the 2nd generation prototype, a rough technique of ensuring synchronization could be investigated.

3.3 Recommendation for Topology

It is our recommendation that the use of wireless links between the sensors and gateway be retained in the East African prototype.Power savings should be achieved by using sleep/wake up cycles on the sink node.
The use of a redundant sink node for reliability and selected algorithms for preserving data integrity can be investigated starting with the 2nd generation prototype and there is ample time

to perfect these during the testing and manufacturing phases of the AWS.

It is also our recommendation that the current motes in use, the Universal IEEE 802.15.4 mote, be retained. There are several reasons for this:

i. There is evidence of a great deal of design effort and testing that has already been put into the device. There have been two board revisions and several firmware revisions.We believe that there is not much improvement an alternative design would bring to the table compared to the man-hours it would need to make such a design which could be spent on other AWS design issues.
ii. Free ADC channels make the device scalable and several sensors can be mounted on a single mote, thus reducing the total number of required motes
iii. The mote has already met several safety and RF approvals on the European continent.
iv. The mote has already been extensively tested in several similar deployments in-field and there is evidence of it running continually in an AWS prototype for several months without major issues. We can infer some degree of reliability from this.
v. It is small, portable and has very good power consumption during active and sleep periods. In one application, it has been shown to go for six weeks on a single supercapacitor [5].
vi. It is based on a common microcontroller (ATMEGA128) with free development tools and a large community of users.

4 Gateway

To arrive on an appropriate choice of gateway to use in the EA prototype, we discussed from scratch the following issues:

i. the function of the gateway
ii. the desirable properties the gateway should have

4.1 Functions

The following are the functions of the gateway.

i. Receive and store data from local sensors
ii. Send data from local sensors to (remote) server
iii. Receive data from server. These data can be commands, acknowledgement information, firmware updates, etc.

4.2 Desirable properties

The following properties are desirable to have in the gateway considering that the bulk of the stations will be located in remote areas, away from immediate human intervention.

1. **Low power consumption.**
   This is an extremely important property. Energy is a constrained resource.
   - The consumption should be as low as possible compared to the ability of the power source.
   - The power consumption during active mode should be low but more important is its consumption during inactive mode (see item 2).
2. The gateway should have an **on-demand low-power inactive mode** (standby/hibernate/shutdown) that can be entered into and exited from using software or hardware. This directly affects power consumption and is desirable because measurements and transmissions of data will be happening continually and not continuously.
3. The gateway **boot and wake-up time** should be as low as possible. This is because bulk of the power consumed in active mode should be for “real work”, such as data reception and transmission, and not initialization overheads.
4. In case the uplink to the server is down, the gateway should have a **local data storage** facility.
5. In general, the gateway’s form factor should be conducive to the whole design process, starting from placement of peripherals to selection of the housing.
6. **Perform diagnostic service and give reports** of itself and sensors connected to it. This property is desirable as it allows for remote monitoring and troubleshooting
7. The gateway should be a well-tested product from a credible manufacturer.The gateway should support **networking and a file system** at the bare minimum. This is because of its storage and communication functions.
8. The gateway should be **affordable**accordingto the WIMEA-ICT budget.

4.3 Matters arising

The desirable properties will now be weighed against the current gateway in the Bergen prototype, the Raspberry Pi, and with other possible contenders, and recommendations given.

1. **Power consumption**: While the consumption of the Pi is quite low compared to other devices used in personal computing (which was main design philosophy behind it [6]), there are several devices that could be programmed to achieve the functions mentioned in 4.1 while running at a fraction of its power. More importantly, since the gateway willbe performing its functions in a small amount of time and be inactive for the larger part, the power consumption in this period also matters. The Pi, per se, does not have a sleep mode (see item 2) and when idle consumes about 1100mW [7]. Moreover, the sink node is always on thus adding onto the total consumption. A gateway that operates in the microwatt ranges during idle time, such as those based on the STM32 family [8], can be achieved. In fact, several 32 bit ARM Cortex devices are good contenders as they come with various sleep modes and consume ultra-low currents during active and sleep modes. Examples are the FM4 family from Cypress Semiconductor [9] and the EFM32 family from silicon labs [10].

2. **Low power inactive mode**. Except for shutdown, or complete power off which can be done easily, such a mode (suspend/sleep/hibernate) does not exist in the Raspberry Pi. But the real issue to deal with waking up from his mode. The Pi's processor and architecture doesn't have a Real Time Clock as do other processors.In the current prototype, the sink node has a relay that, with slight modification, can be used to connect or disconnect power to the Pi, while the node goes to sleep itself. The challenges in this case would be how to synchronize the sleep-wake up patterns of all the components, and whether the reboot time of the Pi is at least partially deterministic for us to be confident that it will be awake when a node needs to transmit data. Apart from the sink-node solution, anysolution involving extra hardware adds onto the number of peripherals, and thus affects initial design time, portability, aesthetics, points of failure and cost.
3. **Wake up time**: In this case, wake up essentially means boot if the Pi is used as suggested in (ii). The boot time on a Linux system depends on the size of the OS, the services enabled during boot as well as the boot management system in use. Some users have reported boot times (both kernel and user space) of about 8 seconds running by using `systemd` instead of `sysvinit`[11] and others of upto 22 seconds running Minibian [15]. Such times are of course much larger than those achievable using "bare metal" development. The same STM32 family talked about earlier has wake up times in the order of microseconds [8].
4. **Local data storage**: The Pi's memory is sufficient for this application. Almost all contemporary microcontrollers support some kind of file system or at least an SPI interface to which an SD card can be attached. As an example, the sensors.dat file (which holds data received from the sensors) is currently at just under 90MB after

several months of data collection in the Bergen prototype. A 4GB removable card could be sufficient to hold several years of data. As the file size increases, its access time also increases so a good idea could be to store the data in separate files up to a given maximum. Alternatively, once we can confirm that the data was successfully uploaded to the server, it could be deleted after a specified time and data redundancy could be delegated to RC2.

5. **Form factor**: This is a challenge given that there are several interfaces on the Pi that shall not be useful in this application. The HDMI port, 2 or 3 USB ports, the audio jack and the Ethernet port are all unnecessary—the last being due to the fact that in a remote area, the uplink will most probably be a cellular or other wireless service. If we are to design for portability and proper mechanical support, the gateway microcontroller, its power supply and other peripherals should be on the same PCB. This also improves aesthetics and eliminates the chances of lose connections and conductor breakage at interface points. Designing the PCB for chip level connections will also enable flexibility of component placement around the processor.
6. **Diagnostic Services**: It is possible to log onto the Pi in the Bergen prototype and perform some diagnostic and troubleshooting operations. If the gateway will be going to sleep for some time and waking up again, the connection procedure becomes more complex as it will have to be timed accurately. Hence, a possible solution would be to schedule regular diagnostic reports from the gateway, the contents of which can be agreed upon later (online motes, voltages, RSSI values, dead sensors etc). Also, there will be several AWS installed. It will be much faster looking for faults by querying and data mining the report information sent to the server rather than attempting to log onto individual AWSs.
7. **Credible Manufacturer**: The Pi's comes from a credible manufacturer features open source hardware and software and a vibrant community. This is also the same case with many other computing platforms.
8. **Networking and a file system.** The Pi is OS based and so supports these as a basic function. The Bergen prototype currently uses an Ethernet connection to the internet and this would necessitate a TCP/IP stack. However, with the use of a GPRS/3G module, low level networking operations are left to the module's processor and connection from the gateway to the module are at a system level. File system support for common removable devices (FAT16/FAT32) is also a basic feature on several contemporary microcontrollers.
9. **Affordability:** The Raspberry Pi is rather cheap since in this application no general purpose computing peripherals are needed. Several microprocessors, already on their development boards, cost just as much as the Pi, or vary only slightly. The cost of the gateway itself, however, is not representative of the total cost of developing the gateway system which will include PCB design and electronic peripherals that may even be more expensive than the gateway device itself.

4.4 Gateway Uplink

The issue of selecting the type of uplink the gateway will have is of major contention, because the Bergen prototype uses TCP/IP over Ethernet, which will be next to impossible in several remote areas of East Africa. The following list contains all the options available, regardless of possibility of use.

i. Cellular Service (GSM, GPRS or 3G)
ii. Other Radio Frequency uplinks (VHF/UHF/Microwave)
iii. Optical Fibre
iv. Satellite Systems

Satellite systems cannot be used for obvious reasons. They are very expensive, and require very large or directional terrestrial antennas.

Optical fibre is a choice to ponder on. Most East African countries have some kind of optical fibre backbone program going on with the intention of bringing fast internet inland, either by themselves or private companies like Google. The major problem with optic fibre would be the last mile network to the AWS— and in this case, the option would be either (i) or (ii).

Other Radio wave communication options stated in item (ii) have expensive terminal equipment. Large antennas are required for both the receiver and transmitter and the geographical distance that can be covered is quite limited. A feasible option in this case would be to involve one of the TV or radio signal carriers to relay the AWS data, which may have licensing costs.

Using cellular service is perhaps the cheapest (per AWS), most portable, most available and thus only realistic option currently. All the East African countries have extensive mobile coverage by at least one of several service providers. The terminal equipment at the AWS is very small and cheap. There is support for data services as well as text messaging (SMS) and voice calls. The operational costs are quite low and the NMAs have the option of entering a MoU with national service providers for even lower costs.

4.5 Recommendation

It is our belief that the primary functions of the gateway can be performed at a fraction of the power currently being consumed by the Raspberry Pi while implementing several of the desirable properties within the remaining time frame of the Bergen visit.

We suggest that power consumption be considered the primary parameter in selecting the gateway. Power consumption is also important because there may be a possibility of panel-less installations in order not to attract vandals, a major problem seen with the AWSs installed in

Uganda have faced [12]. As such, an experiment should be conducted using devices whose datasheets suggest a consumption of much less power, such as those suggested in [8],[9],[10] and yet meet the majority of the desirable properties, against which the Pi has also been evaluated. The manufacturer provided data for such devices, on paper, is enough to justify such an experiment.

Of the devices we reviewed, the STM32 is the best contender by a significant margin. It has free development tools (compliers and IDE), extensive software library from the manufacturer and also from other parties, cheap manufacturer developed boards that are quick to start with for both beginners and professionals such as [16], as well as several open source "bare" boards developed by various users for professional use such as [17]. It supports programming in C as well as Object Oriented programming in C++, C# and Java (The last two, over a runtime at the expense of some performance). It is also directly supported by the GCC compiler for ARM. There is no extra hardware needed for programming.

Such a device will be programmed to achieve the bare minimum AWS functionality (receive data, store and transmit), and its power consumption benchmarked against the Raspberry Pi. If the savings are significant then it is our recommendation that the alternative be taken.

We also recommend that the only type of uplink to be tested and used, in the meantime, be a GSM/GPRS/3G service until there is sufficient reason to review an alternative.

5 Other issues

We now move our discussion to other design issues that we felt would be important for evaluation.

5.1 RS Mote Power Supply

The nodes are currently being powered by solar panels charging ultracapacitor batteries. The current solar panels in use are quite sizeable, and would find a number of household uses in rural East Africa which may cause the vandalism spoken about in [12]. A design to consider is to use very small 2.5-3.5V, 40-100mA solar panels (~ 60mm x 60mm or similar)with a basic charge controller to the capacitor. Such small panels would deter vandals—and solar energy resource in East Africa is available for long durations throughout the day on many days so small charging currents are not necessarily a disadvantage.

5.2 Node Enclosure

The IP rating for the enclosure casings currently in use to protect the motes and internal circuits from dust and water is not available but from our recent experiences, seems satisfactory. The main worry is how this rating is affected by any drilling done to permit the passage of conductors. We suggest the use of standard drill bits and standard cable glands to create IP68 compliant enclosure boxes for the electronics.

5.3 Conduction and Radiation shielding

The manufacturers of the SHT2X sensor family have some general guidelines for best practices while using their sensors [13]. It would seem that some of these practices are intended for high precision measurements and their disregard may probably not cause issues that later calibration will not fix. However, it seems to be a general guideline to thermally decouple the sensor from heat sources on the PCB (usually, other electronics) by mounting the sensor on its own board and then connecting this board to the main board or using a flex in the PCB—see figure 8 in [13]. We suggest this as an issue to discuss, especially with RC1.
We are aware that the RC3 advisors have developed a radiation shield for this sensor and we await a demonstration of this.

5.4 Frame

The Prototype currently has no Frame onto which the gateway and some sensor nodes will be mounted. In the initial deployment, some sensors were mounted on an already existing AWS from another manufacturer. Generally, the frame should be weather resistant and physically strong. Our research has revealed several materials that could be used, including but not limited to:

i. Stainless steel
ii. Lightweight Carbon fibre
iii. Aluminium
iv. A mixture of any of these

It may be necessary for this team to make a specific survey of few weather stations installed in Bergen directed to study the materials used and other general frame design principles.

References

1. Mahmood M.A et al, Reliability In Wireless Sensor Networks: A Survey and Challenges ahead, *The International Journal of Computer and Telecommunications Networking*, vol. 79 (2015) pp. 166–187, Online. [6 January 2015].
2. Pehrson, B., Olsson,R., *WIMEA-ICT Research Component 3, Progress Report 2014*, pp.5
3. Toscano, E., Lo Bello, L., Cross-Channel Interference in IEEE 802.15.4 Networks, IEEE Workshop on Factory Communication Systems, May 2008, pp. 139-148
4. Osslon, R., A practical Guide to Wireless IEEE 802.15.4 Monitoring. http://hejulf.se/products/WSN/sensors/wsn_practical_guide.html (Accessed: 28 Aug 2015)
5. Olsson, R., Perhson, B., Powering devices using ultra-capacitor batteries, AFRICON 2015
6. https://www.raspberrypi.org/help/what-is-a-raspberry-pi/ (Accessed 27 Aug 2015)
7. Nungu, A., et al, Towards single watt nJoule/bit Routing, UbuntunetConnect2014, Lusaka (+ Addedum: in Email from bpehrson@kth.se to authors, Jul 18 2015)
8. ST Microelectronics AN4365 Application Note, Page 8. http://www.st.com/st-web-ui/static/active/en/resource/technical/document/application_note/DM00096220.pdf (Accessed 27 Aug 2015)
9. http://www.spansion.com/Products/microcontrollers/32-bit-ARM-Core/Pages/Default.aspx (Accessed September 1 2015)
10. http://www.silabs.com/products/mcu/lowpower/Pages/32-bit-microcontroller-technology.aspx#low-power (Accessed September 1 2015)
11. http://www.samplerbox.org/article/fastbootrpi (Accesed 27 Aug 2015)
12. Nsabagwa, M., Byamukama M., Report On The Status Of Weather Stations In Uganda, May 2015
13. Sensirion AG, SHTxx Design Guide, http://www.sensirion.com/nc/en/products/humidity-temperature/download-center/?cid=881&did=115&sechash=f65f6b63 (Accessed 27 Aug 2014)
14. Photomultiplier Tubes: Basics and Applications, Hamamatsu Photonics K.K., Chapter 13, pp.3 Available at http://www.hamamatsu.com/resources/pdf/etd/PMT_handbook_v3aE.pdf (Accessed 15 Sep 2015)
15. https://minibianpi.wordpress.com/2013/07/04/minibian-minimal-raspbian-image-for-raspberry-pi/ (Accessed 16 Sep 2015)

16. http://www.st.com/web/catalog/tools/FM116/CL1620/SC959/SS1532/LN1848/PF252419 (Accessed Sep 14 2015)
17. http://hackaday.com/2013/11/28/breadboard-friendly-arm-board-based-on-stm32f4/ (Accessed Sep 14 2015)
18. Value obtained from Raw counting of ASCII characters in a single frame from a 10m node in sensors.dat file