Unguided Energy (IEEE 802.11, GSM, & GPS) Child Saving Sensor Network

Derek Scarborough

Institution: DeVry University, Department of Computer & Electronics Engineering Technology

Team Members: Felix Astacio
Mary Feightner
Derek Scarborough
Justin Snyder

Faculty Advisor: Professor John Golzy, P.E.

Abstract - There is a dire technological necessity for a vehicle child saving system. As will be noted throughout this document, there have been one too many young children who have met vehicular fatalities. This DeVry Senior Capstone Project entailed designing, building, simulating, and testing an embedded sensor network that utilizes IEEE 802.11b/g unguided energy (as well as guided energy measures to initiate audio and visual cues/alarms) to inform individuals/passersby/authorities of a child left alone in a vehicle (with the sole purpose of saving said child). Specific attention is given to current temperature and time the child has been left alone. The Vehicle Integrated Kid in Emergency (VIKIE) system utilizes various input sensors to detect if a child is present and also to detect if said child has been left alone. Consequently, if VIKIE has determined that a child has been left alone, its real-time system begins monitoring and taking various measures to ensure child's safety via the aforementioned IEEE 802.11 unguided (and guided) energy initiating alarms and/or informing/soliciting others for child-emergency assistance.
Keywords: VIKIE, Vehicle Integrated Kid In Emergency, CAN, Patent-pending

I. INTRODUCTION

This document provides a stepwise description of the design, construction and implementation of VIKIE (Vehicle Integrated Kid In Emergency) by applying the IEEE 802.11a/b/g standard. VIKIE began as a senior capstone project (is currently patent-pending) and was designed to address vehicular child-safety and use modern technology to prevent and/or eliminate vehicle-related child fatalities. Research[1] has revealed that 700+ children have met fatal consequences from 1991 to present due to vehicle heat strokes alone (not including other vehicle incidents). This averages to approximately 30 children per year suffering senseless deaths due to vehicle heatstroke. Henceforth, VIKIE is an engineered technological invention to fill this gaping necessity. As there are no systems like VIKIE on the market, the team chose to focus on how to develop a system that could be engineered to seamlessly integrate into vehicles. This integrated approach ensures a system that is always available; ergo, there is no accidentally forgetting a child, there is no instructing a child to "stay put for a minute" (unless child meets weight requirements and can fend for themselves), and in fact there is a measure of non-repudiation (against the operator) achieved as a result (as disarming/disabling VIKIE requires specific and distinct steps).

This report will also provide the results of VIKIE's implementation. VIKIE is an embedded microcontroller system (coded in C) that senses for multiple phenomena (child present, child alone, environmental heat if child present, time child left alone, sound, motion, and more). In other words, no one sensor can trigger VIKIE to begin monitoring and initiate any alarm procedures. For example, the load cell detects a child's weight (up to 70 pounds) but this one factor will only alert VIKIE to the possibility of a child present (as it could be a pet or any other object within the weight limit). It will take multiple sensors being monitored by VIKIE's programmed real-time interrupts to determine that a child is actually present. Next, VIKIE was/is a proof-of-concept. The decision was made to develop a system that could be modified by any and all car manufacturers and/or after-market shops. That being said, VIKIE is also a working prototype. VIKIE's microcontroller and various sensors only require wiring (preferably to the CAN, but again, modifications can be made) inside of a vehicle to traverse from prototype to actual working system (wiring would be no different from an audio system, car alarm system, etc.). Also, if modifications dictate it, the code can simply be uploaded to pre-existing vehicle computer chips which would then only require wiring of the various sensors.

Those who have learned from an early age the anguish experienced from heat-related injuries know the reverberating profoundness that heat injuries cause. It simply feels as if the pain will not subside. VIKIE was unanimously agreed upon as a senior capstone project to address the tragic suffering of little ones via being trapped in a "car-oven." Now, this report will explain without revealing confidential and patented information the relevance of VIKIE. Firstly, VIKIE is always ready to be triggered. What does this mean? Multiple key-wakeup interrupts (load cell and seat belt) inform VIKIE of a child presence. This causes VIKIE to enter into a non-stop real-time monitoring service. If previously determined present child is subsequently left alone, VIKIE begins monitoring temperature and time left alone. Once predetermined limits have been exceeded, VIKIE informs another piece of code: Spartan. Spartan is in charge of the alarm system to protect the child in distress. Once Spartan starts it will cycle through its various alarms (texting driver, flashing lights, horn & flashing lights, PA announcing child in distress, and finally texting 911 with GPS coordinates). Spartan will not stop unless a door is opened. Spartan must fully acknowledge that the child is being retrieved/rescued to reset. VIKIE utilizes a supercap array to supply twice the amount of time needed for VIKIE to exceed safe limits; thus, irrespective of vehicle battery power (main power to VIKIE normally) availability, VIKIE (and Spartan) will function. VIKIE represents a technological solution long overdue to save children (even one child lost to heat suffering is too many).

The 802.11 b/g communication standard was selected because of its suitability with achieving the goals and objectives of the project (section VII will provide reasons of the selection). This standard is the key cog that allows VIKIE to save children. In particular, IEEE 802.11 is used at least twice and sometimes more (as discussed throughout the document. Again, VIKIE's code and input sensors allow technological advances in detecting distressed children; whereas it is VIKIE's communication system (namely 802.11) that allows VIKIE to reach outside help to actually save children therein giving VIKIE its real power and societal usefulness.

II. VIKIE PROJECT REQUIREMENTS

1) Research the feasibility of VIKIE.
2) Define the problem VIKIE is addressing and develop design specification accordingly.
3) Plan VIKIE's solution (Gantt chart used).
4) Develop design schematics of load cell, supercap battery backup, and buck regulator.
5) Complete custom layout and etch PCB(s) and/or use COTS materials.
6) Develop pseudocode, software flowchart, FSM, and VIKIE code.
7) Build/construct prototype.
8) Program Android and iPhone apps for remote disarm/disable capability.
9) VIKIE will communicate wirelessly via IEEE 802.11b/g, GSM, and GPS.
10) Test hardware.
11) Test networking.
12) Test and debug software.
13) Demonstrate working proof-of-concept demonstration.
14) Device will remain extremely customizable per individual car manufacturing/modeling.

III. VIKIE COMPONENTS

A. Power Supply

VIKIE can employ 3.3VDC and 5VDC which will be supplied via a buck voltage regulator. The buck regulator will utilize the vehicle's main battery power supply at its input. The buck regulator will output required power to the VIKIE microcontroller. The microcontroller can utilize between 3.15V to 5.5V (with emphasis on using 3.3V for energy saving when possible). All components used for current VIKIE prototyping use 5V which is supplied by the primary microcontroller. These include the motion sensor, electret, piezo (stand-in for PA and horn), second microcontroller, WiFi shield, GSM shield, and GPS shield. The buck regulator will also output charge to VIKIE's supercapacitor battery array with a transient charge of approximately 180s and a discharge time of approximately 1800s (30 min.).

B. Microcontroller(s)
The primary microcontroller, Freescale HCS12 (S12G128 Tower), was chosen due to its built-in automotive communication capabilities. CAN and LIN communications are integrated into the HCS12. Also, the HCS12 is actually labeled as an automotive microcontroller. The Tower offered enough memory, scalability, plenty of interrupts, and a host of other powerful features, as listed below:

- 100-pin LQFP
- 16-bit architecture
- 128K flash memory
- 8K RAM
- 1 CAN 2.0A/B
- 3 SPI
- 3 SCI (LIN communication support)
- 8 16-bit timers
- 86 GPIO

For prototyping purposes only, a secondary Arduino Uno R3 microcontroller (controlled by the HCS12, as PCBs are further developed, all components that are prototyped via shields will be integrated into a single PCB and attached to primary MCU) was chosen due to its simple and powerful scalable prototyping capabilities.

- 8-bit architecture
- 32K flash memory
- 2K RAM
- 20 I/O pins (14 digital I/O, 6 analog input)
- Large array of ready-made shields

C. Input devices
The current prototype model utilizes the following input devices which are all driven by the primary Freescale HCS12 microcontroller:

- NovoTech HC-SR501 pyroelectric infrared sensor (motion sensor)
- Meco LM393 electret (microphone sensor)
- Meder MK03 reed sensor (embedded seat belt sensor)
- Microchip 3-Pin TO-92 MCP9700 thermistor IC (temperature sensor)
- Custom-design load cell based off of schematic below (there are multiple versions, as the load cell is one of the main components that must be modifiable due to various rear seat construction/assembly)

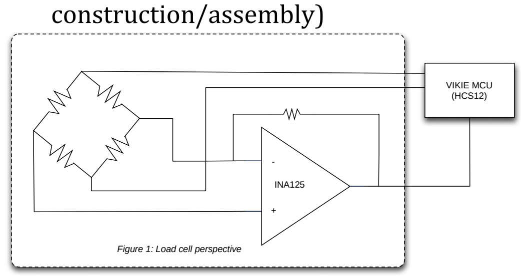

Figure 1: Load cell perspective

D. Output devices
The current prototype model utilizes the following output devices which are all driven by the primary Freescale HCS12 microcontroller:

- Amico 5VDC active-low piezo module (vehicle horn emulation)
- PA speaker
- LEDs (vehicle headlights/taillights emulation)
- LCD (prototype demonstration purposes only, LCD not part of final system)

The following output devices are wireless technologies which are driven by the secondary microcontroller (which in turn is driven by the primary microcontroller):

- Arduino WiFi shield (802.11b/g)
- Arduino GSM shield
- Itead GPS shield

E. Miscellany
In addition to the aforementioned I/O devices, the following options are used to communicate directly with VIKIE as needed. Of course, the microcontroller can be attached to a programmer for hard-wired updates and maintenance. The programmer depends on the microcontroller used. For the current prototype microcontroller, a Mini-B to Type A

USB cable and computer will suffice (where other MCUs may require a programmer device).

Also, the operator/consumer of VIKIE has the option to install the VIKIE app on either an Android device or iOS device which communicates with VIKIE via 802.11b/g (if available) or GSM (as a last and more expensive resort if 802.11b/g is unavailable). The VIKIE app allows the owner to enter contact information (which is used for first contact in child emergency) and also the ability to disarm (temporary off - perhaps transporting pets or so) or disable (permanent off - perhaps child has outgrown VIKIE or car is sold to childless owner) via GSM and/or WiFi. Note that the VIKIE app cannot reset the system if alarm is going off due to child left alone; it can only alter system if system is idle (not currently monitoring a child in vehicle). The VIKIE app will be available for Android phones and watches, as well as iPhones and Apple's watch.

IV. VIKIE HARDWARE DESIGN

A. Vehicle Implementation/Interfacing

There are three ideal hardware scenarios for VIKIE implementation/interfacing. The most desirable is the ability to embed VIKIE/Spartan code inside a pre-existing vehicle computer chip (as allowed/dictated by individual car manufacturers) and wire the various sensors accordingly.

The second (and maybe even more ideal than the first) hardware scenario consists of a single integrated microcontroller which is to be installed in the vehicle (under seat, dash, etc.). This could be the current HCS12 prototype microcontroller with all components (WiFi chip, GSM chip, GPS chip, and so forth) integrated on one or more PCBs. This second scenario is actually a great solution in regards to any possible SPOF (Single Point of Failure) that could occur. This scenario if met with malfunction would actually be the SPOF, versus a malfunction occurring inside of the complex vehicle network chip (requiring complex troubleshooting measures). As the VIKIE system is so inherently important, this second scenario if riddled with complex problems (unable to quickly troubleshoot and repair) is ideal for an expeditious system swap (much like a car battery - whereas, replacing an integrated vehicle chip is not ideal, if even possible) to ensure VIKIE is up and running at all times.

Finally, the current prototype (used by the team for demo purposes) represents the more complex and costly of the vehicle-VIKIE interfacing. It entails a primary microcontroller (currently the HCS12) which houses the logic of VIKIE (and Spartan). From here, if Spartan (recall, Spartan does the actual child protecting) is started by VIKIE, Spartan then utilizes a secondary microcontroller (currently the Uno) which contains specific logic to utilize its attached WiFi, GSM, and GPS shields. Hence, there are no less than two microcontrollers and at least three additional ICs/shields creating a complex network to troubleshoot.

B. Power management

VIKIE will receive power via the vehicle's power system. The present prototype draws on average between 500mA and 700mA and at maximum, 1.1A at 5VDC (recall the 5VDC is stepped down from 12VDC via a buck regulator). The maximum amount represents a full-blown system which is engaged in saving a child. In other words, every component is firing including the GSM and GPS which together can draw a maximum of 2A alone under heavy use. VIKIE's use of these components will always be intermittent. For example, the final alarm to 911 cannot occur without GPS coordinates. Ergo, VIKIE will use the GPS chip to receive coordinates and store. Immediately thereafter, VIKIE will initiate the GSM to dial 911 and send a predetermined message ("VIKIE 911: Child in distress, Car model: Audi, Car color: Silver, License: 123-SOS, Lat: 33.3806716, Lon: -84.799657"). The longitude and latitude are what's gathered via the GPS sequence. Also, it must be noted that while VIKIE is simply monitoring that the current draw is less than 300mA (and completely negligible when idle). The above maximum and averages are for a system that is using all of its components to save a child and is not (and hopefully never will be) the norm for VIKIE's power requirements.

VIKIE also employs a dynamic battery backup: a super capacitor array. As noted continuously throughout this document, VIKIE is highly modifiable; thus, the super cap array in one vehicle may be a single super cap in another or an ultra-cap in yet another vehicle. It may be feasible in some vehicles to also use lithium, or a type of SLA rechargeable battery system. VIKIE's requirements are quite low and even if a vehicle battery lacks amperage to start a car, it would be a rare occurrence for the vehicle battery to be in such a state as to not have enough voltage to power VIKIE. That being said, what is important is that VIKIE will have backup power either via a weakened vehicle battery or a capacitor array (or other means). This ensures availability of the ever-important VIKIE.

C. PCB Design plans

The current VIKIE prototype consists of COTS (Commercial off-the-shelf) materials (microcontrollers, shields/PCBs, etc.). As VIKIE has been proven, design plans entail developing custom PCBs. One major PCB will be designed to contain WiFi, GSM, and GPS.

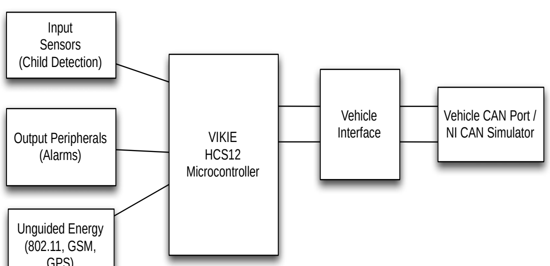

Figure 2: VIKIE black box perspective

V. VIKIE SOFTWARE DESIGN

A. Design Requirements

Unlike the hardware design, wherein COTS materials were available to design VIKIE, such is not the case with VIKIE software requirements. There is no COTS software available (or close enough to study/emulate) for VIKIE. Thus, custom embedded software is necessary. VIKIE's entire system was designed and is predicated on interrupts (real-time and key wake-up). VIKIE is written entirely in C (with future plans to also benchmark against assembly). VIKIE's main software flow/logic monitors actions via input sensors and subsequently utilizes time and temperature; hence, through thorough multiple phenomena, VIKIE can with certainty determine that a child is present and commence with real-time monitoring. Once VIKIE's logic has determined a child is present, VIKIE will not stop monitoring in real-time until said child is no longer in vehicle (whether other occupants are present or not). If VIKIE determines child is left alone, Spartan begins its FSM alarm system. Spartan is coded as such that only one action will reset it: opening a door AND child is no longer detected OR occupant(s) over weight requirement is present AND/OR vehicle is in action (moving, brake engaged, etc. - any action proving child is removed or not alone). Along with VIKIE's interrupts and Spartan's FSM, one other important (and obvious) notion is heavily utilized: binary semaphores (shared flags). These flags work in tandem with VIKIE's interrupts, Sparatan's FSM, and other aforementioned multiple phenomena. Example, a child is either present (1) or not (0). The child is either alone (1) or not (0) and so forth.

VIKIE's software that is embedded in the secondary microcontroller simply continuously loops and awaits specific signals from VIKIE. Code in the secondary when instructed sends unguided energy to inform others. This code controls the WiFi, GSM, and GPS systems. The code in the secondary simply loops awaiting input from VIKIE. If VIKIE determines a child is in distress, VIKIE signals that to Spartan (which is also in the primary) and Spartan signals the secondary. Once the child has been saved, VIKIE determines this as well and subsequently signals that to Spartan and secondary as well. All logic is in the primary under the auspice of VIKIE. The secondary simply responds to Spartan which is, again, controlled by VIKIE.

B. Communication Protocols

The VIKIE system requires multiple wireless access technologies to send "child in distress" messages and to allow communication with operator/consumer. For the ever-important child-saving services, IEEE 802.11b/g was chosen for wireless TCP/IP as these IEEE wireless protocols are ubiquitous via WiFi hotspots. If free WiFi service is available an SMS will be sent via WiFi-to-SMS system. If WiFi service is not available, or signal is too weak (slow response, timer has begun), or even a WiFi jungle (multiple WLANs available, again possibly causing slow response due to possible cycling channels for strength, etc.) VIKIE will rollover to GSM.

In addition, IEEE 802.11b/g is the protocol of choice for the operator/consumer to enter contact information (name, phone number, vehicle data, etc.). The ability to do so is made possible via Android or iOS devices. The "smart devices app" has three coded functions: 1) contact information, 2) disarming, and 3) disabling. The contact information may be gleaned from the device's integrated contacts directory and/or entered manually. The disarming code allows the operator the ability to temporarily disarm the VIKIE system. Upon the second car ignition, VIKIE will rearm automatically. Disarming is handy when the operator knows in advance, various phenomena may cause VIKIE to monitor needlessly. VIKIE still will not start without correct determination, but will at the very least begin steps continuously to detect and monitor. Disarming will avoid this as the operator may have a pet transporting to the vet and may use the seatbelt on pet/pet carrier. The disabling function is a very important piece of VIKIE's code. Once a child outgrows VIKIE, the operator can then completely disable VIKIE. VIKIE will simply be a dormant system which can be re-enabled if need be at a later date. For example, a car sale to a family would constitute re-enabling VIKIE or having young grandkids visiting for a summer, and so forth.

In addition to WiFi, GSM, and GPS communication protocols, the CAN (Controller Area Network) is a necessity. The CAN is a serial data communications bus that can be used for real-time applications, such as VIKIE's embedded real-time software. The CAN may be the most important protocol for VIKIE. Recall the door opening informs VIKIE of such an action. This resets the system. VIKIE can determine if the car is actually moving via the CAN bus by gathering information about the vehicle's speed. VIKIE can quickly in real-time use the brake engaged to determine a driver occupant is present. VIKIE can monitor if airbag systems are engaged due to driver and/or passenger occupants; which helps VIKIE determine occupant(s) are present. And more. VIKIE's CAN access is limited to and by each individual vehicle manufacturer and of all of VIKIE's customizable systems must remain the most modifiable. CAN signals per manufacturer are proprietary and are not public knowledge and indeed for car safety must be kept so. Henceforth, VIKIE software per vehicle manufacturer CAN signals must be updated continuously per each car manufacturer and/or model.

VI. APPLICATION OF NI CAN

A. CAN (Controller Area Network)

Per the aforementioned proprietary CAN signals discourse, how can VIKIE be programmed and intrinsically know what signals represent what car code if car CAN codes are proprietary and confidential? National Instruments of course. NI provided our team with a top-of-the-line CAN simulator. NI donated the NI USB-9862 CAN Module, NI cDAQ-9171, CAN Breakout Box, and NI-XNET Software.

The application of the CAN simulator cannot be understated. It allows VIKIE to emulate receiving vehicle CAN signals which can occur at any given time. This is important to understand. Having specific knowledge of what each CAN signal means is crucial. This knowledge is transferrable to applying real-world CAN signals inside of VIKIE's code. To be honest, the NI CAN simulator is still being explored and is a crucial work in progress. That's key however as the CAN codes will be different for each vehicle; hence, coding the CAN will be a continuous work in progress. The learning curve of the NI system in and of itself has been nothing but a positive success. The software lead had never worked with CAN systems/signals prior to the NI simulator. Simply learning this system alone has ingrained the CAN system. For example, being able to set up custom CAN signals taught an invaluable lesson: CAN messages are identified by their contents and not their addresses. Each message has a unique ID. Recall that car CAN codes are proprietary and confidential. So via NI's CAN simulator, VIKIE was able to be monitored for a specific door open signal.

Additional study of this powerful tool is ongoing. The CAN simulator while learning it was assumed to only work with LabVIEW and NI-XNET. However, further study reveals that the software lead can indeed use a C API for

NI-XNET. As powerful as LabVIEW is, the use of C will allow even more powerful and portable solutions.

To reiterate, the intrinsic understanding of CAN signals and their required implementation into VIKIE was made possible due to NI and their donations. Simply learning NI's CAN simulator led to additional knowledge of the primary HCS12 microcontroller. How should VIKIE be coded to get those CAN signals off the CAN bus? Lo and behold, the HCS12 has a programmable listen-only CAN bus mode and a programmable CAN key wake-up interrupt. These important revelations and more were made solely as a course of study (in learning NI's CAN system) to implement CAN signals into VIKIE which was tangibly made possible due to NI's donations, for which the team is forever grateful.

VII. APPLICATION OF IEEE STANDARDS

A. IEEE 802.11b/g[2]

The need for simple and ubiquitous wireless access to communicate in VIKIE is inherently clear. There are myriad wireless options such as IEEE 802.15.1 (Bluetooth), IEEE 802.15.4 (ZigBee), and others. Research immediately ruled out all, save IEEE 802.11. Bluetooth and ZigBee are more geared for a WPAN (Wireless Personal Area Network) and very few consumer establishments offer this service to consumers anyway. Also, the range of a Bluetooth piconet (approximately 10m - without any amplifier), and the 802.15.4 (100m - 120m max. and point-to-point) eliminated these wireless technologies almost immediately. Another major factor is the "plug-and -play" factor of 802.11. Most Bluetooth devices need some form of pairing and 802.15.4 needs specific channels and IDs set as it is a point-to-point protocol. Therefore, 802.11 was a unanimous choice. IEEE 802.11 is ubiquitous and growing. 802.11b/g offer 140m max range; however, this does not take into account repeaters and WAPs companies offer to its consumers. In other words, 802.11 hardware for routing, signal strength, and free use is prevalent in most establishments and city/county areas (and growing where not).

The 802.11 component used in the VIKIE prototype was an Arduino WiFi shield. This shield offers IEEE 802.11b/g standards. Software was written to instruct the shield to attempt to access a free available network when required.

Due to the "plug-and-play" functionality of most 802.11 WLANs, VIKIE from the beginning had a "way to work." In addition to the WiFi shield, the XBee S6B was coded and used. All worked seamlessly thanks to IEEE 802.11's rock-solid standards. IEEE 802.11 also led to other unguided energy ideas. 802.11 covers terrestrial waves whereas GSM & GPS works up to satellite waves. Throughout the entire development of VIKIE, 802.11 was used as a baseline to communicate with Android devices, iOS devices, additional radio devices (which emulated being 911 or the driver/operator), and more. When code on smart devices was changed and needed testing, for efficiency's sake, 802.11 was the first (and sometimes only) to test. For example, GSM works great, however, GSM definitely has a higher cost and is much, much slower than 802.11. Code for 802.11 could be programmed and tested in less than 1 minute literally in some cases; whereas, any changes to GSM code (or smart devices) could take as long as 10 - 13 minutes to send an SMS in some cases. VIKIE is a truly unguided energy sensor network that saves children lives, in large part due to IEEE 802.11.

The first important use of 802.11 is the need for the vehicle owner/driver to enter contact information into VIKIE. This is done almost exclusively via 802.11. If 802.11 is not available, costly and expensive GSM may be used. Also, one may actually buy a special adapter and get under the hood or rear seat to physically access the VIKIE system. These latter two options present cost and accessibility problems to be sure. Henceforth, 802.11's easy wireless access (in most cases, free to nominal cost) to the system from inside one's home to their car's VIKIE system is only the first testament to the power of IEEE 802.11.

Next is another very important 802.11 use: VIKIE's first alarm attempt. This occurs in 3 minutes after leaving a previously detected child in the vehicle. Thus, upon a driver exiting a vehicle, who innocently and inadvertently left/forgot a child, a text message will be sent via VIKIE's use of 802.11 to email-to-SMS (or a host of other options) alerting them of a "child left in vehicle." This occurs via 802.11 if available in the first 3 minutes before any danger befalls the child.

Yet another important need of 802.11 for VIKIE is its need to contact help if a child is in danger/left alone for too long. For instance, if the aforementioned driver does not respond to the first alarm which is simply a friendly text, other alarm measures (horn, lights flashing, and PA announcement) will take place. This last and most unfortunate use occurs in the 15 minute range where the child's danger is reaching criticality. 802.11 is again invoked to search for an available network. This time to send emergency messages to the authorities (Police, EMS, Fire & Rescue) which entail car info and GPS coordinates. Here as time becomes of the utmost importance, tests revealed that 802.11 can send messages in under 30 seconds whereas the GSM can take 60 seconds and up to 180 seconds to get a complete message delivered. The efficient efficacy of 802.11 is paramount in VIKIE. In an almost instantaneous manner, VIKIE can use unguided energy to contact others.

The IEEE 802.11 offers extreme simplicity and robustness.

B. IEEE 1490[3]

VIKIE was developed while adhering to 7 of 10 knowledge areas of IEEE 1490 project management. The VIKIE team met twice a week (and in some cases 3), emailed, video-conferenced, and utilized Google Drive (for a bevy of documentation) for eight plus months. VIKIE was met with success due to the following management areas that were utilized throughout:

1. Integration management
2. Scope management
3. Time management
4. Cost management
5. Communications management
6. Risk management
7. Procurement management

In particular, each team member was required to wear many hats for this project, integration management, thus, was a key element in parsing task objectives (divvying per member) or compiling them (unifying them for one member). One member may have had the task to completely manage the team's BOM for one session and the next session all members may have been parsed the duties of individual components that had been procured via our initial procurement guidelines (another important KA where the team followed cost management and utilized a BOM and specific manufacturers). Integration management lent much needed coordination to the team's processes due to all members' fluctuating school/work scheduling and workloads.

Another project management tenet was our complete adherence to our scope management initiative. For VIKIE, our scope set forth to develop a system to address heat-related child-car incidents. Throughout development, a bevy of other topics were constantly brought up by outsiders (cold-weather, pets, and other niceties). All of these were outside of the scope at the time and were placed in our "parking lot" for later review. These topics were and will be addressed; however, we agreed as a team to adhere to our scope management guidelines. And this in turn (remaining within scope) led to the team being able to meet cost management guidelines to keep VIKIE within budget. VIKIE was also accomplished ahead of schedule, thereby meeting time management initiatives.

These and the other aforementioned IEEE 1490 management areas were specifically addressed and gleaned from the Guide to Project Management Body of Knowledge to ensure completion of a viable invention/product.

C. IEEE 12207-2008 (basis for SWEBOK V3)[4]

VIKIE was also developed while adhering to no less than 5 knowledge areas of SWEBOK V3 software engineering paradigm. The software lead of VIKIE studied and learned a necessary body of knowledge for the software engineering profession and applied said knowledge from knowledge areas as noted:

1. Software requirements
2. Software design
3. Software construction

4. Software testing
5. Software maintenance

An example of the use of IEEE 12207-2008 involved consistent attention to software requirements. This embodied the entire software life cycle of VIKIE. It began simple enough with a simple analysis of needs the software had to perform. From there software design entailed a flowchart and pseudocode. The construction and testing took place thereafter and concluded by revisiting the software requirements KA which as a final course of action entailed testing and validation of software and its feasibility for VIKIE.

VII. CONCLUSION Based on our test results on a live network, VIKIE met the design goals and objectives. VIKIE's microcontroller was able to consistently communicate with the coded smartphone app designed to work with VIKIE (both of which contained a chip with integrated 802.11). This 802.11 communication was tested and verified and subsequently used to completely test VIKIE's communication system. Also, VIKIE's software adhered to IEEE 12207-2008. Appropriate software KAs were tested and verified via test equipment (logic analyzer) and visual demonstration. VIKIE's inputs and outputs were compared against the designed flowchart (software design) to ensure VIKIE's operations worked accordingly and completely met software requirements. In so meeting software KAs, this afforded the team to meet our IEEE 1490 project management initiatives. In verifying a working communication system (IEEE 802.11) and software requirements (IEEE 1490), project management (IEEE 12207-2008) guidelines were subsequently met.

VIKIE clearly demonstrates a successful integration of technology that can be used to save children. Through the use of embedded microcontrollers, sensors, and especially via the use of the aforementioned IEEE standards (IEEE 802.11, IEEE 1490, IEEE 12207-2008, and an IEEE student grant to boot) VIKIE was accomplished and will fill a much needed and dire societal need.

ACKNOWLEDGEMENTS

- NI (National Instruments) for $1500 donation (various CAN hardware)
- IEEE for $500 grant which assisted project BOM
- Professor John Golzy, P.E. DeVry University for insight on seeking, acquiring, and vouching for external donations and assistance
- Professor Amin Karim for technical review and feedback

REFERENCES

[1] *http://www.kidsandcars.org/statistics.html*
[2] *IEEE Standard for Information technology—Telecommunications and information exchange between systems Local and metropolitan area networks—Specific requirements, 802.11, 2012.*
[3] *A Guide to the Project Management Body of Knowledge, 3rd Edition, 2004.*
[4] *SWEBOK V3.0 Guide to the Software Engineering Body of Knowledge, 2014.*