BASA: Building Automation and Security on Android

João Miguel Marques Sampaio

Instituto Superior Técnico, Universidade de Lisboa
joao.mm.sampaio@tecnico.ulisboa.pt

***Abstract*—Systems capable of reducing energy consumption are needed in order to reduce monetary energy costs for companies. At the same time, studies show that temperature can influence human productivity, thus simply removing Heating, ventilation and air conditioning (HVAC) systems could impact negatively a company's employees work productivity.**

Nowadays Building Automation Systems (BAS) are used to manage a building, however traditional systems face a big problem, they have high monetary costs associated with installation and hardware. At the same time, the market is flooded with cheaper tablet devices with built-in sensors, connectivity and visual display.

In this document, we propose a system that utilizes an Android tablet, mounted in the room wall, running an application to achieve building automation. Our solution addresses the energy consumption problem, increases occupant comfort and offers security, including intrusion detection notification and video monitoring, to the user. The proposed system differs from traditional centralized BAS by offering a distributed architecture with nodes deployed in every office and a user mobile application capable of interacting with the system.

I. Introduction

Buildings represent a large portion of the total energy consumed. According to the International Energy Agency, buildings represent 32% of total final energy consumption [**?**]. One way to contribute to reducing the energy consumption is to use of BAS to improve their efficiency. Unfortunately, not all buildings are equipped with such systems. Most times, these systems offer limited functionality and can only be controlled by the building manager.

Human behavior influences the amount of energy a building requires. Depending on the occupant behavior, the building's energy cost can increase or decrease by one-third of its design performance [**?**]. Simple actions such as leaving the lighting system always on, even after the occupant has gone home, has an impact on the wasted energy used by the building. The lack of occupant detection systems in buildings prevents further energy savings, as by knowing when a room is unoccupied it is possible to shutdown unnecessary electric systems.

Nowadays, consumer smart home systems are becoming more popular. The ability to remote control the house lighting and electric devices is very appealing to consumers. At the same time smart phones and tablets flood the market at very accessible price ranges.

In this thesis, we preset a BASA (Building Automation and Security on Android) a system that uses existing Android devices to provide a BAS capable of managing a room. Our system consists in using a wall mounted Android tablet, called Hub, to control other devices. It also allows the user's phone to interact with the Hub using a mobile application.

We designed BASA with a set of requirements in mind: The system must offer remote control of the room's lighting and HVAC systems, allow users to automate tasks, it must be user and motion aware, it must have a affordable cost and offer good usability.

Our system is user aware and is capable of adjusting the lighting and HVAC systems in a energy efficient way. At the same time it offers the user a If This Then That (IFTTT) system (trigger actions based on events) including voice recognition, for a personalized smart office experience. Finally, it provides the user with a security system. The Android camera is used to detect motion, when movement is detected and no registered person is present in the room a notification is sent to the user and a 30 second video is recorded to the cloud for latter viewing.

By using a tablet as a BAS we are able to reduce the monetary cost of our system in comparison to traditional alternatives. The tablet offers several sensors, access to Wireless fidelity (WiFi) and Bluetooth (BT) networks, microphone, sound speaker and a touch screen. We are able to leverage the tablet sensors including illuminance, temperature and camera sensor for automation. If the tablet does not have a temperature sensor it is possible to interact with external sensors to overcome the lack of the sensor.

The below listed examples represent some automation actions possible with our IFTTT system that contribute to decrease energy consumption and increase user comfort:

- If no user is present in the office then turn off the lights.
- If user arrives at building then set temperature to 24 ^{o}C (pre-heating).
- If lights are turned on and illuminance is above 120 lux then turn off the lights. (If there is sunshine and the lights are on, we turn them off).
- If no user is present in office and motion is detected then say "Hi! You are being recorded, smile!".

The above examples are not hard-coded into the system. They are created by the user. This ability offers great flexibility to our system to provide a personalized feel to the room.

II. Background and related work

BAS are distributed control systems capable of monitoring and controlling a multitude of individual systems in a building. Usually they are used to control a building's HVAC, lighting, security and access control system (SAC). The objectives of building automation are the reduction in energy consumption,

operating cost, improvement of occupant comfort, and efficient operation of building systems.

Until recently, there was no standard industry network protocol for building automation. BAS manufacturers developed unique, proprietary communication protocols and users had to choose between many different systems. Today, we have reached a place where there are a few major platforms used for BAS to choose from: BACnet[**?**], [**?**], [**?**], [**?**], LonWorks[**?**], [**?**] and few other. They were designed for specialized tasks, which limits the possibilities and capabilities of every node. The automation control is usually performed on a centralized server, commonly specified to as the Gateway. Installation may be a very complex task, requiring personalized hardware and/or software to be configured.

More recently other standards for building automation were created: ZigBee [**?**] and Z-wave [**?**][**?**]. These new protocols vary from the previous ones by being designed to use wireless mesh communication networks and allowing different manufacturers to produce products designed to to operate with these protocols. Nowadays many home automation products use ZigBee or Z-wave as their wireless communication standard allowing interoperability between products from different manufacturers.

Modern Home Automation

The popularity of home automation has been increasing in recent years due to higher affordability and simplicity. Home automation may include centralized control of lighting, HVAC, appliances, security locks of gates and doors and other systems. Vendor solutions often rely in wireless technologies to connect the various devices, thus eliminating the need to rewire the house.

Some companies like Philips have developed smart lighting products that allow remote control over the lighting system without the use of conventional light switches. The Philips hue[1] is a wireless lighting system. This system is quite simple, you replace existing lights with Hue light bulbs and use a device called Hue bridge to communicate with the lights using a mobile application. Both Philips hue and similar smart lights suffer from a problem: the user cannot use a regular light switch to switch them off. By doing that the lights become disabled, the user won't be able to use the mobile application to turned them back on, requiring the user to flip the light switch back up.

One other popular product is the Nest[2] Learning Thermostat. It is an electronic, programmable, and self-learning WiFi thermostat. It uses machine learning algorithm to optimize heating and cooling of homes. Studies[**?**] show this thermostat is capable providing savings equal to about 10%-12% of heating usage and electric savings equal to about 15% of cooling usage in homes with central air conditioning.

Amazon Echo[3] is a smart speaker developed by Amazon[4]. The device is capable of voice interaction, music playback, making to-do lists and other useful features. It can also control several smart devices using itself as a home automation hub. The main characteristic of Echo is it's voice interaction capability, any automation system would benefit by having such functionality.

Finally there are companies that provide web-based services that allows users to create simple conditional statements, which are triggered based on changes to other web. One such company is IFTTT, it allows the user to trigger actions based on services such as Weather channel, Facebook, Gmail, Amazon Echo and Nest thermostat. This service allows the user for example to automate their Philips Hue lights, Nest thermostat and many other services.

There are many products for home automation in the market, in this paper we merely discussed some that contribute to our final solution. We learned smart lights have some setbacks. The Nest thermostat offers a good solution to control a HVAC system. The only problem is it has a high monetary cost so it won't be used in our solution. Echo increases user comfort by allowing it to serve as an automation hub and allow voice interaction. We chose to implement a simple voice interaction in our final solution, as it offer many advantages. Finally we decided to implement a similar feature to the service provided by the company IFTTT. We allow the user to create conditional statements that trigger actions based on events in the office.

Occupancy detection

Human behavior influences the amount of energy a building requires. Depending on the occupant behavior, the building's energy cost can increase or decrease by one-third of its design performance [**?**]. By knowing and tracking when a room is occupied, we can provide the system with relevant information that in turn my help in taking important actions such as switching off the lights if no one is present.

There are several different methods to determine if a person is in the room, they range from Radio Frequency Identification (RFID), Passive Infrared (PIR), Vision-based, WiFi and Bluetooth, among other [**?**].

Yuvraj Agarwal and his co-authors proposed using a mixture of PIR sensor and a simple magnetic contact switch to track when the door opens and closes, this solution provides better result in regard to a PIR only solution[**?**].

Using WiFi it is possible to estimate the number of occupants in the area. Occupants usually have mobile devices connected to the building's WiFi, by knowing the devices currently connected to the AP it is possible to estimate their relative location. Furthermore since it does not require additional equipment, it is an economic solution. Bluetooth Low Energy (BLE) technology is also an available solution for room user detection, leveraging the user's phone to do the detection, or asking the user to carry a BLE beacon and have a system listening for the signals.

A camera and image analysis software are able to identify movement between video frames. Noise detection could also provide valuable input to determine occupation, yet it is less

[1] www2.meethue.com, last accessed on January 6^{th}, 2016

[2] https://nest.com/

[3] https://www.amazon.com/Amazon-Echo-Bluetooth-Speaker-with-WiFi-Alexa/dp/B00X4WHP5E, accessed 10/10/2016

[4] https://www.amazon.com/

reliable as external sounds could induce false positive occupant detection.

Sensor application in Automation Systems: Sensors can have important contributes to BAS, they can provide higher comfort to the occupants of the building as well as save on energy costs. Comfort can be improved using temperature sensors by regulating the building temperature to maximize occupant productivity while at the same time only spending energy when required.

Luminosity Sensors can be used to determine if lights need to be turned on or if there is enough natural light to turn off the lights, or at the very least dim the light intensity.

III. Architecture

Nowadays it is affordable to built a device to provide BAS functionality to a small area. This allows the room to be personalized to the occupants needs and habits.

We designed a system, named BASA, capable of controlling the lighting/HVAC systems and offer room security. The system is described in Figure 1 and consists of two mobile applications, one is the Hub app that runs in a tablet mounted in a wall in the room, the other is the User app that the occupant of the room can install on his personal smartphone.

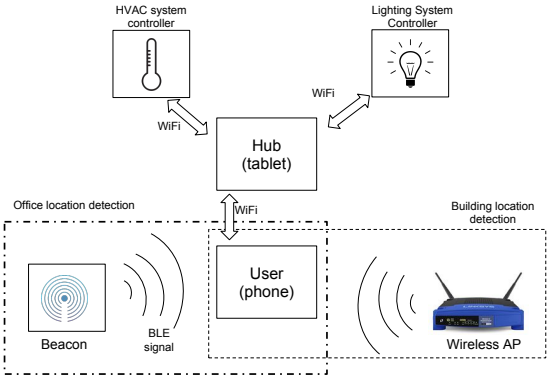

Fig. 1. Overview architecture of the system

The Hub app is responsible for controlling the HVAC and lighting systems in the office, enforcing energy saving policies, improving occupant comfort and providing security monitoring when the user is away from the office.

The main visual features of the Hub application are: a simple graphical user interface (GUI) to control the lights and heating/cooling system, an IFTTT automation system that allows the user to create personalize trigger/actions rules. In the background it also provides motion detection and video recording for occasions when the user left the room but movement is detected.

The User app runs in the user's mobile device and offers remote control of the lighting, HVAC and security systems provided by the Hub. Besides improving user comfort by allowing remote access to the Hub, the user app also helps the Hub with user detection, allowing the system to know when an authorized user is inside the building or room.

A. Hardware Architecture

To achieve building automation, our solution requires a tablet to act as a central control unit, a beacon device used for user detection inside the room and two other devices capable of interacting with the lighting and HVAC systems. In Figure 2 we describe the main hardware components of our system and the communication protocols used.

The tablet is where the Hub app is executed. It provides the sensors, the connectivity, the storage as well as a camera, microphone, speaker and a touch screen.

The beacon is used for user detection, this will be explained latter in Section III-B.

To control the lighting and HVAC systems we require devices capable of interacting with existing systems that offer a way to remotely control these existing systems. These devices can be for example microcontrollers. A microcontroller allows the digital world to interact with the real world through the use of actuators that convert electric signal into mechanical actions. The microcontroller is connected to the actuators (relay) and is able to switch on/off the lights as well as controlling the HVAC. This component is needed because typically, lighting and HVAC systems do not have any type of connectivity other than the electric wires.

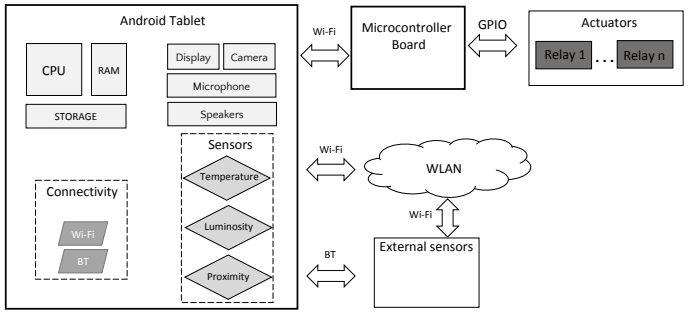

Fig. 2. Hardware architecture of the Hub

B. Software Architecture

In this section we present the software architecture for the mobile apps and explain some of the core functionalities of our solution.

To achieve **building automation** we propose a IFTTT system. The user is able to create personalized chains of conditional statements, called "recipes", they are triggered based on events relevant to the office. With this system we could for example create a recipe that turns on the lights automatically when the user enters the room or shut them down when he leaves.

One of the goals of our solution is **user detection**, in order to accomplish this, we leverage two different types of location systems: WiFi and BLE beacon location. The location detection is handled by the user app. We use the building's WiFi network to determine if the user is inside the building by comparing the media access control (MAC) address of available Access Point (AP)s to a set of known addresses.

The other location system we use is beacon location. There can be a beacon emitting BLE signals and when this signal is detected, we know the user is near the beacon source (the office).

Regarding **office security**, we propose using the tablet's camera and analyzing the consecutive frames for changes. When no user is present in the room and a large enough number of pixels are different, we assume movement has occurred. We can then notify the user that someone is inside the room.

1) Hub App Architecture: We divided the Hub app into several managers, each responsible for several task. Figure 3, represents the architecture of the Hub App.

The User Interface (UI) must have a set of screens that offer:

- Manual control of the lighting and HVAC systems.
- User registration.
- Creation of automation rules.
- Real-time ambient sensors readings.
- Security settings, enable/disable security monitoring and notification email.
- General settings, activate voice control, allow sound.

Communication Layer: The Hub app implements a web server offering an representational state transfer (REST) application programming interface (API) that the user app can use to interact with.

The Event Manager offers publish/subscribe service. It allows other managers to register their interest in certain events and when those happen they are notified. There can exist several different types of events: temperature reading, motion detected, user detection, etc.

The Event Manager is also responsible for the automation logic by enforcing the IFTTT rules. The Automation Manager will check if the condition for the rule has been satisfied and trigger the action. Finally it also has a scheduler responsible for executing repetitive tasks at regular time intervals or at fixed moments.

The User Manager handles user registration, authentication and detection. It tracks the whereabouts of the user inside the building, in association with the User app so that, it is notified when the user enters the building or office.

The Sensor Manager allows access to the tablet's sensors as well as the external sensors. It abstracts external sensors into virtual sensors. It is responsible for a virtual motion detection sensor, by using the camera to detect changes that would indicate motion, as well as other data such as the opening and closing of the door.

The Lighting Manager is responsible for managing the lighting system. It communicates with the lightning controller (microcontroller) and synchronizes the lights state with the UI and vice versa.

The Temperature Manager periodically reads the room temperature and is able to communicate with the HVAC controller (microcontroller) to turn on/off the system when required.

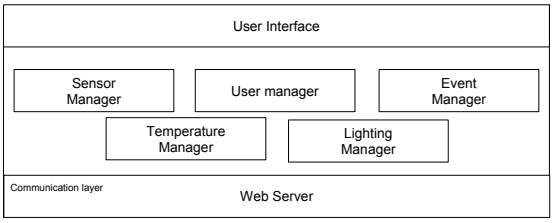

Fig. 3. Hub APP architecture

2) App Architecture: The User App is simpler than the Hub App, the app is divided in UI screens. There are no managers like the Hub architecture, the business logic is handled by each screen separately.

The main screens are the home, lighting, temperature, security and registration screens.

Since the user app can control several Hub apps, we require a simple registration process to add the user to the Hub. We propose showing a QR-Code in the Hub registration screen, this code possesses the Internet Protocol (IP) address for the Hub and a temporary token used for registration. The user app can use the phone camera to scan the QR-Code and automatically register the user in the Hub. Then a virtual representation of the Hub is shown in the home screen.

For user detection to work when the app is not being used, we use a background service to monitor the WiFi and beacons nearby the phone. The interval between scans are one every five seconds for BLE scans and one minute for WiFi, these values are adequate times between scans to be conservative regarding the user's mobile device battery.

IV. Implementation

This section addresses the main decisions adopted regarding the implementation of BASA. Thus, the following sections cover the hardware and software choices made in the development process of BASA.

A. Hardware Architecture

In Figure 4 is shown the required hardware components of our system. We require tablet to run the HUB app, an optional user phone with a User app installed, a device to control the HVAC system (two supported Arduino HVAC or PerOMAS[**?**]), a Edup smart light switch to control the lights. We also need a BLE beacon for user detection inside a room and require the building's WiFi network for user detection inside the building. To allow communication between different components in the system we use the building's WiFi network for local communication and use Firebase servers to allow external access from outside the building's network.

Tablet: We decided to use the Nexus 7 (2013) Android tablet as the base of our solution. We compared IOS, Android and Windows devices and concluded that an Android tablet fitted our needs.

HVAC control: In order for our BAS solution to control an HVAC system, we need to either communicate with a building central controlling unit or physically control the system like a

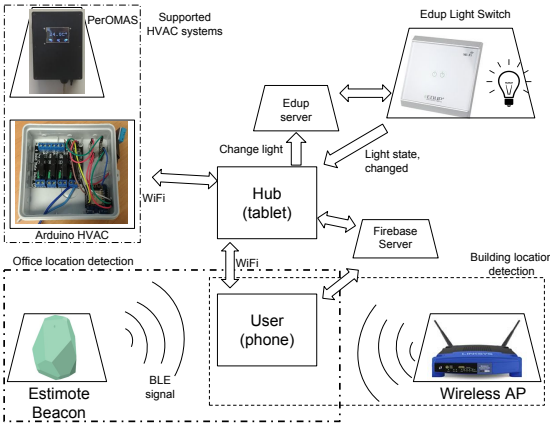

Fig. 4. Hardware implementation architecture

thermostat does. Since most buildings don't have a central control point running some kind of server where you can control it using the Internet, we went with the thermostat route.

We compared several thermostat alternatives with WiFi capabilities and concluded there was non that fitted the requirements of low cost. We decided to make a Arduino based device with built in WiFi, a relay to drive the electric wires and a temperature and humidity sensor.

Besides the mentioned Arduino based solution some offices in our University have installed the PerOMAS automation system that allows remote control of the installed HVAC system. Due to time constrains the Arduino base solution was not installed and we used PerOMAS for HVAC control instead.

Lighting control: Besides controlling the room HVAC, we require control over the lighting system. After some research we had three options: using wireless controllable light bulbs, using some kind of Arduino/relay solution (like the thermostat) or using a wireless controllable light switch.

Our lighting solution should be affordable, have a physical light switch and an API to control it. The smart lights were not supported by our office (fluorescent tube lamp) for these reasons we excluded them, also the Arduino based solution implied assembling the components and that should be kept to a minimum to make BAS more easily deployable.

The lighting option chosen was the Edup Smart Light Switch. It provides a physical light switch, supports the existing infrastructure in the buildings (as long as there is a phase and neutral wire in the wall socket) and is not too expensive.

Originally we proposed to have, the lighting system integrated with the Arduino/HVAC using relay actuators. In the end we decided go with the Edup light switch after searching for lighting products we could control, it is a better solution because of the touch control, the ability to continue to use the lights in case of system failure (no Internet or control mobile application crash) and no need for assembly as described previously.

Unfortunately the wall socket where the light switch was going to be installed was missing the neutral wire and the wait time for electricians to wire the neutral wire to the wall socket would take too long. The alternative found was to use the light switch to control a lamp, as shown in Figure 5.

Fig. 5. Edup Light Switch connected to a lamp to simulate room lights

Bluetooth Beacon: We require a way to determine if a user is near a certain point in the building such as the office.

From the alternatives seen in Section II, we determined that BLE beacons are an option to achieve room level location in a building.

There are many kinds of devices capable of emitting BLE packets: android devices(available in version 5.0), some microcontrolers and commercial beacon products.

We decided to use an Estimote beacon. One reason it was chosen was because we had access to one and did not need to buy it. This beacon also has the advantage of providing temperature readings if needed. Since it is portable we can position it anywhere in the room, enabling better coverage.

B. Software Architecture

Our solution is divided into two separate Android applications, the Hub and the User apps. Both apps work side by side in order to achieve the following functions: user detection, control over HVAC and lighting systems, task automation and finally video monitoring & security. These functions are the building blocks which enables our solutions to achieve building automation, increase user comfort and security monitoring.

1) User Detection: One energy wasting problem is having lighting and HVAC systems turned on when no one is present.

In order for our system to take informed actions to cut wasted energy consumption, we need to know the location of users inside the building.

The location detection is handled solely by the User app. After it confirms the user is inside the building or office it communicates with the Hub. The Hub on the other hand has a timeout system. When a user fails to communicate for a certain amount of time, we presumed he is outside the office or building.

In the building

The building location detection consists in scanning the available WiFi networks and checking if the MAC address of the AP matches a list of pre-scanned APs. Scanning for APs is battery draining and should be kept to a minimum. We decided to start scanning only when the phone connectivity changed. For example when a user enters a building his phone connects to a known WiFi network. We use this event to start scanning if the user is inside the target building. If after the first five

scans (5 minutes) no matching MAC address is found, the scanning is disabled until a new connectivity change happens.

In the office

There is an Estimote beacon in the office broadcasting a BLE signal. The User app is scanning for these signals and when one is found, it checks if the namespace in the beacon package matches the known office namespace. After a correct match is found the User app informs the Hub the user is near the office.

While BLE is a lower-energy system than traditional BT, scanning is still a fairly power-intensive operation. To limit the battery drain, we only start scanning when the user is in the building and stop after he leaves.

2) Lighting control: The lights in the room are controlled by the Edup Light Switch. The switch must be connected to a WiFi network and automatically establishes a connection to Edup server to enable remote control over the switch. There is no official API available to control the switches. In order to understand how the light switch worked, we used Wireshark to sniff the traffic created by the Edup app and tried to replicate the messages.

To control the light switch we must open a Transmission Control Protocol (TCP) socket send a packet with the data shown in Figure 6 to Edup server.

Since there is a physical light switch interface we must update our application status if a button is pressed outside our app. The Edup Light Switch sends a User Datagram Protocol (UDP) multicast message when the lights are changed. We implemented an UDP server to listen for those messages.

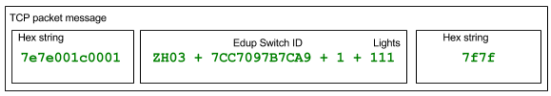

Fig. 6. Edup packet to set the light state.

3) HVAC control: Our HVAC solution uses an Arduino with a webserver to control a relay in order to interface with the existing HVAC system.

The Arduino we implemented to control the HVAC system was written using C++ programming language. It has four jobs: run a webserver to allow remote control, implement Simple Service Discovery Protocol (SSDP) to enable discovery, read temperature/humidity values from a DHT11 sensor and control a four relay module to control the HVAC system.

After the device is first powered up it becomes an AP. The user must connect to this new network and use the browser to choose the WiFi network he want the Arduino to connect.

There is a discovery mechanism to add the Arduino IP address automatically to the Hub app. Our Arduino based HVAC they is programmed to be discoverable using the SSDP protocol using a search target equal to "schemas-basa-pt:service:climate:1".

BASA can also control the PerOMAS system installed in some offices in our campus, we use the provided web-server to interact with the system.

4) Automation: One of the goals of BASA is enabling the automation of lighting and HVAC systems. A simple example of an automation task is turning off the lights when no occupant is inside the office. With our system we aimed to give the user high control and flexibility over the automation process. We allow the user to create chains of simple conditional statements, called "recipes", which are triggered based on events relative to the office.

The recipes contain a list of triggers and actions. A recipe can have one or multiple triggers, when there are multiple triggers we use the logical AND operator, all the triggers must be true in order for the recipe to be triggered.

There are seven triggers available:

- **Temperature:** This trigger allows the user to trigger an action when the office temperature is bellow or above a value specified.
- **Motion:** The motion sensor is triggered when something changes in the field of view of the Hub or no movement detected in X seconds.
- **Speech:** Say the keyword "my assistant" and after the beep you say the phrase specified in the trigger.
- **Time:** The timer trigger allows the user to run certain actions at a specific time(Scheduler).
- **User location:** Allows actions to be triggered based on the user location
- **Light sensor:** The light sensor triggers an action if the light level is bellow or above the value specified.
- **Light state:** This trigger allows the user to trigger an action when the lights are in a specific state.

The actions available are:

- **Light:** Sets the state of the lights.
- **Temperature:** Changes the target temperature and turns on the HVAC if needed.
- **Speech:** This action uses the speakers to say the text specified (text to speech).

In order to show the flexibility of our solution we created the following example recipes:

- If a user is in the office and the temperature is above 25^{o}C then we turn on the cooling.
- If no user is inside the office then turn off all the lights and the HVAC.
- If the user says "turn on lights" then turn on all the lights.
- if the user location status is "user arrives at office", then say "Welcome back".

5) Communication: BASA has several devices that need to communicate with each other as well as external services. To allow communication we implemented the following services:

Rest server - Hub app: We implemented the server using java and the SPARK 1.1.2 framework [5] running at port 5002.

The server exposes a Rest API that allows the User app to communicate with the Hub.

Rest client - Hub and User app : The Rest client uses the Retrofit Android[6] library to allow our apps to initiate

[5]http://sparkjava.com/

[6]https://square.github.io/retrofit/

communication with a server. It is used by the User app to interact with the Hub app server and used by the Hub to communicate with the Arduino, PerOMAS and Weather forecast API.

Firebase - Hub and User app

Originally the plan was to only use the web server, but in our building we can't run web servers in the WiFi network. When we tried to contact the Hub it did not receive any request. Then we changed to a private wireless network it all worked normally.

The solution we found for connecting to the Hub when the User app is outside the wireless local area network (WLAN) was Google Firebase. It is a platform that offers multiple services. We use two services, the real-time database and the storage. We use their real-time database to synchronize application data across clients and store it on Firebase's cloud. The application data we store are the temperature, the temperature the thermostat is set to, the light state, the beacon namespaces and MAC addresses for user location and finally data related with live camera view of the office and recorded videos storage location URLs. The storage service is used to upload the recorded videos and live preview images, captured by the security system.

Firebase solves us two problems, one is users can access our solution from home and access the live camera, Firebase is used as a middle man to communicate. The second problem it fixed was user detection, since our building is big, the private network where the tablet is connected can't be access in most of the building. A user entering the building could not communicate with our Hub and notify of it's presence since they were in different networks. We use Firebase to add our location messages that are then received by the Hub to achieve and building level user detection.

6) Security - Motion detection, video recording, Live view: With our office being like an extension of our home, we felt the need to have some sort of video surveillance system. We created a system capable of detecting if a object is moving inside the office, identify if a user is inside the office and if not, send a notification to the user and record a small video.

Motion detection: The motion detection works by comparing two camera frames. If a percentage of pixels are different we assume motion has happened. Both the percentage of different pixels and the detection period are configurable using the UI.

In order for a camera frame pixel to be considered different the RGB value of the pixel must vary more than 50 in a scale between 0 and 255. The percentage of different pixels for a positive detection is user defined, the default is 0.5%.

Video recording: When motion is detected and no user is inside the office we record 30 seconds videos. We do not record constantly, just after motion is detected. These videos are uploaded and stored to Firebase servers, where the user can latter watch them using the user app.

Live Camera

We can watch in real time pictures from the office camera. When the user is using the User app live camera functionality, the Hub saves a camera frame every second to the Firebase server. We have seen a minimum delay of 4 seconds between the live pictures and the time they were taken.

7) Hub app: The Hub app suffered a few changes from the initial architecture with the addition of more functionalities.

We added two more managers to the Hub app: **The Speech Recognition Manager** was inspired by Amazon Echo product and offers our user voice recognition. This new feature is currently used with our automation solution to allow the user to trigger recipes using voice commands.

This manager uses the CMU Sphinx[7] offline keyword speech recognizer. When it detects the keyword, it shuts down and we launch Google online speech recognition. We use this other speech recognition because it offer better results, the reason the manager uses two different speech recognizes is because using only Google recognizer has higher bandwidth cost since we would be constantly sending audio.

The Video Manager does two jobs, it records 30 seconds videos when no user is present in the office and motion is detected. The other job is offering a live preview of the office, an user can use the user app to watch in real time the office.

User interface: The final UI of the Hub app is composed of three main screens: a lighting screen, a temperature screen and general menu that contains several advanced features.

The **lighting screen**, allows the user to control the lighting environment in the room.

The **temperature screen**, shown in Figure 7 gives visual information about the current weather outside. There is also a virtual thermostat where the user can select the temperature he desires.

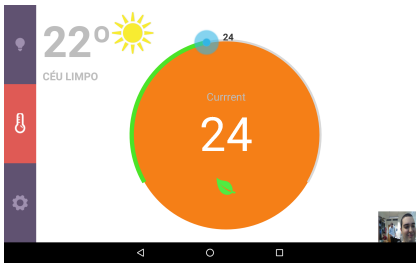

Fig. 7. Temperature fragment.

There is a **general menu** that contains the other screens that enable additional features. This menu contains:

- **User registration:** Allows a person to setup an account in Hub.
- **Recipe management:** Contains a list of the current recipes in the Hub and allows the user to create, edit, disable/enable and delete them.
- **Event history** Shows a list events ordered chronologically.
- **Statistics screen** Shows temperature, lighting status, luminosity and occupation graphics.
- **Settings screen** Allows the user to configure the Hub app.

[7]http://cmusphinx.sourceforge.net/wiki/

The **user registration screen** allows the user to register using the user app or an email address. If the user uses the Android user app, a visual registration token is shown in the form of a QR-Code. The QR-Code contains a temporary id and the IP address of the Hub server. This temporary id is latter needed when the user registers and exists to ensure only people inside the office can register and not just anyone that knows the IP address. If the user chooses to register with just the email we use Maingun[8] API to send a email to the provided email address with a identification QR-Code for user login.

In the **recipe screen**, the user can create new recipes, first the trigger/s is selected, only then can the action/s be picked. Figure 8 shows one of the screens to create a recipe.

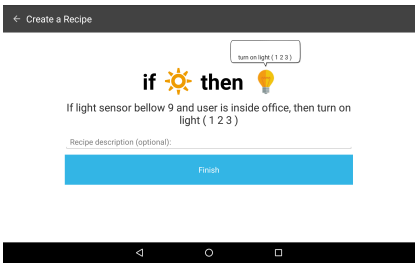

Fig. 8. Final recipe creation fragment.

The **settings screen** allows the user to configure many different parameters in the app, this screen also allows the user to:

- Specify an area in the camera frame where motion is ignored.
- Scan for Arduino/HVAC, shows the list of available Arduino based HVAC in the network.
- Enable Kiosk mode, this mode disables the home and power button of the Android device. It prevents a user from closing the app.

8) User app: The User app allows user detection, remote control over the office, access to live image preview of the office, notifications when movement is detected, and video history of recorded videos.

The User app allows the user to create zones, in a zone the user can add and control Hub devices. To add a device the user uses a screen with a camera and scans the Hub app Qr-Code available in the Hub's registration screen. After the Hub device is added three icons representing three services appear in the zone screen: control lighting, HVAC and security systems. This is shown in Figure 9.

The lighting and HVAC screens are straightforward, they allow the user to remotely control a Hub's lighting and HVAC systems. The security screen enables live camera visualization of the Hub's camera and access to a list of recorded videos. When motion is detected by the Hub all registered users are notified via Android notification.

[8]http://www.mailgun.com/

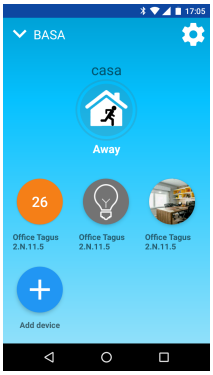

Fig. 9. HomeFragment screen with the added Hub.

V. Evaluation

In order to evaluate the developed solution, we deployed the BASA system in one office. Our goal was to validate the basic operation of the system and it's ability to provide energy savings through user detection and automation. Provide it's users with their desired temperature settings. Test the motion detection system developed. Finally evaluate the UI of the automation solution developed.

All the tests were made using an Android nexus 7 2013 tablet, a motorola moto G3 Android phone, the Arduino based HVAC DHT11 temperature sensor and a Estimote BLE beacon.

User detection - Room: We measured the time it took the User app to detect the BLE beacon inside the office, the user was initially outside the office entered the room. The cumulative distribution function for the test is shown in Figure 10.

We observed that in most cases our system detected the user presence very quickly within few seconds but in other occasions it seemed to take up to 30 seconds. To We noticed once it detected the Estimote beacon it found it correctly the consecutive times but seems to take some time for the initial discovery. We consider the time to detect the user presence is adequate for turning on the HVAC system in the user presence but not perfect for turning on the lights when the user enters the room.

For cases when the user wants the lights to be on when he enters the room and if the detection time is considered insufficient, a automation recipe can be created for turning on the light when movement is detected and the user is inside the building. Nevertheless we consider the room detection time adequate for rooms with ambient light from a window but not for dark rooms.

User detection - Building: Several measurements were taken to determine the response time the application takes to identify the user is inside the building. The cumulative distribution function from the measurements is shown in Figure 11.

Building user detection does not need very quick response time unlike room detection, building location will be mostly

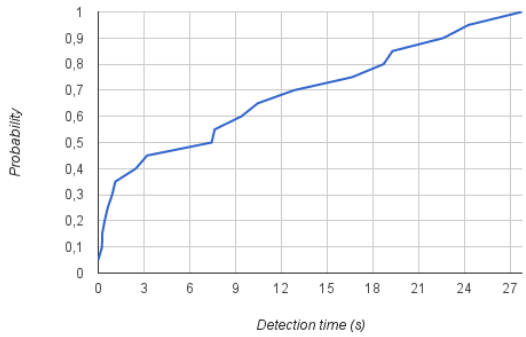

Fig. 10. Cumulative distribution function of the time it takes to detect a user when user enters room

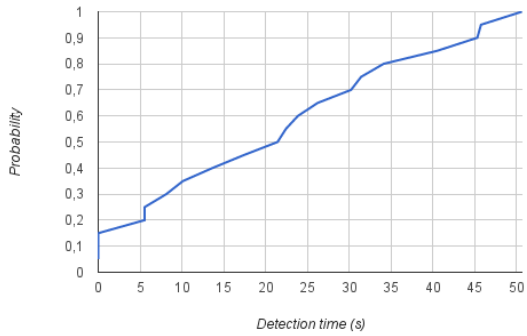

Fig. 11. Cumulative distribution function of the time it takes to detect a user after he enters the building

used to preheat the room before the user arrives. The response time observed shows an acceptable response time.

Motion detection: To test motion detection, we conducted tests with three different people with different heights. Each person entered the room at different speeds, the obtained results are shown in Table I.

Our system was able to correctly identify a person walking inside the room every time, Figure 12 shows the different pixels in the camera frames identified by our algorithm. Regarding false positives, they don't seem to be a big problem since our system allows the creation of a no monitoring area in the camera frame to use in cases of rooms with window view.

The motion test was performed during day hours, in these conditions the camera is able to capture good frames. We did not perform test during the night time but likely our system will have problems detecting movement in completely dark rooms. To fix this problem we can install sensors to detect if the office door is open. If the door is opened and there is not enough light our system could turn on the lights momentarily to enable motion detection.

	Person 1 detection	Person 2 detection	Person 3 detection
Slow walk	20/20	20/20	20/20
Normal walk	20/20	20/20	20/20
Run	20/20	20/20	20/20

TABLE I
MOTION DETECTION TESTS FOR THREE PEOPLE WITH DIFFERENT HEIGHTS, 20 MEASUREMENTS EACH.

Fig. 12. Motion detection - Frame before and after, motion detected

Temperature and Luminosity: In order to test the functionality of the luminosity and temperature sensors in the system, these were used to collect ambient data.

To test the luminosity sensor behavior we turned on the lights in the room and allowed the Hub to record the values for a while, then we turned off the lights and waited for a few minutes then repeated the same steps one more time. In Figure 13 we can observe two different gaps where the lights were off. As expected the Android luminosity sensor operates as intended and provides the system with correct data.

To test the external temperature sensor the HVAC system was turned on and the temperature was recorded. In Figure 14 we can observe a small decline in the temperature after the HVAC system was turned on. The temperature was lowered by 2°C and then went back to the 24 °C after the HVAC system was turned off.

Automation: To evaluate the usability of the IFTTT interface (automation recipes), user usability tests were performed with a total of 20 users. The users who performed the tests were not familiar with our system and had no previous experience with similar IFTTT solutions.

The users were asked to create two distinct recipes:

- If the user left the office, then tun off all the lights.
- If the temperature is above 25 °C, then set temperature to below 20 °C.

The test results were then compared to an expert user

Fig. 13. Luminosity Readings

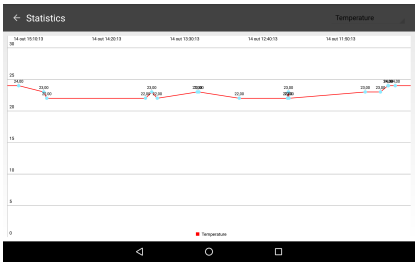

Fig. 14. Temperature Readings

to determine the usability of the system. Comparing the results presented in Table II and Table III we observe that an inexperienced user took on average 19 seconds more to complete the first task in comparison with the experienced user. On average the inexperienced user took more 9 seconds to complete the second task. These values are to be expected, as first time users always under perform expert users. The extra time the first time users took is reasonable. We conclude there were no major problems in the UI used to complete the tasks.

	Task 1	Task 2
Mean	36	27.8
Standard deviation	9.31	5.96

TABLE II
USER USABILITY TESTS, MEAN AND STANDARD DEVIATION

	Task 1	Task 2
Mean	17	18.4
Standard deviation	0.89	1.49

TABLE III
EXPERT USER USABILITY TESTS, MEAN AND STANDARD DEVIATION

CONCLUSION

In this document, we present BASA a system that uses regular Android tablet devices running an Android application to create a BAS. Our system aims to improve the comfort of office users, minimize wasted energy consumption of the building and provide office security monitoring. This is achieved by using a tablet (Hub) mounted in a wall in the office acting as a BAS, as well as, other devices to control the lights and HVAC system. We added an IFTTT functionality that allows regular users to personalize their office automation to improve comfort and reduce wasted energy. We mitigate some problems related to user neglect by detecting the user presence in the room or building. This allows the IFTTT automation component of our system to take certain actions based on the room and building occupancy, for example turning off the lights when no user is present inside the room.

The results presented in this thesis provide a strong foundation for future work in BAS leveraging Android devices. One aspect this thesis did not focus and is pertinent to its future is the ability to connect multiple Hub instances (rooms) and take coordinated actions to improve energy efficiency and user comfort across entire sections of a building.

Another future improvement to the system is the ability for the tablet itself to act as a BLE beacon. This means we can use the Android device running the Hub app to achieve room user detection and can skip the external beacon device.

During the development of our prototype, we were unable to install the Edup light switch in the wall due to a missing neutral wire. A neutral wire will have to be wired to the to the wall in order to deploy the prototype as originally intended.

One aspect of our system we were not completely happy with was the user detection time in a room. The detection time should have been much lower, further research and testing must be done to determine if it is the Android system who is not receiving the BLE signals due to energy saving policies or if it is the library by Estimote that we used and is not operating as it should.

REFERENCES