

5-1-2015

Tracking and Monitoring Unmanned Aircraft Systems Activities with Crowd-Based Mobile Apps

Tejaswini Papireddy

University of Nevada, Las Vegas, papiredd@unlv.nevada.edu

Follow this and additional works at: <https://digitalscholarship.unlv.edu/thesesdissertations>



Part of the [Computer Sciences Commons](#)

Repository Citation

Papireddy, Tejaswini, "Tracking and Monitoring Unmanned Aircraft Systems Activities with Crowd-Based Mobile Apps" (2015). *UNLV Theses, Dissertations, Professional Papers, and Capstones*. 2408.
<https://digitalscholarship.unlv.edu/thesesdissertations/2408>

This Thesis is protected by copyright and/or related rights. It has been brought to you by Digital Scholarship@UNLV with permission from the rights-holder(s). You are free to use this Thesis in any way that is permitted by the copyright and related rights legislation that applies to your use. For other uses you need to obtain permission from the rights-holder(s) directly, unless additional rights are indicated by a Creative Commons license in the record and/or on the work itself.

This Thesis has been accepted for inclusion in UNLV Theses, Dissertations, Professional Papers, and Capstones by an authorized administrator of Digital Scholarship@UNLV. For more information, please contact digitalscholarship@unlv.edu.

TRACKING AND MONITORING UNMANNED AIRCRAFT SYSTEMS
ACTIVITIES WITH CROWD-BASED MOBILE APPS

By

Tejaswini Papireddy

Bachelor of Engineering, Computer Science

Chaitanya Bharathi Institute of Technology, India

2013

A thesis submitted in partial fulfillment
of the requirements for the

Master of Science - Computer Science

School of Computer Science
Howard R. Hughes College of Engineering
The Graduate College

University of Nevada, Las Vegas

May 2015



We recommend the thesis prepared under our supervision by

Tejaswini Papireddy

entitled

**Tracking and Monitoring Unmanned Aircraft Systems Activities with
Crowd-Based Mobile Apps**

is approved in partial fulfillment of the requirements for the degree of

Master of Science in Computer Science

Department of Computer Science

Juyeon Jo, Ph.D., Committee Chair

Yoohwan Kim, Ph.D., Committee Member

Ajoy K Datta, Ph.D., Committee Member

Venkatesan Muthukumar, Ph.D., Graduate College Representative

Kathryn Hausbeck Korgan, Ph.D., Interim Dean of the Graduate College

May 2015

Abstract

Tracking and Monitoring Unmanned Aircraft Systems Activities with Crowd-Based Mobile Apps

By

Tejaswini Papireddy

Dr. Ju-Yeon Jo, Examination Committee Chair

Associate Professor, Department of Computer Science

University of Nevada, Las Vegas

An Unmanned Aircraft System(UAS) is an aircraft without human pilot onboard. They can be controlled remotely from a ground control station (GCS) or can fly autonomously based on pre-programmed flight plans. Tracking and monitoring these UASs activities in the National Airspace System (NAS) is one of the major challenges related to UAS. This system tracks a set of UASs at a given point of time based on the user-submitted tracking information. This thesis presents the basic knowledge of UAS, types of UAS, advantages, and challenges with UASs. The implemented Android application allows the users to submit the observed UAS information to a tracking database server by taking a picture of the UAS. The monitoring system is implemented in Android for retrieving and displaying the UAS activity data from the database server. This system is expected to promote safer airspace by providing the UAS activity information to manned aircraft pilots and other UAS operators to avoid collision.

Acknowledgements

I would like to express my sincere gratitude to my committee chair, Dr. Ju-Yeon Jo for her strong support and guidance throughout my thesis. It gives me great pleasure to acknowledge her valuable assistance.

I am extremely grateful to Dr. Yoohwan Kim for his indispensable advice, information, and valuable guidance on different aspects of my research work.

I would like to thank my parents, Anitha and Sreenivasulu Reddy for their continuous support and motivation. Their valuable thoughts and immense cooperation was imperative to my completion of this degree. I owe thanks to my friend Sai Priyank for his encouragement.

I would also like to thank Dr. Venkatesan Muthukumar for being a part of my committee. I convey my special thanks to the graduate coordinator, Dr. Ajoy K Datta for all his support and guidance throughout my Master's Program.

Table of Contents

Abstract	iii
Acknowledgements	iv
Table of Contents	v
List of Tables	ix
List of Figures	x
Chapter 1 Introduction	1
1.1 Unmanned Aircraft System.....	1
1.1.1 Unmanned Aircraft.....	1
1.1.2 Data and Control link Functions.....	1
1.1.3 Ground Control Station	2
1.2 Applications	2
1.2.1 Government.....	3
1.2.2 Fire Fighting.....	3
1.2.3 Energy Sector	3
1.2.4 Agriculture, Forestry and Fisheries	3
1.2.5 Earth Observation and Remote Sensing.....	3
1.2.6 Communications and Broadcasting.....	3
1.3 Advantages	4
Chapter 2 Background	5
2.1 Background about UASs.....	5
2.2 Popular Civilian UASs.....	5

2.2.1 Parrot Bebop	5
2.2.2 3D Robotics Iris+	6
2.2.3 DJI Phantom Inspire	7
2.2.4 Turbo Ace Matrix	7
2.2.5 Hubsan x4	8
2.2.6 Blade 350 Qx	9
Chapter 3 Motivation	10
Chapter 4 Challenges faced by UAS	12
4.1 Safety.....	12
4.1.1 Collision Avoidance	12
4.1.2. System Reliability.....	14
4.1.3 Human Factors.....	14
4.1.4 Weather.....	15
4.2 Security.....	15
4.2.1 Ground Infrastructure	15
4.2.2 Communications Signal Security	16
4.2.3 Data Security	16
4.3 Air Traffic	16
4.3.1 Air Traffic Management.....	17
4.3.2 Information Networks.....	18
4.3.3 Navigation	18
4.3.4 Emergency Flight Recovery and Termination Systems	18

4.4 Regulation	19
4.4.1 Definition and Classification Schemes	19
4.4.2 Standards	19
4.4.3 Pilot Certification	20
Chapter 5 UAS Tracking and Monitoring Android Application	21
5.1 About Application	21
5.2 Technologies Used	22
5.2.1 Android	22
5.2.2 Eclipse and Android Development Tools	25
5.2.3 mariadb	27
5.2.4 Tomcat Server	27
5.2.5 RESTFUL Web Services	27
Chapter 6 Software Architecture	29
6.1 Use-case diagram	29
6.2 Control Flow Diagram	30
6.3 Sequence Diagram	32
Chapter 7 User Interface	34
7.1 User Interface Controls	34
7.2 Screenshots of the application Functionality	35
Chapter 8 Database	39
8.1 Connection between Android application and Database	40
Chapter 9 Web Services	42

9.1 RESTFUL.....	44
9.1.1 HTTP Methods	45
9.2 Advantages of REST over SOAP.....	46
Chapter 10 Conclusion and Future Work.....	53
Bibliography	54
VITA.....	59

List of Tables

4.1 Analysis of the onboard Systems	13
8.1 UAS Register Database.....	40

List of Figures

1.1 Unmanned Aircraft System Model	2
2.1 Parrot Bebop	5
2.2 3D Robotics	6
2.3 DJI Phantom Inspire	7
2.4 Turbo Ace Matrix	8
2.5 Hubsan x4	8
2.6 Blade 350 Qx	9
4.1 Non-cooperative Onboard Systems	13
4.2 Cooperative Onboard Systems.....	13
5.1 Folders in an android Application.....	23
5.2 Application Launcher Icon	25
5.3 AVD Selection	26
6.1 Use-case diagram	29
6.2 Control flow diagram to add UAS	30
6.3 Control flow diagram to select UAS.....	31
6.4 Sequence diagram to add UAS	32
6.5 Sequence diagram to select UAS.....	33
7.1 Opening Screen.....	35
7.2 UAS Dropdown Menu	35

7.3 UAS on Google Maps.....	36
7.4 UAS Description.....	37
7.5 Time Menu.....	37
7.6 Activity to add UAS.....	38
9.1 Communication Via Web Service	43
9.2 Communication using REST	45

Chapter 1

Introduction

1.1 Unmanned Aircraft System

The system that describes the aircraft, data link and control station is called an unmanned aircraft system. Fig 1.1 describes the Unmanned Aircraft System Model.

1.1.1 Unmanned Aircraft

Unmanned Aircraft is defined as a powerful system, that does not carry a human operator, uses aerodynamic forces to provide vehicle lift, can fly autonomously or be piloted remotely, can be expendable or recoverable, and can carry a payload [1].

Our focus in this thesis is civilian UASs.

A civilian UAS is an unmanned aircraft system which has the ability to fly autonomously or through a remote control device. It allows humans, surveillance of hazardous locations and eliminates the risk of physical harm [2].

1.1.2 Data and Control link Functions

- Uplink from the ground station or a satellite to send control data to the UAS.
- Downlink from the UAS to send data from the onboard sensors and telemetry system to the ground station.
- A means for allowing measurement of azimuth and range from the ground station and satellite to the UAS to maintain good communication between them [3].

1.1.3 Ground Control Station

At the Ground station, UAS is operated remotely by a team of two: a pilot and a sensor operator. The pilot's primary function is flying the plane, while the sensor operator monitors the performance of the many different sensor systems utilized by the UAS. They look at screens that show what the UASs cameras are seeing, as well as maps of the area and normal instrument readings. They fly the plane with joysticks just like the ones we would see on a video game.

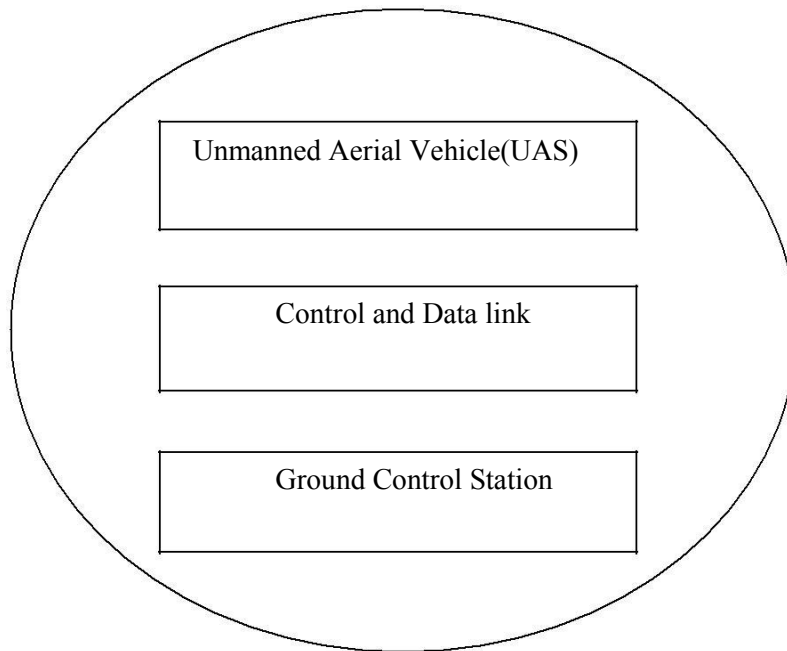


Fig 1.1 Unmanned Aircraft System Model

1.2 Applications

UASs are used in wide range of applications. They are increasingly being employed for non-military applications [4].

1.2.1 Government

- Law Enforcement (Police, civil security)
- Border Security
- Coastguard

1.2.2 Fire Fighting

- Forest Fires
- Emergency Rescue (e.g. - Mountain Rescue)

1.2.3 Energy Sector

- Oil and gas industry infrastructure
- Electricity grids/ distribution networks

1.2.4 Agriculture, Forestry and Fisheries

- Environment Monitoring
- Crop Dusting
- Optimizing use of Resources

1.2.5 Earth Observation and Remote Sensing

- Climate Monitoring
- Aerial photography, mapping and surveying
- Seismic Events

1.2.6 Communications and Broadcasting

- VHALE platforms as proxy-satellites
- MALE/ S/ MUAS as short term, local communications coverage [5]

1.3 Advantages

The advantages of using an Unmanned Air Vehicle, compared to use of a manned aircraft, are that:

- They do not contain, or need, a qualified pilot on board.
- They can enter any environment that is dangerous to human life.
- They reduce the exposure risk of the aircraft operator.
- They can stay in the air for up to 30 hours, performing a precise, repetitive raster scan of a region either in day or at night or in any other environmental condition under computer control.
- They can be programmed to complete the mission autonomously even when contact with its GCS is lost [6].

Chapter 2

Background

2.1 Background about UASs

UASs were mainly launched for reconnaissance and are still used with a similar intent. One of the recorded usages of drones for the first time was by Austrians on August 22, 1849. They launched around 200 pilotless balloons mounted with bombs against the city of Venice. The next such usage was after no later than two decades, balloons were flown in the U.S. Civil War in 1862, to find the information about the enemy.

In the first world war, aerial surveillance was extensively used. Analysts tried with different means to find out the enemy's movement [7].

2.2 Popular Civilian UASs

2.2.1 Parrot Bebop

Parrot Bebop Drone is robust and safe. It is known for the emergency function that automatically stops the propellers and lands the drone in case of emergency [8].



Fig 2.1 Parrot Bebop

- Duration: 11-12 Minutes
- Speed: 13 meters/sec
- Weight: 0.903 lbs
- Price: \$500 (Approximately)

2.2.2 3D Robotics Iris+

The 3DR IRIS+ is used primarily for shooting photos and video from new perspectives. IRIS+ is a robot that will automatically fly itself where we tell it to go, while keeping a camera always steady with two-axis gimbal stabilization(allows rotation of an object) [9].



Fig 2.2 3D Robotics

- Duration: 16-22 Minutes
- Speed: 13 meters/sec
- Weight: 2.82 lbs
- Price: \$750 (Approximately)

2.2.3 DJI Phantom Inspire

The Inspire One rocks a three-axis gimbal mounted camera capable of shooting 12 megapixel images and 4K video at 30, 25 and 24fps. It is primary known for shooting videos with 4K video [10].



Fig 2.3 DJI Phantom Inspire

- Duration: 18-22 Minutes
- Speed: 22 meters/sec
- Weight: 6.47 lbs
- Price: \$3000 (Approximately)

2.2.4 Turbo Ace Matrix

This UAS is used in aerial video and surveillance applications. It is used to take ultra stabilized videos at 18,000 feet elevation to chase scenes exceeding 60mph to preprogrammed long distance autonomous flights in 40mph winds [11].



Fig 2.4 Turbo Ace Matrix

- Duration: 48 Minutes
- Speed: 26.82 meters/sec
- Weight: 8 lbs
- Price: \$4000 (Approximately)

2.2.5 Hubsan x4

It is the best and light weight micro quadcopters. It is capable of rotating 360 degrees [12].



Fig 2.5 Hubsan x4

- Duration: 7-8 Minutes

- Speed: 11.17 meters/sec
- Weight: 1.2 lbs
- Price: \$50 (Approximately)

2.2.6 Blade 350 Qx

It is one of the more popular, affordable hobbyist models available to the public [13].



Fig 2.6 Blade 350 Qx

- Duration: 10-15 Minutes
- Speed: 31.29 meters/sec
- Weight: 1.49 lbs
- Price: \$470 (Approximately)

Chapter 3

Motivation

In terms of a normal aircraft, the first generation of communication relied solely on radio and telephone communication. At established points along the airways, pilots were expected to report their time of arrival and altitude and their estimated time of arrival over the next checkpoint. In the Air Traffic Control center, controllers wrote the message on a blackboard and tracked flights by moving a marker on a tabletop map. In a later improvement, paper strips marked with flight data were posted in the order of their estimated arrival at each reporting point or airway intersection.

The second generation of communication came with the introduction of radar after World War II. In the 1950's, airport surveillance radars were introduced at major airports to provide data on arriving and departing aircraft within roughly 50 miles [14].

For a normal aircraft, as it travels through a particular airspace division, it will be monitored by one or more air traffic controllers responsible for that division. The controller will monitor this plane and give instructions to the pilot. As the plane leaves that particular airspace division and enters another division, the air traffic controller passes it off to the controllers responsible for the new airspace division [15].

Since there is a significant role played by the pilot for the flight to travel, there are many challenges that need to be addressed in order to integrate UASs with National Airspace system.

In the FAA Modernization and Reform Act of 2012, the Congress directed FAA to propose a rule to integrate small UASs into the National Airspace System by the end of fiscal year 2015.

UAS operations poses various threats to the stability of the National Airspace in terms of the safety of planes, passengers on the ground, and people and places near airports. There are several risk scenarios that could leave aircraft vulnerable to mid-air or runway collisions with other aircraft or collisions with stationary structures, leading to injuries, deaths, and property damage. Although this risk is present with manned aircraft, it is expected to increase with UAS given the lack of an on-board human operator and the resulting inability to directly sense and avoid other aircraft. Furthermore, UASs are susceptible to malicious hijackers who aim to perform terrorist attacks on communication networks, buildings, and people. To promote proper integration of the UAS within the NAS, the U.S. government has so far invested nearly 2 billion dollars in the NextGen project [16,17,18].

As a part of this Federal Aviation Administration proposed rules that are designed to improve safety standards for the operation of unmanned aircraft weighing less than 55 pounds. One of those rules proposed on February 15, 2015 is that the aircraft should be registered with FAA [19].

Looking at above problems, I felt that there is a need to create an application to track UASs, at the same time it should be easy to use. Since Android is the best source to implement all these, I have designed an android application to track and monitor UAS activities.

Chapter 4

Challenges faced by UAS

4.1 Safety

Successful integration of UASs in NAS requires assurances that they can safely operate within the constructs of a commonly shared aviation system and environment. UASs must demonstrate that they do not pose any hazard to other aircraft or to any persons on the ground. It means they must provide for an equivalent level of safety to manned aircraft. The UAS community is keenly aware of the safety concerns, specially concerning the poor reliability track record of UAS systems, and is in a verge to improve this record [20].

The following are the safety concerns.

4.1.1 Collision Avoidance

There is an increased use of unmanned aircraft systems in the National Airspace System [21]. The problem of detecting and avoiding aircraft and other objects for UASs is a difficult challenge. To avoid collisions, UASs must have a “see and avoid” capability that allows them to detect and safely move on.

To avoid collisions, special onboard systems are integrated in the UAS. The onboard systems of UAS can be either cooperative or non-cooperative.

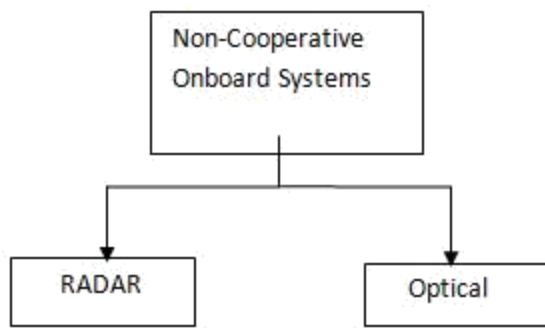


Fig 4.1 Non-cooperative Onboard Systems

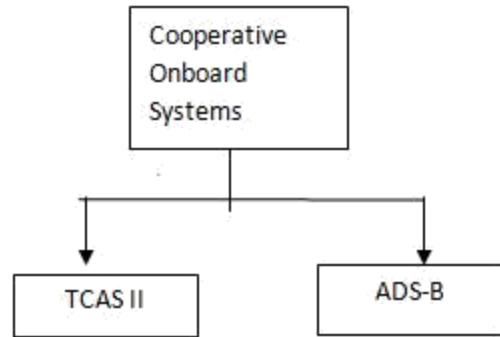


Fig 4.2 Cooperative Onboard Systems

A Non-cooperative aircraft in Fig 4.1 refer to those not possessing or using cooperative systems. A Cooperative aircraft in Fig 4.2 refers to those aircraft possessing systems capable of announcing their position to aircraft and/or the air traffic system [22,23].

	Non-Cooperative		Cooperative	
Equipment	RADAR	Optical	TCAS II	ADS-B
Coverage(in kms)	7	1	33.3	400
Right/Left (in degrees)	180	100	360	360
Upper/Lower(in degrees)	100	100	360	360
Advantage	Weather-proof	Cheap	Cheap	Cheap
Disadvantage	Expensive	Not weather-proof	Opponent needs transponder	Opponent needs ADS-B

Table 4.1 Analysis of the onboard Systems

Table 4.1 explains the equipment used in different onboard systems, their coverage, the amount of rotation, advantages and disadvantages of the non-cooperative and cooperative onboard systems.

4.1.2 System Reliability

The poor reliability record of UAS systems is one among the principal inhibitors to the integration and wide-spread acceptance of UASs. If no improvements are made, this issue will probably stand as the greatest obstacle. UASs have a high accident rate when compared to manned aircraft. According to a recent Defense Science Board review, nearly 85 percent of all UAS accidents are a result of equipment failure [24].

4.1.3 Human Factors

According to the OSD UAS Reliability Study, 17 percent of UAS accidents in the military are caused by human failures compared to 85 percent for manned aircraft [24]. The report also states that "Adaptation of the cockpit environment to the ground control station is more difficult than predicted". So one among the objectives in designing a UAS control station will therefore be to increase situational awareness. This is an approach being considered by the Air Force Research Labs where they are developing logic in UASs that will permit them to make decisions autonomously, even when instructed otherwise by a pilot if it is known by the system that an instruction will lead to an unnecessary hazard [25]. The assumption is that the vehicle will have a better awareness of itself and its environment than a distant operator.

4.1.4 Weather

UASs are lighter, slower, and fragile consequently are more uniquely sensitive to certain meteorological events such as surface/terrain-induced (boundary layer) winds, turbulence, icing, extreme cold, and precipitation. In most UAS weather accidents, the vehicles were not equipped with sensors and the ground control was not fully aware of the hazardous meteorological conditions that existed at the time. Despite the weather vulnerabilities of UASs, they do have more real-time, ground based weather information available to them than pilots of most manned aircraft. This suggests that improvements in ground-based weather detection and information distribution systems will improve the ability of UAS operators to forecast, detect, and avoid hazardous weather.

4.2 Security

Security requirements of the vehicle, ground control station, data link infrastructure and even the data must be a fundamental consideration in system design. In addition to being vulnerable to security breaches, UASs themselves also pose a security threat.

4.2.1 Ground Infrastructure

The operation of UASs is conducted from ground-based facilities. These facilities can vary from small mobile units to elaborate, interconnected, global systems. This becomes a more complex issue as the control functions and infrastructure of some ground operations may be distributed in various locations within the U.S. and around the world. Apart from the UAS control facility, the communication infrastructure will need to have redundancies and alternate paths.

4.2.2 Communications Signal Security

Ground based links are used for controlling the vehicle, monitoring, and air traffic communications. These links are subject, to varying degrees, vulnerable to jamming, spoofing, and interference. To prevent this from happening, a system of high-integrity, secure data links between the aircraft, the ground control stations, and air traffic facilities will be a fundamental requirement in approving UAS operation in the NAS.

4.2.3 Data Security

Aviation data will be used by UAS operations to plan flights. To prevent the possibility of intentional corruption of the data safeguards must be assured. To address data security concerns, the Department of Defense is developing programs to prevent cyber attacks and a suite of technologies, while providing managers of the information system an ability to see, counter, tolerate, and survive such attacks [26]. Security of the ground control stations and data link infrastructure is also a critical requirement for UAS integration[27].

4.3 Air Traffic

To assure the safe and efficient integration of UASs into air traffic operations, it will require UASs to operate within the constraints of the evolving air traffic system. To assess the potential impact of UASs on air traffic operations we will have to depend on the UAS types, numbers, operating environments, frequency of flights and performance characteristics.

4.3.1 Air Traffic Management

To accommodate UASs, operational procedures are required to enable consistent handling by UAS pilots and Air Traffic Controllers. In addition, the pilot(who controls UAS from the ground) of a UAS must have the capability to sense and avoid other aircraft [28].

4.3.1.1 Potential NAS-wide Impacts

The varied vehicle performance and flight characteristic of UASs may prove to be challenging to air traffic service providers and their supporting systems. Because very few UASs have interacted with the air traffic system till date, it is difficult to predict their impacts. The extent of UAS impacts on air traffic management will be dictated as much by UAS performance as by market developments.

4.3.1.2 Controller Impacts

Controller roles may also affect UAS operations, in instances where controllers have handled UASs. Another issue concerning UASs is the registration number. Because a controller is only interested in speaking with the pilot of a UAS, there will probably be a requirement for the ground control station to have a registration number separate from the vehicle (or vehicles).

4.3.1.3 Contingency Procedures

Contingency and emergency procedures will also have to be understood by controllers in case of lost communications or if a vehicle malfunction affects a change in the planned route. The procedures to be taken by the vehicle will need to be communicated or predictable to the controller.

4.3.1.4 Wake Turbulence Separation

A final issue for air traffic management and controllers is to consider the effect of turbulence on UASs.

4.3.2 Information Networks

UASs operating autonomously are reliant on accurate and timely data for navigational guidance, flight path optimization etc. In addition, data is needed by UAS ground control stations for planning, tracking of aircraft movements, weather conditions and traffic avoidance. Ideally, these data requirements will align with the data being processed, distributed, and communicated for manned flights by the air traffic system. The System Wide Information Management (SWIM) system concept is being developed to acquire, process, store, and spread information on traffic, systems, and weather conditions. SWIM will provide an information sharing infrastructure to support enhanced situational awareness and improved collaborative decision making.

4.3.3 Navigation

For UASs to operate in the future airspace, they will likely require enhanced navigational capabilities in order to meet the Required Navigational Performance (RNP) and reduced vertical separation minima (RVSM) expected for manned aircraft. For navigational guidance, the global positioning system (GPS) is the most reliable and affordable technology as other avionics are heavy to be equipped into the UAS.

4.3.4 Emergency Flight Recovery and Termination Systems

Emergency flight recovery and termination systems are an essential part of operating safely in civil airspace. These systems act to avoid or minimize the consequences

associated with accidents such as on-board system failures or lost communication. Emergency flight recovery systems typically have a preprogrammed set of instructions that tell the vehicle to take a specific action based on the emergency. Flight termination systems refer to self-destructive devices. Emergency recovery and termination actions can be initiated by the vehicle or the ground-based pilot. A final issue pertains to controller training on how UASs will react during lost links or other emergencies that result in a flight termination procedure. This information should be ideally included in the flight planning process and the information should be rapidly accessible by air traffic controllers, to assist them in keeping traffic clear of the UAS during its recovery operation [27,29].

4.4 Regulation

Regulatory requirements define the operational boundaries of UAS certification, flight operations, and operator qualifications.

4.4.1 Definition and Classification Schemes

Regulations will require clear definitions and distinct classifications among UAS types. As of now, there is no universally accepted definition or standard classification for UASs.

4.4.2 Standards

There are currently no published standards specific to UAS systems, operations or operator qualifications. There are several organizations working to develop standards for UAS systems and operations. Few are summarized here:

- Beginning in 1999, the U.S. DoD, in coordination with NATO, has developed 15 Standardized Agreements (STANAGs) that apply to UAS control, data and communications [30].
- In July 2004, the UK UAS Safety Subcommittee (a MoD/industry group) began work on the development of design and airworthiness standards for UAS systems. Within the next five years, the group seeks to work on a “classification scheme that will segregate UASs according to appropriate airworthiness requirements and the harmonization of requirements with international UAS policy and standards” [31].

4.4.3 Pilot Certification

Developing certification criteria for UAS pilot operations is complicated by the diversity in size, autonomy level, and potential uses of UASs.

There are two primary requirements for a UAS pilot

- 1) He/she must be fully capable of controlling the vehicle and must be able to interact safely with air traffic in the operating environment.
- 2) Drawing the line between who should or should not require a license, or what licensing levels might apply, will continue to be a challenge for UAS standards developers and regulators [20,32].

Chapter 5

UAS Tracking and Monitoring Android Application

5.1 About Application

The main purpose of the application is to fulfill the role of UAS Observer and UAS client. Our Android application allows the users to track the UASs based on time and name. It also allows the users to submit the observed UAS information to a tracking database server by taking a picture of the UAS. The monitoring system is implemented in Android for retrieving and displaying the UAS activity data from the database server. Application locates the UASs on the Google map and provides the related UAS details to the users.

This application serves two roles

- 1) UAS Observer
- 2) UAS Client

The role UAS Observer is responsible for taking pictures of any UAS, user comes across and upload it to the database by clicking on the "Add UAS" button provided in the start page of our application.

The role UAS Client is responsible for watching UAS movements based on the query provided by the UAS Observers. If incase they feel suspicious about any UAS movements they can take a note of them.

5.2 Technologies Used

5.2.1 Android

5.2.1.1 Android Application

An Android application is a software application that runs on the Android platform. Because this platform is built for mobile devices, an Android app is designed for a smart phone or a tablet PC running on the Android OS. Android apps are written in java and they use core java libraries.

Android Operating System has many Advantages over other operating systems and major of them are listed below:

- Open Source framework
- Use of tools is simple
- Availability of apps(Most of them are free in Google Play Store)
- Integrated applications and features
- Inbuilt support for flash

5.2.1.2 Android Application structure

Android Application structure plays a major role in its easy development and resource isolation and management. Any Android project contains things such as applications source code and resource files. Some are generated for us by default, while others should be created if required. Lets us have a look at android eclipse project directory structure shown in the Fig 5.1 below [33].

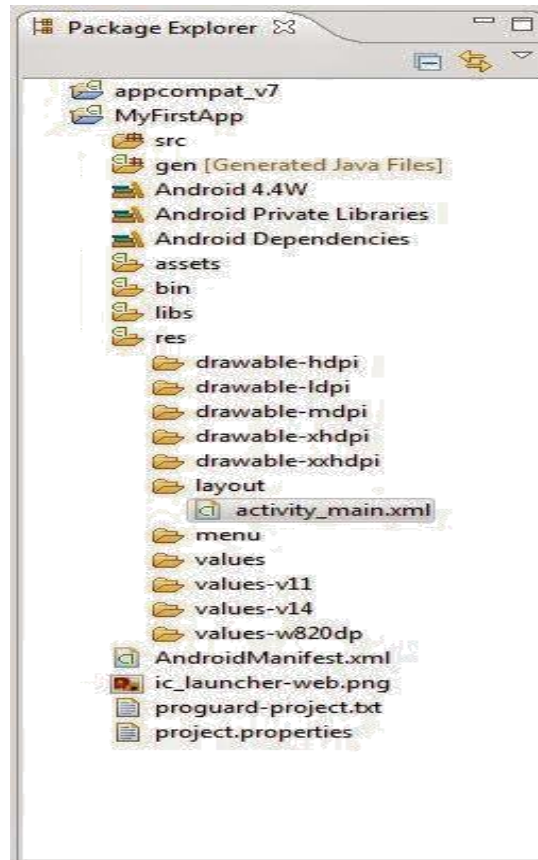


Fig 5.1 Folders in an android Application

Src: User specified java source code files are available here.

Gen: The gen directory in an Android project contains auto generated files. We can see R.java inside this folder which is a generated class, it contains references to certain resources of the project. This is automatically created by the Eclipse IDE and any manual changes to this file are not necessary.

Assets: This folder contains raw hierarchy of files and directories, with no other capabilities. It's just an unstructured hierarchy of files, which allows us to put anything we want there and later retrieve as raw byte streams.

Bin: This folder is used by the compiler to prepare the files to be finally packaged to the application's APK file. The following are performed by the compiler,

- Compiling our Java code into class files
- Putting your resources (including images) into a structure to be zipped into the APK

This folder is basically the output directory of the build. This is where we can find the final .apk file and other compiled resources.

Libs: External library files are placed in this folder. If we want to use any external library in our project we should place the library jar inside this folder and it will be added to the classpath automatically.

Res: Android supports resources like images and certain XML configuration files, these things can be kept separate from the source code. All these resources have to be placed inside the res folder. This res folder will contain sub-folders to keep the resources based on its type.

/res/values: Used to define strings, dimensions, styles etc. By convention each type is stored in a separate file. For example, strings are defined in the res/values/strings.xml file.

/res/layout: This folder contains the layouts to be used in the application. These are resource directories in an application that provides different layout designs for different screen sizes

AndroidManifest.xml: All the android applications will have an AndroidManifest.xml file in their root directory. This file contains essential information i.e., the information the system must have before it can run any of the application's code. This file describes the nature of the application and each of its components.

ic_launcher-web.png: This is an icon to be used in Google play Store to represent the application and this icon is used to represent a particular app. We have used the following icon.



Fig 5.2 Application Launcher Icon

proguard-project.txt: The ProGuard tool shrinks and optimizes your code by removing unused code. The result of this is a smaller sized .apk file that is more difficult to reverse engineer.

project.properties: This is the main project's properties file which contains information such as the build platform target and the library dependencies. This file is integral to the project.

5.2.2 Eclipse and Android Development Tools

Eclipse is an integrated development environment (IDE). It contains a base workspace and an extensible plug-in system for customizing the environment. It is used

to develop applications. Android Development Tools (ADT) is a plugin for the Eclipse IDE that is designed to give us a powerful, integrated environment in which we can build Android applications. This ADT extends the capabilities of Eclipse to let us quickly set up new android projects, create an application User Interface, add components based on the Android Framework API, debug our applications using the Android SDK tools, and even export

.apk files in order to distribute your application.

This is the best and fastest way to get started, so developing a project in Eclipse with ADT is highly recommended. With the guided project setup eclipse provides, as well as tools integration, custom XML editors, and output pane for debugging, ADT gives us an incredible boost in developing Android applications.

The Android Virtual Device Manager is an easy to use user interface to manage our AVD (Android Virtual Device) configurations.

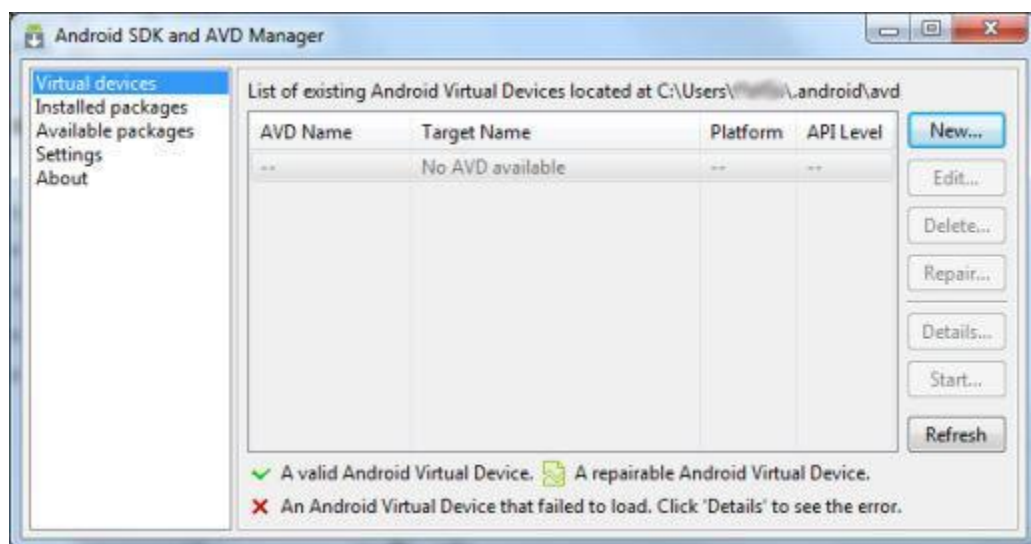


Fig 5.3 AVD Selection

AVD is a device configuration for the Android emulator that allows us to model different configurations of Android devices. When we start the AVD Manager in Eclipse or run the android tool on the command line, we will see the AVD Manager as shown in Fig 5.3.

5.2.3 mariadb

It is an enhanced replacement of MySQL. It was developed to maintain high compatibility with MySQL, ensuring a "drop-in" replacement capability with library binary equivalency and exact matching with the APIs and commands of MySQL [34].

5.2.4 Tomcat Server

Usually a web server is a place to store, process and deliver web pages to clients. The communication between client and server takes place using the Hypertext Transfer Protocol (HTTP).

In our application we are hosting web services on the web server, since it is a place where web services are hosted and administered. To host our web service we are using tomcat server 7. This host allows us to deploy our web application on the URI, the administrator chooses and serves its clients. In our application Tomcat Server runs on port no 8087.

5.2.5 RESTFUL Web Services

A technique by which two applications, regardless of platform or their programming language can communicate with each other is called a web service. A web service requires some data or argument to be passed to it, the service performs some kind of

processing on that data and finally the web service returns the data, in a particular format that is defined in the web service's programming.

Here in our application we are using RESTFUL Web Services.

REpresentational State Transfer (REST) is a stateless client-server architecture in which the web services are viewed as resources and are identified by their URIs. Web service clients who want to access these resources, access through globally defined set of remote methods which describe the action to be performed on the resource.

Chapter 6

Software Architecture

We describe the software architecture and its operational procedure in UML diagrams.

6.1 Use-case diagram

A use case diagram is a graphical representation of the interactions among the elements of a system.

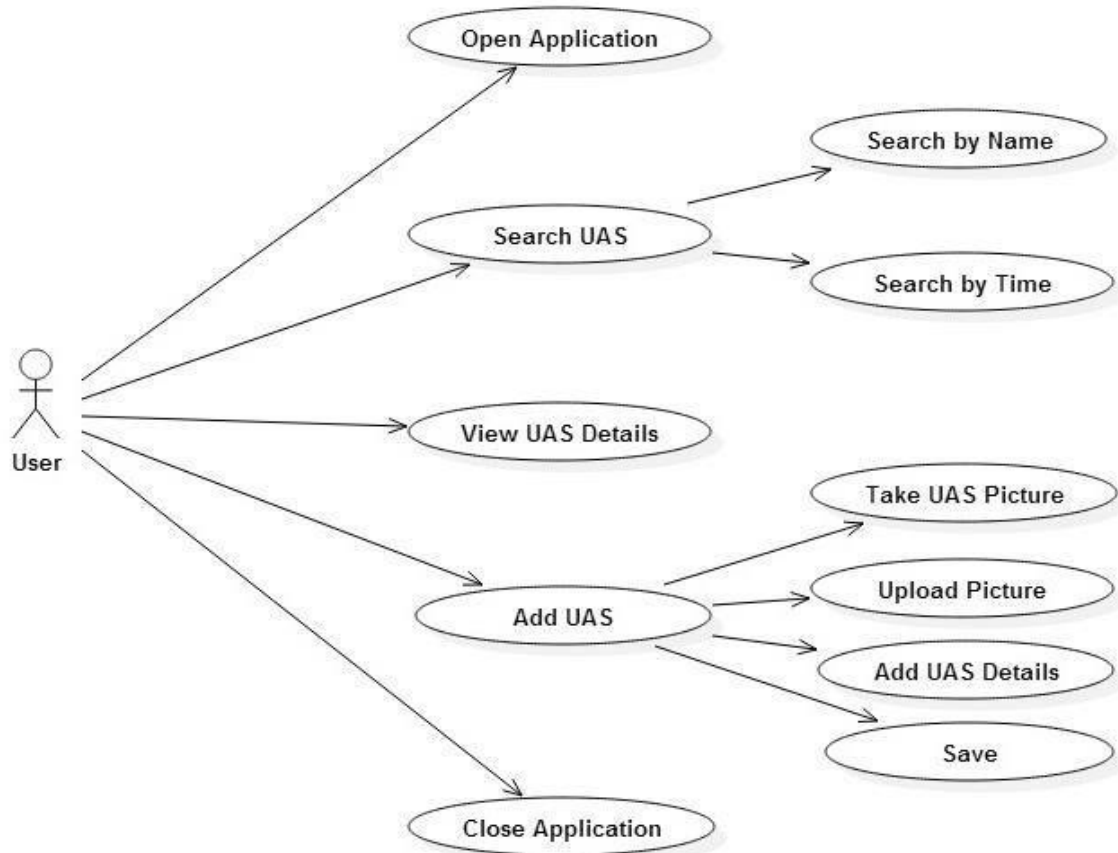


Fig 6.1 Use Case Diagram of UAS User

The use case diagram in Fig 6.1 shows how the user will interact with the system for searching UASs and adding UAS.

6.2 Control Flow Diagram

A control flow graph (CFG) consists of all the paths that might be traversed through a program during its execution. Fig 6.2 shows the control flow when user start adding an UAS to the database. In this process user will click add UAS icon on the screen, take a picture of the UAS, enter UAS details and clicks the save button.

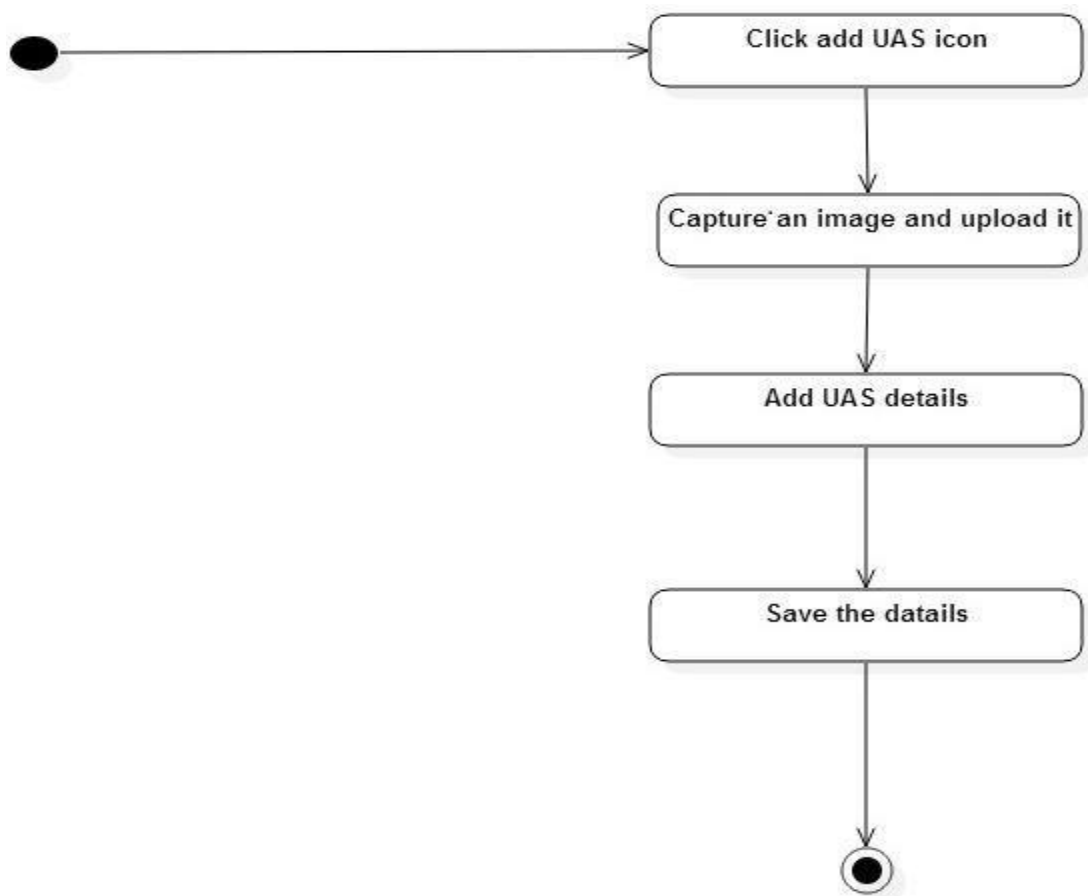


Fig 6.2 Control flow diagram to add UAS

These details have to be added manually to the UAS details table once they are verified. Once they are added to the table a UAS client can perform the following as described in the Fig 6.3. The figure shows the control flow where user is trying to search one or more UASs. User will open the app, select the UAS(s) from the drop down list or enter the UAS(s) names and clicks search button. Then user will get output on Google maps wherein clicking on the icons will get details of UAS .

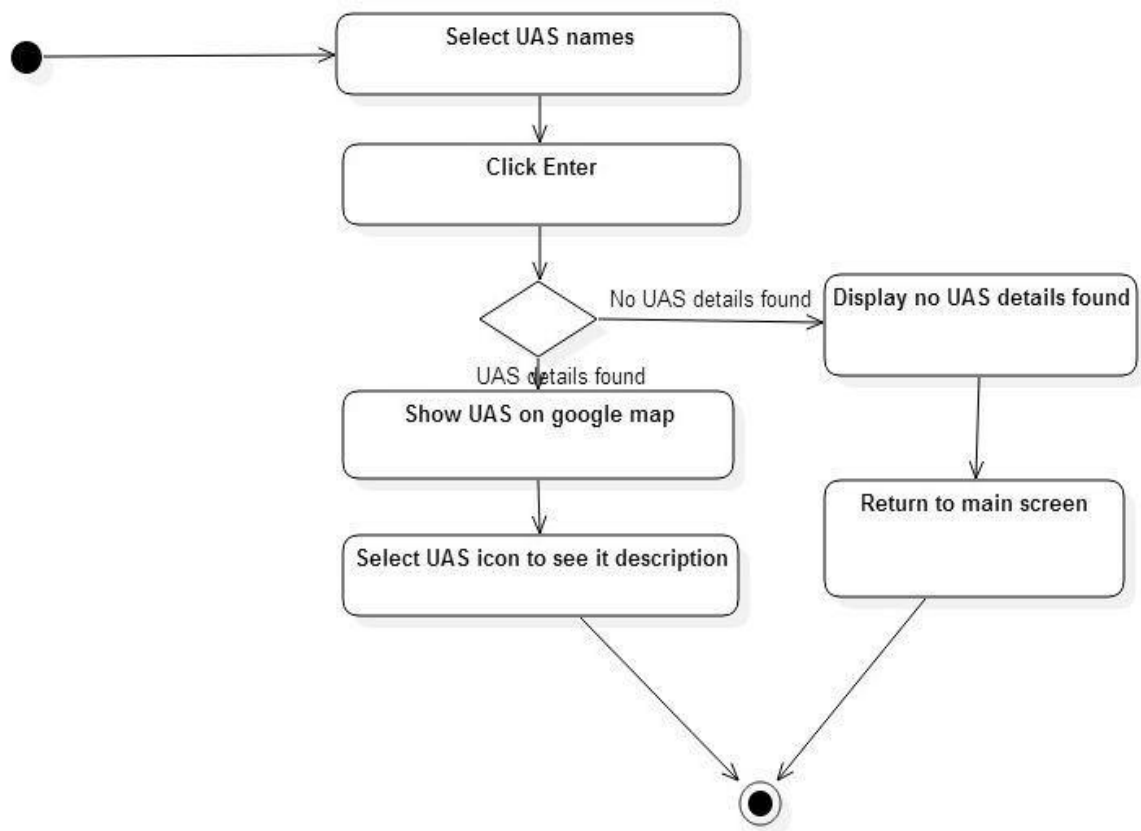


Fig 6.3 Control flow diagram to select UAS

6.3 Sequence Diagram

A Sequence diagram is an interaction diagram that shows how processes operate with one another and what is their order.

Fig 6.4 shows the sequence flow of saving UAS details into the database. First user will click add UAS button on the interface and interface will pop up displaying the form, where user will add the image and details of the UAS. Then interface will call the save method of web service and web service will add the details into the database.

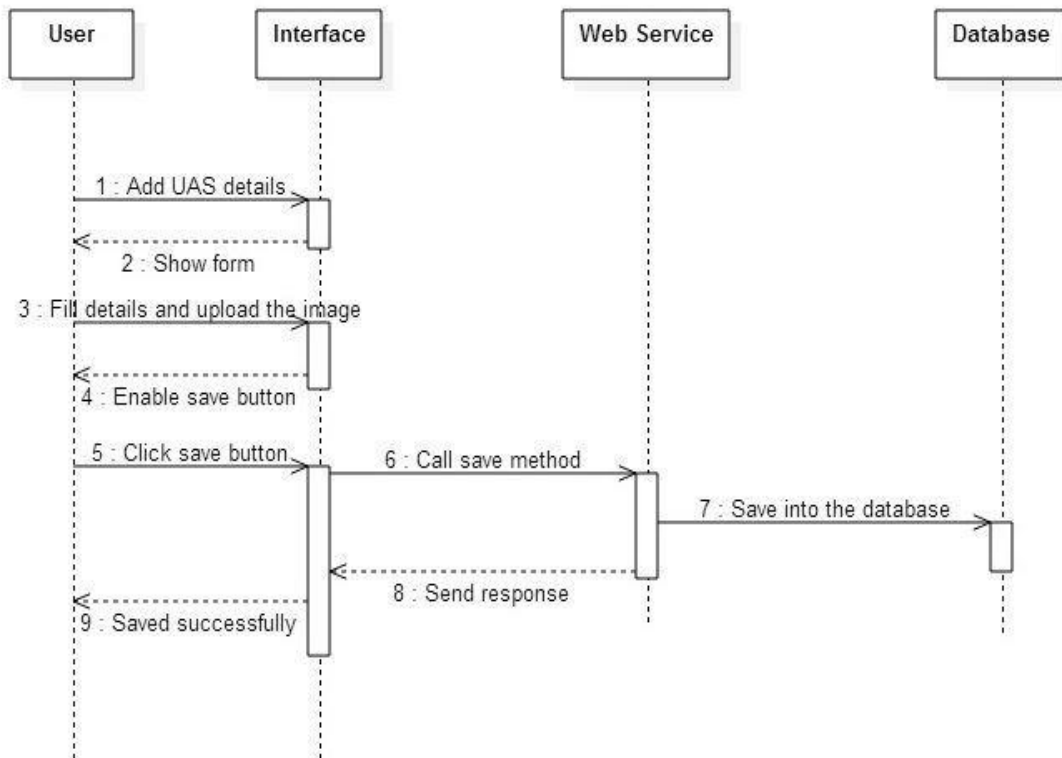


Fig 6.4 Sequence diagram to add UAS

Fig 6.5 shows the sequence flow where user will select UAS on the interface, interface will call, get selected UAS method of web service which in turn queries the data base and returns the response of database back to the user interface. User interface will point the details on the google map.

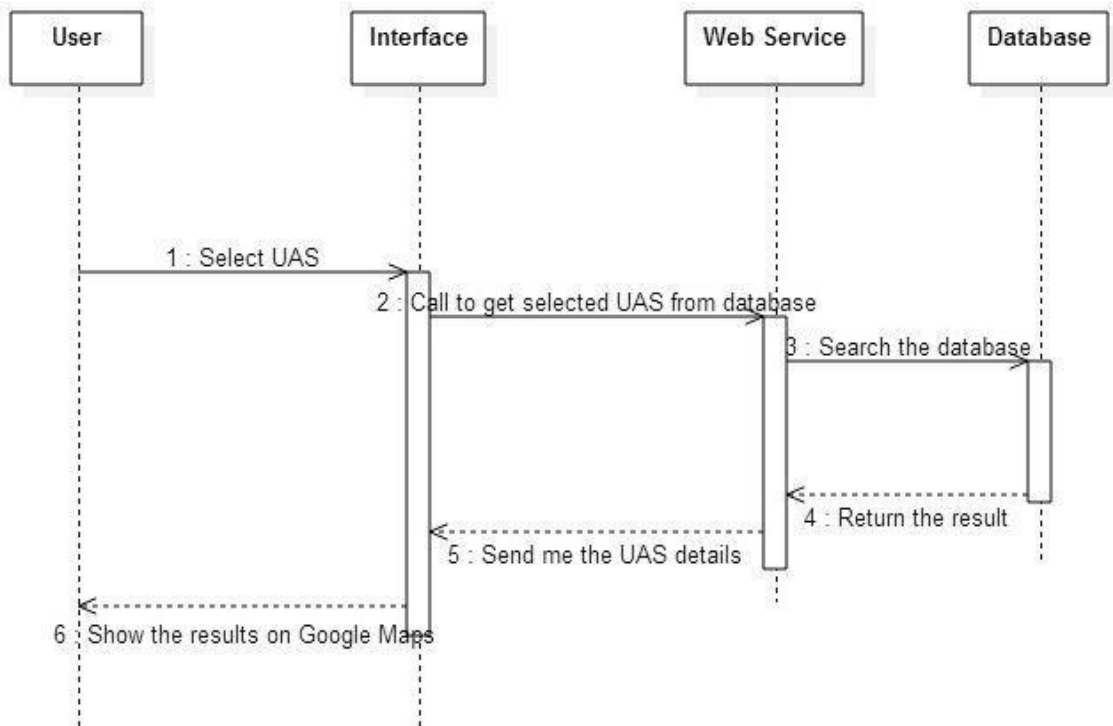


Fig 6.5 Sequence diagram to select UAS

Chapter 7

User Interface

User interface of this application is very attractive and self-explanatory. User interface will have Google map with points marked on it. Application will show points on Google map. User will track the Flight details based on **Time** and **Name** of the flight.

7.1 User Interface Controls

Dropdown Checkbox list: This control will have list of flight names. User can select one or more flight names by checking the Checkbox beside the name of the flight.

Search Button: This search button beside the dropdown list searches the flights from the database with the selected flight names. When user clicks this button Map must get refreshed with new data. If user has not selected any flight name user must get a pop up message asking him to select flight name/names before proceeding to search.

Date and Time Selection Control: This control allows the user to chose a particular date and time based on which UAS(s) shown on the map.

Search Button: This search button beside the date and time control allows the user to search for what all UASs are present at any place at that point of time. On selection of time at which no flights are present, user will get a pop up saying no flights are present at that time.

Time Level Control: This control will have a + and - sign. This control allows the user

to see a particular flight at different points of time.

Flight Icons: Depending on the type of the UAS, that particular icon is pointed on the map.

7.2 Screenshots of the application Functionality



Fig 7.1 Opening Screen

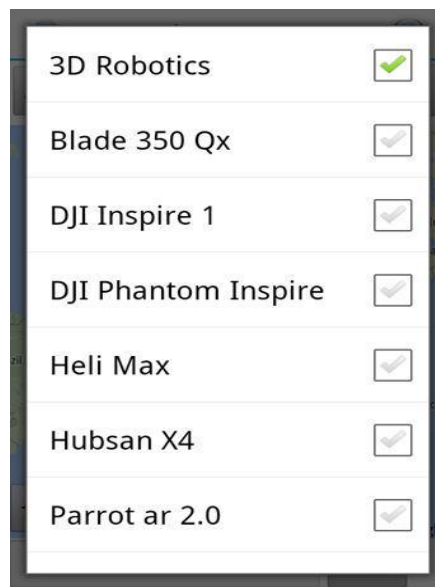


Fig 7.2 UAS Dropdown Menu

This screen in Fig 7.2 is shown while user selects the drop down list button. It allows the user to select as many UASs as he wishes to track.

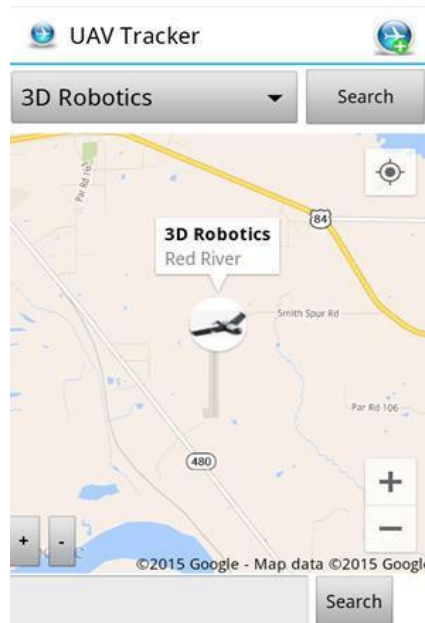


Fig 7.3 UAS on Google Maps

Upon selection of the UASs, user gets to see all the selected UASs on the map as shown in Fig 7.3.

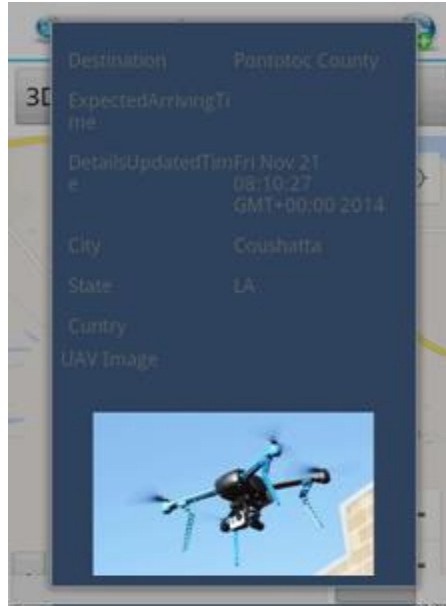


Fig 7.4 UAS Description

If the user wants to know the details of a particular UAS as in Fig 7.4, user has to touch on that particular UAS so that the details like description, current location, Destination, Expected Arrival Time along with the UAS image will be displayed on the screen.

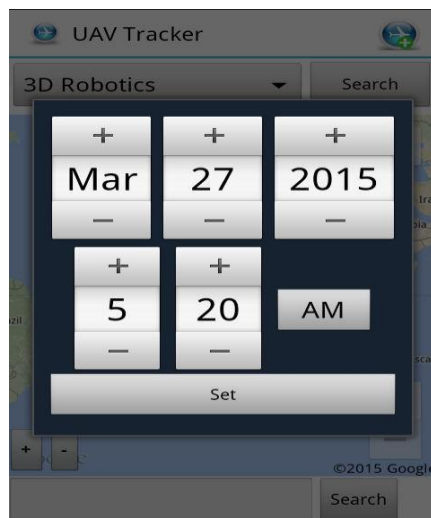
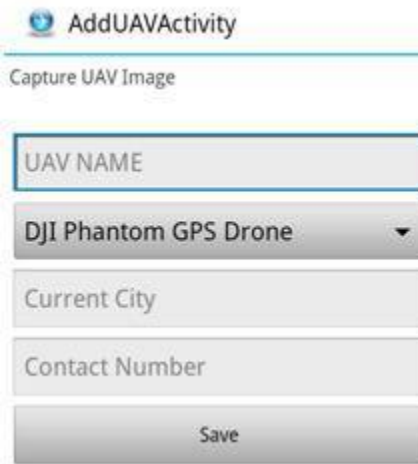


Fig 7.5: Time Menu

If the user wants to see the UASs at a particular point of time as in Fig 7.5, he could do so by selecting the date and time. There are '+' and '-' signs provided near the time control. They allow the user to switch between current position of the UAS and the previous position of the UAS.



The screenshot shows a mobile application interface for adding UAV activity. At the top, there is a title 'AddUAVActivity' with a globe icon. Below the title is a subtitle 'Capture UAV Image'. The form consists of several input fields: a text field for 'UAV NAME', a dropdown menu currently showing 'DJI Phantom GPS Drone', a text field for 'Current City', and a text field for 'Contact Number'. At the bottom of the form is a 'Save' button.

Fig 7.6 Activity to add UAS

User uses the screen shown in Fig 7.6 to add UAS when he comes across the UAS.

Chapter 8

Database

We used database mariadb for our application. It runs on port number 8085. We are storing all the details related to the UASs including their ids, names, images, a detailed description, time of start, expected time of reaching particular point and latitude and longitude values at particular time.

We are using two tables in the database, one table will store UAS details which our UAS client will be able to search from the app. Second table will save the unprocessed details that users of the application submits. Second table will maintain user details such as phone number, time at which image is added, place from where user added the image. Once the UAS details are processed and validated, they will be added to the UAS details table.

Column Name	Type	Description
Id	Int	It is the unique id for the entries of this table
UAS_name	Varchar	It is the name given by user to his UAS
UAS_type	Varchar	It is the type of UAS
Lat_long	Varchar	The latitude-longitude values are extracted using Global Positioning System (GPS) of the user's smartphone which can then be used to show the UASs current location
Mobile_number	Varchar	It is the user's mobile number
Image	MediumBLOB	It is the user's mobile number
Time	Timestamp	Time at which user captured the image
City	Varchar	Place from which user captured the image

Table 8.1 UAS Register Database

8.1 Connection between Android application and Database

Below two lines of code allow our android application to establish a connection with the database through Web service.

```
Class.forName("com.mysql.jdbc.Driver");
con = DriverManager.getConnection(
    "jdbc:mysql://localhost:8085/flight", USER_NAME, PASSWORD);
```

This connection is achieved when the application is hosted on Tomcat Server and when Web service runs on it. As long as this Web service runs on the server, our application will be able to receive data from the database.

Chapter 9

Web Services

A web service can be treated as a communication between two computers. Instead of text and graphics, a web service transmits code requests that interact directly with another computer, without any human intervention. Computers can use these web services to communicate directly with one another at a low level like for sending and receiving requests for raw data or sharing functions and methods. In this sense, a web service is treated as the back end part of the world wide web.

For example, let us imagine that we want to be able to access stock price data. Let us suppose it to be stored on an external server in raw form, so there might be some simple algorithm on that server that will download the data, convert it into something useful and then send it to us. If the algorithm is public and can be run by external computers, then it is a web service. We might wonder why web services are necessary. Couldn't we just download a function for stock data conversion from the site once and then never have to access the data again? Well, there are a number of disadvantages with such a strategy.

If the functions are ever updated, then we will have to download them again to update our local copy. If the database we are accessing changes, then we have to make a note of that change and rewrite our code to interface with the new database. In such a situation a web service helps us to leave key functions on the source server instead of pulling those down to the local browsers. This makes it easier to update and modify those changes from a centralized location. But this process is not as simple as copying and pasting a function to another computer. Within a single program, communication between

functions is simple and straightforward. The communication between two machines via web service is shown in Fig 9.1 below.

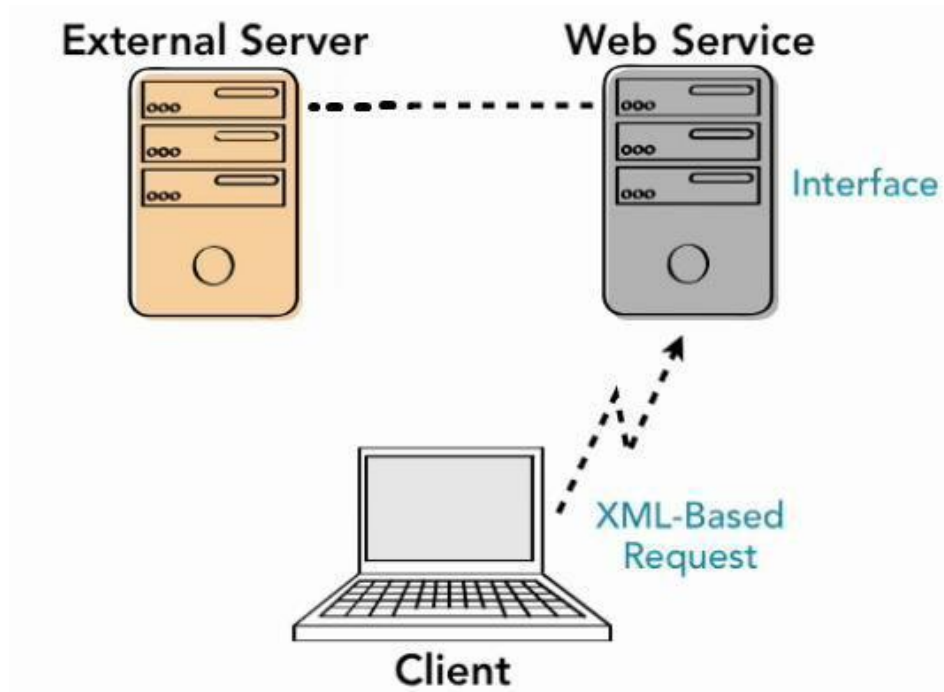


Fig 9.1 Communication Via Web Service

Java programs support certain data types universally. So any method in a java program can communicate with any other method in java program by passing in data structures like integer and arrays. Similarly, any method written in C, C++, or any other programming language can communicate with other methods in its language using common data structures. However, since the Internet consists of billions of different computers interacting with each other through a variety of different protocols and functions. And there is no guarantee that any two programs that need to interact are written in the same language.

Web services bridge these gaps by providing compatibility across multiple languages and operating systems. Instead of requiring the user to know the details and code structure of every program they access, a web service will provide a set of standard information in a format that is universally readable. Programmers call this type of programming as the black box method of programming [35]. For maximum flexibility and compatibility, users should be able to effectively interact with the program with as little information as possible. There exist many web services for achieving above discussed features. We are using most popular web services in our application which is known as RESTFUL Web Service.

9.1 RESTFUL

RESTFUL Web Service consists of two components and they are as follows,

- REST server - provides access to the resources.
- REST client - access and modify the REST resources.

In the REST architecture style, clients and servers exchange representations of resources by using a standardized interface and protocol. REST is not protocol specific, but when people talk about REST they usually mean REST over HTTP.

The response from the server is considered as the representation of those resources taken from the server.

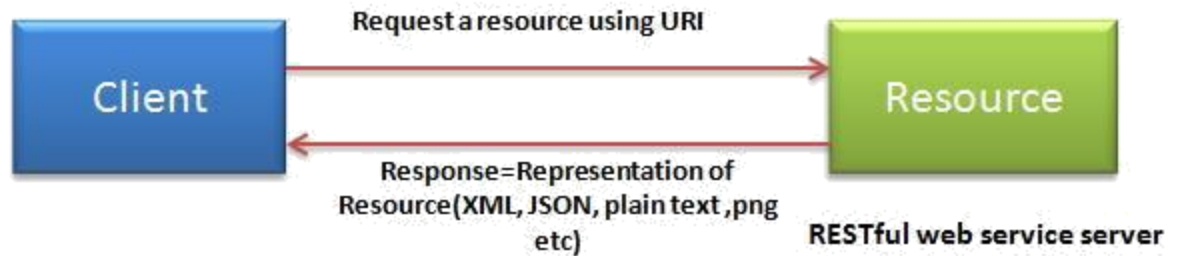


Fig 9.2 Communication using REST

Above Fig 9.2 shows the communication between client and server using REST protocol wherein

- A client requests for a resource using URI, over HTTP
- Server finds the resource using URI and processes the client request
- The response is sent to the client in the requested format

REST allows resources which have different representations, like xml, json etc. The rest client can get response in specific representation via the HTTP protocol [36].

9.1.1 HTTP Methods

GET: The GET method is used to retrieve information from the server using a given URI. Requests using GET method should only retrieve data and should have no other effect on the data.

PUT: This method replaces all current representations of the target resource with the uploaded content.

DELETE: This method removes all current representations of the target resource given by a URI.

POST: A POST request is used to send data to the server [37].

9.2 Advantages of REST over SOAP

A SOAP [38] client works like a custom desktop application, tightly coupled to the server. There is a rigid contract between client and the server, and everything is expected to break if either side changes anything. We need constant updates following any change.

A REST client is more like a browser. It is a generic client that knows how to use a protocol and standardized methods, and an application should fit inside that. We should not violate the protocol standards by creating extra methods, we leverage on the standard methods and create the actions with them on our media type. If done right, there's less coupling, and changes can be dealt more easily.

A client is supposed to enter a REST service with almost no knowledge of the API, except at the entry point and the media type. In SOAP, the client needs knowledge on everything they will be using, otherwise it won't even begin the interaction.

REST is protocol independent. It is not coupled to HTTP. We can even follow an ftp on any website, a REST application can use any protocol for which there is a standardized URI scheme.

REST is not mapping CRUD(Create, Read, Update and Delete operations) to HTTP methods. Security and authentication is standardized [39].

Illustration of invoking and working of web services in our application:

This is the method used in the main activity class to call the web service

```
new GetFlightNamesWS(this).execute(Constants.GET_ALL_FLIGHT_NAMES);
```

There is a file called constants in our application which has the URI, so this method GetFlightNamesWS is invoked at this path.

```
public static final String GET_ALL_FLIGHTS = GET_IP_ADDR +  
"/getAllFlight"; public static final String GET_IP_ADDR=IP_ADDR+  
":8087/Flightwebapp/ws/flightws;  
public static final String IP_ADDR="http://172.20.10.3";
```

So the URI for our application will be

```
http://172.20.10.3:8087/Flightwebapp/ws/flightws/getAllFlight
```

Here 172.20.10.3 is the ip address on which we are connected.

8087 is the port number on which our server is running.

Rest of the part is a file hierarchy in our application.

The following function is supposed to create a new dialog and ask the user to wait until some action is performed.

```

public GetFlightNamesWS(MainActivity activity)
{
    this.activity = activity;

    dialog = new ProgressDialog(this.activity);
    dialog.setTitle("Please wait...");
}

```

When the user is waiting, the following action is performed in the background.

```

@Override
protected Object doInBackground(Object... params)
{
    HttpClient httpClient = new DefaultHttpClient();
    String responseString = null;
    HttpResponse response;
    String error_message;
    StatusLine statusLine;

    try
    {
        HttpPost post = new HttpPost(params[0].toString());
        post.setHeader("Accept", "application/json");
        post.setHeader("Content-type", "application/json");
        response = httpClient.execute(post);
        statusLine = response.getStatusLine();
    }
}

```

```
    if (statusLine.getStatusCode() == HttpStatus.SC_OK)
    {
        Log.d("statuline", "ok");

        ByteArrayOutputStream out = new
        ByteArrayOutputStream(); response.getEntity().writeTo(out);
        out.close();

        responseString = out.toString();
    }
    else
    {
        Log.d("statuline", "not ok");

        ByteArrayOutputStream out1 = new ByteArrayOutputStream();

        try
        {
            response.getEntity().writeTo(out1);
            out1.close();
            error_message = out1.toString();
        }
        catch (IOException e)
        {
            e.printStackTrace();
        }
    }
}
```

```
}  
  
catch (UnsupportedEncodingException e)  
  
  {  
  
    e.printStackTrace();  
  
  }  
  
catch (ClientProtocolException e)  
  
  {  
  
    e.printStackTrace();  
  
  }  
  
catch (IOException e)  
  
  {  
  
    e.printStackTrace();  
  
  }  
  
  
return responseString;  
  
}
```

Before the web service execution takes place the following method is invoked.

```
@Override
protected void onPreExecute()
{
    dialog.show();
    super.onPreExecute();
}
```

After the execution of web service the following method is invoked.

```
@Override
protected void onPostExecute(Object result)
{
    System.out.println(result);
    if (result != null)
    {
        ArrayList<String> names = (ArrayList<String>) new
Gson().fromJson(result.toString(), ArrayList.class);
        activity.initializeSearch(names);
    }
    if (dialog != null && dialog.isShowing())
    {
        try
```

```

        {
            dialog.cancel();
        }
        catch (Exception e)
        {
        }
    }
}

```

This is a method in that is in the web service, which is invoked when the web service is called.

```

@Path("/getFlightsByName")
@POST
@Consumes(MediaType.APPLICATION_JSON)
@Produces("application/json")
public Response getFlightsByNames(String names)
{
    List<Flight> flights = null;
    if (names != null)
        flights = new FlightDao().getFlightsByName(names.split(","));
    return Response.status(200).entity(new Gson().toJson(flights)).build();
}

```


Chapter 10

Conclusion and Future Work

In the previous sections the challenges faced by UAS and the need for UAS integration into National Airspace System have been discussed. Keeping that in mind an Android application is developed which allows the users to submit the observed UAS information to a tracking database server by taking a picture of the UAS, along with the monitoring system which is implemented in Android for retrieving and displaying the UAS activity data from the database server. This system is expected to promote safer airspace by providing the UAS activity information to manned aircraft pilots and other UAS operators to avoid collision.

The future work may extend the functionality of the application by allowing the server to relay all the UAS information to the Air Traffic Management(ATM) as well as other UAS and enable two way communication between UAS operator and ATM Controller.

The application could be published in Google Play Store and tested with live audiences.

Bibliography

[1] "Unmanned aerial vehicle", wikipedia.

Available: http://en.wikipedia.org/wiki/Unmanned_aerial_vehicle.

[2] "Civilian UAV", AeroVironment Glossary.

Available: http://www.avinc.com/glossary/civilian_uav.

[3] Suraj G. Gupta, Mangesh M. Ghonge, Dr. P. M. Jawandhiya, *Review of Unmanned Aircraft System (UAS)*, International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) Volume 2, Issue 4, April 2013.

[4] "Eyes of the Army, US Army UAS Roadmap 2010-2035", Barclay J., US Army Aviation Center of Excellence, Fort Rucker, AL, 2010. Available: <http://fas.org/irp/program/collect/uas-army.pdf>

[5] "Study Analysing The Current Activities in the Field of UAV", European Commission Enterprise and Industry Directorate-General, ENTR/2007/065; 2007. Available:

http://ec.europa.eu/enterprise/policies/security/files/uav_study_element_2_en.pdf

[6] "Advantages of UAV", Unmanned Aerial Vehicle Systems Association. Available: <https://www.uavs.org/advantages>.

[7] J. M. Sullivan, *Revolution or evolution? the rise of the UAVs*, Technology and Society Magazine, IEEE , Volume 25, Issue 3, 2006.

- [8] Parrot Bebop Specifications, Available: <http://www.parrot.com/usa/products/bebop-drone/>
- [9] 3 D Robotics Iris+ Specifications, Available: <http://3drobotics.com/iris/>
- [10] DJI Phantom Inspire One Specifications,
Available: <http://www.theverge.com/2014/11/12/7210621/dji-inspire-one>
- [11] Turbo Ace Matrix Specifications, Available: <http://www.turboace.com/>
- [12] Hubsan x4 Specifications, Available: http://www.hubsan.com/productinfo_16.html
- [13] Blade 350 Qx Specifications, Available: <http://www.bladehelis.com/350qx/>
- [14] "National Airspace System", Airport and Air Traffic Control System, January 1982, pg.25-42.
Available: <https://www.princeton.edu/~ota/disk3/1982/8202/820205.PDF>.
- [15] "How Air Traffic Control Works", Craig Freudenrich, Ph.D. Available:
<http://science.howstuffworks.com/transport/flight/modern/air-traffic-control.htm>
- [16] Beason, R.C., & Seamans, T.W., *Runway incursions: a call for action*, Air Line Pilots Association, International, 2007.
- [17] Geyer, C., Singh, S., & Chamberlain, L. , *Avoiding collisions between aircraft: state of the art and requirements for UAVs operating in civilian airspace*. Carnegie Mellon University. Retrieved form, January 2008. Available:
<http://www.frc.ri.cmu.edu/projects/senseavoid/Images/CMU-RITR-08-03.pdf>

[18] Sampigethaya, K., Poovendran, R., & Bushnell, L. (2010), *Assessment and mitigation of cyber exploits in future aircraft surveillance*, Aerospace Conference, IEEE 2010.

[19] "New rules governing drone journalism are on the way", Matt Waite.

Available: <http://www.niemanlab.org/2015/02/new-rules-governing-drone-journalism-are-on-the-way-and-theres-reason-to-be-optimistic/>

[20] Matthew T. DeGarmo, *Issues Concerning Integration of Unmanned Aerial Vehicles in Civil Airspace*, The MITRE Corporation, November 2004.

[21] Jill Kamienski, Elliott Simons, Steven Bell, and Steven Estes, *Study of Unmanned Aircraft Systems Procedures : Impact on Air Traffic Control*, Digital Avionics Systems Conference (DASC), IEEE 2010.

[22] Hyeon-Cheol Lee, *Implementation of Collision Avoidance System using TCAS II to UAVs*, Digital Avionics Systems Conference, 2005.

[23] Adrian MURARU, *Some Aspects Regarding "Sense and Avoid" Requirements for UAV Integration in the National Air Space*, INCAS BULLETIN, Volume 2, Number 4/ 2010.

[24] "OSD UAV Reliability Study", Office of the Secretary of Defense, February 2003.

Available: <http://www.uadrones.net/military/research/acrobat/0302.pdf>

- [25] Robert Smith, *Empowering UAVs with Responsibility for Own Safety*, AUVSI conference, 17 July 2003.
- [26] A Compendium of DARPA Programs, Defense Advanced Research Projects Agency, August 2003, pg. 30.
- [27] "*Unmanned Aerial Vehicles: Examining the Safety, Security, Privacy and Regulatory Issues of Integration into U.S. Airspace*", Evan Baldwin Carr.
Available: <http://www.ncpa.org/pdfs/sp-Drones-long-paper.pdf>
- [28] John Villasenor, "*Drones*" and the Future of Domestic Aviation, Proceedings of IEEE, Volume 102, Issue 3, pp. 235-238, 2014.
- [29] K. Dalamagkidis, K. P. Valavanis, and L. A. Piegl, *Current status and future perspectives for unmanned aircraft system operations in the us*, Journal of Intelligent and Robotic Systems, vol. 52, no. 2, pp. 313-329, 2008.
- [30] Article in "NATO Standardization Agreement 4586 – Leading the Way to NATO UAV Systems Interoperability", November 2003.
- [31] "Priorities Defined for UK UAV Standards", New update on UAVworld News, UAVworld.com, July 2004.
- [32] Lacher, Andy, Andrew Zeitlin, Chris Jella, Charlotte Laqui, Kelly Markin, *Analysis of Key Airspace Integration Challenges and Alternatives for Unmanned Aircraft Systems*, MITRE report, 2010.

[33] " Welcome to Android ", Gaurav Bhor .

Available: <https://pensieveoverflow.wordpress.com/>

[34] "MariaDB versus MySQL - Compatibility", MariaDB KnowledgeBase.

Available: <https://mariadb.com/kb/en/mariadb/mariadb-vs-mysql-compatibility/>

[35] Martín Garriga, Andres Flores, Alejandra Cechich, Alejandro Zunino, *Testing-based Process for Service-oriented Applications*, Computer Science Society (SCCC), 30th International Conference of the Chilean, 2011.

[36] Charl van der Westhuizen and Marijke Coetzee, *A Framework for Provisioning RESTful Services on Mobile Devices*, Adaptive Science and Technology (ICAST), 2013.

[37] " RESTful Web Services ", Venkata Subbaiah.

Available: <http://pvenkat99.blogspot.com/2014/03/restful-webservice.html>

[38] Lanthaler, M. ; Gutl C. , *Towards a RESTful service ecosystem*, Digital Ecosystems and Technologies (DEST), 4th IEEE International Conference, Page(s): 209 - 214, 2010.

[39] Mohammed Husain Bohara, Madhuresh Mishra, Sanjay Chaudhary, *RESTful Web Service Integration Using Android Platform*, Computing, Communications and Networking Technologies (ICCCNT), Fourth International Conference, 2013.

VITA

Graduate College

University of Nevada, Las Vegas

Tejaswini Papireddy

Degrees:

Bachelor of Engineering in Computer Science, 2013

Chaitanya Bharathi Institute of Technology.

Master of Science in Computer Science, 2015

University of Nevada Las Vegas.

Thesis Title:

Tracking and Monitoring Unmanned Aircraft Systems Activities with Crowd-Based Mobile Apps.

Thesis Examination Committee:

Chair Person, Dr. Ju-Yeon Jo, Ph.D.

Committee Member, Dr. Yoochwan Kim, Ph.D.

Committee Member, Dr. Ajoy K Datta, Ph.D.

Graduate College Representative, Dr. Venkatesan Muthukumar, Ph.D.