MIDDLE EAST TECHNICAL UNIVERSITY

SOFTWARE REQUIREMENT SPECIFICATION DOCUMENT

v.1.0

Group Name: S.T.A.L.K.E.R.S.

Group Members: Burak ÖZALP

Bülent ÖZGÜNER

Fazilet ÖZER

Hüsnü ŞENER

Advisors: Selim TEMİZER

Ayşe Gül YAMAN

General Purpose Intelligent Flying Camera with Quadcopters

Version	Date	Changed	A/D/M	Brief Description
1.0	30/11/14	-	-	Initial Version

Table of Contents

1. Introduction

In this section, the content of SRS document is explained briefly. This document contains all specifications about General Purpose Intelligent Flying Camera with Quadcopters.

1.1 Problem Definition

In our daily life, cameras has a role for sharing our precious moment with our families and friends. However, there are some limitations on classical cameras such as having restricted mobility.

That is, it is usually difficult to obtain the right angle for specific tasks. To solve this problem, we have a solution approach that is using quadcopter's flexible movement capability. Therefore, our goal is simply creating an easy-to use flying camera. This camera enables users to capture events that are difficult to capture by classical ones.

1.2 Purpose

The aim of SRS document is to specify the software requirements of the project "General Purpose Intelligent Flying Camera with Quadcopters". Basically, it gives detailed description of the project. Moreover, the goal and the qualification of the system, the constraints and interfaces of the system and what the system will do are clarified in this document. The explanation of the functions that the software will perform helps to see whether the software meets its requirements. This document specifies all the requirements pre-design, during design, programming and testing.

1.3 Scope

The project that is stated in this document, is aims to create an easy-to use flying camera basically.

This camera can manage to do many different tasks. For instance, users can control it both automatically and manually. To control it automatically, microcomputers will be used. Giving it a predefined path or a target to follow are the options that are considered to control the camera automatically. For target tracking, some image processing algorithms will be used to detect the target. Other option for target tracking is to obtain the position of the target from mobile phones.

On the other hand, manual control will enable users to have precise control on the camera.

1.4 Definitions, acronyms and abbreviations

Terms	Definitions
SRS	Software requirements specification
Quadcopter	It is a multirotor helicopter that is lifted and propelled by four rotors.
AR.Drone	A quadcopter that has a built in camera produced by French company Parrot.
ROS	Robot Operating System, a framework for robotics applications.
SDK	Software Development Kit
ardrone_autonomy	A driver for AR.Drone 2.0 based on AR.Drone SDK and works over ROS Framework.
Use Case Diagram	Use case diagram shows users' interaction with a system
Class Diagram	A special diagram that presents classes of a system, attributes of the classes, methods of the classes and the relationships between objects.
OpenCV	Open Source Computer Vision Library
IEEE	The Institute of Electrical and Electronics Engineers
ARM	A RISC based instruction set architecture for CPUs, mostly used on mobile devices
x86	A CISC based instruction set architecture developed by Intel, mostly used on PCs.
Raspberry Pi	A small sized ARM-based programmable microcomputer.

1.5 References

The resources listed below are references used in requirement analysis;

IEEE Standard Documents:

- IEEE STD 1233-1998, IEEE Guide for Developing System Requirements Specifications
- IEEE STD 830-1998, IEEE Recommended Practice for Software Requirements Specifications

1.6 Overview

There is detailed information about the project in this document. In the first section, there is general information about the projects such as the definition of the problem which this project's aim is to solve, purpose of this SRS document and scope of the project are explained briefly. Second section of the document, gives information about product perspective, product functions, constraints, assumptions and dependencies of the product. In next section, interface, functional and non-functional requirements of the project will be clarified. In fourth section of the document, data models and descriptions of them will be explained by showing their relationships. Next section will give information about the behavioral model and its description. In section 6, team structure, basic schedule and process model will be explained. Finally, in the conclusion part, a brief summary of this document will be given.

2. Overall Description

In this section, product perspective, product functions, constraints, assumptions and dependencies will be explained in detail.

2.1 Product Perspective

For now, the product is independent, in other words it isn't a component of any larger system. However, there can be extra features to be added in the future according the requests from the users. Scope of project can be changed according to new requests.

2.1.1 User Interfaces

Here are 6 sample user interface mock-ups to give a basic idea about how user interaction is planned. They are not the final designs, whole UI is open for new ideas.

2.1.1.1 Initial Interface

In this state, quadcopter is connected to PC and ready for take off. The status of quadcopter is "ready".

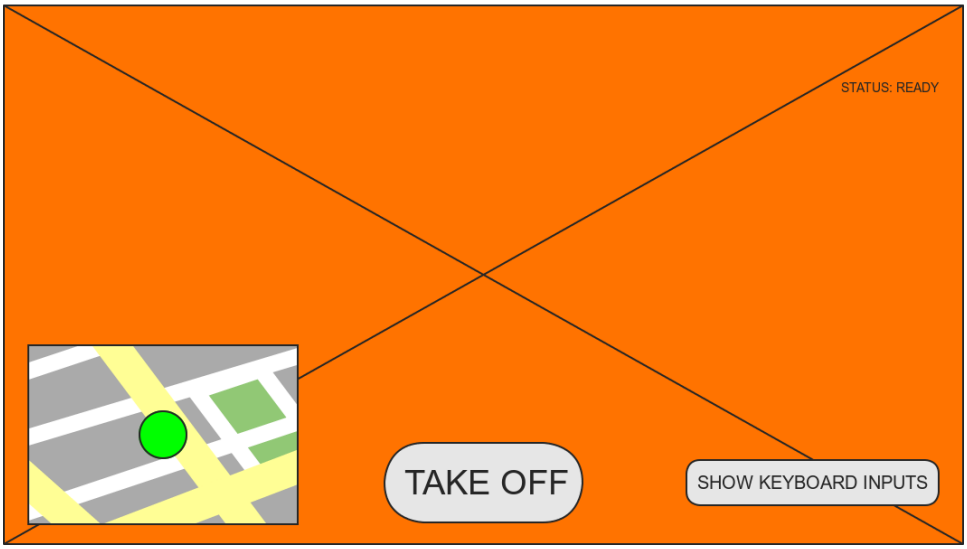

Figure 1: Initial Interface

2.1.1.2 Interface of Flying

In this state, take off operation is done. Recording and target tracking mode is off.The status of quadcopter is "flying". Thanks to the UI, user can also see the attitude of this quadcopter.

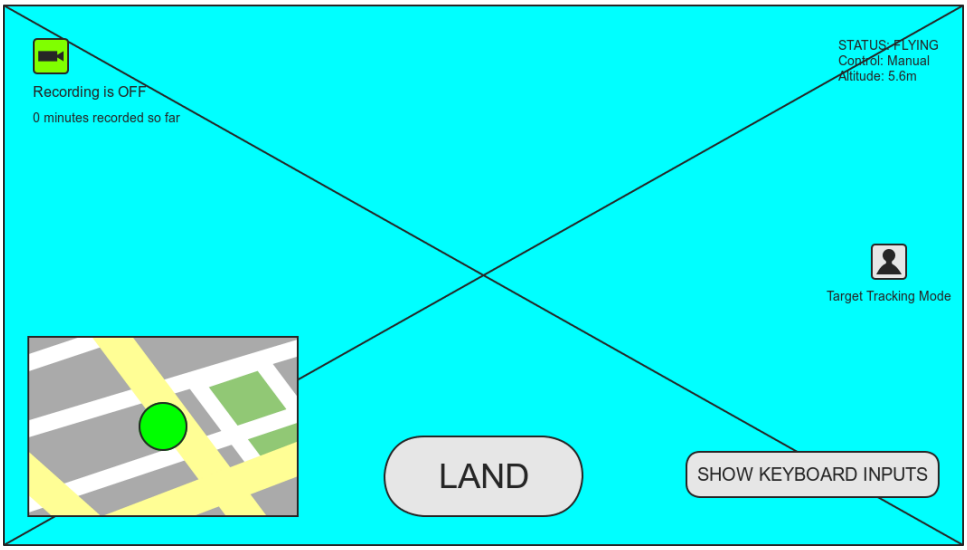

Figure 2: Flying Interface

In the next figure the recording is on, since the user click on the camera button. User also can observe the recording time.

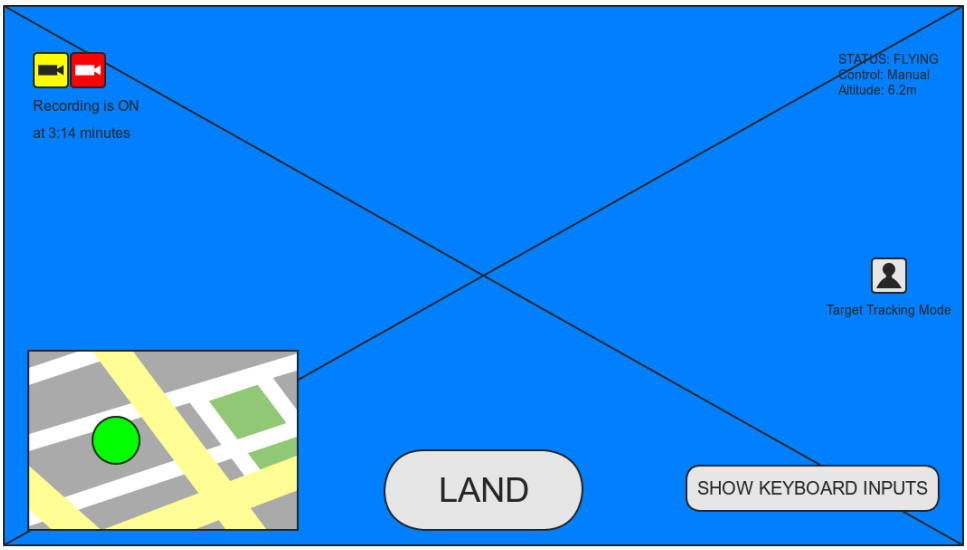

Figure 3: Recording Mode Interface

The user can pause the video recording by clicking on the yellow camera button. As we can see in the figure below, recording is paused.

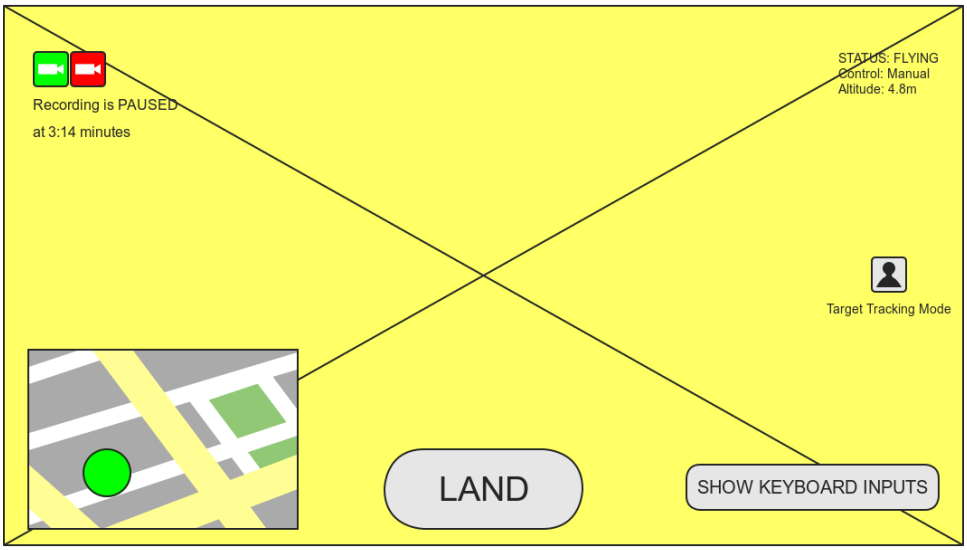

Figure 4: Pause Recording Mode Interface

Now the main purpose of this project becomes clear. User can select the target area for tracking.

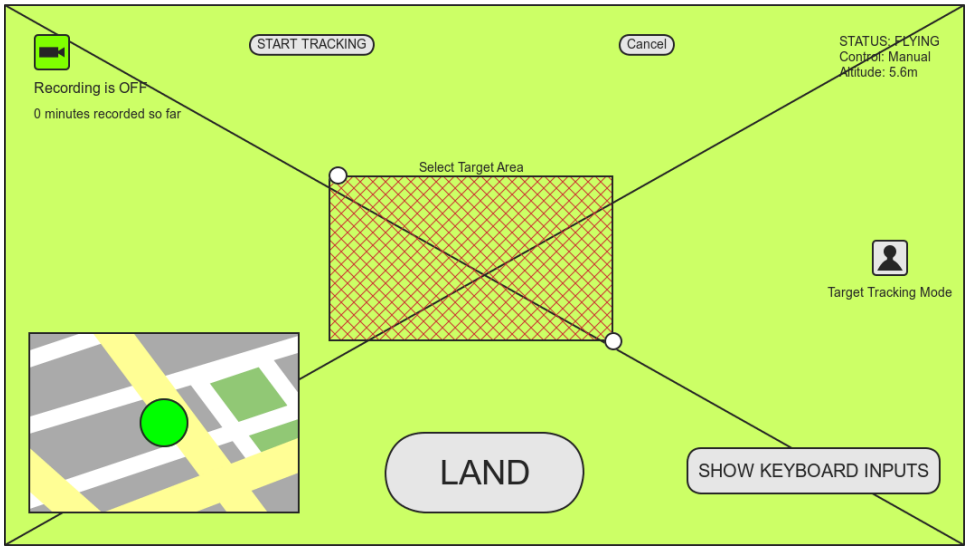

Figure 5: Selecting Target Mode Interface

If the area selected, target tracking mode is on. In this state, quadcopter should track the selected target area. And started to track it.

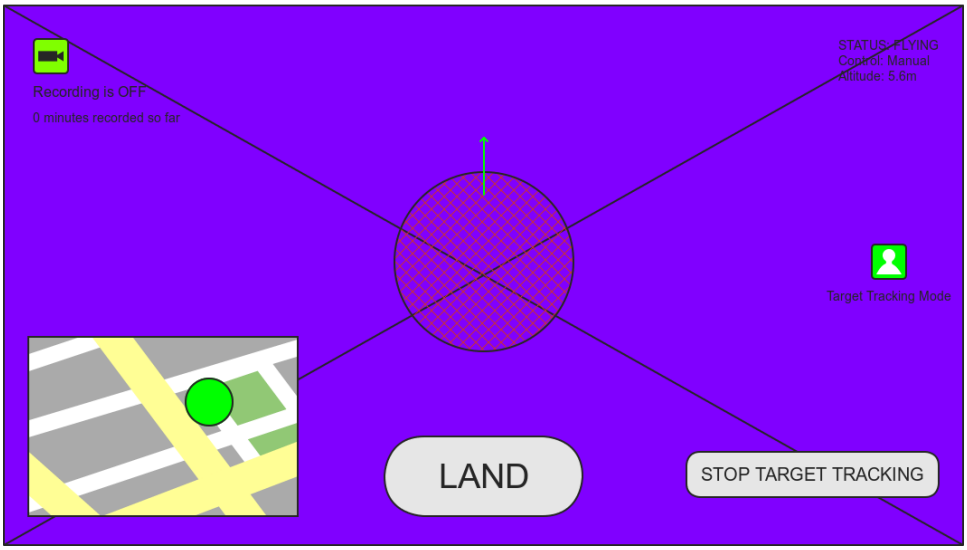

Figure 6: Target Tracking Mode Interface

2.1.2 Hardware Interfaces

As quadcopter with camera, AR.Drone 2.0 will be the hardware chosen for this project. No other quadcopter will be supported by default.

As controller hardware a decent x86 PC with Ubuntu 14.04 installed is the default configuration. But also for portability, a version of controller software that would work on Raspberry Pi Model B+ will be prepared as a complete system image. Operating System choice for Raspberry Pi image will be done latter, but a Debian-based distro for Raspberry Pi is the very first choice (due to similarity with Ubuntu).

2.1.3 Software Interfaces

Below, information about other necessary software products is provided.

Robot Operating System

- ROS
- Indigo
- www.ros.org

Ubuntu OS

- Ubuntu
- 14.04.1 LTS
- www.ubuntu.com

Open Source Computer Vision Library

- OpenCV
- 2.4.9
- www.opencv.org

Below, there will be information about interfaces with other application systems.

AR.Drone Autonomy

- This software connects AR.Drone 2.0 Controller and our software over ROS Platform.
- Standard ROS Topic is used for bi-directional communication interface.
- Source codes and detailed documentation could be found over https://github.com/AutonomyLab/ardrone_autonomy

2.1.4 Communication Interfaces

In this project, IEEE802.11n standard will be used for wireless communication with AR.Drone.

2.1.5 Memory

Although the system has no predefined memory restriction, available memory space will directly affect maximum recording time. Memory configuration is left to the user, but 8 GB USB Memory or equivalent free space on internal memory is recommended for a smooth user experience.

2.1.6 Operations

Below, existing mode of operations are listed. Some extra operations might be added to next version of the document.

- For user organization, there will be several modes of operations as stated in detail in subsection 2.1.2 User Interfaces.
- Since video recording will be done instantaneously, there will be no concern about data loss. Therefore, there will be no need for backup and recovery operations.

2.2 Product Functions

2.2.1 Common Use Case

Common use case of all functionalities is showed below. They will be explained in detail in upcoming sections.

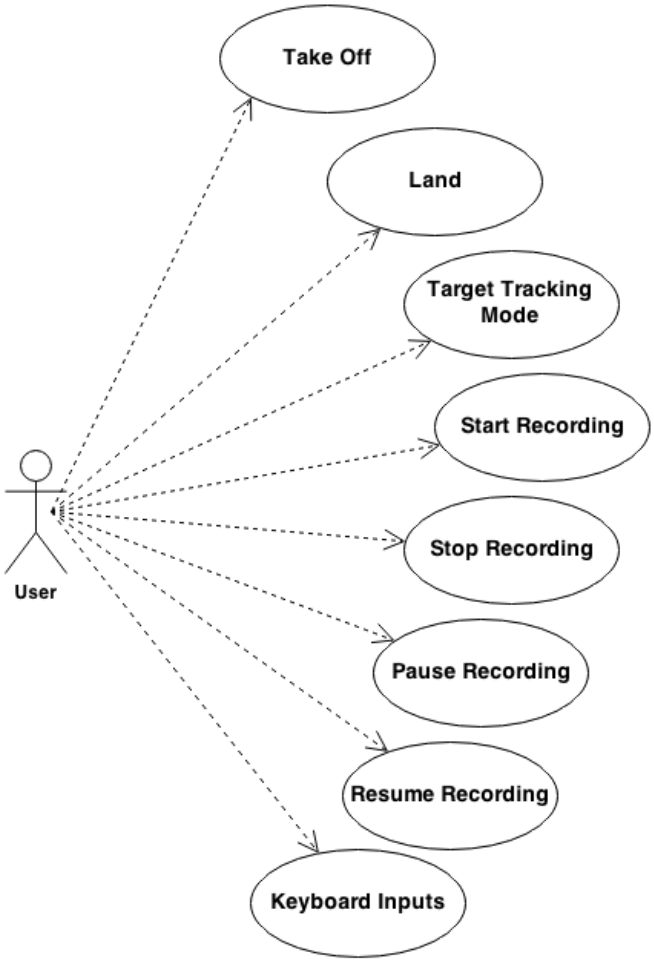

Figure 7: Common Use Case Diagram

2.2.2 Take Off Use Case

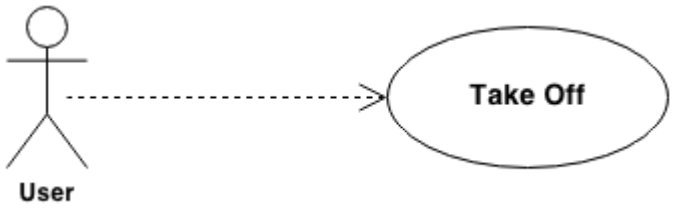

Figure 8: Take Off Use Case Diagram

In this case the user will push the "Take Off" button on the user interface screen. And thanks to this button, required command will be given to the quadcopter and the quadcopter will take off smoothly. And after taking off the user can choose Target Tracking Mode or he/she can control the quadcopter manually.

2.2.3 Land Use Case

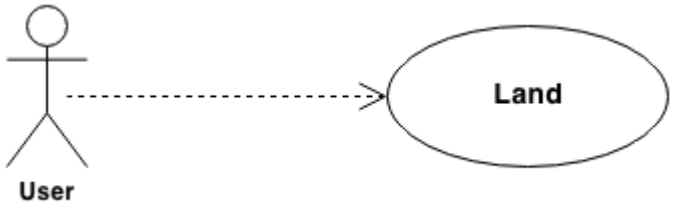

Figure 9: Land Use Case Diagram

In this case, If the user push the "Land" button on the user interface, the required command for landing will be given directly to the quadcopter. And then quadcopter will loose its altitude and when it comes to enough altitude, it will land.

2.2.4 Target Tracking Mode Use Case

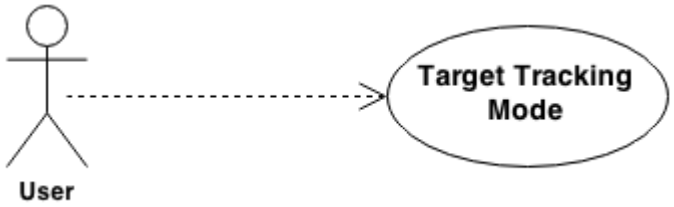

Figure 10: Target Tracking Mode Use Case Diagram

When the user push the button named "Target Tracking Mode" on the user interface, then the tracking mode will be activated. In this mode the user should select the target area and the quadcopter will find the target in this area by using image processing technics or geolocation technics. And it will track the target without expecting any commands from the user. And also in target tracking mode the user use the recording mode which will be given in detail in below sections.

2.2.5 Start Recording Use Case

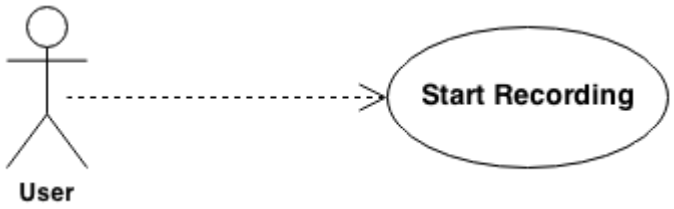

Figure 11: Start Recording Use Case Diagram

The user can activate this mode from the upper-left side of the user interface which is shown as small green camera image. When the user click on this image, the recording will start and there will be two different colored camera image on the screen. First one is yellow and the second one is red. And also there will be information about the situation of the recording and record time below these images. In this case the situation will be "Recording is ON".

2.2.6 Stop Recording Use Case

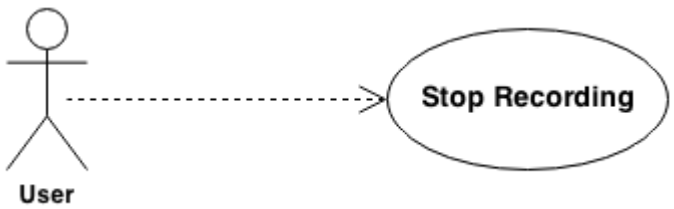

Figure 12: Stop Recording Use Case Diagram

The user can stop recording from the user interface by clicking on the red camera image which is next to the yellow camera image. When the record is stopped, then the situation of the record will be "Recording is OFF" .

2.2.7 Pause Recording Use Case

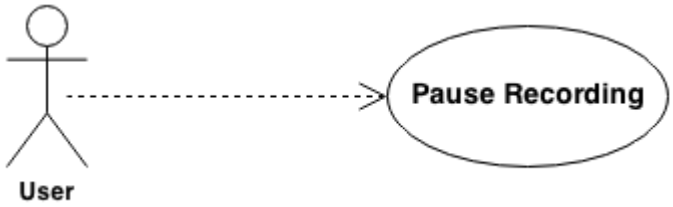

Figure 13: Pause Recording Use Case Diagram

The user can pause recording from the user interface by clicking on the yellow camera image which is next to the red camera image. When the record is paused, then the situation of the record will be "Recording is PAUSED" . When the pause mode activated the yellow button will turn to bright green.

2.2.8 Resume Recording Use Case

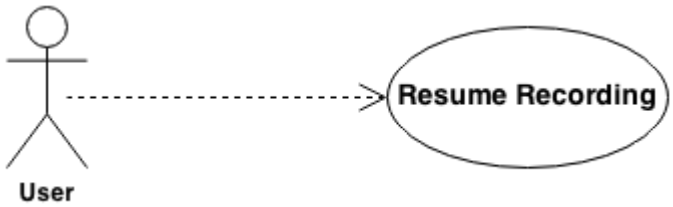

Figure 14: Resume Recording Use Case Diagram

When the user wants to resume recording, he/she should click on the bright green camera image. And then the record will continue, from where it is left. Then the situation of the recording will be "Record is ON" again.

2.2.9 Keyboard Inputs

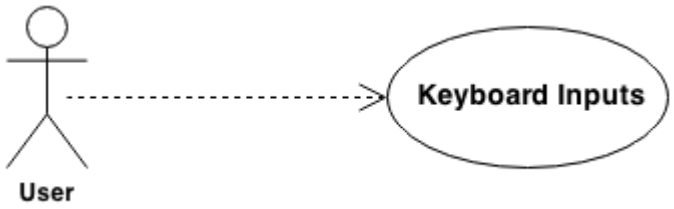

Figure 15: Keyboard Inputs Use Case Diagram

If the user wants to control the quadcopter from the computer, he/she should click on the "SHOW KEYBOARD INPUTS" button and then the keyboard inputs will show up on the screen. Then the user control the quadcopter easily by using these buttons.

2.3 Constraints

The system may suffer from the following major problems;

a) Limited Video Record Time : Since the memory is limited, the video record time also will be limited.

b) Battery Lifetime(Power) : Since AR.Drone has a limited battery, it's lifetime will be limited and it will affects the efficiency of the quadcopter.

c) Environmental Issues: Having lots of obstacles and the weather conditions can affects the performance of the quadcopter negatively.

2.4 Assumptions and Dependencies

To use the end product, the user should know basic Graphical User Interface utilization. Also for recording video, 8 GB USB Flash Disk is recommended. To control the inputs and also quadcopter, Rasperry Pi with mouse and keyboard will be used.

3.Specific Requirements

In this section, there will be detailed information about the software requirements as needed to enable the development team to satisfy those requirements for the system and testers to test whether those requirements satisfy.

After this point, no design specifics will be provided. Because this document is a software requirement specification document, all materials that will be given including diagrams should be considered from analysis point of view. Understandability is more important concept than appropriate application of designing and modeling standards.

3.1 Interface Requirements

In this subsection, there will be more information that was stated in section 2.2 which is Product Functions. However, there will be no repeated information.

In this project, we will use Java Graphics libraries for UI. UI will have simple buttons for basic tasks (Take Off/Land,Start/Pause/Stop Recording and Enable/Disable Tracking Mode) and will use keyboard inputs for manual control over AR.Drone. Keyboard shortcuts may be changeable by the user. While whole background will be live-view of AR.Drone's camera, there will be also a small map at the bottom left for giving user geolocation datas in a better way.

3.2 Functional Requirements

In this subsection, each major software functions will be described in detail.

3.2.1 Using interface for giving commands to AR.Drone

Basically, users will be able to control AR.Drone by using this interface. Normally, AR.Drone can be controlled by giving shell commands to it. However, this attitude is not good for the users since it is not easy for them to use shell commands. Instead of this, an interface will be designed for them. In this interface, basically, there will be buttons for taking off and landing. In addition to these, there will be buttons for directions such as right, left, up and down. Also, buttons for rotating it will exist in the interface.

3.2.2 Target tracking

In the project, there will be option of following a target with a quadcopter. Target will be determined by the user at the beginning and quadcopter will start recording video in auto mode. In order to detect the target continuously, some image processing algorithms will be implemented. OpenCV Library will be used during implementation of such image processing algorithms.

3.2.3 Manual Control During Video Recording

Users will have an option of recording events by controlling AR.Drone manually. With this option, users will have almost full control over AR.Drone, only gyroscope based stabilization will be left to the flight computer.

3.2.4 Wireless Transmission of Video Simultaneously

From the first second of an established communication with AR.Drone, every frame the camera captures will be transmitted to the controller computer. Such a live-view feature will be extremely helpful for the user especially in Manual Control mode. Also simultaneous recording will make video recordings more reliable (no data loss at a crash) and more practical (user can use it according to his/her desire from the very first moment with no waiting time). For that feature, Ros Topic Commands will be given to AR.Drone Autonomy driver at the background.

3.2.5 Cooperating with other quadcopters

While recording video, there can be other quadcopters capturing the event from different aspects. To be able to do that in most efficient way, connection and cooperation between those quadcopters should be established.

3.3 Non-functional Requirements

In this subsection, system's non-functional requirements such as performance, availability, reliability and hardware constraints will be explained.

3.3.1 Performance Requirements

For the performance issue, target tracking is the most important functionality of the project to be considered. In order not to loose the target's location while following it, detection of the target must be finished as fast as possible. In the project, it is considered to be over within 200 ms. Other than that, bandwidth of the communication between the quadcopter and the controlling computer should be large enough to transmit 720p@30fps video with a proper encoding.

3.3.2 Design Constraints

In this project, Java will be used as programming language. In addition to the programming language, OpenCV libraries will be used for target tracking.

For hardware constraints, Ubuntu 14.04 is used as operating system. Moreover, X86 architecture is used and ARM also will be supported. In addition to these, there will be memory limit for video recording which is 8 GB.

The system will be available as long as a single charge of AR.Drone lasts. 10 minutes is enough for AR.Drone to be available for usage. The will be no hot-swapping option.

The system's reliability is based on the capabilities of AR.Drone in the case of encountering with any obstacle in the environment. In this obstacle case, AR.Drone should escape from all obstacles it sees. Other issue to think for the system's reliability is for the case of loosing connection of AR.Drone. In this case, it should have emergency landing.

The project will be open source and all codes will be documented.

4. Data Model and Description

4.1 Data Description

4.1.1 Data Objects

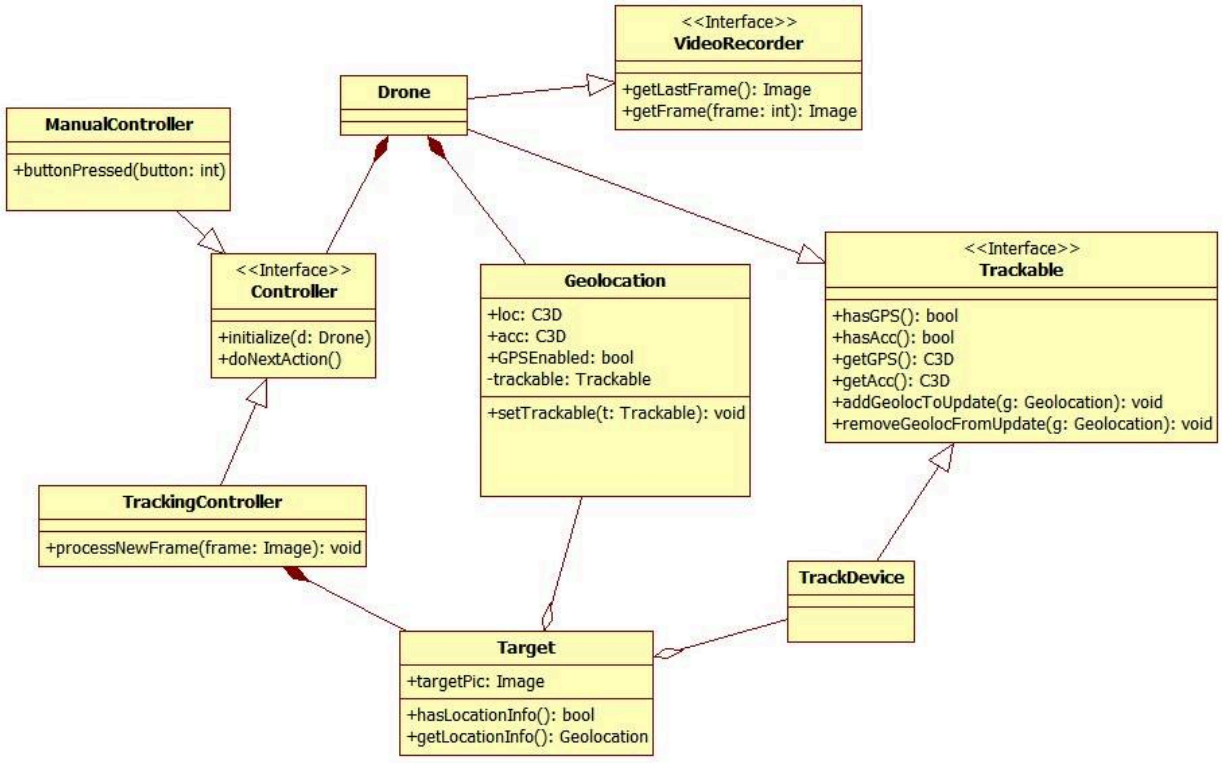

Figure 16: Class Diagram

Trackable: Common interface for trackable objects with some sensor data which provide some geolocational information (GPS or Accelerometer data). Target tracking with Computer Vision is out of this interface's scope.

VideoRecorder: Common interface for objects with an image stream.

Controller: Common interface for controlling algorithm classes, part of the "Strategy Design Pattern"

ManualController: Controlling algorithms for manual controlling with keyboard, part of the "Strategy Design Pattern"

TrackingController: Controlling algorithms for target tracking with Computer Vision and/or Geolocational Tracking, part of the "Strategy Design Pattern"

Target: Target information for TrackingController objects, may contain image information for Computer Vision based tracking and geolocation info for Geolocation Sensors based tracking.

TrackDevice: Software representation of optional tracking device for target (for example: mobile phone of the target).

Geolocation: Location data representation for objects. Location info could be fed by an Trackable object.

Drone: Software representation of AR.Drone, a Trackable, VideoRecorder object with a Controller, as a part of "Strategy Design pattern"

4.1.2 Data Dictionary

Geolocation: Real-World location of an object. GPS, Radar or Accelerometer based identifications are some example ways to obtain geolocation data.

GPS: Global Positioning System, a satellite powered navigation system for obtaining geolocation data. Based on measuring distance between the object and multiple satellites (at least 4) via signals.

Accelerometer: A sensor which measures applied acceleration on an object. Acceleration data may be used to calculate geolocation at any time if initial geolocation is known.

5. Behavioral Model and Description

This section presents a description of the behavior of the software.

5.1 Description for Software Behavior

When the users wants to control AR.Drone by themselves, they can use the friendly user interface. Therefore they can easily control the AR.Drone without using shell commands. When the users push some button on this interface, the software will call required commands for them.

And if the user choose the target tracking mode, the software of the AR.Drone will start working without taking commands from the user. It will detect the target by image processing technics or some geolocation technics and then will start to record the target which could be dynamic.

Users also can record video in both cases with two option. First option is recording to the usb flash drive by plugging into the AR.Drone. This will be done automatically by the AR.Drone. Another option is taking video stream from the quadcopter and recording this video to the computer by wireless transmission. This will be done with Ros Topic Commands and it will be given to AR.Drone Autonomy driver at the background.

When the number of quadcopters increase, they should cooperate synchronized. In order to do that we select one of them as master and the others will work according to it. When they recording same event, they will record the event from different aspects by changing directions or their degrees.

5.2 State Transition Diagrams

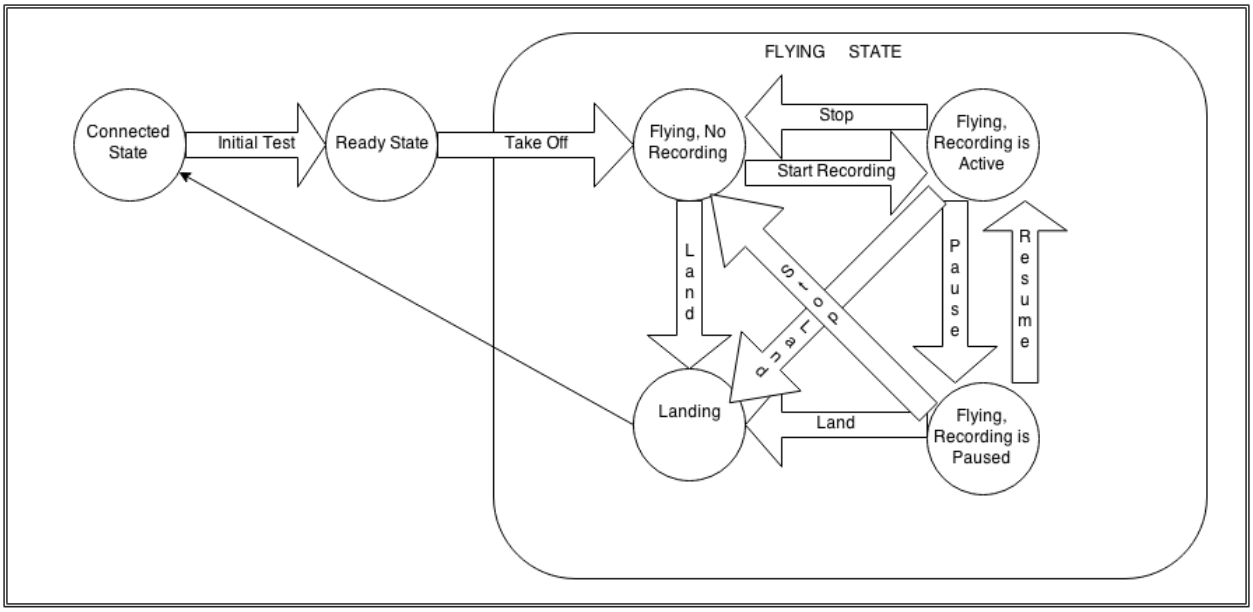

Figure 17: State Transition Diagram

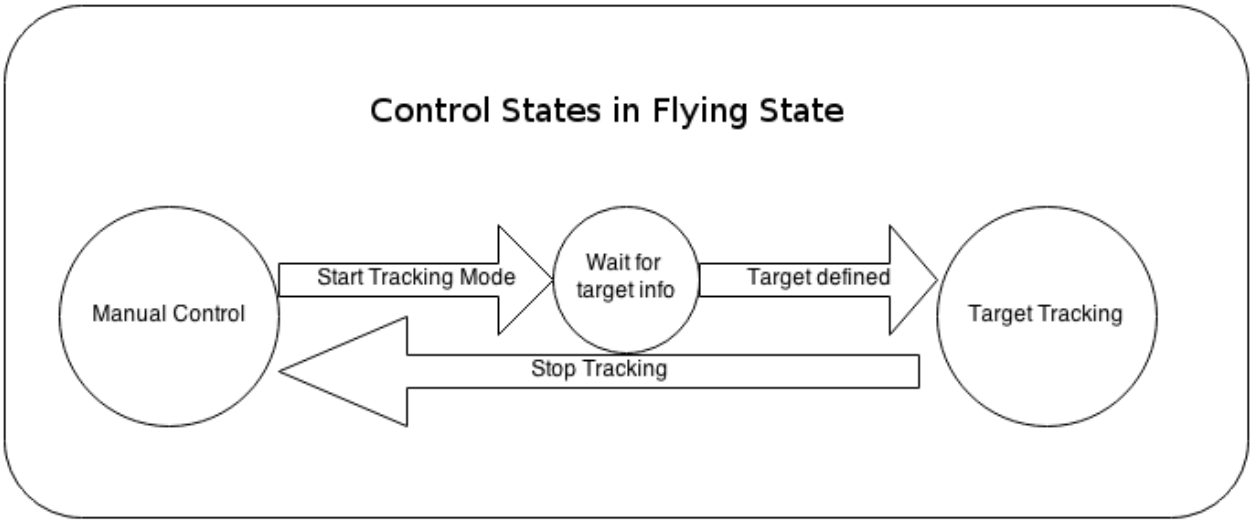

Figure 18: Control States in Flying State

6. Planning

In this section, the team structure of the project, estimated basic schedule and the process model will be shown.

6.1 Team Structure

Asst. Prof. Selim TEMIZER - Project Advisor

Ayşe Gül YAMAN - Advisor

Burak ÖZALP - Developer

Bülent ÖZGÜNER - Developer

Fazilet ÖZER - Developer

Hüsnü ŞENER - Developer

6.2 Estimation (Basic Schedule)

In this semester, mainly design process and documentation process will be handled.We have decided to meet weekly to achieve our goals.The development and enhancement will be main focus of the next semester and rest of the this semester.

	Task Name	Duration	Start	ETA
1	Team Management	2 day	22.09.14	25.09.14
2	Project Selection	12 days	24.09.14	05.10.14
3	Project Idea Proposal	5 days	06.10.14	10.10.14
4	User Story and Proposal Document	9 days	11.10.14	19.10.14
5	Literature Resarch	7 days	20.10.14	26.10.14
6	Installations for Communication with AR.Drone	12 days	27.10.14	09.11.14
7	Creating sample Interface for sending commands	5 days	10.11.14	14.11.14
8	Software Requirement Spesifications Document	14 days	15.11.14	28.11.14
9	Implementation of the Tracking Software	12 days	29.11.14	10.12.14
10	Implementation of the Controlling Software	9 days	11.12.14	20.12.14
11	Merging Final Software	9 days	20.12.14	28.12.14
12	Software Design Description Document	7 days	29.12.14	04.01.15
13	Final presentation	5 days	05.01.15	09.01.15
14	Demo	2 days	10.01.15	11.01.15

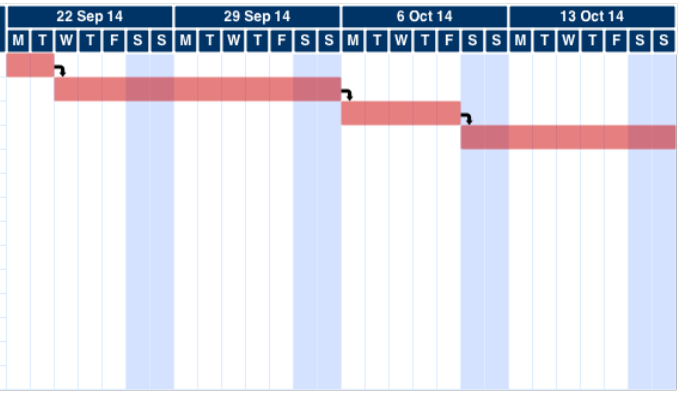

Figure 19: Schedule of the first month of the term

	Task Name	Duration	Start	ETA
1	Team Management	2 day	22.09.14	25.09.14
2	Project Selection	12 days	24.09.14	05.10.14
3	Project Idea Proposal	5 days	06.10.14	10.10.14
4	User Story and Proposal Document	9 days	11.10.14	19.10.14
5	Literature Resarch	7 days	20.10.14	26.10.14
6	Installations for Communication with AR.Drone	12 days	27.10.14	09.11.14
7	Creating sample Interface for sending commands	5 days	10.11.14	14.11.14
8	Software Requirement Spesifications Document	14 days	15.11.14	28.11.14
9	Implementation of the Tracking Software	12 days	29.11.14	10.12.14
10	Implementation of the Controlling Software	9 days	11.12.14	20.12.14
11	Merging Final Software	9 days	20.12.14	28.12.14
12	Software Design Description Document	7 days	29.12.14	04.01.15
13	Final presentation	5 days	05.01.15	09.01.15
14	Demo	2 days	10.01.15	11.01.15

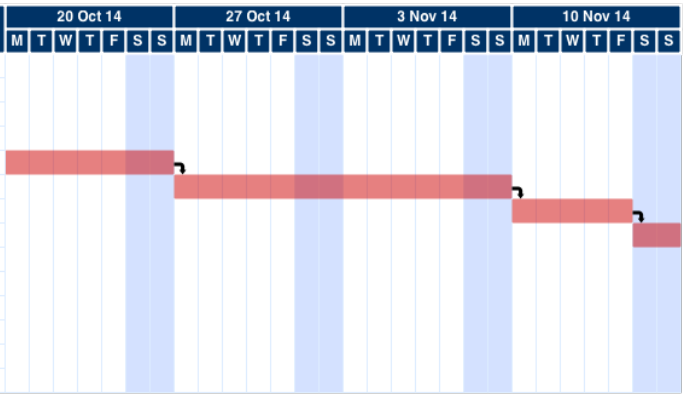

Figure 20: Schedule of the second month of the term

	Task Name	Duration	Start	ETA
1	Team Management	2 day	22.09.14	25.09.14
2	Project Selection	12 days	24.09.14	05.10.14
3	Project Idea Proposal	5 days	06.10.14	10.10.14
4	User Story and Proposal Document	9 days	11.10.14	19.10.14
5	Literature Resarch	7 days	20.10.14	26.10.14
6	Installations for Communication with AR.Drone	12 days	27.10.14	09.11.14
7	Creating sample Interface for sending commands	5 days	10.11.14	14.11.14
8	Software Requirement Spesifications Document	14 days	15.11.14	28.11.14
9	Implementation of the Tracking Software	12 days	29.11.14	10.12.14
10	Implementation of the Controlling Software	9 days	11.12.14	20.12.14
11	Merging Final Software	9 days	20.12.14	28.12.14
12	Software Design Description Document	7 days	29.12.14	04.01.15
13	Final presentation	5 days	05.01.15	09.01.15
14	Demo	2 days	10.01.15	11.01.15

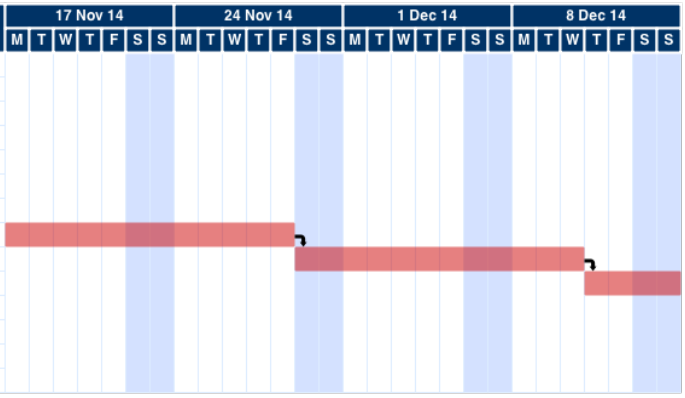

Figure 21: Schedule of the third month of the term

	Task Name	Duration	Start	ETA
1	Team Management	2 day	22.09.14	25.09.14
2	Project Selection	12 days	24.09.14	05.10.14
3	Project Idea Proposal	5 days	06.10.14	10.10.14
4	User Story and Proposal Document	9 days	11.10.14	19.10.14
5	Literature Resarch	7 days	20.10.14	26.10.14
6	Installations for Communication with AR.Drone	12 days	27.10.14	09.11.14
7	Creating sample Interface for sending commands	5 days	10.11.14	14.11.14
8	Software Requirement Spesifications Document	14 days	15.11.14	28.11.14
9	Implementation of the Tracking Software	12 days	29.11.14	10.12.14
10	Implementation of the Controlling Software	9 days	11.12.14	20.12.14
11	Merging Final Software	9 days	20.12.14	28.12.14
12	Software Design Description Document	7 days	29.12.14	04.01.15
13	Final presentation	5 days	05.01.15	09.01.15
14	Demo	2 days	10.01.15	11.01.15

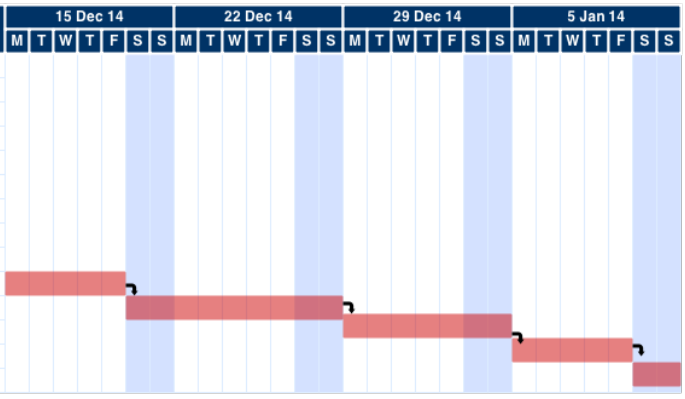

Figure 22: Schedule of the first month of the term

6.3 Process Model

In this project, we will use an scrum process with agile development, so that system can respond quickly to changing requirements without excessive rework. In this model we will have sprints and after every sprint will last 4 weeks. We will have Scrum meetings in each week and we will discuss what we have done and what we will do and also we will have daily scrum meetings. Once we generate the initial version of the product, then our product will be developed according to deficiency of the product. In every sprint we will add some new functionalities to old version.

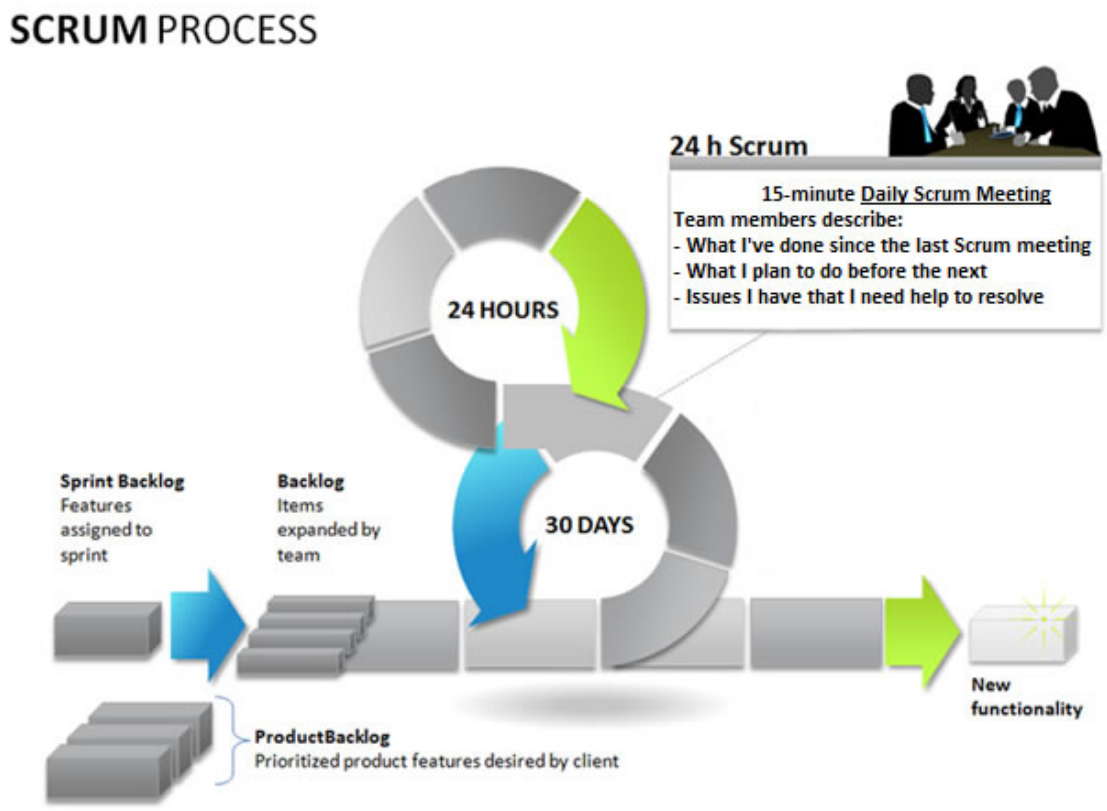

Figure 23: Scrum Process

Image taken from: https://www.cprime.com/resources/what-is-agile-what-is-scrum/

7. Conclusion

The aim of this document to provide information about the project "General Purpose Intelligent Flying Camera with Quadcopters" which enables users to capture events that are difficult to capture by classical cameras. Firstly, the problem is described, and after that our solution to this problem is clarified in this document. In order to clarify, we explain functional and non- functional requirements and also some constraints such as memory, performance and design for this project. Relationships between users and functions, user characteristics and roles are shown by diagrams. Moreover, necessary assumptions that sustain the accuracy are determined to be able to maintain a credible product. After all, our team members and structure, estimated basic schedule and our process model are shown.