CENG 491

Computer Engineering Design

SOFTWARE REQUIREMENT SPECIFICATIONS

Venture Co.

Efe Suat ERDUR

Basrican ŞEN

Yunus Emre AKBABA

Fırat AKYILDIZ

Table of Contents

1 Introduction

This software requirements specification report is created by Venture Co. members to be submitted to METU Computer Engineering Department in the concept

of design lecture. The document includes software requirements for AnelArge. An overview of the problem, specific requirements, data model and descriptions are explained in this document.

1.1 Problem Definition

Perimeter Surveillance Systems is a class of radar sensors that monitor activity surrounding or on critical infrastructure areas such as airports, seaports, military installations, national borders, refineries and other critical industry and the like. They mostly use static cameras and sensors. These systems are known as Perimeter Surveillance Radar (PSR). Such radars are characterized by their ability to detect movement at ground level of targets such as an individual walking or crawling towards a facility. Such radars typically have ranges of several hundred meters to over 10 kilometers. These radar systems are quite efficient if the guarded area is well known and it is eligible to construct such platforms, which are exampled above. On the other hand, if the guarded area is unknown and can be changeable, constructing static sensor solutions are useless. Low cost unmanned air vehicles have increasing popularity day by day and can also be used to provide situation awareness in such environments. In the concept of this project, we aimed to solve this surveillance problem in the unknown areas, by creating a multi agent based unmanned air vehicle simulation system.

1.2 Purpose

The purpose of this document is to give detailed information about the overall description of all the functions and specifications, the requirements, data model and behavioral model of our multi agent based unmanned air vehicle simulation system.

1.3 Scope

The scope of this project basically includes the expectations of our sponsor from our group. In addition to this, it includes functionalities, constraints, assumptions, dependencies, behavioral and data models of the project.

1.4 User and Literature Survey

As the result of our researches, we see that the existing projects are tried to solve our problem by using static sensors and cameras. It is quite fair to solve the problem if the area is well known and it is eligible to construct such platforms. But considering exploring a unknown and changeable area, these solutions become deficient and our researches about the solutions of this conditions came to naught. We believe that our 'Perimeter Search and Patrolling Using Limited Capacity Unmanned Air Vehicles' system will be able to solve this problem after it is ported into the real life and get a considerable success on military purposes.

We also made some research on the existing multi agent simulation frameworks. Some of these frameworks are explained below.

1.4.1 Mason

“MASON is a single-process discrete-event simulation core and visualization toolkit written in Java, designed to be flexible enough to be used for a wide range of simulations, but with a special emphasis on “swarm” simulations of a very many (up to millions of) agents. The system is open-source and free.” [7]

1.4.2 OpenMap

OpenMapTM is a Java BeansTM based toolkit for building applications and applets needing geographic information. Using OpenMap components, you can access data from legacy applications, in-place, in a distributed setting. At its core, OpenMap is a set of Swing components that understand geographic coordinates. These components help you show map data, and help you handle user input events to manipulate that data. [9]

1.5 Definitions and Abbreviations

SRS	:Software Requirements Specification
UAV	:Unmanned Aerial Vehicle
MASON	:Multi-Agent Simulator Of Neighborhoods

1.6 References

[1] Towards a Standardization of Multi-Agent System Frameworks, by Roberto A. Flores-Mendez

[2] Bradshaw, J.M. An Introduction to Software Agents. In: Software Agents, J.M. Bradshaw (Ed.), Menlo Park, Calif., AAAI Press, 1997, pages 3-46.

[3] The Affective Reasoner : A process Model of Emotions in a Multi Agent System, by Clark Davidson Elliot, June 1992

[4] Genesereth, M. An Agent-based Framework for Interoperability. In: Software Agents, J.M. Bradshaw(Ed.), Menlo Park, Calif., AAAI Press, 1997, pages 317-345.

[5] Nwana, H.S. Software Agents: An Overview. In: The Knowledge Engineering Review, October/November 1996, Volume 11, Number 3, pages 205-244.

[6] Shoham, Y. An Overview on Agent-oriented Programming. In: Software Agents, J.M. Bradshaw (Ed.), Menlo Park, Calif., AAAI Press, 1997, pages 271-290.

[7] MASON: A New Multi-Agent Simulation Toolkit, Sean Luke, Claudio Cioffi-Revilla, Liviu Panait, and Keith Sullivan

[8] Sean Paus. Floodland: A simple simulation environment for evolving agent behavior. Technical report, Department of Computer Science, George Mason University, Fairfax, 2003.

[9] http://ostatic.com/openmap-2

1.7 Overview

This document includes software requirements for our multi agent based

unmanned air vehicle simulation system. To create a perceptible document, the functionalities, behavioral and data models are explained in detailed by using use cases, flow-charts and diagrams. The overview of the document can be described step by step in this way;

In chapter 2, the general factors that affect the product and its requirements are explained. The structure of the project and major functions are described in this chapter by using use-case diagrams. In addition to this, the assumptions and dependencies that affects the requirements,hardware limitations, safety and security considerations and some similar concepts are tried to be explained in this chapter.

In chapter 3, specific requirements are explained under three subtopics. Interface requirements, functional requirements and non-functional requirements are described under these subtopics.

Chapter 4 includes detailed information about our data model. Data objects of the project and their attributes are described in this chapter. Moreover, the relationships among these data objects are described in this chapter in a superficial method. In the end there also exist an electronic data dictionary.

Chapter 5 is related to the general behaviour of the project. The possible events and states, in which state what is going to be the behaviour of the system are described in this chapter.

Chapter 6 is the planning part. There is a brief information about group members, estimated time schedule of the project progress and some information about process model.

In chapter 7, there is a conclusion part.

2 Overall Description

We aim to create a simulation framework that will consist of building a control and distributed task execution framework to be used with unmanned air vehicles. Besides the basic properties of the project, which will be explained in detailed in the following chapters, we are encouraged to make the communication and transmitting the video images in a P2P environment available to be ported on real systems with no or a

little modifications and be implemented realistically.

2.1 Product Perspective

We are planning to construct our project in five basic modules. The first module was, no doubt, will be the basic part and will be responsible for the construction and implementation of separate agents doing similar duties and be in a communication with each others during the whole cycle of program. Constructing these agent classes, their behavior under different situations like number of agents, size of the environment, weather conditions or battery levels of the agents (in real life this will refer to the quadrotors) are going to be defined in this module.

Since the user wants to get the data during the patrolling period and afterwards, wants to interfere the conditions and maybe the controls of the agents, we are going to create a user interface that presents the user an environment that he can apply this desires. This part is going to be our second module of the project. It will start at the beginning of the program, ask user to enter some data about the conditions then let the agents to be created and implemented. For this part, the company does not expect us a very developed interface of the simulation since the system will be ported to the real life as soon as it successfully passes the test conditions at the end of the year. Some desired data on the console is enough for now; on the other hand, due to the fact that we are planning to use some features of Mason in the base module, we are also planning to use it's 2D visualization features to create a java based interface which can be used more efficiently.

The other module will be scenario. This module is used to hold the important information about the environment and the nature.

Fourth module will be workplan which will be used for storing the workplan of an agent. This module will also be used to construct a cycle which continues working since the goal is achieved or the program terminates.

The last module, which is the most important will be core. This module will be directly or indirectly connected to the each agent on the program and be responsible for creating plans for the agents.

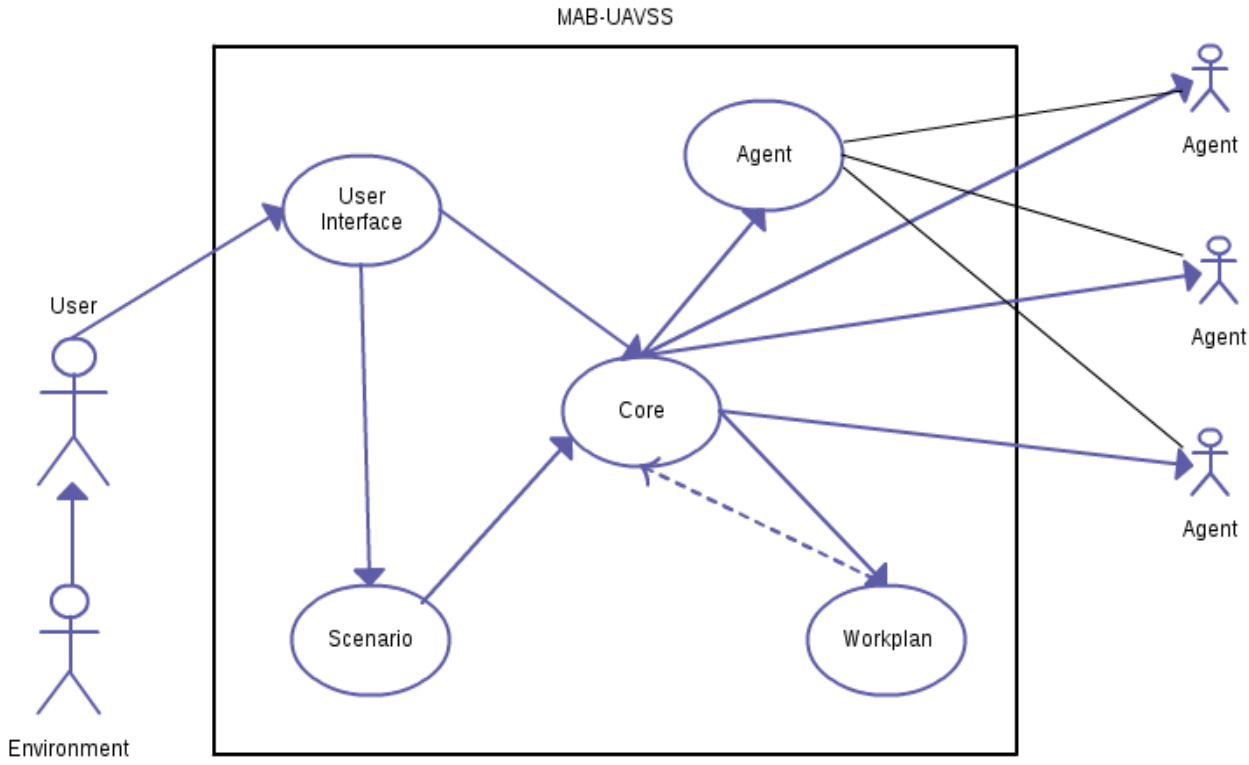


Figure1

2.2 Product Functions

Scenario Module Functions : New, Save, Load,Update,Delete

Users can create new scenarios to simulate on. After creating a scenario,user has to fill the required fields(the scenario details). At this point, user can execute the newly created scenario. The software also includes a save/load mechanism, so that the scenarios can be saved and loaded at a later time. Also, a user can update an existing scenario or delete it.

Core Module Functions : Begin,Pause,Resume,Stop,Open,Close,Save,Load

After creating a new scenario or loading an existing one, user can start the simulation. At any time during the simulation, user has the option to pause or stop the simulation. If paused, user can change some settings of the according scenario and resume the simulation. By this way,user can see how the UAV's behave in a changing and possibly unexpected environment. In addition, user has the option to save the simulations, apart from the scenarios,so that a continuing simulation can be stopped and resumed at a later time.By this way, simulations which take too much time can be finished with many sessions. This option also saves a lot of time if some other unexpected (and unwanted) incidents(by this unexpected incidents, we mean something that is not related to our software of course, such as electricity shortcut or OS crash) happen during the simulation, which would otherwise lead to the loss of data and time.

2.3 Constraints, Assumptions and Dependencies

The project will be used by AnelArge to be integrated in real life with real vehicles. It will probably be a part of military service project, so it is not going to be released under open-source free software domain.

The only available language of the system will be English.

3. Specific Requirements

3.1. Interface Requirements

The simulation will have menu interface, 2D visualisation interface. Users can select some properties like they want in menu interface. Menu interface will has resume, options, load and exit buttons. In 2D visualisation interface users can see what the agents doing, where they've gone, agents' sight area, agents' positions and also all the environmental effects like wind, weather, fog...

The area that our agents covered will be shown with different colors. Also agents will have a sight area shown a circle around them with different colors. So users can easily see what they are doing at that moment.

3.2. Functional Requirements

3.2.1. USERS

Users can start a new simulation, set parameters, load scenarios, stop simulation, resume simulation, change settings, save simulation, see the details of agents at any time.

3.2.1.1. Start a New Simulation

This feature allows users to start a default, new simulation.

Data Flow

i. User opens simulator.
ii. Menu is opened.
iii. User selects “Start a New Simulation” button.
iv. A window opened which has all the settings and parameters that can be changed.

3.2.1.2. Set Parameters

This feature allows users to setting all the parameters of environment and agents.

Data Flow

i. User opens simulator.
ii. Menu is opened.
iii. User selects “Start a New Simulation” button.
iv. A window opened which has all the parameters that can be changed.
v. User changes the values of parameters like he/she wants.
vi. User chooses apply button when he/she finishes with setting values.

3.2.1.3. Load Scenarios

This feature allows users to load saved scenarios or load some other scenarios that exists.

Data Flow

i. User opens simulator.
ii. Menu is opened.
iii. User chooses Load Scenarios button.
iv. A window opened which shows files in the Scenarios file of our program.
v. User can choose any scenario file from anywhere of the computer.
vi. User choose select button after he/she decided to choose the scenario.
vii. A window opened which has all the settings and parameters that can be changed.

3.2.1.4. Stop Simulation

This feature allows users to stop the simulation that is in progress.

Data Flow

i. User starts a new simulation or load a simulation.
ii. When simulation continues user chooses “Stop Simulation” button.
iii. Simulation stopped until user will choose “Resume Simulation” button.

3.2.1.5. Resume Simulation

This feature allows users to resume the stopped simulation.

Data Flow

i. User stops the simulation.
ii. User can change “Resume Simulation” button now.
iii. User chooses “Resume Simulation”.
iv. Simulation continues where it has been stopped.

3.2.1.6. Change Settings

This feature allows users to change some settings of the simulation like some visual settings and some general settings.

Data Flow

i. User opens simulator.
ii. User selects “Settings” button in menu.
iii. A window opens which has radio buttons for enable or disable some settings and sliders for change the values of some settings.
iv. User can change sliders or can enable or disable radio buttons.
v. When he/she finishes changing he/she chooses “Apply Settings” button.
vi. All the settings that he/she changed will be applied when the simulation starts.

Alternative Data Flow

i. When the simulation has already been running user chooses “Menu” button. (When simulation continues if user selects menu then simulation automatically paused.)
ii. User chooses “Settings” button at the menu.
iii. A window opens which has radio buttons for enable or disable some settings and sliders for change the values of some settings. (but user can not change all of the settings when he/she chooses to change settings while simulation is running)
iv. User can change sliders or can enable or disable radio buttons(not

all of them).

v. When he/she finishes changing he/she chooses “Apply Settings” button.
vi. User chooses “Resume Simulation” button.
vii. All the settings that he/she changed will be applied when the simulation continues.

3.2.1.7. Save Simulation

This feature allows users to save their simulations which is in progress at that time so they can continue that simulation another time.

Data Flow

i. When the simulation has already been running user chooses “Menu” button. (When simulation continues if user selects menu then simulation automatically paused.)
ii. User chooses “Save Simulation” button from menu.
iii. A window opens that displays the “Saves” file of simulation.
iv. User can choose any directory from that window.
v. After choosing directory user selects “Save”.
vi. The simulation saved to the directory that user selected.

3.2.1.8. Details of Agents

This feature allows users to see the every detail of the agents anytime they want.

It shows statistics of agents like how much area they covered, how much battery left, how much area it can cover more with that remaining battery...

Data Flow

i. When the simulation is running user chooses any agent that he/she wants to see the details.
ii. A small window opens and it displays the details of agent that is selected.
iii. The details change as the simulation continues, so users can see the changing at anytime.
iv. Users can close the details window with selecting the close (x) button of the details window.

3.3. Non Functional Requirements

3.3.1. Performance Constraints

Since we are at the beginning of the project we do not know the exact performance requirements now. But it's a simulation project so we will have agents moving and while they are moving all the values and calculations changing. So it is not hard to guess that it will require a good hardware and software. The simulation's performance will depends on the computer's performance which is running simulation.

3.3.2. Design Constraints

3.3.2.1. Software Requirements

1. Simulation can be executed in all of the operating systems.
2. All of the coding will be written with Java Language.

3.3.2.2. Recommended Hardware Requirements

1. Pentium 4 or more Processor
2. 2-3 GB DDR2 Ram
3. 128 MB Graphics Card
4. 2 GB Hard disk

(They are not the exact requirements but it will recommended for users to use simulation without any problems. Otherwise they may use it with lower performances.)

3.3.2.3. Software Attributes

Scalability : Only one user can use it on a computer.

Availability : Prototype must be available to its users every time.

Reliability : Simulation's reliability must be high because after this simulation project company wants to integrate it into a real life project.

Usability : Prototype can be used by any user easily because it will have a useful interface so that users can do what they want without having problems.

4 Data Model and Description

This section describes information domain for the MAB-UAVSS.

4.1 Data Description

This section describes the most important data objects that will be used and managed in MAB-UAVSS. It also includes a part where the relationships among the objects are discussed,and a complete data model which would allow the reader of this document to have a better picture of the software. At the end of this section, a data dictionary is provided for reference.

4.1.1 Data objects

Data objects of MAB-UAVSS and their major attributes are described below.

Agent : This data object is for creating agents in our multi agent based simulation. Instances that are created from this data object will operate in the simulation to get the desired results. It will also include the properties of the UAV's.

Interface : This data object will be used for creating an interface so that the user of the software can describe a scenario to be carried out by MAB-UAVSS. Also,t he simulation progress and the results will be available to the user via this interface.

Scenario : This data object will contain everything about the nature of the assignment that is expected for MAB-UAVSS to solve. First, it holds the environment upon which UAV's are going to operate on (for instance a 2D map). It also includes all the environmental variables which may or may not affect the simulation. Finally, it holds the details about the problem which will be solved by MAB-UAVSS.

Workplan : This data object will be used for storing the workplan of an agent. For every agent,there will be an according workplan at a given time.In this workplan,t he behaviour of an agent for a certain period of time will be determined.It will describe everything that an agent has to do in a certaion time period. Upon receiving this object,an agent should know what it has to do and when, so that it will operate independent of the remaining modules of the software. By this way, an agent can go beyond the transmission range of other agents and/or headquarters. After carrying out its orders,it will then come back to a place(which is also included in the workplan) within range of another agent or headquarter, where it is again connected to the network. So it should inform the headquarter(directly or indirectly) that the operations it has been tasked has been executed(or there is some error occured). After that,it will take another workplan for further assignment. This cycle is repeated until all the scenario is completed.

Core : This will be the most important data object in our software since it will act as a headquarter for our agents.Having been connected (directly or indirectly) to all the agents, the Core object will create plans for agents' movements and behaviours.It will also include all the settings that will vary the simulation.

4.1.2 Relationships

Relationships among data objects are described in this section using an ERD-like form. No attempt is made to provide detail at this stage.

Core module will use interface module to supply user with relevant information and take the settings of the Scenario,as in Figure 2 below.

Figure 2

Core module will be using the Scenario module to create distinct plans for every agent, as in Figure 3 below.

Figure 3

Core module will create and submit Workplans, as in Figure 4

below.

Figure 4

Core module will be connected to the agents(or at least one agent) to provide their Workplans.

Figure 5

Agent modules may(and probably will) transact with each other so that the agents far away from the “Headquarters” (the core module) may receive orders and/or send data, as in Figure

6.

Figure 6

4.1.3 Complete data model

Below is a diagram for the software.

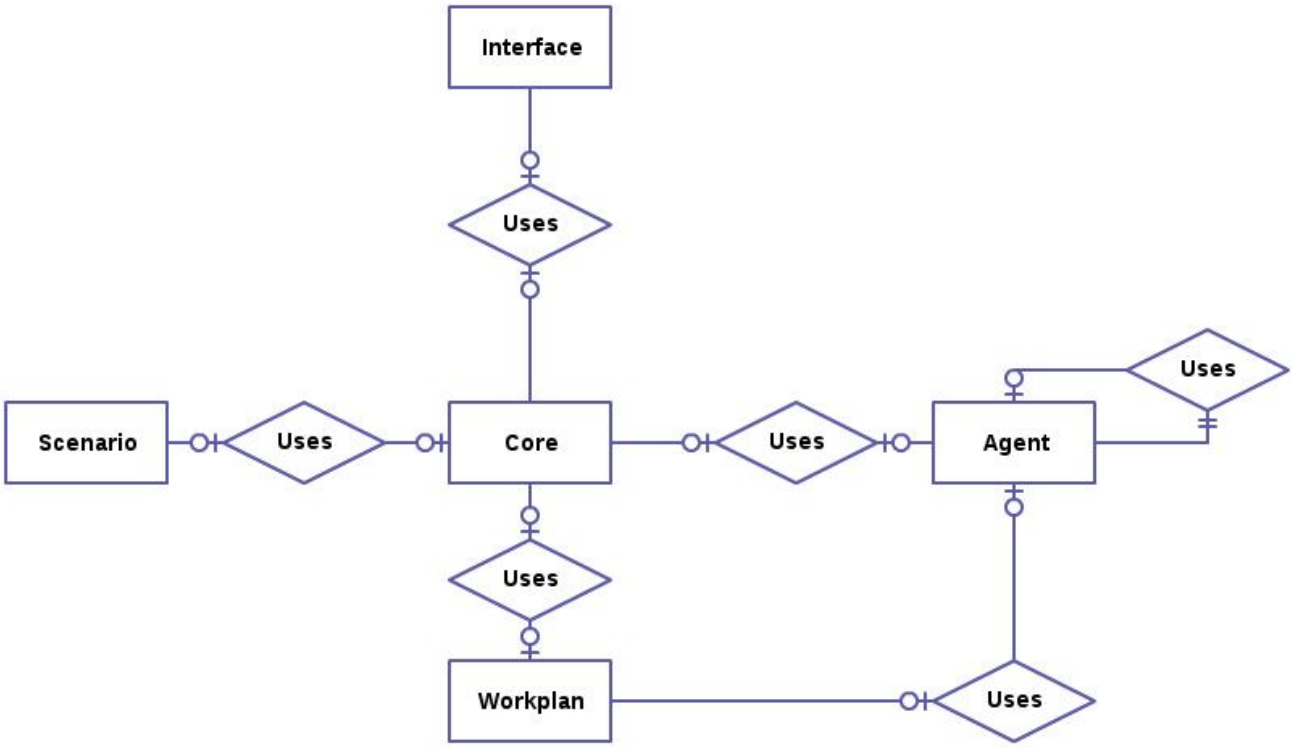


Figure 7

5. Behavioral Model and Description

MAB-UAVSS (Multi Agent Based Unmanned Air Vehicle Simulation System)

is a tool which is designed for perimeter search and patrolling using limited capacity unmanned air vehicles. MAB-UAVSS has three behaviors ;

- Configuration
- Analyze and problem solving
- Simulating the solution

5.1 Description For Software Behavior

5.1.1. Configuration

First of all, our problem is basically depend on patrolling previously specified area with unmanned air vehicles. Therefore, before applying an algorithm we have to be well informed about all specifications of the area,like land forms above the specified area, size of the field , exact coordinates, various environmental aspects that our unmanned air vehicles could be faced with. Apart from these , we have to be well informed about the specification about our unmanned air vehicles. For example; how many vehicle are we going to use to patrol the area, whats the battery level of vehicles or whats the maximum time that the vehicle could stay in the air, or whats the maximum height that vehicle could reach. Therefore at the beginning we have to set up the configurations. MAB-UAVSS expects specification file of the area and additionally the detailed information about the unmanned air vehicles.

5.1.2. Analyze and Problem Solving

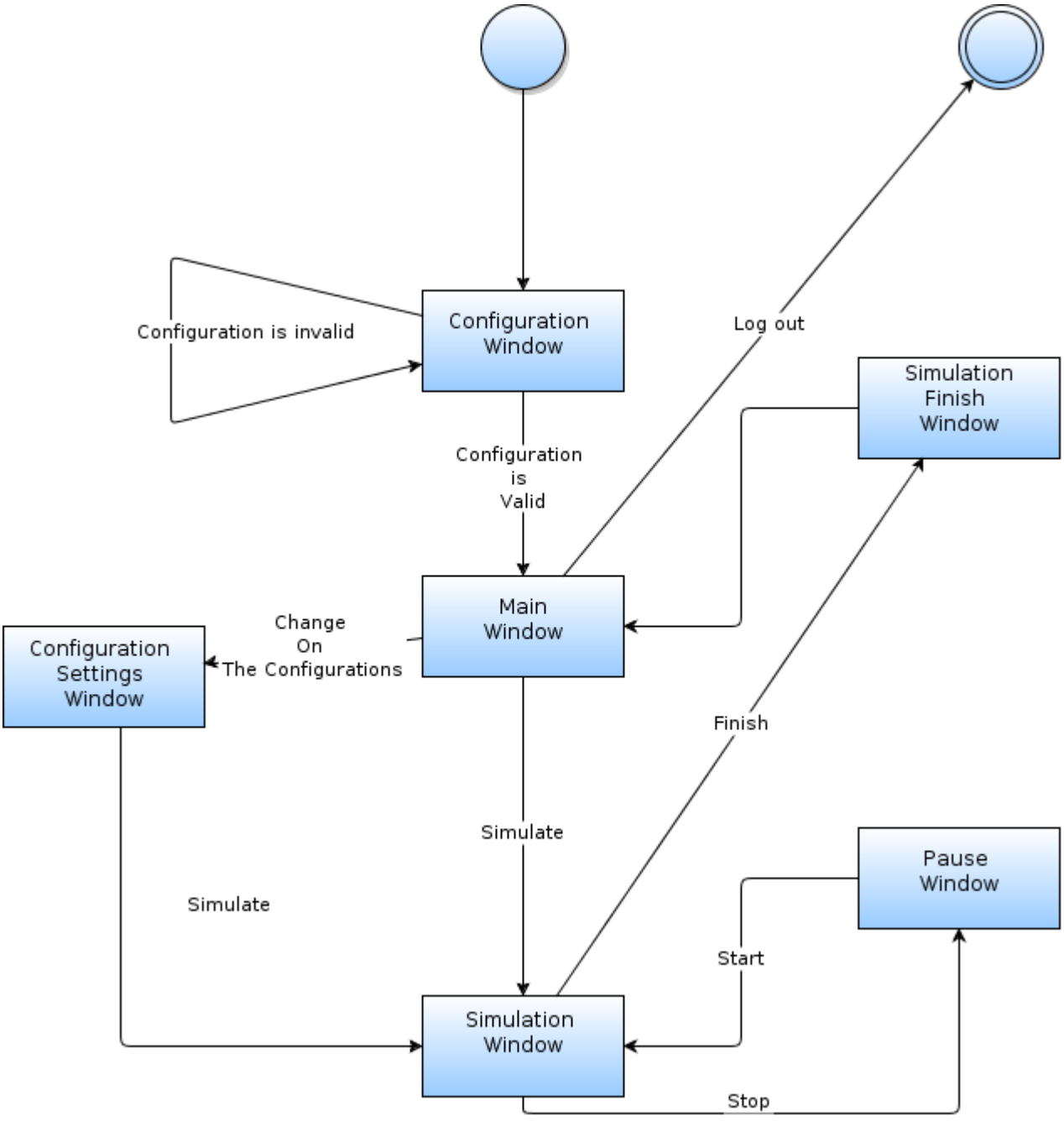
According to the specification file of the area , and the detailed information about unmanned air vehicles, MAB-UAVSS is going to generate several scenarios that accomplish the patrolling job. Of course all of the scenarios won't be efficient, or or applicable. Hence , MAB-UAVSS is going to eliminate the non-applicable ones and from the applicable ones the most efficient one is going to be chosen as solution of the problem.

5.1.3. Simulating The Solution

After analyzing the configuration files and choosing the best solution from all scenario, MAB-UAVSS is going to present the main scenario to the user. according to the steps of the scenario, MAB-UAVSS is going to create an artificial intelligence for all vehicles and command them what to do in the whole patrolling way. Additionally simulate it in 2D ,or 3D to the user.

5.2 State Transition Diagrams

The diagram below depicts the overall behavior of the system.



6. Planning

6.1. Team Structure

During producing **MAB-UAVSS** (Multi Agent Based Unmanned Air Vehicle Simulation System) , we have a sponsored company which is AnelARGE located at

Hacettepe Technopolis, and an adviser from sponsored company is guiding us about course of events. Additionally , we have also an advisor who is teaching assistant from our department. Apart from these, about our team structure;

- Efe Suat Erdur, is the leader of our group, due to his active role in our sponsored company, he is responsible from the organization , and role of agent between AnelARGE and our team.

- Every decision is made by participation of all team members.

- Team have scheduled meetings on every Thursday with the participation of our teaching assistant advisor. Each member has to attend the meetings on time.

- The subjects of the scheduled meetings are responsibilities of each members and the evaluation of the accomplished responsibilities.

-Role of Each Team Members

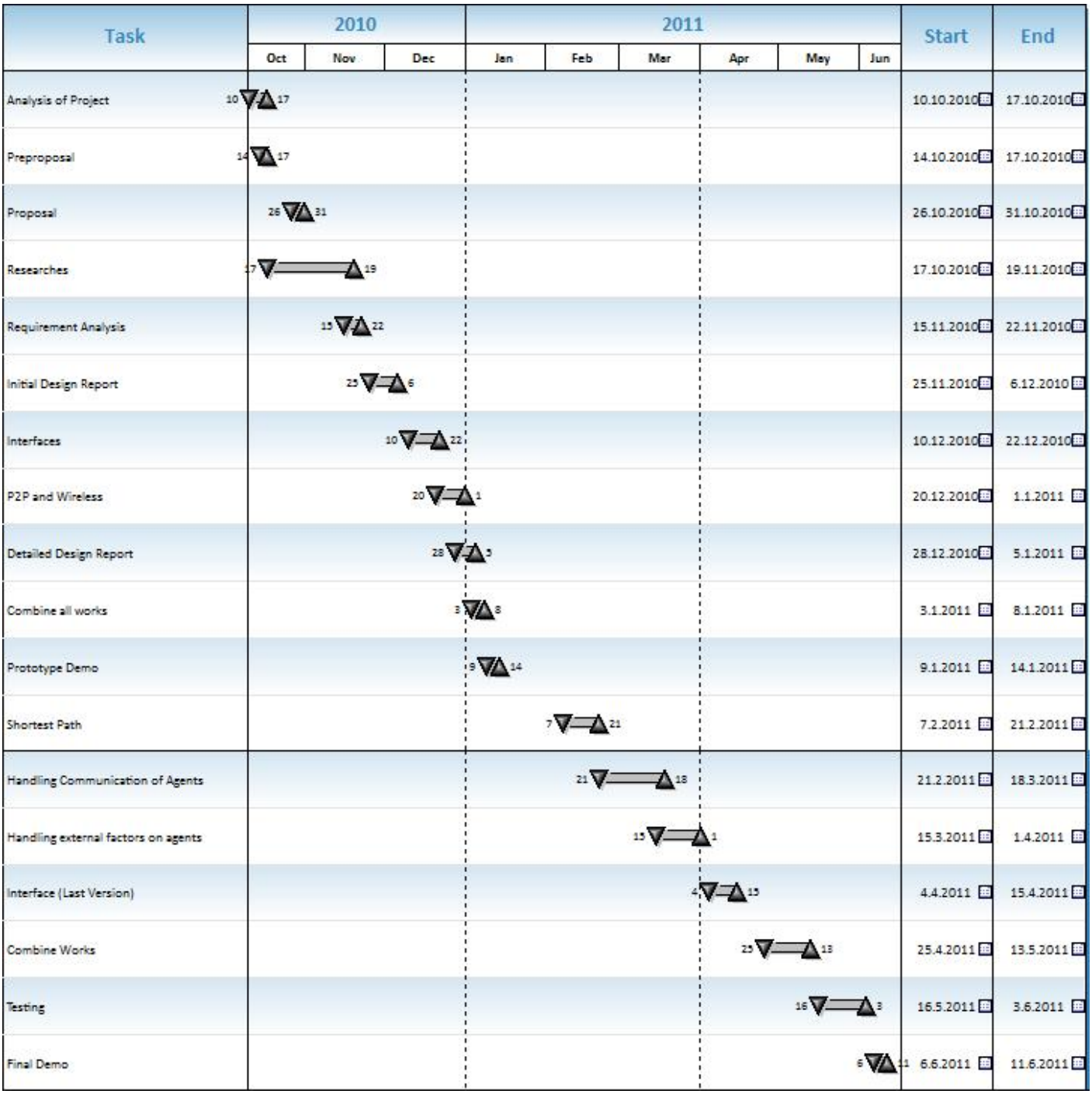
- Efe Suat Erdur : Team Leader, Developer & Group Member
- Fırat Akyıldız : Group Member ,Developer & Report Organizer
- Basrican Şen : Developer & Group Member
- Yunus Emre Akbaba : Developer& Group Member

6.2. Estimation (Basic Schedule)

1.	Analyses of Project	Oct, 10, 2011 - Oct, 17, 2011
2.	Preproposal	Oct, 14, 2011 - Oct, 17, 2011
3.	Proposal	Oct, 26, 2011 - Oct, 31, 2011
4.	Researches	Oct, 17, 2011 - Nov, 19, 2011

5.	Requirement Analyses	Nov, 15, 2011 - Nov, 22, 2011
6.	Initial Design Report	Nov, 25, 2011 - Dec, 06, 2011
7.	Interfaces	Dec, 10, 2011 - Dec, 22, 2011
8.	P2P & Wireless	Dec, 20, 2011 - Jan, 01, 2012
9.	Detailed Design Report	Dec, 28, 2011 - Jan, 05, 2012
10.	Combine all works	Jan, 03, 2012 - Jan, 08, 2012
11.	Prototype Demo	Jan, 09, 2012 - Jan, 14, 2012
12.	Shortest Path	Feb, 07, 2012 - Feb, 21, 2012
13.	Handling Communication of Agents	Feb, 21, 2012 - Mar, 18, 2012
14.	Handling External Factors of Agents	Mar, 15, 2012 - Apr, 01, 2012
15.	Interface (Last Version)	Apr, 04, 2012 - Apr, 15, 2012
16.	Combine Works	Apr, 25, 2012 - May, 13, 2012
17.	Testing	May, 12, 2012 - Jun, 03, 2012
18.	Final Demo	Jun, 06, 2012 - Jun, 11, 2012

Our basic timeline is depicted below in a gantt chart ;



6.3 Process Model

In our project, we will use waterfall as process model. Waterfall model is separate and distinct phases of specification and development.

It is most widely used model in the field of software development since it is a linear model and simple to be implemented, documentation is produced at every stage which brings simple understanding of design procedure and after each step of coding, testing is done to check the code so it enables to eliminate possible errors at each stage. Due to the advantages mentioned above, we have also decided to use this model in our project.

7 Conclusion

All in all, perimeter search and patrolling using limited capacity unmanned air vehicle is a detailed and multi-directional project. **MAB-UAVSS** (Multi Agent Based Unmanned Air Vehicle Simulation System) is going to have the power or solving all the solutions about these problems.

So far, we did framework investigation and specified our route in this way. By the help of this route we are going to overcome the problems and generate the software in time. During this development, report, and demonstration period, we expect that we are going to face with difficulties and problems, and we are sure that we are going to handle them with devotion and the help our advisers.