

FIREBOT SOFTWARE

CS 350 SOFTWARE ENGINEERING II

SPRING 2018 PROJECT

ALI KALKANDELEN

SAMUEL OWUSU-BINEY

"Firebot" Fire Extinguishing Robot

There are more than 20,000 house fires and more than a dozen wildfires in the United States each year. Most of the time these fires are not immediately controlled or rescued due to lack of resources and personnel.

The firebot is a fire-extinguishing robot designed to assist firefighters in times of wildly spreading, uncontrollable fires, where seconds and minutes will make the difference between life and death. Firebot will be a self-sufficient, portable, powerful robot able to fight even the highest of blazes.

The robot will essentially be a water or foam tank attached to a rover along with a hose to spread the liquid evenly among the fire. The water or foam will depend on the situation and will easily be able to get filled through the firetruck or hydrant. The hose will be special stainless steel pipe with a custom tip that can change the spread of the solvent as appropriate for the situation.

The robot will detect the fire using changes in heat signatures on its high field of view camera. The camera will also detect high concentrations of carbon dioxide, which is a common molecule found in almost all fires. Once detected the robot will engage the fire using its reserve and hose as necessary and calculate the best angle and procedure for the extinguishing method.

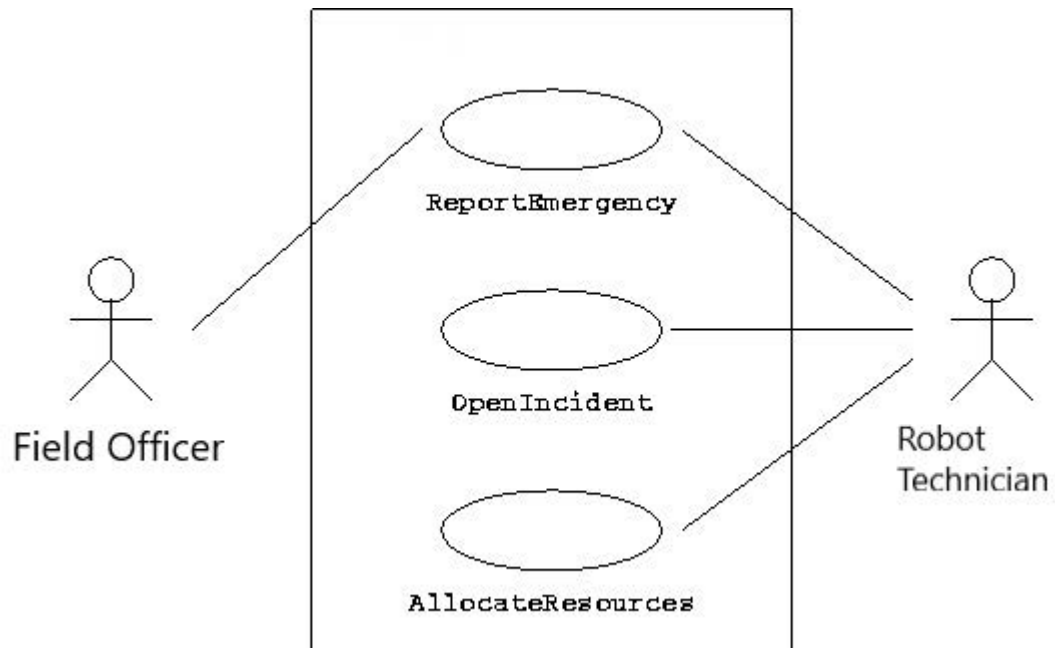
Methods

- o Movement:
 - Directional Movement (int robotID, float xPos) : void
 - Yaw Movement (int robotID, float yPos) : void
 - Call Gimbal Movement (int robotID, float pitch) : void
 - Call Hose Movement (int x, int y, int z) : void
- o Data Storage:
 - Db Connection (int robotIdNum, int DbIdNum) : boolean
 - Get Data (int robotIdNum, char item) : char
 - Put New Data (int robotIdNum, char item) : boolean
 - Replace Data (int robotIdNum, char item) : boolean
- o User Interface:
 - Data Display (int robotIdNum, int UiID) : void
 - Settings Display (int robotIdNum, int settingID) : void
 - Logging (int robotIdNum, char filename): boolean
 - User Menus (int robotIdNum, int menuID) : void
- o Processing:
 - Process HeatSig (int robotId, int sigStrength) : boolean
 - Process ChemTrail (int robotId, int chemStrength) : boolean
 - ProcessPower (int robotId, float powerLvl) : float
 - ProcessUI (int robotId, int uiID) : void

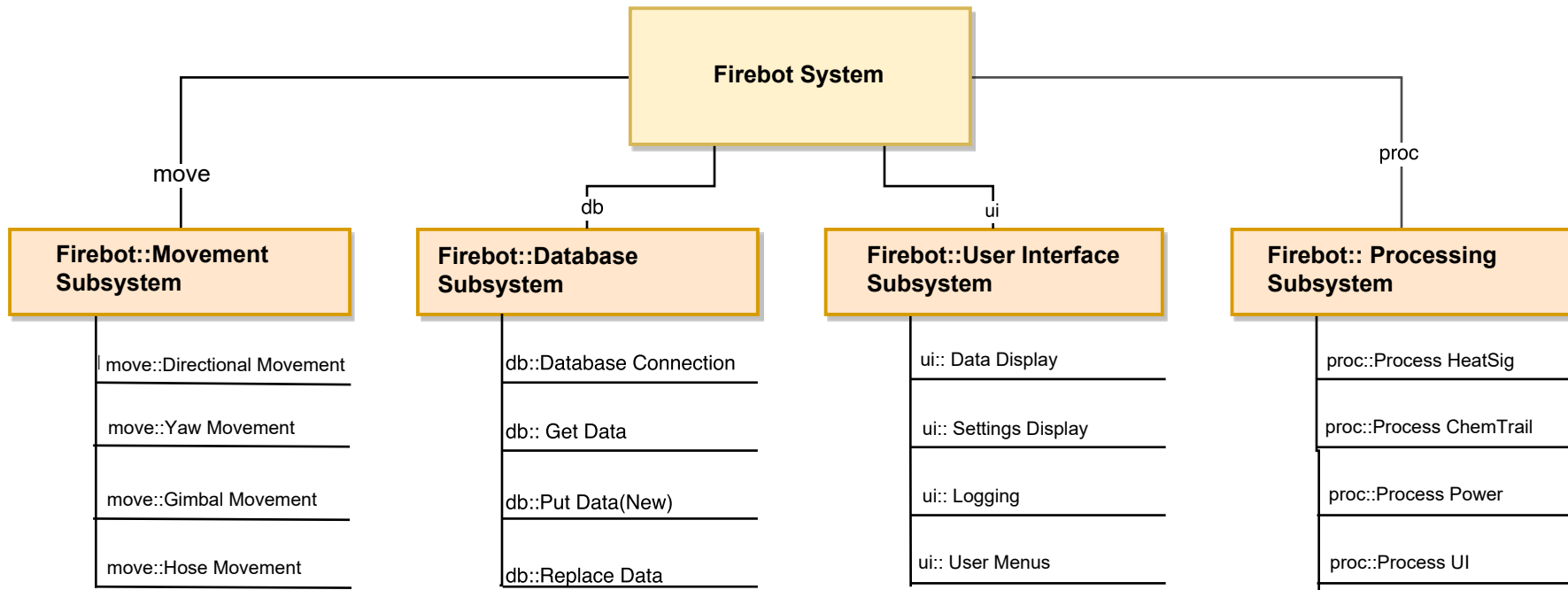
Stake Holders

- **Government**
- **Fire Departments**
- **Municipalities**
- **Rescue Teams**
- **Non-Profit Organizations**

4. USE CASE



5. IDENTIFY SUBSYSTEMS



6 – Interfaces

<u>Movement</u>		
Field Name	Data Type	Data Shape
Yaw	float	xxx.xx
Pitch	float	xxx.xx
Roll	float	xxx.xx
Speed	int	xx
Wheel_RPM	float	xxx.xx
Hose_Speed	float	xxx.xx
Robot_Turn	float	xxx.xx
<u>DB</u>		
dataID	int	xxx
dataName	varchar	xxxx xxxx
heatSigVal	int	xxx xxx xxx
userID	varchar	xxxxxx
userName	varchar	xxxxxx
userPhone	varchar	xxx-xxx-xxx
userRegion	varchar	xxxxxx
connectionID	int	xxxxxx
<u>UI</u>		
menuID	int	xxx
menuName	varchar	xxxx
batteryLvl	int	xxxx
hoseAngle	int	xx.xxxx
statusID	int	xx.xxxx
statusName	varchar	xxx
emergencyStopTrig	int	x
<u>Process</u>		
settingID	Int	xxx
settingName	varchar	xxx
radiusCalc	int	xxx
chemToHeatFactor	int	xxx

7 – Functional Requirements

o Movement:

- **Directional Movement** - Forward and back movement of the robot
- **Yaw Movement** - Rotational movement of the robot
- **Gimbal Movement** - 3-axis movement of the built in camera
- **Hose Movement** - Vertical movement of the attached waterhose

o Data Storage:

- **Db Connection** - allows the connection and setup of the database
- **Get Data** - allows information to be retrieved from the db
- **Put New Data** - allows new information to be put into the db
- **Replace Data** - allows existing information to be replaced

o User Interface:

- **Data Display** - displays essential data about the robot to the user
- **Settings Display** - displays various settings options to the user
- **Logging** - automatically logs information for review and history
- **User Menus** - determines the layouts of various menus available to the user

o Processing:

- **Process HeatSig** - Processes heat variables detected by the camera
- **Process ChemTrail** - process variables received from the fire detection system
- **ProcessPower** - make sure power is properly distributed among the robot system
- **ProcessUI** - processes inputs put by the user in various menus

8 – Non-Functional Requirements

- **Performance:** The robot will run on a 4000maH lithium Iode battery that will last about half an hour. The battery will be easily replaced. The directional speed of the robot will max out at 15 mph and a rotational speed of 15rpm. The instruction(system) to action speed(response time) is about 1/60 s.
- **Reliability:** Since the system is directly connected to the robot a reliability of 99% can be achieved. Maintenance on the caching system plugins will need to be performed once every 3 months. System may be down for 15 minutes a month which will be back in 15 minutes.
- **Capacity:** The robot will be equipped with a 20 liter tank for either water or fire fighting foam. The tank will be easily refillable through a firetruck or a hydrant. The software will need to be installed individually to each robot.
- **Security:** Since the robot is used under very specific circumstances, with a supervision of at least 1 officer, security systems will not be put into place for the software. But the user will need to be authenticated and verified before the software is initiated and ultimately robot is used.
- **Portability:** A copy of the software will need to be in each robot individually and not be tampered with or installed without the consent of a professional technician. The robot will be easily portable due to its ability to move and the petite size to fit into trucks and beds.
- **Response Time:** System can turn on in 15 seconds. Remote to Robot connection is 1/60 seconds and system can turn down in 3 seconds.

9- Select a Process Model

Pick: Agile-EP

About Agile:

Agile is mostly described as an iterative waterfall model. This is because the software is delivered in iterations as opposed to delivering it as a whole. Software is developed in Sprints that last from 1 week to 4 weeks, and then presented and evaluated with the client before proceeding to the next sprint. This way, the changes or errors in development can be done much earlier, saving a lot of money from post-delivery maintenance.



Why choose Agile for Project:

- Made mostly for small groups working in a tight environment on a single project.
- Extreme Programming allows building a software without knowing what the client wants.
- Less prone to errors and bugs after delivery important when time and effort after delivery is not possible.

10 – Project Management Plan

1 Overview.

1.1 Project Summary.

1.1.1 Purpose, Scope, and Objectives. The objective of this project is to develop a software product that will operate a robot that extinguished fires. The firebot is a fire-extinguishing robot designed to assist firefighters in times of wildly spreading, uncontrollable fires, where second and minutes will make the difference between life and death. Firebot will be a self-sufficient, portable, powerful robot able to fight even the highest of blazes.

1.1.2 Assumptions and Constraints. Constraints include the following:

The deadline must be met.

The budget constraint must be met.

The product must be reliable.

The architecture must be open so that additional functionality may be added later.

The product must be user-friendly

1.1.3 Project Deliverables. The complete product, including user manual, will be delivered 6 weeks after the project commences.

1.1.4 Schedule and Budget Summary. The duration, personnel requirements, and budget of each workflow are as follows:

Requirements workflow (1 week, two team members, \$740)

Analysis workflow (1 weeks, two team members, \$480)

Design workflow (1 weeks, two team members, \$480)

Implementation workflow (2 weeks, three team members, \$6,830)

Testing workflow (1 weeks, three team members, \$1,220)

The total development time is 6 weeks, and the total internal cost is \$9,750.

1.2 Evolution of the Project Management Plan. All changes in the project management plan must be agreed to by Ali before they are implemented. All changes should be documented to keep the project management plan correct and up to date.

2 Reference Materials. All artifacts will conform to the company's programming, documentation, and testing standards.

3 Definitions and Acronyms. Firebot - Refers to the Fire Extinguishing Robot and all of its components

4 Project Organization.

4.1 External Interfaces. All the work on this project will be performed by Ali, Samuel, and Hugh Mungus. Ali will meet weekly with the client to report progress and discuss possible changes and modifications.

4.2 Internal Structure. The development team consists of Ali (owner), Samuel, and Hugh Mungus.

4.3 Roles and Responsibilities . Sam and Hugh will perform the design workflow. Ali will implement the class definitions and report artifacts, Samuel will construct the artifacts to handle investments and operating expenses, and Hugh will develop the artifacts that handle mortgages. Each member is responsible for the quality of the artifacts he or she produces. Ali will oversee integration and the overall quality of the software product and will liaise with the client.

5 Managerial Process Plans.

5.1 Start-up Plan.

5.1.1 Estimation Plan. As previously stated, the total development time is estimated to be 10 weeks and the total internal cost to be \$9,750. These figures were obtained by expert judgment by analogy, that is, by comparison with similar projects.

5.1.2 Staffing Plan. Ali is needed for the entire 6 weeks, for the first 3 weeks in only a managerial capacity and the second 3 weeks as both manager and programmer. Samuel and Hugh are needed for the entire 6 weeks, for the first 5 weeks as systems analysts and designers, and for the second 1 weeks as programmers and testers.

5.1.3 Resource Acquisition Plan. All necessary hardware, software, and CASE tools for the project are already available. The product will be delivered to the Customer of choice installed on a Firebot Robot that will be leased from our usual supplier.

5.1.4 Project Staff Training Plan. No additional staff training is needed for this project.

5.2 Work Plan.

5.2.1–2 Work Activities and Schedule Allocation.

Week 1. (Completed) Met with client, and determined requirements artifacts.

Inspected requirements artifacts.

Weeks 2. (Completed) Produced analysis artifacts, and inspected analysis artifacts. Showed artifacts to client, who approved them. Produced software project management plan, and inspected software project management plan.

Weeks 3. Produce design artifacts, inspect design artifacts.

Weeks 4-6. Implementation and inspection of each class, unit testing and documentation, integration of each class, integration testing, product testing, and documentation inspection.

5.2.3 Resource Allocation. The three team members will work separately on their assigned artifacts. Ali's assigned role will be to monitor the daily progress of the other two, oversee implementation, be responsible for overall quality, and interact with the client. Team members will meet at the end of each day and discuss problems and progress. Formal meetings with the client will be held at the end of each week to report progress and determine if any changes need to be made. Ali will ensure that schedule and budget requirements are met. Risk management will also be Ali's responsibility.

5.2.4 Budget Allocation. The budget for each workflow is as follows:

Requirements workflow \$ 740

Analysis workflow 480

Design workflow 480

Implementation workflow 6,830

Testing workflow 1,220

Total \$9,750

5.3 Control Plan. Any major changes that affect the milestones or the budget have to be approved by Ali and documented. No outside quality assurance personnel are involved. The benefits of having someone other than the individual who carried out the development task do the testing will be accomplished by each person testing another person's work products.

At each meeting, Samuel and Hugh will present the day's progress and problems Ali will determine whether they are progressing as expected and whether they are following the specification document and the project management plan. Any major problems faced by the team members will immediately be reported to Ali.

5.4 Risk Management Plan. The risk factors and the tracking mechanisms are as follows:

There is no existing product with which the new product can be compared. Accordingly, it will not be possible to run the product in parallel with an existing one. Therefore, the product should be subjected to extensive testing.

5.5 Project Close-out Plan. Not applicable here.

6 Technical Process Plans.

6.1 Process Model. The Agile Process will be used.

6.2 Methods, Tools, and Techniques. The workflows will be performed in accordance with the Agile Process. The product will be implemented in Aduino Sketch.

6.3 Infrastructure Plan. The product will be developed using Arduino running under Linux on a personal computer.

6.4 Product Acceptance Plan. Acceptance of the product by our client will be achieved by following the steps of the Agile Process.

7 Supporting Process Plan

7.1 Configuration Management Plan. CVS will be used throughout for all artifacts.

7.2 Testing Plan. The testing workflow of the Unified Process will be performed.

7.3 Documentation Plan. Documentation will be produced as specified in the Unified Process.

7.4–5 Quality Assurance Plan and Reviews and Audits Plan. Sam and Hugh will test each other's code, and Ali will conduct integration testing. Extensive product testing will then be performed by all three.

7.6 Problem Resolution Plan. As stated in 5.3, any major problems faced by the team members will immediately be reported to Ali.

11 – SAFETY CONCERNS

Safety Concern #1: Software Compromised

Implications:

- Hacker can access database from an external network
- Hacker can add and delete data from database including entire table
- Hacker can steal user sensitive information such as full name and address.

Defense:

- Symmetric Database Encryption
- Hashing sensitive data
- Transparent Data Encryption

Safety Concern #2: Radio Signal Compromised

Implications:

- Wrong information is provided to the user
- Hacker can disrupt the communication between robot and operator

Defense:

- Create proper Firewall protocols.
- Encrypt Signals before being sent out
- Monitor signal disruptions

Safety Concern #3: Robot gets too close to fire

Implications:

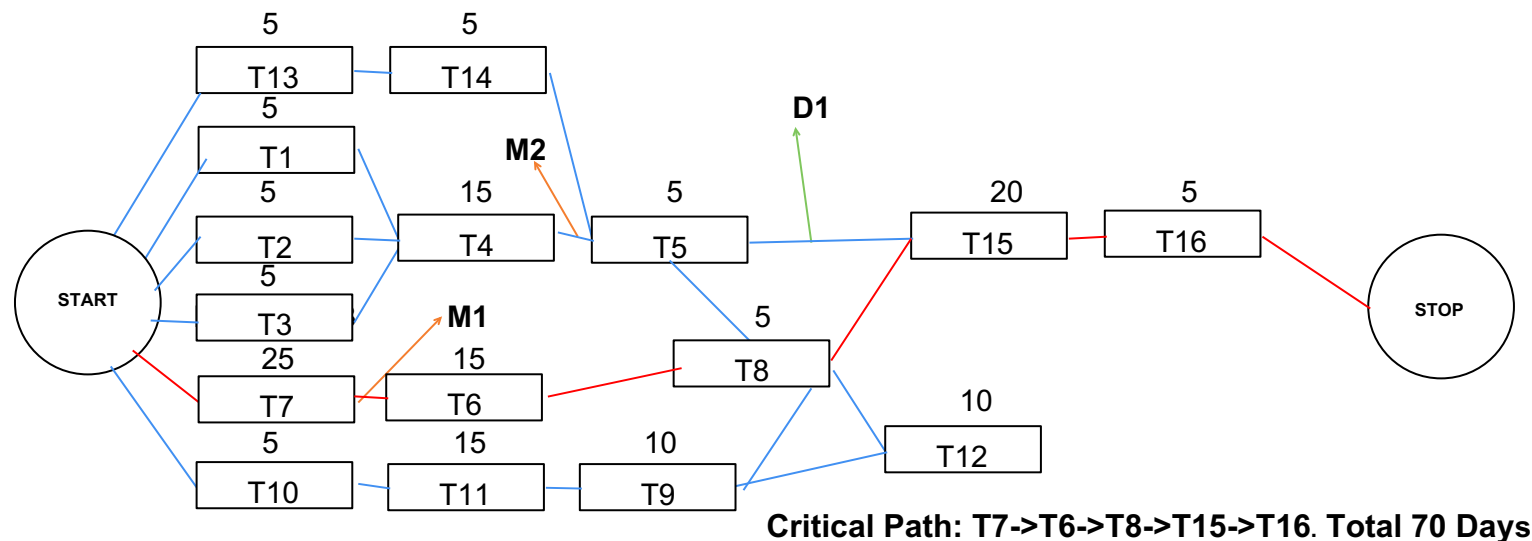
- The metal can melt releasing lead into the air
- The Pressurized tank inside of robot can burst
- The battery can explode

Defense:

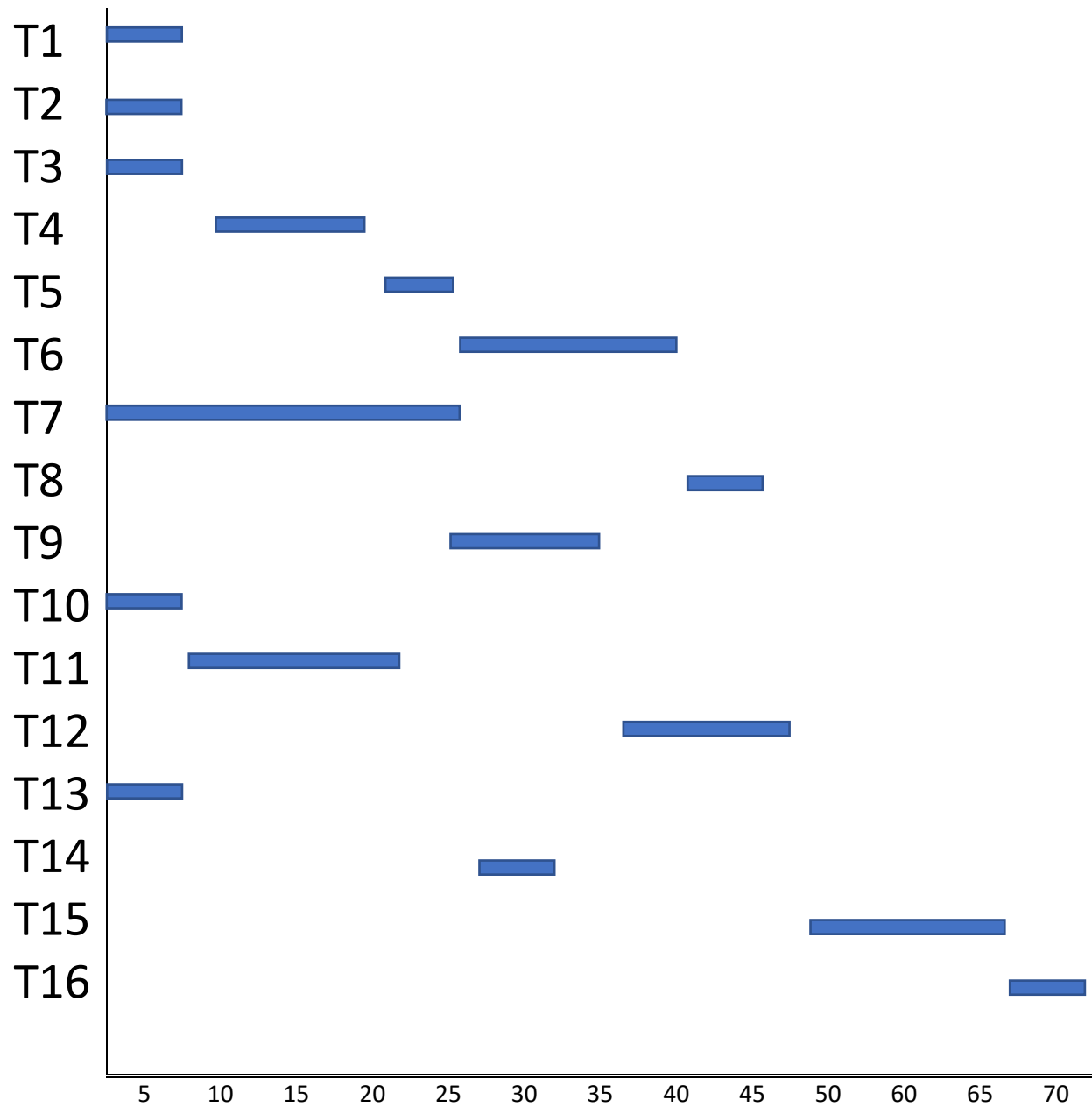
- Explosion proof casing of the battery and water tank.
- Calculating distance to heat signature and alerting if too low.

12 & 13 - ACTIVITY CHART, MILESTONES, DELIVERABLES, CRITICAL PATH

Task	Description	Duration (days)	Dependencies
T1	Directional Movement	5	
T2	Yaw Movement	5	
T3	Gimbal Movement	5	
T4	Setup Database Connection	15	T1,T2,T3
T5	Hose Movement	5	T4
T6	Set dabatase systems and variables	15	T7
T7	Code out processing algorithms for the heat signatures and chemTrail signatures	20	
T8	Process algorithm on data and setup organization of data in db	5	T5,T6,T7
T9	Code script to process user input from controller	10	T11
T10	Design the user interface outlook	5	
T11	Code the menus according to design	15	T10
T12	Code out displafy of robot information such as battery or power	10	T8,T9
T13	Setup connection with 3rd Parties	5	
T14	Code authentication of user using 3rd party softwares	5	T13, T5
T15	Create Settings Framework(Back-End)	20	T5,T8,T9,T14
T16	Code Encryption, Analytics, and maintenance code	5	T5,T9



12 – BAR CHART



15 - COCOMO

Nominal Effort: $3.2 \times (\text{KDSI})^{1.05}$

Estimated DSI = 4000

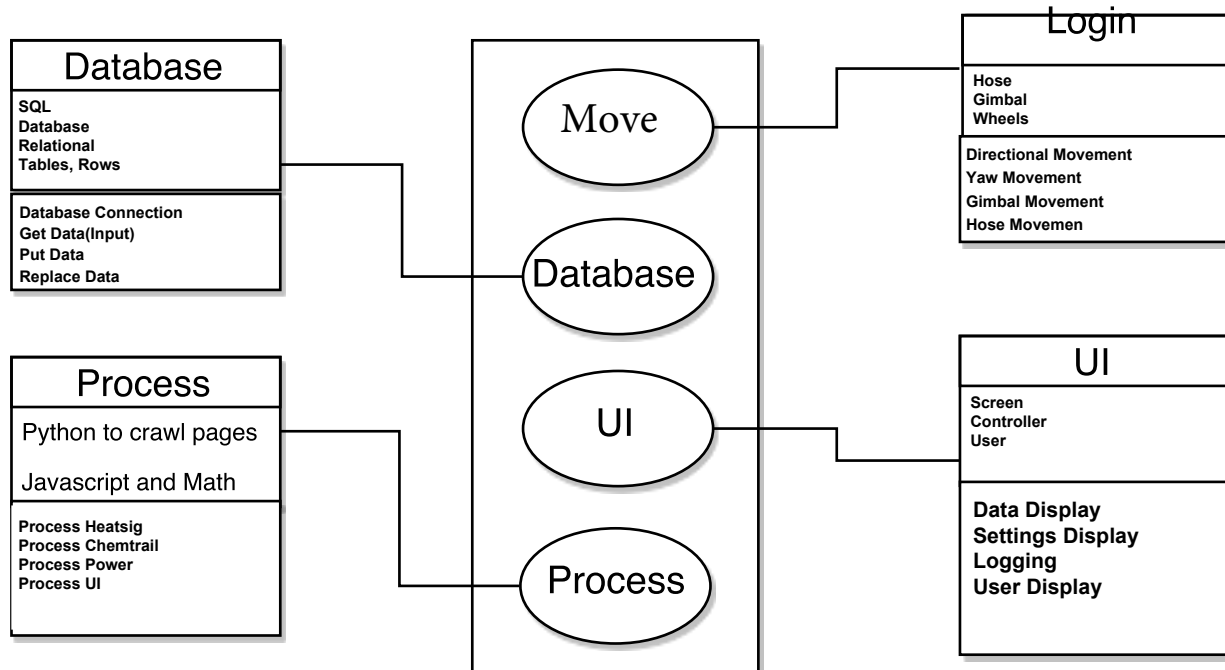
$$3.2 \times (4)^{1.05} = 13.72$$

$$(1.15)(1.08)(.85)(1)(1)(1)(.87)(1.19)(.91)(.86)(1.1)(.95)(.82)(.91)(1.1)$$

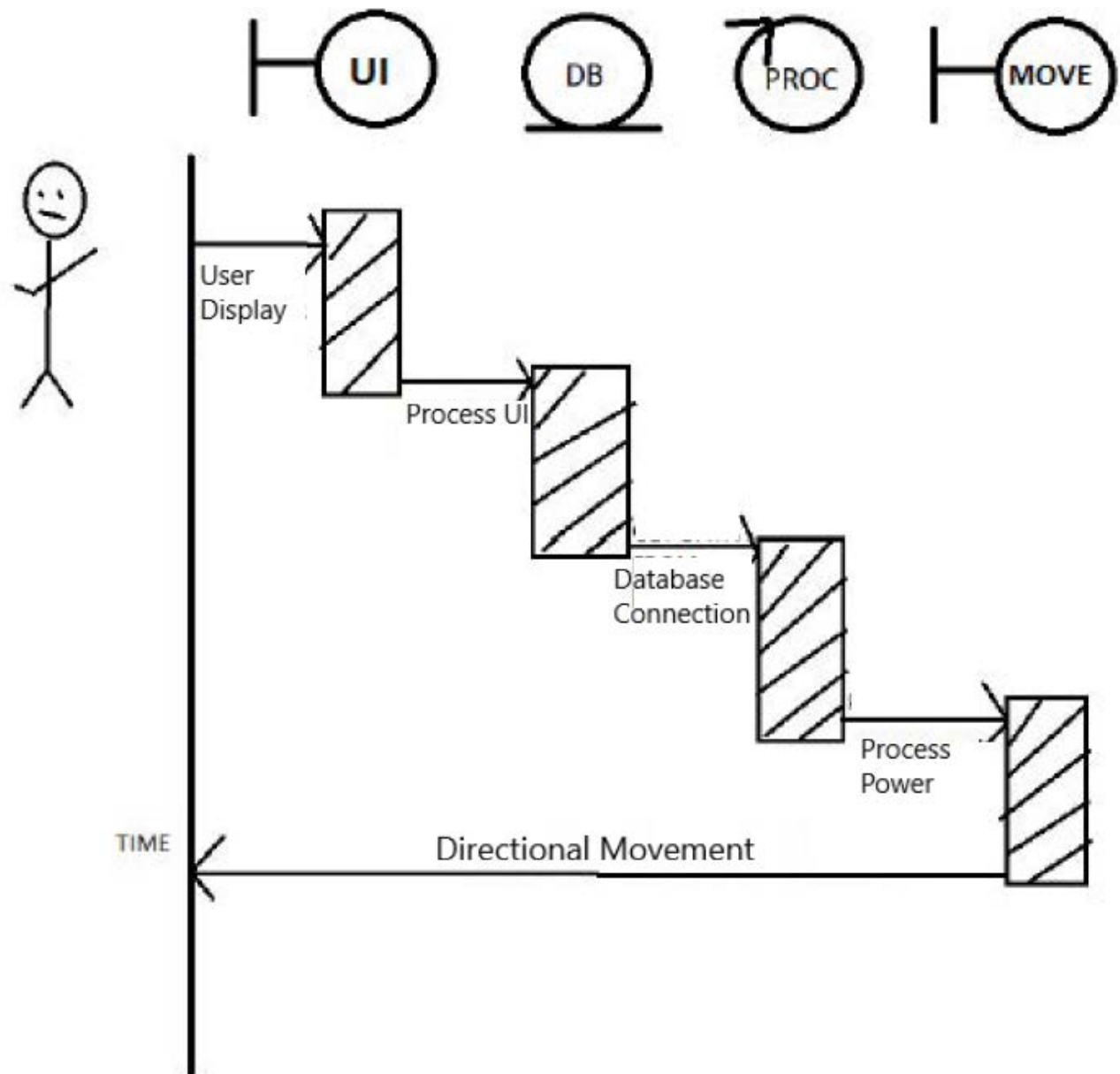
$$= .74$$

$$13.72 \times .74 \approx 11 \text{ Person-Month}$$

#17 - UML



18 - SEQUENCE DIAGRAM



19 – PROGRAM DESCRIPTION LANGUAGE

```

Class Movement{
void Process_directional_movement (int robotID, float x){
    robot.move(db.speed,direction)

}
void Process_yaw_movement (int robotID, float y){
    robot.turn(rotation)
    process_directional_movement();
}
void process_hose_movement (int robotID, float pitch)
{
    if (get_hose_pitch() is equal to pitch )
        print "error 1: already in position";
    else if(pitch less than 30 or more than 120 )
    {
        print "error 2: pitch degree impossible to achieve";
    }
    else
        set hose_pitch_deg equal to pitch;
    Move Hose To hose_pitch_deg.  }
}
void gimbal_movement (int x, int y, int z){
    cam.turn(x,y,z);
}
}

Class Database{
void Database_Connection(int robotIdNum, int DbIdNum){
    db.connect("username", pass);

}
void Get_Data(int robotIdNum, char item){
    Select item from db
    //return okay
}
void put_Data(int robotIdNum, char item){
    if (exists )
        print "error 1: already in position";  else
        {
            insert into db (item)
        }
    else
        {}
}
void replace_data (int robotIdNum, char new){
    insert into db with overwrite over old
    //return okay
}
}

Class UI{
    void data display(int robotIdNum, int UiID){
        for each item
            get_data(ui data)
            display it
    }
void settings_display(int robotIdNum, int settingID){
    get_data(ui settings)
    display it
}
void Logging(char data){
    //insert data into log
    {
        print "error 2: pitch degree impossible to achieve";
    }
    else
        print "error"  }
void User Display(int robotIdNum, int displayID)
{
    for each item
        get_data(ui data)
        display it
}
}

Class Process
    void process_Heat(int robotId, int sigStrength){
        If sigStrength > 100
            //alert the robot to move
    }
void Process ChemTint (robotIdNum, int chemStrength){
    If chemStrength > 100
        //alert the robot to move
    }
void processPower(char item){
    if (exists )
        print "error 1: already in position";
    else
        {
            insert into db (item)
        }
    else
        {}
}
void rprocessUi (int robotId, char new){
    db.connect
    ui.process
}
}

```

20 – TEST PLAN

- **System:**
 - Click Login System
 - Print OK
 - Click Movement system
 - Print OK
 - Click User Interface Systems
 - Print OK
 - Click Processing/Calculations.
 - Print OK
- **Subsystem:**
 - **Movement:**
 - Call Directional Movement
 - Call Yaw Movement
 - Call Gimbal Movement
 - Call Hose Movement
 - **Data Storage:**
 - Call Db Connection
 - Call Get Data
 - Call Put New Data
 - Call Replace Data
 - **User Interface:**
 - Call Data Display
 - Call Settings Display
 - Call Logging
 - Call User Menus
 - **Processing:**
 - Call Process HeatSig
 - Call Process ChemTrail
 - Call ProcessPower
 - Call ProcessUI
- **Unit (Login System):**
 - U: Enter "John Smith"
 - S: Error "john smith" is not recognized
 - U: Enter "JohnSmith"
 - S: Ok. Enter Password
 - U: "*****"
 - S: Error "Must be longer than 8 digits"
 - U: "*****"
 - S: OK.

21 - Hardware, FF,MUX,BUS,ALU,CU,CPU,Control Memory,state diag,Q,N,S

FF: Form feed is a button or command on the printer that allows the advancement of a printer page.

MUX: The multiplexer is a combinational logic circuit designed to switch one of several input lines to a single common output line by the application of a control logic.

BUS: A Bus is a single path for multiple sources to be routed and processed. Multiple Devices can be connected to the bus simultaneously

ALU: The arithmetic logic unit (ALU) is responsible for all logical and mathematical operations in the system.

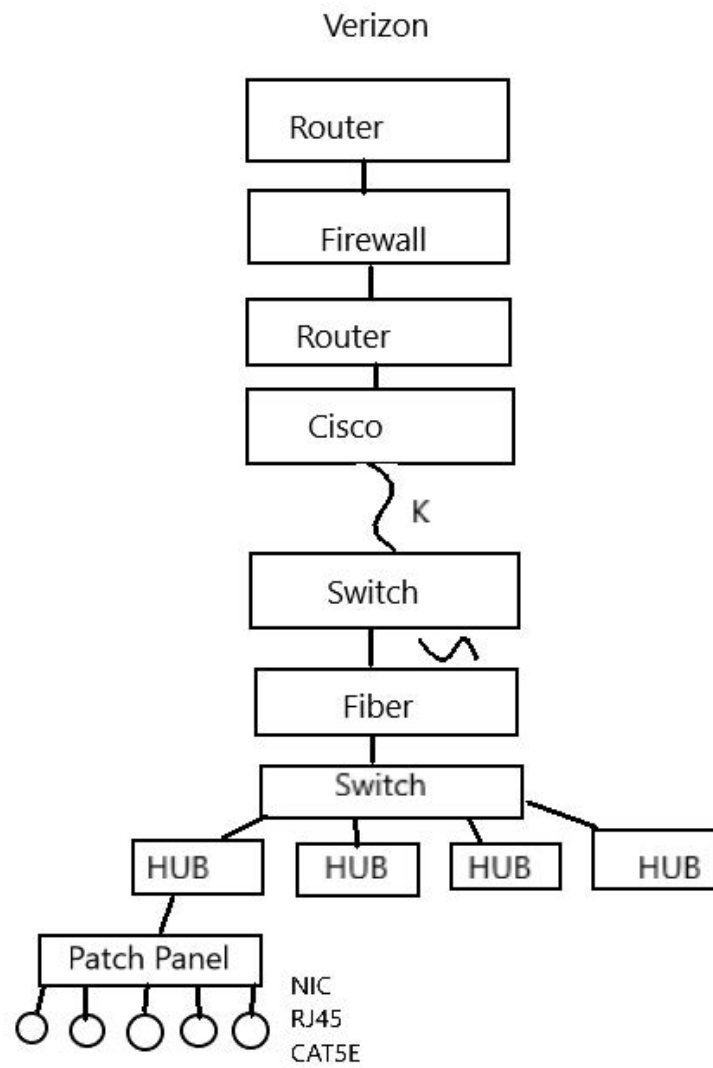
CU: The Control unit controls the operations of all parts of the computer but does not carry out any actual data processing operations.

CPU: A CPU is an electronics circuit used in a computer that fetches the input instructions or commands from the memory unit, performs arithmetic and logic operations and stores this processed data back to memory.

CONTROL MEMORY: Control memory is a memory that is part of control unit

STATE DIAGRAM: Its a diagram showing different nodes known as states and the transition between different states including how they are triggered and what needs to be translated from one node into the next.

22 - Telecomm. 2000pc-1,Hub,switch,corning,router,gateway,fireWall



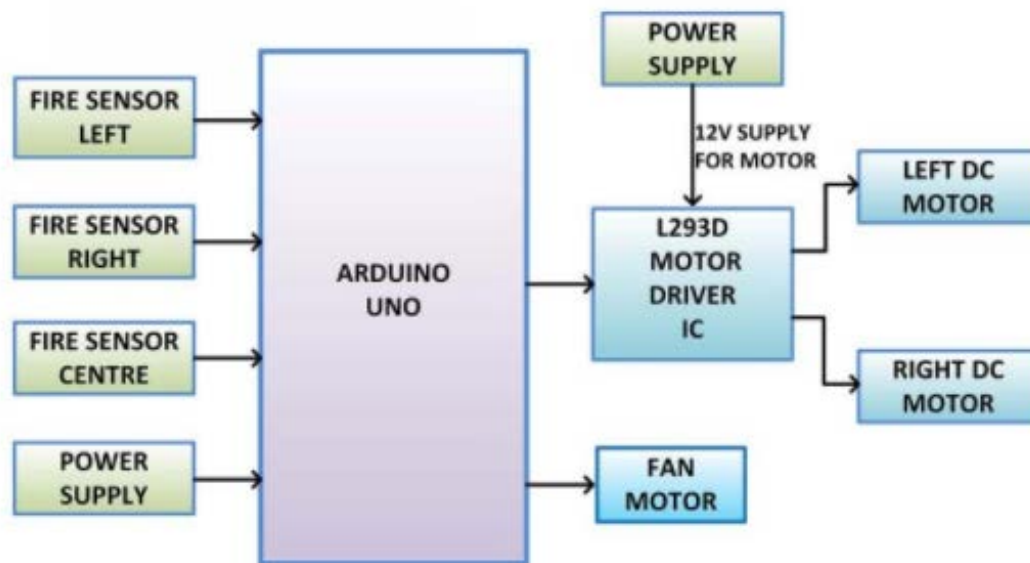
#23 - Robotic Hardware, Software, network, operating system, Algorithm..

Power Supply - 5V regulated DC for Circuit. 12V DC for motor. A 12V NIMH battery is used as the primary source of power.

Arduino UNO - Computer Board. Built-in arduino boot loader. Atmega 328 based controller board w/ 14 GPIO pins, 6 PWM pins, 6 Analog inputs and on board UART, SPI and TWI interfaces.

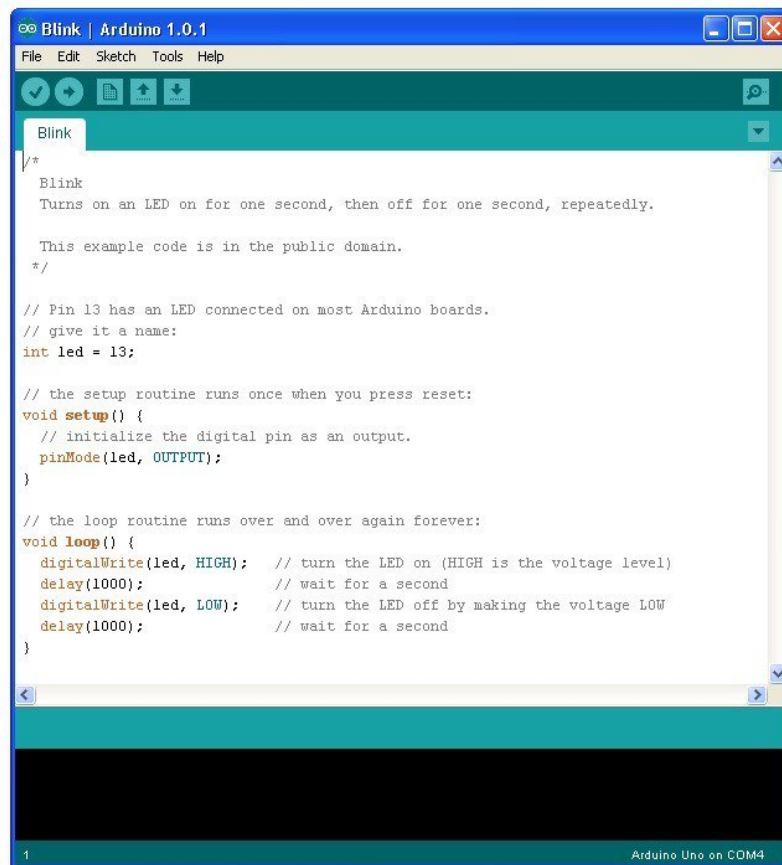
L293D DC Motor Driver IC - Dual H-bridge motor driver integrated circuit (IC). 16 pins.

Fire Sensors - A Fire Detection Sensor. wavelengths 760 nm to 1100 nm. 60 degrees detection point, distance about 1 M to 2 M. IR receivers are used as fire detection sensors in the circuit.



23 – Software and State Diagram

Arduino Sketch Code written and then compiled on Arduino IDE on Windows 10.



```
Blink | Arduino 1.0.1
File Edit Sketch Tools Help

Blink
/*
  Blink
  Turns on an LED on for one second, then off for one second, repeatedly.

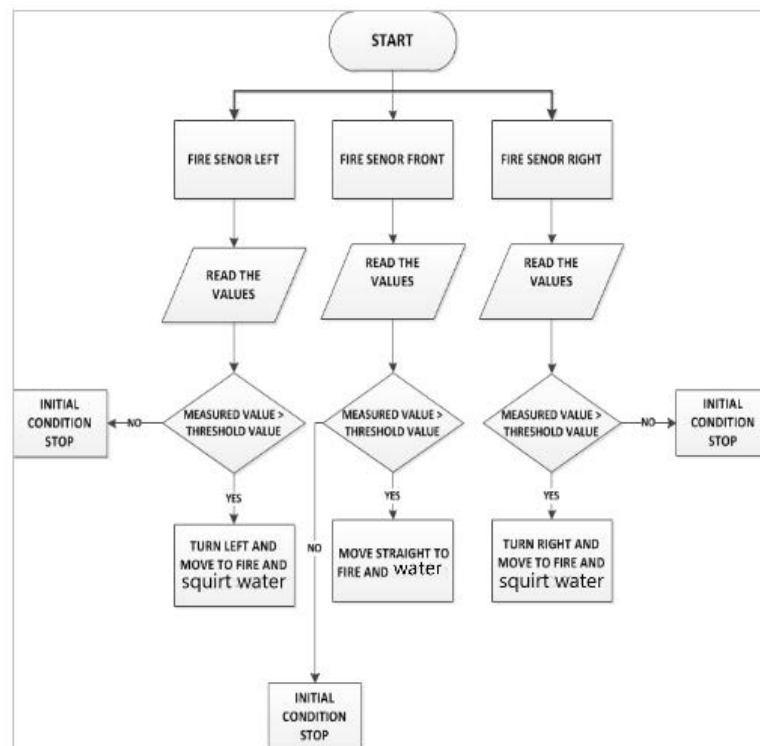
  This example code is in the public domain.
  */

// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;

// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  digitalWrite(led, HIGH); // turn the LED on (HIGH is the voltage level)
  delay(1000);             // wait for a second
  digitalWrite(led, LOW);  // turn the LED off by making the voltage LOW
  delay(1000);             // wait for a second
}
```

STATE DIAGRAM



24 – MAINTENANCE PLAN

Contract Maintenance

- Hire Contractors to do Maintenance
- More expensive
- No headache because you don't have to do maintenance yourself
- You have no direct control of the maintenance.

In-House Maintenance

- Train a maintenance crew yourself
- Less expensive
- More control on how the maintenance goes
- Can take longer or shorter depending on training

Corrective: Corrective maintenance is when changes are made to repair defects in the design, coding, or implementation of the system

Adaptive: Adaptive maintenance is making changes to an information system to evolve its functionality to changing business needs or to migrate to a different operating environment.

Perfective: Perfective maintenance is making enhancements to improve processing performance, interface usability, or to add system features.

25 – CODE USED/CHANGED

AliKalkandelen.github.io/CS351

```
<html lang="en-US"><head>
<meta charset="utf-8">
<meta http-equiv="X-UA-Compatible" content="chrome=1">

<title>CS350PROJEKT</title>
<meta property="og:title" content="CS350">
<meta property="og:locale" content="en_US">
<meta name="description" content="CS 350 PROJECT FALL 2017">
<meta property="og:description" content="CS 350 PROJECT FALL 2017">
<meta property="og:url" content="https://alikalkandelen.github.io/CS350/">
<link href="https://fonts.googleapis.com/css?family=Arvo:400,700,400italic" rel="stylesheet"
type="text/css">
<link rel="stylesheet" href="https://pages-
themes.github.io/dinky/assets/css/style.css?v=46349262f43f70d5dc78cf83775850b0466aa573"
>
<script src="https://pages-themes.github.io/dinky/assets/js/scale.fix.js"></script>
<meta name="viewport" content="width=device-width, initial-scale=1, user-scalable=no"> <!--
[if lt IE 9]>
<script src="//html5shiv.googlecode.com/svn/trunk/html5.js"></script>
<![endif]-->
<style>
header{
padding: 34px 80px 22px 50px;
}
</style>
</head>
<body>

<div class="wrapper">
<header>
<h1 class="header">BRAINSTREAM</h1>
<p class="header">CS350 PROJECT FALL 2017</p>
<ul>

<li class="download"><a class="buttons" href="combinepdf.pdf">Download PDF</a></li> <li
class="download"><a class="buttons" href="projectfiles.zip">Download ZIP</a></li>
```


<p>Ali Kalkandelen
 Samuel Owusu-Biney
 Stephen Kilnisan</p>

</header>

<div id="results" class="hidden"></div>

<section id="example1">

HELLO WORLD

</section>

<script src="pdfobject.min.js"></script>

<script>

var options = {

pdfOpenParams: {

pagemode: "thumbs",

navpanes: 1,

toolbar: 1,

statusbar: 0,

view: "FitV"

}

};

var myPDF = PDFObject.embed("project.pdf", "#example1", options);

var el = document.querySelector("#results");

el.setAttribute("class", (myPDF) ? "success" : "fail");

el.innerHTML = (myPDF) ? "" : "The PDF embed Didn't Work. Please try another browser or deleting your cache and trying again.";

</script>

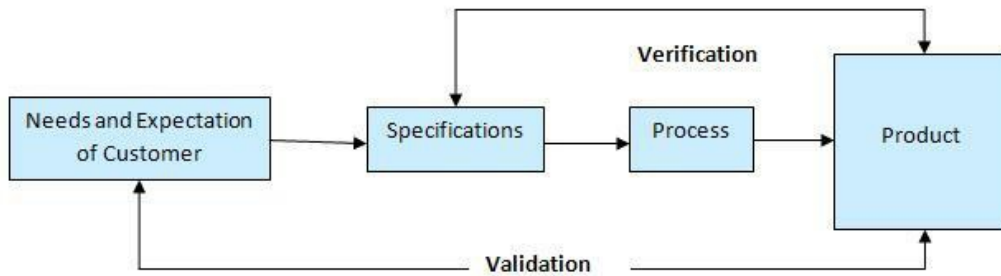
</div>

<!--[if !IE]><script>fixScale(document);</script><![endif]-->

</body>

</html>

26 - VERIFICATION AND VALIDATION



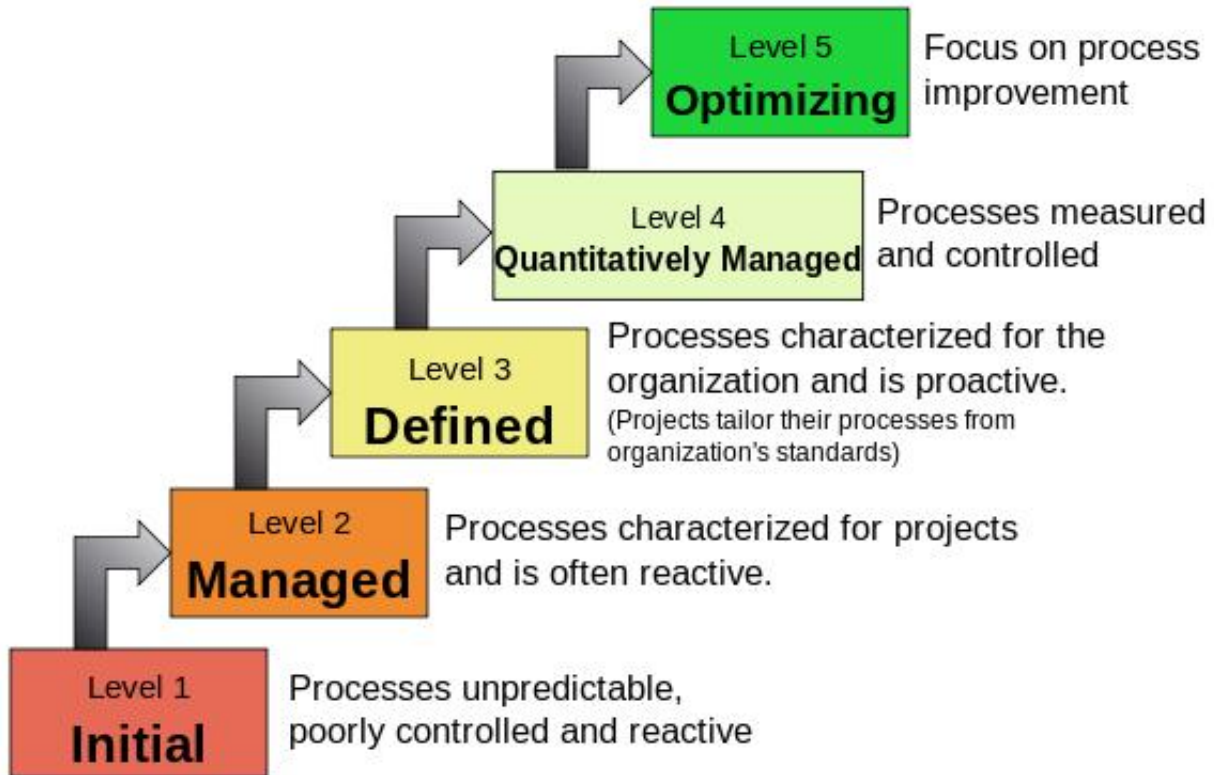
	Validation	Verification
Questions:	<ul style="list-style-type: none"> Does the final product meet the business needs of the customer? Does the product meet all the testing requirements? 	<ul style="list-style-type: none"> Does the product according to the specifications? Does the implementation meet the design? Does the product follow the proper cycle?
To do:	<ul style="list-style-type: none"> Execute the software Test executed software (alpha,beta,FAT) Validate product with business 	<ul style="list-style-type: none"> Review specifications and implementation, making sure nothing is missed. Review the Requirements and match it with the product

Verification: Meets the Specifications

Validation: Meets The Needs of the Customer at the time of Delivery

27 - CAPABILITY MATURITY MODEL

Characteristics of the Maturity levels



CMM LEVEL PICKED: **LEVEL 2 - MANAGED**

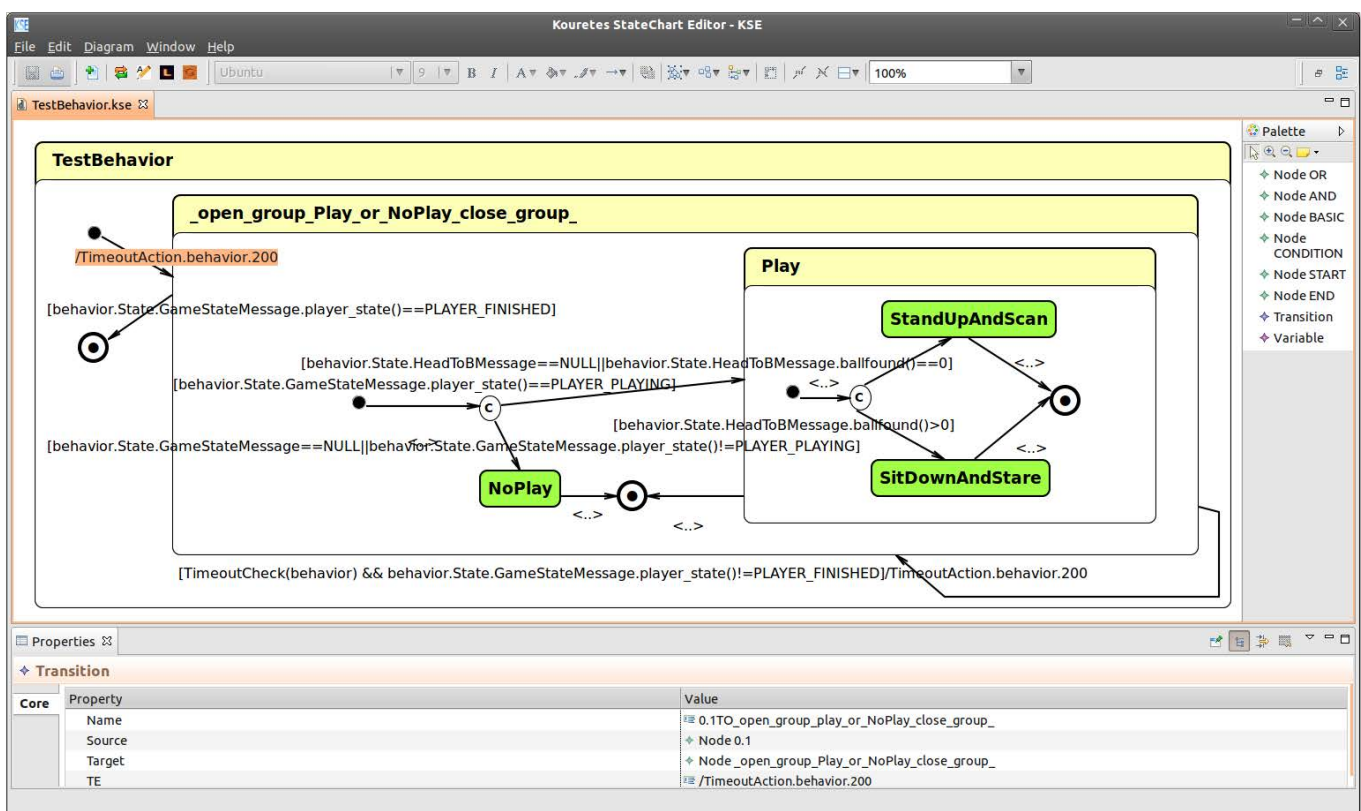
Reasons:

- The characterization of projects are well defined
- Processes for the project are clear and explanatory.
- Software is from a brand new group, so no organizational standards are set.
- Processes are defined but not well controlled.
- Processes have no definition for improvement or proactive for the future.

COST PER PERSON : \$20,000 Per Month Per Person. Based on 11

person month * 20K = **\$220,000 For Project.**

The development of high-level behavior for autonomous robots is a time-consuming task even for experts. **Kouretes Statechart Editor (KSE)** is a **Computer-Aided Software Engineering (CASE)** tool, which enables the developer to easily **specify a desired robot behavior as a statechart model** utilizing a variety of base robot functionalities (vision, localization, locomotion, motion skills, communication). A statechart is a compact platform-independent formal model used widely in software engineering for designing software systems. KSE adopts the Agent Systems Engineering Methodology (ASEME) model-driven development approach. ASEME is an Agent-Oriented Software Engineering (AOSE) methodology applied, through this work, to robotic teams behavior development. KSE guides the developer through a series of analysis and design steps within a graphical environment that leads to automatic source code generation.



29 - Decommissioning Plan

- Determine Retirement Strategy
 - Is the program in full retirement or migrating to a newer system?
 - Will the retired program be backed up or purged forever?
 - How many people are involved in the current system?
 - Size of Database? Size of Code? Size of Traffic?
- Database:
 - Database Refactoring to newer software, or complete, safe deletion if full retirement.
 - Backups are deleted or migrated depending on the decision
- Documentation:
 - Docs are updated to include decommissioning portions and plans
- Users:
 - Users are informed of migration or removal.
 - Users are directed to a new system in a seamless manner
 - Users receive proper training into understanding new system
- Implementation:
 - Remove all code from server.
 - Delete proper files and folder structures.
 - Backup to another PC if needed
- Hardware:
 - Determine if or how the hardware will be used.
 - If reused replace all batteries.
 - Clean and replace motors if shown signs of metal fatigue or damage
 - If not reused dispose recycle or trash accordingly.

31 - Java/Python/Corba/JVM

Java:

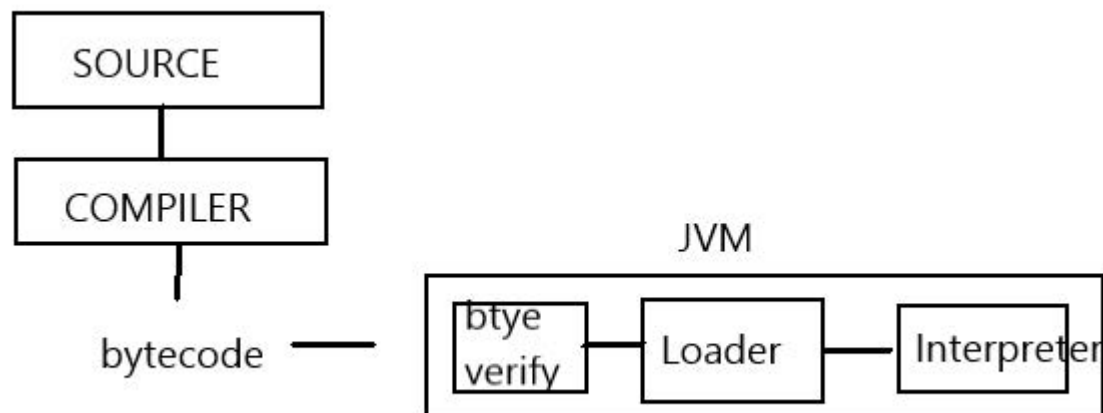
1. Built in multi-threading
2. Built in garbage collection
3. Built in error management
4. Simplified code in ORB/CORBA
5. Ideal lang for writing client & Server
6. Can built object infrastructure
7. JDB Connectivity JDBC: Set of classes that provide interface to SQL

Python:

1. Third Party modules
2. Extensive support libraries
3. Open source and community development
4. Learning ease and support
5. User friendly data structures
6. Productivity and speed

CORBA: The standard for Object request broker

JVM



32 – Postdelivery maintenance types,corrective,perfective,adaptive

Corrective: Corrective maintenance is when changes made to repair defects in the design, coding, or implementation of the system

Adaptive: Adaptive maintenance is making changes to an information system to evolve its functionality to changing business needs or to migrate to a different operating environment.

Perfective: Perfective maintenance is making enhancements to improve processing performance, interface usability, or to add system features.

33- EMERGING TECHNOLOGIES

Robot can able to communication with each other, also can commotion with people in case the is the fire in the house, car, office and school.

Robot can remember us to find our ways by fire.

Robot can open our doors for us by the fire.

Robot desire for a more comfortable, responsive prosthetic led to developing the BiOM ankle, a dense collection of sensors and circuitry controlling an artificial calf muscle that propels users forward with a natural, easy gait.

34 - DATED LOG

WHO	WHAT	WHEN
Operations Manager	Set-Up Orgnization structure and personnel	01/01/2018
Database Analyst	Made priliminary database structures	01/20/2018
Programmer	Tech stack setup and analysis	01/22/2018
Senior Analyst	Major change in life cycle method	02/10/2017
DIRECTOR Programmer	Changed tech stack to accomodate project	02/20/2017
Project Manager	Added developers to project to speed up prod	03/31/2017
Vendor	Changed Specifications (2.3)	11/01/2017
Secretary	Made updates to office space	11/02/2017