Software Engineering and Project

Software Requirements Specification

UG-14

a1647816 Anthony Scaffidi
a1693014 Jingbin Zhang
a1710974 Jinshan Wang
a1727491 Lincoln Phillips
a1687940 Mitchell Phillips
a1696673 Nanye Zou
a1669907 Shaun Kraemer

0.5

21/10/2018

Table of Contents

Version History

Date	Version	Changer	Description
19/8/2018	0.1	Anthony Scaffidi	Completed first draft of the SRS document
14/10/2018	0.2	Jingbin Zhang Nanye Zou Jinshan Wang	Modify document layout
16/10/2018	0.3	Anthony Scaffidi	Updated structure of requirements
16/10/2018	0.4	Jingbin Zhang Jinshan Wang	Completed overall description
16/10/2018	0.4	Nanye Zou	Modify Product features and User classes
16/10/2018	0.4	Jingbin Zhang	Updated overall description and fixed
21/10/2018	0.5	Anthony Scaffidi	Updated requirement descriptions and classifications
21/10/2018	0.5	Lincoln Phillips	Proofreading and spell checking

1. Introduction

1.1 Scope

The purpose of the project is to develop suitable java source code to control an unmanned autonomous rescue robot using the Lego NXT EV3 kit in conjunction with leJOS. The prototype itself will be equipped with a number of onboard sensors to facilitate effective obstacle avoidance and navigation around the defined situation map ensuring efficient finding of survivors.

The rescue robot is required to effectively navigate a map the size of an A0 piece of paper and shall be capable of detecting map boundaries, avoid obstacles and differentiate between obstacles and survivors to ensure the latter are all located within a 20 minute time frame. The prototype will not utilise any proprietary GPS painting technology to achieve its task but will instead use sensor input to provide mapping output and report survivor locations. The produced system will be capable of outputting to the user the results of the pathfinding operation as well as the status of survivors and locations of obstacles with respect to the discovered map.

1.2 Intended audience

This software requirement specification documentation is intended for viewing by the members of the 2018 Software Engineering & Project group UG-14 as well as the client - The University Natural Disaster Search and Rescue Service (UNDSRS), and Software Engineering Project teaching staff.

1.4 Glossary

Definitions of any acronyms or terms used in this document are as follows:

leJOS	Java-based firmware for use on Lego NXT development kits
GPS	Global Positioning System
UG-14	Undergraduate Group 14
UNDSRS	The University Natural Disaster Search and Rescue Service
JVM	Java Virtual Machine
API	Application Programming Interface

2. Overall description

2.1 Product perspective

The following SRS documentation will detail the requirements and use cases for the developed autonomous EV3 rescue robot and associated software developed within the leJOS java interface for Lego EV3 products. The ultimate goal of this project is the development of an unmanned prototype for search and rescue operations on a predefined map without the use of GPS painting technology using strictly Lego EV3 sensors and components.

2.2 Product features

The main capabilities and client constraints of the product for development include:

- Autonomous pathfinding to develop a map of the disaster area in real-time.
- Dynamic survivor/obstacle locations dictated by the client.
- Ability to accurately perform pathfinding operations via precise localisation.
- Robot must halt operation upon discovery of entire map.
- Detect and differentiate between obstacles and survivors.

2.3 User classes

A number of user classes are important in the context of the project. Primarily among these are the clients/customers and the users and/or administrators. The overarching functional requirements of the produced system are derived and detailed based on the specification provided by the client. Upon delivery, the developed system must facilitate ease of use by the customer or users who may not necessarily have the experience or expertise of the overall administrators who manage or control the systems and perform operations such as changing specific software features of applying software fixes.

Ultimately a clear set of functional as well as user requirements must be detailed enabling users to interact with the developed system to achieve the necessary functions detailed within the specification.

2.4 Operating environment

The EV3 robots will be developed using java using leJOS to provide access to the various inbuilt packages and interfaces necessary to control EV3 componentry. LeJos provides a Java virtual machine (JVM) and a comprehensive Application Programming Library(API) allowing users to run Java programs on EV3 robots. The API library can access the functions of EV3 robots necessary to control key functionality of attached components such as motors, sensors and other electromechanical devices. The leJOS libraries are accessible through all major operating systems such as Windows, iOS and linux allowing for ease of development between administrators and users on various systems.

2.5 Design and implementation constraints

It must be acknowledged that there are a number of constraints on the project especially with respect to time-sensitivity and component availability. The latter is particularly limiting as the limited components within the Lego EV3 kit also feature a number of hardware limitations including restricted memory, low-resolution, lack of effective accuracy and calibration and assembly quality.

3. Client requirements

3.1 Autonomy

CA0001: Autonomous operation

Description The developed rescue bot shall be capable of performing its search and rescue role entirely autonomously. The system must be capable of instantly beginning its search and locating all the survivors within the designated time frame without user input beyond the initial powering on of the robot. Upon locating all survivors, the robot shall return to the boundary of the devised map and halt movement.

Rationale Autonomous operation was specified as a key client requirement and is essential to the expected performance of the rescue robot as defined by UNDSRS.

Acceptance criteria The requirement can be verified when the user can confirm that the robot can perform repeatable successful passes over the map and locate all the survivors in the necessary time frame each test.

Source Sourced from client specifications and first client meeting.

Priority High.

3.2 Navigation

CN0001: Navigation

Description One of the fundamental requirements for the search and rescue functionality of the robot is navigation about the designated map. The system shall be capable of fully autonomous forward motion as well as left/right turning capability of both 90 degree turns as well as stationary pivoting. The robot shall be capable of navigating an A0 map while locating and avoiding any obstacles within its path.

Rationale The requirement of motion and accurate navigation and thus avoidance of obstacles is essential for ensuring the robot can explore the entire A0-sized map and locate any survivors in an effective manner. Stationary turning capability is ideal for obstacle avoidance within tight confines; while 90 degree turning radius limitations ensure that the robot motion is simplified and more likely to conform to the idealised grid-based search path.

Acceptance criteria Verification will be performed by the establishing of basic tests by the administrator in which the robot will be capable of turning a designated direction and angle upon encountering an obstacle whilst simultaneously not making contact with said object. If the user can perform a single pass over the entire map without the robot missing locations or leaving the bounds of the map then the requirement has been met.

Source Sourced from first client meeting

Priority High.

CN0002: Boundary detection

Description The system shall be capable of recognising the boundary of the map designated by coloured tape of varying colour. This shall be achieved via use of light-sensitive colour sensor and used to denote the limits in which the robot can navigate between and the edge in which to park itself once the search and rescue operations have been completed.

Rationale This requirement is essential as the robot is expected to navigate a predefined map of A0 paper size and thus a clear boundary must designate the searchable area in which the robot can perform its search operations.

Acceptance criteria User can place robot within a user-defined boundary (of varying shape) constructed from coloured tape, and observe whether the robot remains within the confines of the boundary and alters its trajectory upon encountering a boundary line. If pathfinding behaviour is as user expects then the robot correctly detects boundaries.

Source Sourced from first client meeting.

Priority High.

CN0003: Boundary detection (light sensitivity)

Description In addition to R0007, the rescue robot shall also be capable of detecting the boundaries even in the case of low or varying lighting conditions. A secondary sensor system shall be used to confirm the object recorded by the colour sensor is a legitimate boundary and also to act as a contingency in the event the colour sensor cannot recognise the presence of a boundary.

Rationale Sensitivity of the sensors within the Lego NXT EV3 kit are quite limited in sensitivity and their input - particularly light-sensitive variants such as the colour sensor - can be compromised by varying ambient conditions such as differing lighting conditions in testing locations.

Acceptance criteria The user will be capable of defining predefined maps utilising varied light conditions i.e low light, or highly illuminated terrain and observing whether the robot's movement path remains within the boundaries despite these conditions. If performance is unaffected despite the user-defined conditions then the requirement has been verified.

Source Sourced from second client meeting.

Priority Medium.

CN0004: Virtual mapping (storage)

Description The system shall be capable of storing searched locations of the discovered map within memory to ensure that the rescue robot does not repeat passes on previously searched regions within the map. This will be most easily facilitated by the development of a grid-based search system in which the robot can cross reference already explored regions against.

Rationale This requirement will improve overall system efficiency by removing repeated passes and thus ensuring the robot is more likely to locate the survivors within the designated time limit. A grid-based search will also ensure the robot has a clear series of checkpoints it must reach to ensure it has searched the terrain in its entirety compared to randomised and intersecting search paths.

Acceptance criteria The user can analyse the produced graphical output of the robots path and ensure repeated passes over the same region are not completed and that the explored area covers the entirety of the map produced.

Source Sourced from second client meeting.

Priority Medium.

3.3 Detection and obstacle avoidance

CD0001: Obstacle avoidance

Description The system shall be capable of avoiding obstacles within its search path and automatically adjusting its trajectory to search a new location after reporting the obstacle's location. Avoidance shall be performed largely via use of the robot's stationary turn capabilities to ensure that deviation from the ideal search path is not as substantial as would be the case with the added turning radius necessary for a 90 degree turn.

Rationale It is a system requirement that all obstacles are avoided and the robot does not make contact with anything over the course of its search and rescue operations.

Acceptance criteria The user can observe whether the robot can accurately locate obstacles and whether it can maneuver itself out of the path of the obstacle without physically touching it.

Source Sourced from first client meeting.
Priority High.

CD0002: Obstacle vs. survivor differentiation

Description The system shall be capable of differentiating between encountered objects and classifying them as either obstacle or survivor ensuring that it knows when to avoid the encountered obstacle and altering its search path or report the location of the survivor discovered.

Rationale The client specification details the importance of avoiding any obstacles as a key navigational requirement while also emphasising that all survivors must be located.

Acceptance criteria The user can define the number of survivors and obstacles present on the map. If the reported graphical output denotes the correct classification of each kind of object then the system is functioning as required. If the user can run repeated runs of varying scenarios and receive the correct output then system robustness can be confirmed.

Source Sourced from first client meeting.

Priority High.

4. User requirements

4.1 Connectivity

UC0001: Remote connectivity

Description Robot shall be connected to a UI which will allow for sending/listening of data allowing for start operation commands wirelessly as well as retrieval of data for map output to provide necessary robot localisation and task updates.

Rationale Wireless control of connection allows for the user to initiate an autonomous search and rescue operation without further user input. The existence of an ongoing connection between the user's laptop and the robot ensures that information can be displayed to the user as well as allowing for failsafes which shutdown the robot's current operation in the event of a disconnection or inability for the user to communicate with the robot.

Acceptance criteria User should be capable of connecting/disconnecting and thus starting or ending operation of the robots current task via use of UI buttons or command line commands. If the robot can be initialised or disconnected from user input then connectivity between the user and robot has been achieved as required.

Source Requirement sourced from initial group design meeting detailing key robot functional requirements.

Priority High.

UC0002: Manual failsafe

Description Robot shall facilitate manual override functionality to allow the user to manually and instantaneously switch off robot in the event of unexpected robot behaviour or for testing requirements.

Rationale Safety is integral during development and operation thus to protect both the user and hardware the user should be capable of enforcing manual override on the robot whether that be extensive or simply a failsafe on/off capability. This functionality will protect in the case of any potential unexpected robot behaviour.

Acceptance criteria User should be capable

Source First client meeting

Priority Low/Medium.

4.2 Output

UO0001: Graphical user interface (GUI)

Description The produced software shall feature a GUI capable of outputting information important to the user such as the map and robot localisation as well as facilitate key user functionality such as ability to connect/disconnect the robot, update the currently displayed map, or begin a new search and rescue operation.

Rationale Graphical output is integral to verifying robot performance against expected functionality as well as providing the user with the information necessary to correctly interpret the current status of the search and rescue operation despite its autonomous nature.

Acceptance criteria User should be able to connect/disconnect, begin search and rescue operations and update displayed maps by interacting with the produced UI on the laptop. If the robot responds to the button-press events then the system is performing as expected.

Source Requirement sourced from initial group design meeting detailing key robot functional requirements.

Priority High.

UO0002: User output (terrain/mapping)

Description While the system must be entirely autonomous, the system shall be capable of returning graphical output to the user to display its travelled path, discovered obstacles, and also designate survivor locations once located in real time. The map should update automatically as the robot discovers new locations.

Rationale Graphical output will be used by users to confirm successfulness of the rescue bots operation by allowing comparison between real world map and the locations reported by the rescue bot. Likewise, confirmation regarding whether encountered objects were differentiated correctly regarding their status as a survivor or obstacle

Acceptance criteria Verification can be performed by creating basic test cases i.e survivor and obstacle located set distance and angle from the boundary edge and confirming whether the trajectory recorded and locations for obstacle and survivor described via the graphical output match the designed map in reality.

Source Sourced from first client meeting.

Priority High

5. Use Cases

5.1 Case 001 - General operation

Brief Description Rescue robot is located within the map boundary and autonomous search and rescue operation is initiated by the user by first beginning the TCP/IP server connection between the robot and the PC currently running the GUI and then pressing the relevant search and rescue begin button.

Actors The actor within this use case is the user.

Preconditions The use case assumes that the robot has been located randomly on a dynamically constructed map.

Basic Flow

1. User powers on robot and PC running the GUI.
2. User establishes bluetooth pairing between devices.
3. User presses ‘connect’ button on the produced GUI.
4. Robot reports connection success and user presses button to begin search and rescue operation.
5. Robot begins autonomous navigation and pathfinding routine.
6. Map featured on GUI updates dynamically to show user newly discovered locations and their classifications.

Alternate Flows

- If connection attempt fails, robot returns a beep, does not begin any operations and the GUI connect button toggles to red instead of the green that denotes connection success.
- User retries connection using the same method detailed.
- If not successful on retry, user needs to restart the robot.

Exception Flows

- Bluetooth connection fails causing the TCP/IP sockets to fail and the connection to abrupt close.

Post Conditions Connection is correctly established and maintained and robot performs the desired search and rescue operations, returns all updated map information and switches off.

Requirement dependencies CA0001, UC0001, UC0002, UO0001, UO0002

5.2 Case 002 - Robot object/boundary detection failure

Brief Description Rescue robot is performing search and rescue operation when ultrasonic or colour sensor fails to recognise an object or map boundary requiring user intervention to initiate failsafe to protect hardware and user.

Actors The actor within this use case is the user.

Preconditions The use case assumes that the robot has been located randomly on a dynamically constructed map and is currently in the process of performing search and rescue operations and is currently not registering an object or boundary.

Basic Flow

1. Robot moves from current position into a new undiscovered grid.
2. Robot sends sensor output data and updates new grid as empty despite having an object within or being a map boundary.
3. User notes robot is moving to collide with undiscovered object or off edge of map despite receiving an empty classification on the updated GUI map.
4. User kills connection to cause the robot to halt current operation and half search and rescue task.
5. Robot halts safely before collision.

Alternate Flows

- User initiates a map redraw/refresh to check if incorrect classification is an un-updated reference.
- Robot still does not recognise object and collision occurs.
- Alternatively, robot discovers object/boundary and reports the correct classification to the GUI and proceeds to move to a new grid as expected.

Exception Flows

- Robot sensor sample has a false reading and detects a non existent object based on lighting condition causing in incorrect classification as an object or not.
- User fails to notice incorrect sensor reading and does not react in time to terminate robot/GUI connection.

Post Conditions Connection is correctly established and maintained and robot performs the desired search and rescue operations, returns all updated map information and switches off.

Requirement dependencies CA0001, CN0002, CN0003, CN0004, CD0001, CD0002, CUC0002