The University of Adelaide
COMP SCI 3006/7015 Software Engineering and Project

Software Design Document (SDD)

for

Road Closure Marking Robot

Version 0.1

Prepared by Group 2

Contents

Change History

Date	Version	Reason for Change
10th Sept	0.1	Initial draft

1 Introduction

1.1 Purpose

SDD, which is abbreviation for Software Design Document, describes the implementation procedures of a software project. Specifically, it is needed to transfer a specification into an executable system. This document is the SDD for developing a robot to mark road closures in a city, which is destroyed by a natural disaster. This SDD will document the GUI, AI, Robot Move, Map Representation, and Communication that will be utilized in implementing the system for the robot.

1.2 Scope

The aim of this project is to develop a prototype of a robot which may be used to produce a map of the ruined city and mark the road closures in order to prevent people from moving into these areas.

The map includes virtual representations of the closures, disaster area, the obstacles, the intersection and roads, and unexplored areas which includes anything I mentioned above. And the physical map includes using cardboard or some rigid material to represent obstacles and put them randomly, and using black lines as roads. The most important is that the robot should detect the disaster area and mark the road closures with a whiteboard marker. And in this procedure, the robot also has to avoid the obstacles and go forward.

The virtual map will be dynamically displayed on the systems Graphical User Interface (GUI), and it is created and exported in the form of an Extensible Markup Language (XML) document.

Within the map, the robot will move from the start point and explore the whole map so that it can identify where the obstacles are and mark where the road closures are.

For the purposes of this project the city is no larger than A1 paper size.

1.3 Related Documents

This SDD should be related to the other project documents, namely the Software Project Management Plan (SPMP), the Software Requirement Specification (SRS).

1.4 Overview

This SDD consists of the following:

- Introduction introduces the aims and function of the project;
- System Overview briefly gives a broad outline of the architecture of the project;
- System Architecture and Components Design gives a detailed description of the project architecture, including its composite components;
- Data Design describes the a variety of data structures used in the project;
- Design Details includes the Class, State and Interaction Diagrams in this project;
- Human Interface Design describes the perspective of the user to utilize the robot.
- Resource Estimates summaries computer resource estimates required for operating the robot.

1.5 Document Conventions

All diagrams unless otherwise noted follow standard UML conventions.

2 System Overview

3 System Architecture and Components Design

3.1 Architectural Description

- Architectural Pattern: Pipe and Filter

- Control Style: Centralised (Manager Model)

3.2 Component Decomposition Description

- Object Oriented Style

3.3 Detailed Components Design Description

3.4 Architectural Alternatives

3.5 Design Rationale

4 Data Design

4.1 Database Description

Data Design does not occupy as much as other components in this project, as it has not many requirements elicitation, and some of considerations has been set by the client early. We identify the relevant subsections of data design below:

- Map External Representation
- Map Internal Representation

4.2 Data Structures

- Map External Representation
  Map is to be saved in and read from XML format following the DTD presented by the client.

- Map Internal Representation
  The function is implemented by four main class in this project, the Draw class, the ReadMap class, the SaveMap class and the DataDetecting class.
  ReadMap, which is used to read the map file from XML format, absorbs the whole information on the map through loadBoundary method. More specifically, we created two-dimension array in terms of unexplored area, disaster, intersection, obstacle and closure and three-dimension array in terms of road. Then we put the information absorbing from XML file in terms of position and range (something like length, radius) into different arrays, waiting to be utilized in draw method.
  SaveMap, which is to save current map and the whole information on the current map. Through DataDetecting class, it would receive the latest information detected by the robot so that the data would updated on time and save into a new file. So if the information on the map has been modified, all of the modified things will be saved and also, the original is still there. For example, after exploring the unexplored area by the robot, the more roads, obstacles, intersections and so on will be displayed on the map, and all of them can be saved as a new file at any time, distinct from the original file.
  DataDetecting, which is to store the information detected by the robot. It also created a variety of two-dimension arrays in terms of road, obstacle, intersection, closure and disaster to save the value which is modified due to the detection behaviour of the robot, waiting SaveMap class to utilize.
  Draw, is to draw the original map in the GUI. The class includes paint

method to draw things. And different things will be drawn by different colors so that everyone can distinguish them easily. More specifically, it gets data from different arrays in the ReadMap class and then draw the map followed by those data. In addition, the red circle is disaster, green point is intersection, magenta line as road, orange point as obstacle, and gray point as closure.

5 Design Details

5.1 Class Diagrams

UML/softwareArch.png

5.2 State Diagrams

UML/softwareFSM.png

5.3 Interaction Diagrams

6 Human Interface Design

6.1 Overview of the User Interface

6.2 Detailed Design of the User Interface

7 Resource Estimates

8 Definitions, Acronymns and Abbreviations