Mars Robot

Sudhakar Gottam Pim Jager

December 7, 2015

Contents

1 Requirements of Mars Robot

The Mars Robot shall adhere to the following requirements, which are set up using the FURPS model, and are prioritized using the MOSCOW method.

Functional requirements		
Identifier	Priority	Description
F1	Must	The robot must not fall of the table
F2	Must	The robot must be able to detect colors
F3	Must	The robot must be able to detect rocks
F4	Must	The robot must be able to “measure” rocks
F5	Must	The robot must be able to push light-weight rocks
F6	Should	The robot should be able to avoid lakes
F7	Could	The robot could be able to park in a corner
F8	Must	The robot must be able to “measure” lakes
F9	Must	The robot must be able to avoid rocks
F10	Could	The generated programs could work on both robots
F11	Must	The DSL must allow control of the three actuators
F12	Must	The DSL must provide readouts of the the sensors
F13	Must	The DSL must provide a method of controlling actuators based on the values of sensors
F14	Must	The DSL must provide a method of prioritizing different actions
F15	Should	The DSL should abstract over commonly used functionality
F16	Could	The DSL could provide a method to specify and execute user specified subroutines
F17	Must	The DSL must generate a valid Java program which can be run on the mars rover.
F18	Must	The DSL must provide readable and writable global variables
F19	Must	The DSL must provide a method to write global variables based on sensor values
F20	Would	The robot can communicate with an external entity through Bluetooth or WiFi.
F21	Should	The DSL should provide a method to sound beeps and buzzes
F22	Should	The DSL should provide a method to write string to the LCD
F23	Could	The DSL could provide a way to write strings with values of sensor readings to the LCD

Table 1: Functional requirements of Mars Rover

Usability requirements		
Identifier	Priority	Description
U1	Should	The mapping between a generated Java program and a specification in the DSL should be easy to understand
U2	Must	The DSL must be easy to read and understand
U3	Would	The robot prints a trace of behaviors and values on the LCD for easy debugging

Table 2: Usability requirements of Mars Rover

Reliability requirements		
Identifier	Priority	Description
R1	Must	The robot must not be able to enter a deadlock state, unless one is specified in the DSL specification
R2	Should	The robot should have reliable and consistent sensor readings
R3	Could	The robot could be able to recover from driving into a lake

Table 3: Reliability requirements of Mars Rover

Performance requirements		
Identifier	Priority	Description
P1	Must	The robot must be able to complete mission in finite time
P2	Must	The robot must complete missions as fast as safely possible
P3	Should	The robot should reach its targets with as little movement as possible

Table 4: Performance requirements of Mars Rover

Supportability requirements		
Identifier	Priority	Description
S1	Would	The robot can be reprogrammed and debugged without a physical connection to it
S2	Could	The robot could be expanded with additional sensors or actuators

Table 5: Supportability requirements of Mars Rover

2 Proposed layout of Mars Robot

The Mars rover uses two separate bricks to connect all peripherals. Because of this the two bricks need to communicate about the status of the different actuators and sensors.

This communication could introduce small delays, and as stated in Section 1 the most imported requirement for the rover is that is always keeps itself safe. Therefore we propose a layout where the most important sensors related to safety are connected to the same block as the two main motors. This ensures that the robot can always keep itself safe, even when the communication between the two bricks fails.

	Brick 1	**Brick 2**
Actuators	Left Motor Right motor Measurement Motor	
Sensors	Light left Light Right Ultrasonic Front Ultrasonic Rear	Color Sensor Gyro Sensor Touch Sensor Left Touch Sensor Right

Table 6: Connection of the sensors and actuators to the Mars Rover

In this layout Brick 1 is the main control brick, it handles the most important safety related sensors and all actuators. Brick 2 connects the other sensors and can then communicate their readings to Brick 1.

The touch sensor are not considered essential safety sensors as the mars rover is very sturdy and contact with blocks can already be mostly avoided by the ultrasonic sensor on the front.

If it turns out that two ultrasonic sensors on the same brick are problematic then the front ultrasonic sensor of brick 1 will be interchanged with the gyro sensor on brick 2.

2.1 Final Layout

After discussion with the other groups the final layout for the Mars Rover has been decided as shown in Table 7.

This layout has the benefit over our proposed layout that it does not have two Ultrasonic sensors attached to the same brick.

port	Brick 1	Brick 2
A	Left Motor	
B	Right motor	
C	Measurement Motor	
S1	Light left	Left touch
S2	Light Right	Right touch
S3	Ultrasonic Rear	Ultrasonic Front
S4	Gyro	Color Sensor

Table 7: Connection of the sensors and actuators to the Mars Rover

3 Development process of Mars Robot DSL and corresponding Missions

The Mars Robot DSL and corresponding missions will be developed in an agile way; the development is split into small sprints. Each sprint consists of designing, building, integrating and testing a particular function. At the end of each sprint a working product (DSL and code generation) is delivered.

These sprints ensure that the development process won't end in some form of integration hell of all the different sub-parts and shows possible errors in the design of the robot as early as possible. It also allows to check the product quick, and often, with the client, which ensures that the developed product matches the clients expectations.

The goal of the first sprint will be to write an proof-of-concept rover-program in Java to test all the different sensors and actuators and the communication between the two bricks. This has two main benefits, firstly it ensures that the design of the robot is correct and all the basic function of the robot work. Secondly, it provides an example for the implementation of the code generation.

Consecutive sprints will consist of implementing the different sensors and actuators, and writing the mission possible with the sensors and actuators implemented so far.

The selection of the goals for the next sprint is guided by the requirements of Section 1.

3.1 Planning

Table 8 shows the planning for development of the Mars Rover by listing which requirements from Section 1 will be implemented in which sprint.

Week	Goals
Week 1 (9th Dec)	In sprint 1 we will implement a proof of concept to test the various sensors and actuators of the mars Rover and to test the communication between the two bricks.
Week 2 (16th Dec)	The goal of the second sprint is to have a robot which can wonder around and does not fall of the table. • F1 • F11 (only the two driving actuators) • F12 (only the rear ultrasonic and light sensors) • F13 • F14 • F17 • U2 • R1
Week 3 (6th Jan)	The goal of the third sprint is to implement the detection , measuring and pushing of rocks and lakes. • F2 • F3 • F4 • F5 • F8 • F9 • F11 (the measurement actuator) • F12
Week 4 (13th Jan)	The goal of this sprint is to implement the actual missions • F18 • P1 • P2

Table 8: Planning for development of the mars rover

If there is time left after implementing all the requirements with priority ‘must’ then the following requirements with priority ‘should’ shall be implemented first:

- F21
- F22
- U2