April 12, 2013

Team 19 Inspection Document

**Laboratory # 8: Inspection**

**Morgan, Laura**

**Miaw, Jireh**

**Hauser, Steven**

**Dworak, Catherine**

**Bertoglio, David**

***Work Product***

**Documentation of inspection of Group 20’s source code, following three-phase inspection.**

***Document Revision Information***

**April 12, 2013 – Document created**

**Approval Sheet**

**All group members whose names are listed below approve of the document and contributed fairly.**

**Member Names**

**Morgan, Laura**

**Miaw, Jireh**

**Hauser, Steven**

**Dworak, Catherine**

**Bertoglio, David**

**Pledge**

**On my honor, as a student, I have neither given nor received unauthorized aid on this assignment.**

**Names**

**Morgan, Laura**

**Miaw, Jireh**

**Hauser, Steven**

**Dworak, Catherine**

**Bertoglio, David**

Contents

[Inspection Schedule 5](#_Toc354318324)

[Group Member Responsibilities 5](#_Toc354318325)

[Checklists Used 5](#_Toc354318326)

[Phase 1 5](#_Toc354318327)

[Phase 2 6](#_Toc354318328)

[Phase 3 7](#_Toc354318329)

[Inspection and Rework 7](#_Toc354318330)

[Questions about Implementation by Group 19 7](#_Toc354318331)

[Answers about Implementation for Group 20 7](#_Toc354318332)

[Results of Inspection 8](#_Toc354318333)

[Phase 1 8](#_Toc354318334)

[Phase 2 9](#_Toc354318335)

[Phase 3 9](#_Toc354318336)

## Inspection Schedule

Phase 1 – Internal documentation & source-code layout

Monday, April 15 – 3:30 p.m. during in-lab.

Phase 2 – Coding practices

Sunday, April 21 – 1:30 p.m. Rice Hall.

Phase 3 – Functional correctness

Friday, April 26 – 1:00 p.m. Rice Hall.

## Group Member Responsibilities

Inspections

Phase 1 Inspector – Catherine

Phase 2 Inspector – Laura

Phase 3 Inspector – David, Jireh, Steven, Catherine Laura

Rework

Rework of source code after Phase 1 – Steven

Rework of source code after Phase 2 – Jireh

Rework of source code after Phase 3 - David

## Checklists Used

### Phase 1

Internal documentation & source-code layout (single inspector).

Structure

* proper use of indentation for “levels” in code
* proper use of tabbing when declaring variables
* existence of columns of related items
* existence of white space (spaces after commas, variables, between methods etc.)
* use of new line when line is too long
* consistency followed with use of braces {} throughout

Documentation

* sparing use of comments; only used to document unavoidable complexity
* identifiers
  + meaningful
  + underscores used as separators
  + capitalization of types, Classes
  + names indicate purpose
* constants
  + mixed case capitalization
  + no magic numbers (no embedded literals or constants)
  + only symbolic constants used
  + symbolic constants in all capital letters, separated by underscores
  + avoid abbreviations in names
* methods
  + mixed case for name
  + abbreviations avoided
  + names indicate function
  + “get/set” used where attribute is accessed directly
  + “is” used for Boolean methods
  + “find” used for methods that look something up
* variables
  + name should reveal purpose and/or type
  + plural if representing group of objects
  + iterator variables consistent (for example: i and j)
  + abbreviations avoided

### Phase 2

* Switch statements
  + Switch statements used rather than if-else-if blocks
  + Every switch statement has a break in each case statement
* Variables
  + All variables initialized prior to use
  + All variables declared at top of function
  + All loop variables initialized just before loop
  + No variables initialized that are not used
* Classes
  + All classes have complete set of get() and set() methods
* Bounds
  + All arithmetic checked for ranges within bounds
  + All loop bounds are correct
* Methods
  + All method arguments are used in method
  + All functions named after what they return
  + All procedures named after what they do
* Conditionals
  + Complexity of conditionals should be avoided
  + All relational operators correct (> vs >=)
  + Executable statements not included in conditionals
* Miscellaneous
  + No public data
  + Every file is included in given file uses
  + All file open commands checked for failure
  + Type conversion done explicitly

### Phase 3

* Methods
  + All methods return what they supposed to return
  + All methods execute what they are supposed to execute
* Variables
  + All variables exist for the purpose for which they are named
* All inputs are included and implemented
* All outputs are addressed and implemented
* All specified functionality is implemented

## Inspection and Rework

4/15/13 Phase 1 Inspection completed

4/19/13 Rework of Phase 1 completed

4/19/13

4/21/13 Rework of Phase 2 completed

## Questions about Implementation by Group 19

1. Why does decode message and encode message and implement command use a string ArrayList?
2. Why do we include getPadding() function in Message Handler
3. What are the arguments passed in the differential pilot for?
4. Why is the differential pilot passed Motor.B and Motor.C but not Motor.A?
5. Why is the setRotateSpeed of the Pilot originally set to 90?
6. What is COMMAND\_TYPE\_INDEX and why is it initialized to 0?
7. What is DEFAULT­\_RADIUS and why is it initialized to 90?

## Answers about Implementation for Group 20

1. *Why do we utilize a readFlag instead of a while(True) statement in line 48 of Basestation.java*

The readFlag allows an engineer to control whether or not the GUI will read input from the robot more easily than having to find the while(true) statement.

1. *Why do we choose to hardcode values in methods as opposed to creating a communications class and finding the correct message from there as the communication’s document evolves?*

Hardcoding allows for less complexity within the design of the code while still allowing messages to be changed relatively easily if the communications document evolves.

1. *How do we verify that we receive the sensor data after requesting it?*

The returned message is checked through the checksum to verify that it is a valid message. The first three characters of the message are then used to determine what sensor the data is for.

1. *What is the inherent speed limit to our setSpeed method in line 271?*

The inherent speed limit for set Speed is 999.

1. *What happens when the speed limit is set above this number?*

A message that is longer than 11 bytes will be sent.

1. *What does exit robot do?*

An End Connection message will be sent to the robot.

## Results of Inspection

### Phase 1

Date and time: Monday April 15, 2013 4:00 p.m.

Inspector: Catherine

Defects found

* proper use of indentation for “levels” in code

*line 201, else should be on next line*

* existence of white space (spaces after commas, variables, between methods etc.)

*line 58, extra space between (0, 3)*

*in GUI, spaces between “import” lines*

*white space in beginning public class GUI*

* use of new line when line is too long

*line 82 does not need to be on new line (“ + e.toString());”)*

* consistency followed with use of braces {} throughout

*should check consistency. Starting line 199 you being to put { on the same line as the method declaration and the if statement, rather than the next line. These braces should be moved to the next line. Check methods: getTouchValue(), verifyChecksum(), getChecksum()*

* sparing use of comments; only used to document unavoidable complexity

*comment on line 34 runs off screen*

*comment on line 53 doesn’t clarify code*

*unneeded code should be removed lines 95-100*

*comments in moveForward(), moveBackward(), turnLeft(), turnRight(), turn180(), stop() most likely unncecessary*

*in GUI, comment line 20*

*in GUI, line 163, 362, 460, 495, 596, 632, 637, 650*

* constants

*in GUI class, all private variables should be before public*

* + no magic numbers (no embedded literals or constants)

*in setSpeed() what are numbers 10 and 100?*

* methods

*methods between line 163 and 181 – unimplemented or unnecessary?*

* + abbreviations avoided

*getUltraValue() – consider changing to getUltrasonicValue()*

*getMicroValue() – consider not abbreviating*

* variables
  + name should reveal purpose and/or type

*in method setSpeed, int s does not reveal purpose*

* + abbreviations avoided

*variable “ret” – abbreviated for return? name does not indicate purpose, in methods: establishConnection(), getCheckSum()*

Result of rework:

Rework performed by: Steven

Effort used in corrections: rework time,

Approximate number of statements that had to be added:

Approximate number of statements that were changed:

### Phase 2

Date and time:

Inspector: Laura

Defects found:

Result of rework:

Rework performed by: Jireh

Effort used in corrections: rework time, 20 minutes

Approximate number of statements that had to be added: 5

Approximate number of statements that were changed: 10

### Phase 3

Date and time:

Inspector: David, Catherine, Steven, Jireh, Laura

Defects found:

Result of rework:

Rework performed by: David

Effort used in corrections: rework time,

Approximate number of statements that had to be added:

Approximate number of statements that were changed: