Module Interface Specification for Projectile

Samuel J. Crawford

July 9, 2019

Contents

1 Symbols, Abbreviations and Acronyms

See SRS Documentation at https://jacquescarette.github.io/Drasil/examples/Projectile/srs/Projectile_SRS.pdf

2 Introduction

The following document details the Module Interface Specifications for the implemented modules in a program simulating projectile motion. It is intended to ease navigation through the program for design and maintenance purposes.

Complementary documents include the System Requirement Specifications and Module Guide. The full documentation and implementation can be found at [No manual version of Projectile? Should this link be removed or should there be a link? —SC].

3 Notation

The structure of the MIS for modules comes from Hoffman and Strooper (1995), with the addition that template modules have been adapted from Ghezzi et al. (2003). The mathematical notation comes from Chapter 3 of Hoffman and Strooper (1995). For instance, the symbol := is used for a multiple assignment statement and conditional rules follow the form $(c_1 \Rightarrow r_1|c_2 \Rightarrow r_2|...|c_n \Rightarrow r_n)$.

The following table summarizes the primitive data types used by Projectile.

Data Type	Notation	Description
character	char	a single symbol or digit
real	$\mathbb{R}$	any number in (-∞, ∞)

The specification of Projectile uses strings, a derived data type. Strings are lists of characters. In addition, Projectile uses functions, which are defined by the data types of their inputs and outputs. Local functions are described by giving their type signature followed by their specification.

4 Module Decomposition

The following table is taken directly from the Module Guide document for this project.

Level 1	**Level 2**
Hardware-Hiding	
Behaviour-Hiding	Input Parameters Calculations Output Format Control Module Specification Parameters Module
Software Decision	Sequence Data Structure

Table 1: Module Hierarchy

5 MIS of Control Module

5.1 Module

main

5.2 Uses

Param (Section 6), Calculations (Section 7), Output (Section 8)

5.3 Syntax

5.3.1 Exported Access Programs

Name	In	Out	Exceptions
main	-	-	-

5.4 Semantics

5.4.1 State Variables

None

5.4.2 Access Routine Semantics

main():

- transition: Modify the state of Param module and the environment variables for the Output module by following these steps

Get (filenameIn: string) from user.
get_input(filenameIn, Param)
p_{land} := func_p_land(Param)
d_{offset} := func_d_offset(Param, p_{land})
s := func_s(Param, d_{offset})
#Output calculated values to a file.
write_output(s, d_{offset})

6 MIS of Input Parameters Module

The secrets of this module are the data structure for input parameters and how the values are input and verified.

6.1 Module

Param

6.2 Uses

SpecParam (Section 9)

6.3 Syntax

Name	In	Out	Exceptions
get_input	string, Param	-	FileError
input_constraints	Param	-	badSpeed, badAngle, badTargetPosition
v_{launch}	-	$\mathbb{R}$	-
θ	-	$\mathbb{R}$	-
p_{target}	-	$\mathbb{R}$	-

6.4 Semantics

6.4.1 Environment Variables

inputFile: sequence of string *#f[i] is the ith string in the text file f*

6.4.2 State Variables

To Support IM1 and IM2
v_{launch}: $\mathbb{R}$
θ: $\mathbb{R}$
To Support IM3 and IM4
p_{target}: $\mathbb{R}$

6.4.3 Assumptions

- get_input will be called before the values of any state variables will be accessed or verified.

- The file contains the string equivalents of the numeric values for each input parameter in order, each on a new line. Any comments in the input file should be denoted with a '#' symbol.

6.4.4 Access Routine Semantics

get_input(filename, Param):

- transition: The file name "filename" is first associated with the file f. The following procedural specification is followed:
  1. Read data sequentially from f to populate the state variables from FR1 (v_{launch}, θ, and p_{target}) in Param.
  2. input_constraints()
- exception: exc := a file name "filename" cannot be found OR the format of inputFile is incorrect $\Rightarrow$ FileError

input_constraints():

- output: *out* := none
- exception: exc := (
  $\neg(0 < v_{\text{launch}}) \Rightarrow$ badSpeed |
  $\neg(0 < \theta < \frac{\pi}{2}) \Rightarrow$ badAngle |
  $\neg(0 < p_{\text{target}}) \Rightarrow$ badTargetPosition)

Param.v_{launch}:

- output: *out* := v_{launch}
- exception: none

Param.θ:

- output: *out* := θ
- exception: none

Param.p_{target}:

- output: *out* := p_{target}
- exception: none

6.5 Considerations

The value of each state variable can be accessed through its name (getter). An access program is available for each state variable. There are no setters for the state variables, since the values will be set by get_input and not changed for the life of the program.

See Appendix (Section 11) for the complete list of exceptions and associated error messages.

7 MIS of Calculations Module

The secret of this module is how the required values are calculated.

7.1 Module

Calculations

7.2 Uses

Param (Section 6), SpecParam (Section 9)

7.3 Syntax

7.3.1 Exported Access Programs

Name	In	Out	Exceptions
func_p_land	Param	$\mathbb{R}$	None
func_d_offset	Param, $\mathbb{R}$	$\mathbb{R}$	None
func_s	Param, $\mathbb{R}$	String	None

7.4 Semantics

7.4.1 Assumptions

All of the fields Param have been assigned values before any of the access routines for this module are called.

7.4.2 Access Routine Semantics

func_p_land(Param):

- out: $out := \frac{2*v_{\text{launch}}^2*sin(\theta)*cos(\theta)}{g}$
- exception: exc := none

func_d_offset(Param, p_{land}):

- out: $out := p_{\text{land}} - p_{\text{target}}$
- exception: exc := none

func_s(Param, d_{offset}):

- out: $out :=$ (
  $|\frac{d_{\text{offset}}}{p_{\text{target}}}| < \epsilon \Rightarrow$ "The target was hit." |
  $d_{\text{offset}} < 0 \Rightarrow$ "The projectile fell short."|
  True $\Rightarrow$ "The projectile went long.")

- exception: exc := none

8 MIS of Output Module

8.1 Module

Output

8.2 Uses

Param (Section 6)

8.3 Syntax

8.3.1 Exported Access Program

Name	In	Out	Exceptions
write_output	String, String, $\mathbb{R}$	-	-

8.4 Semantics

8.4.1 State Variables

None

8.4.2 Environment Variables

file: The file named "output".

8.4.3 Access Routine Semantics

write_output(s, d_{offset}):

- transition: Write to environment variable named "file" the calculated values s and d_{offset}.
- exception: none

9 MIS of Specification Parameters

The secrets of this module is the value of the specification parameters.

9.1 Module

SpecParam

9.2 Uses

N/A

9.3 Syntax

9.3.1 Exported Constants

From Table 10 in SRS
$g := 9.8$
$\epsilon := 0.02$

9.4 Semantics

N/A

10 Bibliography

Carlo Ghezzi, Mehdi Jazayeri, and Dino Mandrioli. *Fundamentals of Software Engineering.* Prentice Hall, Upper Saddle River, NJ, USA, 2nd edition, 2003.

Daniel M. Hoffman and Paul A. Strooper. *Software Design, Automated Testing, and Maintenance: A Practical Approach.* International Thomson Computer Press, New York, NY, USA, 1995.

11 Appendix

Message ID	**Error Message**
badSpeed	InputError: Speed must be positive.
badAngle	InputError: Angle must be between zero and pi over two radians.
badTargetPosition	InputError: Target position must be positive.

Table 2: Possible Exceptions