Module Interface Specification for Projectile

Samuel J. Crawford

July 3, 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 Output Format Output Verification 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), Temperature (Section ??), Solver (Section ??), Energy (Section ??), verify_output (Section 7), plot (Section ??), 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 Plot and Output modules by following these steps

Get (filenameIn: string) and (filenameOut: string) from user

load_params(filenameIn)

#Find temperature function $(T_W^{\text{Solid}}, T_W^{\text{Melting}}, T_W^{\text{Liquid}}, T_P^{\text{Solid}}, T_P^{\text{Melting}}, T_P^{\text{Liquid}})$, *and energy* (Q_P) *and times of transition between solid, melting and liquid phases* $(t_{\text{melt}}^{\text{init}}, t_{\text{melt}}^{\text{final}})$

$t_{\text{melt}}^{\text{init}}, [T_W^{\text{Solid}}, T_P^{\text{Solid}}]^T := \text{solve}(\text{ODE\_SolidPCM}, 0.0, [T_{\text{init}}, T_{\text{init}}]^T, \text{event\_StartMelt}, t_{\text{final}})$

$t_{\text{melt}}^{\text{final}}, [T_W^{\text{Melting}}, T_P^{\text{Melting}}, Q_p]^T := \text{solve}(\text{ODE\_MeltingPCM}, t_{\text{melt}}^{\text{init}}, [T_W^{\text{Solid}}(t_{\text{melt}}^{\text{init}}), T_P^{\text{Solid}}(t_{\text{melt}}^{\text{init}}), 0.0]^T,$ event_EndMelt, $t_{\text{final}})$

$[T_W^{\text{Liquid}}, T_P^{\text{Liquid}}]^T :=$ solveNoE(ODE_LiquidPCM, $t_{\text{melt}}^{\text{final}}$, $[T_W^{\text{Melting}}(t_{\text{melt}}^{\text{final}})$, $T_P^{\text{Melting}}(t_{\text{melt}}^{\text{final}})]^T$, $t_{\text{final}})$

#Combine temperatures for $0 \leq t \leq t_{\mathit{final}}$

$T_W(t) = (0 \leq t < t_{\text{melt}}^{\text{init}} \Rightarrow T_W^{\text{Solid}} | t_{\text{melt}}^{\text{init}} \leq t < t_{\text{melt}}^{\text{final}} \Rightarrow T_W^{\text{Melting}} | t_{\text{melt}}^{\text{final}} \leq t \leq t_{\text{final}} \Rightarrow T_W^{\text{Liquid}})$

$T_P(t) = (0 \leq t < t_{\text{melt}}^{\text{init}} \Rightarrow T_P^{\text{Solid}} | t_{\text{melt}}^{\text{init}} \leq t < t_{\text{melt}}^{\text{final}} \Rightarrow T_P^{\text{Melting}} | t_{\text{melt}}^{\text{final}} \leq t \leq t_{\text{final}} \Rightarrow T_P^{\text{Liquid}})$

#Energy values $(E_W(t), E_P(t))$ *for* $0 \leq t \leq t_{\text{final}}$

$E_W(t) = (0 \leq t < t_{\text{melt}}^{\text{init}} \Rightarrow \text{energyWater}(T_W^{\text{Solid}}) | t_{\text{melt}}^{\text{init}} \leq t < t_{\text{melt}}^{\text{final}} \Rightarrow \text{energyWater}(T_W^{\text{Melting}}) | t_{\text{melt}}^{\text{final}} \leq t \leq t_{\text{final}} \Rightarrow \text{energyWater}(T_W^{\text{Liquid}}))$

$E_P(t) = (0 \leq t < t_{\text{melt}}^{\text{init}} \Rightarrow \text{energySolidPCM}(T_P^{\text{Solid}}) | t_{\text{melt}}^{\text{init}} \leq t < t_{\text{melt}}^{\text{final}} \Rightarrow \text{energyMeltingPCM}(Q_P) | t_{\text{melt}}^{\text{final}} \leq t \leq t_{\text{final}} \Rightarrow \text{energyLiquidPCM}(T_P^{\text{Liquid}}))$

#Output calculated values to a file and to a plot. Verify calculated values obey conservation of energy.

verify_output(T_w, T_p, E_w, E_p, t_{final})

plot(T_w, T_p, E_w, E_p, t_{final})

output(filenameOut, T_w, T_p, E_w, E_p, t_{final})

6 MIS of Input Parameters Module

The secrets of this module are the data structure for input parameters, how the values are input and how the values are verified. The load and verify secrets are isolated to their own access programs.

6.1 Module

Param

6.2 Uses

SpecParam (Section 9)

6.3 Syntax

Name	In	Out	Exceptions
load_params	string	-	FileError
verify_params	-	-	InputError
v	-	$\mathbb{R}$	
θ	-	$\mathbb{R}$	
p_{target}	-	$\mathbb{R}$	
p_{land}	-	$\mathbb{R}$	
offset	-	$\mathbb{R}$	
message	-	string	

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: $\mathbb{R}$
θ: $\mathbb{R}$
To Support IM3 and IM4
p_{target}: $\mathbb{R}$
From FR4
p_{land}: $\mathbb{R}$

offset: $\mathbb{R}$

message: string

6.4.3 Assumptions

- load_params will be called before the values of any state variables will be accessed.
- 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

Param.v:

- output: $out := v$
- exception: none

Param.θ:

- output: $out := \theta$
- exception: none

Param.p_{target}:

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

Param.p_{land}:

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

Param.offset:

- output: $out :=$ offset
- exception: none

Param.message:

- output: $out :=$ message
- exception: none

load_params(s):

- transition: The filename s is first associated with the file f. inputFile is used to modify the state variables using the following procedural specification:

    1. Read data sequentially from inputFile to populate the state variables from FR1 (v, θ, and p_{target}).
    2. Calculate the derived quantities (p_{land}, offset, and message) as follows:
        - $p_{\text{land}} := \frac{2v^2 sin(\theta) cos(\theta)}{g}$
        - offset $:= p_{\text{land}} - p_{\text{target}}$
        - message :=

$$\begin{array}{ll} |\frac{\text{offset}}{p_{\text{target}}}| < \epsilon & \Rightarrow \text{“The target was hit.”} \\ \text{offset} < 0 & \Rightarrow \text{“The projectile fell short.”} \\ \text{True} & \Rightarrow \text{“The projectile went long.”} \end{array}$$

    3. verify_params()

- exception: exc := a file name s cannot be found OR the format of inputFile is incorrect $\Rightarrow$ FileError

verify_params():

- out: out := none

- exception: exc :=

$$\begin{array}{ll} \neg(0 < v) & \Rightarrow \text{InputError} \\ \neg(0 < \theta < \frac{\pi}{2}) & \Rightarrow \text{InputError} \\ \neg(0 < p_{\text{target}}) & \Rightarrow \text{InputError} \end{array}$$

See Appendix (Section 11) for the complete list of exceptions and associated error messages. [I think the error messages should be more descriptive, and be defined in the Appendix. —SC]

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 and checked by load params and not changed for the life of the program.

7 MIS of Output Verification Module

7.1 Module

verify_output

7.2 Uses

Param (Section 6)

7.3 Syntax

7.3.1 Exported Constant

ADMIS_ER $= 1 \times 10^{-6}$

7.3.2 Exported Access Programs

Name	In	Out	Exceptions
verify_output	$T_W(t) : \mathbb{R} \rightarrow \mathbb{R}, T_P(t) : \mathbb{R} \rightarrow \mathbb{R}, E_W(t) : \mathbb{R} \rightarrow \mathbb{R}, E_P(t) : \mathbb{R} \rightarrow \mathbb{R}, t_{\text{final}} : \mathbb{R}$	-	EWAT_NOT_CONSERVE, EPCM_NOT_CONSERVE

7.4 Semantics

7.4.1 State Variables

None

7.4.2 Assumptions

All of the fields of the input parameters structure have been assigned a value.

7.4.3 Access Routine Semantics

verify_output($T_W, T_P, E_W, E_P, t_{\text{final}}$):

- exception: exc := (

$(\forall t | 0 \leq t \leq t_{\text{final}} : \text{relErr}(E_W, \int_0^t h_C A_C (T_C - T_W(t)) dt - \int_0^t h_P A_P (T_W(t) - T_P(t)) dt) <$ ADMIS_ER$) \Rightarrow$ EWAT_NOT_CONSERVE

|

$(\forall t | 0 \leq t \leq t_{\text{final}} : \text{relErr}(E_P, \int_0^t h_P A_P (T_W(t) - T_P(t)) dt) <$ ADMIS_ER$) \Rightarrow$ EPCM_NOT_CONSERVE
)

7.4.4 Local Functions

relErr: $\mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}$
relErr$(t, e) \equiv \frac{|t-e|}{|t|}$

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
output	fname: string, message : $\mathbb{R}$, offset : $\mathbb{R}$	-	-

8.4 Semantics

8.4.1 State Variables

None

8.4.2 Environment Variables

file: A text file

8.4.3 Access Routine Semantics

output(fname, message, offset):

- transition: Write to environment variable named fname the calculated values message and 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

Table 2: Possible Exceptions

Message ID	**Error Message**
badLength	Error: Tank length must be > 0
badDiam	Error: Tank diameter must be > 0
badPCMVolume	Error: PCM volume must be > 0
badPCMAndTankVol	Error: PCM volume must be $<$ tank volume
badPCMArea	Error: PCM area must be > 0
badPCMDensity	Error: rho_p must be > 0
badMeltTemp	Error: Tmelt must be > 0 and $< Tc$
badCoilAndInitTemp	Error: Tc must be $>$ Tinit
badCoilTemp	Error: Tc must be > 0 and < 100
badPCMHeatCapSolid	Error: C_ps must be > 0
badPCMHeatCapLiquid	Error: C_pl must be > 0
badHeatFusion	Error: Hf must be > 0
badCoilArea	Error: Ac must be > 0
badWaterDensity	Error: rho_w must be > 0
badWaterHeatCap	Error: C_w must be > 0
badCoilCoeff	Error: hc must be > 0
badPCMCoeff	Error: hp must be > 0
badInitTemp	Error: Tinit must be > 0 and < 100
badFinalTime	Error: tfinal must be > 0
badInitAndMeltTemp	Error: Tinit must be $<$ Tmelt
ODE_ACCURACY	*reltol* and *abstol* were not satisfied by the ODE solver for a given solution step.

ODE_BAD_INPUT	Invalid input to ODE solver
ODE_MAXSTEP	ODE solver took $MaxStep$ steps and did not find solution
warnLength	Warning: It is recommended that 0.1 <= L <= 50
warnDiam	Warning: It is recommended that 0.002 <= D/L <= 200
warnPCMVol	Warning: It is recommended that Vp be >= 0.0001% of Vt
warnVolArea	Warning: It is recommended that Vp <= Ap <= (2/0.001) * Vp
warnPCMDensity	Warning: It is recommended that 500 < rho_p < 20000
warnPCMHeatCapSolid	Warning: It is recommended that 100 < C_ps < 4000
warnPCMHeatCapLiquid	Warning: It is recommended that 100 < C_pl < 5000
warnCoilArea	Warning: It is recommended that Ac <= pi * (D/2) ∧ 2
warnWaterDensity	Warning: It is recommended that 950 < rho_w <= 1000
warnWaterHeatCap	Warning: It is recommended that 4170 < C_w < 4210
warnCoilCoeff	Warning: It is recommended that 10 < hc < 10000
warnPCMCoeff	Warning: It is recommended that 10 < hp < 10000
warnFinalTime	Warning: It is recommended that 0 < tfinal < 86400
warnWaterError	Warning: There is greater than x% relative error between the energy in the water output and the expected output based on the law of conservation of energy. (Where x is the value of $ConsTol$)
warnPCMError	Warning: There is greater than x% relative error between the energy in the PCM output and the expected output based on the law of conservation of energy. (Where x is the value of $ConsTol$)