Module Interface Specification for Glass-BR

Spencer Smith and Jingwei Huang

June 16, 2018

Contents

1 Introduction

The following document details the Module Interface Specifications for the implemented modules in a program Glass-BR. It is intended to ease navigation through the program for design and maintenance purposes. Complementary documents include the System Requirement Specifications (SRS) and Module Guide (MG). The full documentation and implementation can be found at https://github.com/smiths/caseStudies/tree/master/CaseStudies/glass.

2 Notation

The structure of the MIS for modules comes from **?**, with the addition that template modules have been adapted from **?**. The mathematical notation comes from Chapter 3 of **?**. 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 Glass-BR

Data Type	**Notation**	**Description**
character	char	a single symbol or digit
integer	$\mathbb{Z}$	a number without a fractional component in (-∞, ∞)
natural number	$\mathbb{N}$	a number without a fractional component in [1, ∞)
real	$\mathbb{R}$	any number in (-∞, ∞)

The specification of Glass-BRuses some derived data types: sequences, strings, and tuples. Sequences are lists filled with elements of the same data type. Strings are sequences of characters. Tuples contain a list of values, potentially of different types. In addition, Glass-BRuses 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.

3 Module Hierarchy

To view the Module Hierarchy, see section 3 of the MG.

4 MIS of InputT 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.

4.1 Module

Param

4.2 Uses

SpecParam (Section **??**)

4.3 Syntax

Name	**In**	**Out**	**Exceptions**
load_params	string	-	FileError
verify_params	-	-	ValueError
a	-	$\mathbb{R}$	
b	-	$\mathbb{R}$	
g	-	GlassTypeT	
$P_{b_{\text{tol}}}$	-	$\mathbb{R}$	
SD_x	-	$\mathbb{R}$	
SD_y	-	$\mathbb{R}$	
SD_z	-	$\mathbb{R}$	
t	-	ThicknessT	
w	-	$\mathbb{R}$	
t_d	-	$\mathbb{R}$	
LDF	-	$\mathbb{R}$	
LSF	-	$\mathbb{R}$	

4.4 Semantics

4.4.1 Environment Variables

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

4.4.2 State Variables

# From R1

a: $\mathbb{R}$
b: $\mathbb{R}$
g: glassType
$P_{b_{\text{tol}}}$: $\mathbb{R}$
SD_x : $\mathbb{R}$
SD_y : $\mathbb{R}$
SD_z : $\mathbb{R}$
t: $\mathbb{R}$
w: $\mathbb{R}$

# From R2
t_d: $\mathbb{R}$
LDF: $\mathbb{R}$
LSF: $\mathbb{R}$

4.4.3 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. The order is the same as in the table in R1 of the SRS. Any comments in the input file should be denoted with a '#' symbol.

4.4.4 Access Routine Semantics

Param.a:

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

Param.b:

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

...

Param.LSF:

- output: $out :=$ LSF
- 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 R1 (L to $ConsTol$).
  2. Calculate the derived quantities (all other state variables) as follows:
     - $V_{\text{tank}} := \pi \times L \times (\frac{D}{2})^2$
     - $m_W := \rho_w(V_t - V_p)$
     - $m_P := \rho_p V_p$
     - $\tau_W := \frac{m_w C_w}{A_c h_c}$
     - $\eta := \frac{h_p A_p}{h_c A_c}$
     - $\tau_P^S := \frac{m_p C_{ps}}{h_p A_p}$
     - $\tau_P^L := \frac{m_p C_{pl}}{h_p A_p}$
     - $E_{P\text{melt}}^{\text{init}} := C_{ps} m_p (T_{\text{melt}} - T_{\text{init}})$
     - $E_{P\text{melt}}^{\text{all}} := H_f m_p$
     - $m_W^{\text{noPCM}} := \rho_w V_t$
     - $\tau_W^{\text{noPCM}} := \frac{m_W^{\text{noPCM}} C_w}{h_c A_c}$
  3. verify_params()

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

verify_params():

- out: out := none

- exception: exc :=

$\neg(L > 0)$	⇒ badLength
$\neg(L_{\text{min}} \leq L \leq L_{\text{max}})$	⇒ warnLength
$\neg(D > 0)$	⇒ badDiam
$\neg(\frac{D}{L}_{\text{min}} \leq \frac{D}{L} \leq \frac{D}{L}_{\text{max}})$	⇒ warnDiam
$\neg(V_P > 0)$	⇒ badPCMVolume
$\neg(V_P \geq \text{minfract} \cdot V_{\text{tank}}(D, L))$	⇒ warnPCMVol
$\neg(V_P < V_{\text{tank}}(D, L))$	⇒ badPCMAndTankVol
$\neg(A_P > 0)$	⇒ badPCMArea
$\neg(V_P \leq A_P \leq \frac{2}{h_{\text{min}}} V_P)$	⇒ warnVolArea
$\neg(\rho_P > 0)$	⇒ badPCMDensity
$\neg(\rho_P^{\text{min}} < \rho_P < \rho_P^{\text{max}})$	⇒ warnPCMDensity

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

4.5 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.

5 MIS of LoadASTM Module

6 MIS of Output Module

7 MIS of Calc Module

8 MIS of Control Module

8.1 Module

main

8.2 Uses

Param (Section ??), Temperature (Section ??), Solver (Section ??), Energy (Section ??), verify_output (Section ??), plot (Section ??), output (Section 6)

8.3 Syntax

8.3.1 Exported Access Programs

Name	In	Out	Exceptions
main	-	-	-

8.4 Semantics

8.4.1 State Variables

None

8.4.2 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)

9 MIS of ConstantsAndTypes Module

Module

ConstantsAndTypes

Uses

N/A

Syntax

Exported Constants

# From Table 8 in SRS
$m := 7$
$k := (2.86)\,10^{-53}$
$E := (7.17)\,10^{7}$
$t_d := 3$
$\text{LDF} := \left(\frac{t_d}{60}\right)^{\frac{m}{16}}$
$\text{LSF} := 1$
$d_{\text{max}} := 5.0$
$d_{\text{min}} := 0.1$
$\text{AR}_{\text{max}} := 5.0$
$w_{\text{max}} := 910.0$
$w_{\text{min}} := 4.5$
$\text{SD}_{\text{min}} := 6.0$
$\text{SD}_{\text{max}} := 130.0$

Exported Types

GlassTypeT = {AN, FT, HS}
ThicknessT = {2.5, 3.0, 5.0, 8.0, 12.0, 19.0, 2.7, 4.0, 6.0, 10.0, 16.0, 22.0}

Exported Access Programs

None

Semantics

State Variables

None

State Invariant

None

10 MIS of FunctT Module

11 MIS of ContoursT Module

12 MIS of SeqServices Module