Module Interface Specification for Glass-BR

Spencer Smith

July 3, 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 (-∞, ∞)
string	$\mathbb{S}$	all finite sequences of symbols from the ASCII character set

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 Input 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. This module follows the singelton pattern; that is, there is only one instance of this module.

4.1 Module

Input

4.2 Uses

Constants (Section 9), GlassTypeADT (Section 10), ThicknessADT (Section 11)

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	
TNT	-	$\mathbb{R}$	
w	-	$\mathbb{R}$	
m	-	$\mathbb{R}$	
k	-	$\mathbb{R}$	
E	-	$\mathbb{R}$	
t_d	-	$\mathbb{R}$	
LDF	-	$\mathbb{R}$	
LSF	-	$\mathbb{R}$	
h	-	$\mathbb{R}$	
GTF	-	$\mathbb{R}$	
SD	-	$\mathbb{R}$	

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: GlassTypeT
$P_{b_{\text{tol}}}$: $\mathbb{R}$
SD_x : $\mathbb{R}$
SD_y : $\mathbb{R}$
SD_z : $\mathbb{R}$
t: ThicknessT
TNT: $\mathbb{R}$
w: $\mathbb{R}$

From R2
m: $\mathbb{R}$
k: $\mathbb{R}$
E: $\mathbb{R}$
t_d: $\mathbb{R}$
LDF: $\mathbb{R}$
LSF: $\mathbb{R}$
h: $\mathbb{R}$
GTF: $\mathbb{R}$
SD: $\mathbb{R}$

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.GTF:

- output: $out :=$ GTF
- 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 (a to w).
  2. Calculate the derived quantities (all other state variables, from R2) as follows:
     - $m, k, E, t_d,$ LDF, LSF as defined in Constants (Section 9)
     - $h = t$.toMinThick() (Section 11)
     - GTF $= g$.GTF() (Section 10)
     - SD $= \sqrt{\text{SD}_x{}^2 + \text{SD}_y{}^2 + \text{SD}_z{}^2}$
  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 :=

$\neg(a > 0)$	$\Rightarrow$ ValueError("dimension a must be positive")
$\neg(a/b \geq 0)$	$\Rightarrow$ ValueError("a must be equal to, or greater than, b")
$\neg(d_{\min} \leq a \leq d_{\max})$	$\Rightarrow$ ValueError("a too large or small")
$\neg(a/b < \mathrm{AR}_{\max})$	$\Rightarrow$ ValueError("Allowable aspect ratio exceeded")
$\neg(b > 0)$	$\Rightarrow$ ValueError("dimension b must be positive")
$\neg(d_{\min} \leq b \leq d_{\max})$	$\Rightarrow$ ValueError("b too large or small")
$\neg(0 < P_{b_{\text{tol}}} < 1)$	$\Rightarrow$ ValueError("$P_{b_{\text{tol}}}$ must lie between 0 and 1")
$\neg(w \geq 0)$	$\Rightarrow$ ValueError("charge weight w must be greater or equal to zero")
$\neg(w_{\min} < w < w_{\max})$	$\Rightarrow$ ValueError("charge weight w is too small or too large")
$\neg(\mathrm{TNT} > 0)$	$\Rightarrow$ ValueError("TNT must be positive")
$\neg(\mathrm{SD} > 0)$	$\Rightarrow$ ValueError("stand off distance (SD) must be positive")
$\neg(\mathrm{SD}_{\min} < \mathrm{SD} < \mathrm{SD}_{\max})$	$\Rightarrow$ ValueError("stand off distance (SD) is too small or too large")

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

Module

LoadASTM

Uses

FunctADT, ContoursADT

Syntax

Exported Constants

None

Exported Access Programs

Routine name	In	Out	Exceptions
LoadTSD	s : string	ContoursT	FileError
LoadSDF	s : string	ContoursT	FileError

Semantics

Environment Variables

infile: two dimensional sequence of text characters

State Variables

None

State Invariant

None

Assumptions

The input file will match the given specification.

Access Routine Semantics

LoadTSD(s)

- output: Create *out* following the following steps. read data from the file infile associated with the string s. Use this data to create a ContoursT object. For each value of w create an object of FunctT and add it to the ContoursT object. Each of the FunctT objects will consist of q versus SD data. The first row of the TSD file contains the values of w. The subsequent columns are grouped in pairs. Each pair corresponds to a column of SD data and a column of q data. There is a pair of columns in this pattern for each value of w.

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

LoadSDF(s)

- output: Create *out* following the following steps. read data from the file infile associated with the string s. Use this data to create a ContoursT object. For each value of J create an object of Funct T and add it to the ContoursT object. Each of the FunctT objects will consist of q^* versus AR data. The first row of the SDF file contains the values of J. The subsequent columns are grouped in pairs. Each pair corresponds to a column of AR data and a column of q^* data. There is a pair of columns in this pattern for each value of J.

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

6 MIS of Output Module

Module

Output

Uses

Input, ThicknessADT, GlassTypeADT

Syntax

Exported Constants

None

Exported Access Programs

Routine name	In	Out	Exceptions
output	$s : \mathbb{S}$, $P_b : \mathbb{R}$, $\text{is}_{\text{safe1}} : \mathbb{B}$, $\text{is}_{\text{safe2}} : \mathbb{B}$, LR: $\mathbb{R}$, $q : \mathbb{R}$, $h : \mathbb{R}$, LDF: $\mathbb{R}$, $J : \mathbb{R}$, NFL: $\mathbb{R}$, GTF: $\mathbb{R}$, $\hat{q} : \mathbb{R}$, $\hat{q}_{\text{tol}} : \mathbb{R}$, $J_{\text{tol}} : \mathbb{R}$, AR: $\mathbb{R}$	-	-

Semantics

Environment Variables

outfile: two dimensional sequence of text characters

State Variables

None

State Invariant

None

Assumptions

None

Access Routine Semantics

output(s, P_b, is_{safe1}, is_{safe2}, LR, q, h, LDF, J, NFL, GTF, $\hat{q}$, $\hat{q}_{\text{tol}}$, J_{tol}, AR)

- transition: write data to the file outfile associated with the string s. The data to output follows:
  - From R4: values from R1 (a, b, g, $P_{b_{\text{tol}}}$, SD_x, SD_y, SD_z, t, TNT, w), values from R2 (m, k, E, t_d, LDF, LSF, h, GTF, SD)
  - From R5: ($\text{is}_{\text{safe1}} \wedge \text{is}_{\text{safe2}} \Rightarrow$ "For the given input parameters, the glass is considered safe" | True $\Rightarrow$ "For the given input parameters, the glass is NOT considered safe")
  - From R6 (s, P_b, is_{safe1}, is_{safe2}, LR, q, h, LDF, J, NFL, GTF, $\hat{q}$, $\hat{q}_{\text{tol}}$, J_{tol}, AR)
- exception: None

7 MIS of Calc Module

Module

Calc

Uses

Input, Constants, ContoursADT

Syntax

Exported Constants

None

Exported Access Programs

Routine name	In	Out	Exceptions
calc_q_hat	$q : \mathbb{R}$	$\mathbb{R}$	-
calc_j_tol	inputs	$\mathbb{R}$	-
calc_pb	inputs	$\mathbb{R}$	-
calc_nfl	inputs	$\mathbb{R}$	-
calc_lr	inputs	$\mathbb{R}$	-
calc_is_safePb	inputs	$\mathbb{R}$	-
calc_is_safeLr	inputs	$\mathbb{R}$	-

Semantics

State Variables

None

State Invariant

None

Assumptions

None

Access Routine Semantics

calc_q_hat(q) # *From SRS DD7 Dimensionless Load ($\hat{q}$)*

- output: $out := \frac{q(ab)^2}{Eh^4 \text{GTF}}$
- exception: None

calc_j_tol() # *From SRS DD9 Tolerable Stress Distribution Factor (J_{tol})*

- output: $out := = \ln[\ln(\frac{1}{1-P_{b_{\text{tol}}}}) \frac{(\frac{a}{1000} \times \frac{b}{1000})^{m-1}}{k((E \times 1000)(\frac{h}{1000})^2)^m \text{LDF}}]$
- exception: None

calc_pb(j) # *From SRS DD7 Dimensionless Load ($\hat{q}$)*

- output: $out := \frac{q(ab)^2}{Eh^4 \text{GTF}}$
- exception: None

calc_nfl($\hat{q}_{\text{tol}}$) # *From SRS DD7 Dimensionless Load ($\hat{q}$)*

- output: $out := \frac{q(ab)^2}{Eh^4 \text{GTF}}$
- exception: None

calc_lr(nfl) # *From SRS DD7 Dimensionless Load ($\hat{q}$)*

- output: $out := \frac{q(ab)^2}{Eh^4 \text{GTF}}$
- exception: None

calc_is_safePb(pb) # *From SRS DD7 Dimensionless Load ($\hat{q}$)*

- output: $out := \frac{q(ab)^2}{Eh^4 \text{GTF}}$
- exception: None

calc_is_safeLr(lr, q) # *From SRS DD7 Dimensionless Load ($\hat{q}$)*

- output: $out := \frac{q(ab)^2}{Eh^4 \text{GTF}}$
- exception: None

8 MIS of Control Module

8.1 Module

main

8.2 Uses

Input (Section 4), 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 Constants Module

9.1 Module

Constants

9.2 Uses

N/A

9.3 Syntax

9.3.1 Exported Constants

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

9.3.2 Exported Types

None

9.3.3 Exported Access Programs

None

9.4 Semantics

9.4.1 State Variables

None

9.4.2 State Invariant

None

10 GlassType ADT Module

From DD6 (Glass Type Factor (GTF))

10.1 Template Module

GlassTypeADT

10.2 Uses

None

10.3 Syntax

10.3.1 Exported Constants

None

10.3.2 Exported Types

GlassTypeT = ?

10.3.3 Exported Access Programs

Routine name	In	Out	Exceptions
new GlassTypeT	$\mathbb{S}$	GlassTypeT	ValueError
GTF		$\mathbb{R}$	
toString		$\mathbb{S}$	

10.4 Semantics

10.4.1 State Variables

g: {AN, FT, HS}

10.4.2 State Invariant

None

10.4.3 Assumptions

None

10.4.4 Access Routine Semantics

new GlassTypeT(s):

- transition: $g := (s = \text{“AN”} \Rightarrow \text{AN} | s = \text{“FT”} \Rightarrow \text{FT} | s = \text{“HS”} \Rightarrow \text{HS})$
- output: $out :=$ self
- exception: $(\neg(s \in \{\text{“AN”}, \text{“FT”}, \text{“HS”}\}) \Rightarrow \text{ValueError})$

GTF():

- output: $out := (g = \text{AN} \Rightarrow 1.0 | g = \text{FT} \Rightarrow 4.0 | g = \text{HS} \Rightarrow 2.0)$
- exception: None

toString():

- output: $out := (g = \text{AN} \Rightarrow \text{“AN”} | g = \text{FT} \Rightarrow \text{“FT”} | g = \text{HS} \Rightarrow \text{“HS”})$
- exception: None

11 Thickness ADT Module

From DD2 (Minimum Thickness from Nominal Thickness)

11.1 Template Module

ThicknessADT

11.2 Uses

None

11.3 Syntax

11.3.1 Exported Constants

None

11.3.2 Exported Types

ThicknessT = ?

11.3.3 Exported Access Programs

Routine name	In	Out	Exceptions
new ThicknessT	$\mathbb{R}$	ThicknessT	ValueError
toMinThick		$\mathbb{R}$	
toFloat		$\mathbb{R}$	

11.4 Semantics

11.4.1 State Variables

$t : T$ where $T = \{2.5, 2.7, 3.0, 4.0, 5.0, 6.0, 8.0, 10.0, 12.0, 16.0, 19.0, 22.0\}$

11.4.2 State Invariant

None

11.4.3 Assumptions

None

11.4.4 Access Routine Semantics

new ThicknessT(x):

- transition: $t := x$
- output: $out :=$ self
- exception: $(\neg(x \in T) \Rightarrow \text{ValueError})$

toMinThick():

- output:

$$\begin{aligned} out := (t = 2.5 \Rightarrow 2.16 \;\;|\;\; t = 2.7 \Rightarrow 2.59 \;\;|\\ t = 3.0 \Rightarrow 2.92 \;\;|\;\; t = 4.0 \Rightarrow 3.78 \;\;|\\ t = 5.0 \Rightarrow 4.57 \;\;|\;\; t = 6.0 \Rightarrow 5.56 \;\;|\\ t = 8.0 \Rightarrow 7.42 \;\;|\;\; t = 10.0 \Rightarrow 9.02 \;\;|\\ t = 12.0 \Rightarrow 11.91 \;\;|\;\; t = 16.0 \Rightarrow 15.09 \;\;|\\ t = 19.0 \Rightarrow 18.26 \;\;|\;\; t = 22.0 \Rightarrow 21.44 \;\;) \end{aligned}$$

- exception: None

toFloat():

- output: $out := t$
- exception: None

12 MIS of FunctADT Module

13 MIS of ContoursADT Module

14 MIS of SeqServices Module