Module Interface Specification for Stoichiometry Mass-Mass Program

Deemah Alomair

December 23, 2019

1 Revision History

Date	Version	Notes
1/12/2019	1.0	First version of document
20/12/2019	2.0	Second version of document

2 Symbols, Abbreviations and Acronyms

See SRS Documentation here

Contents

3 Introduction

The following document details the Module Interface Specifications for Stoichiometry Mass-Mass Program

Complementary documents include the System Requirement Specifications and Module Guide. The full documentation and implementation can be found here

4 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 SMMP.

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 (-∞, ∞)
element	ElementT	set consists of all chemical elements.
Molecule	MoleculeT	tuple of (element : ElementT, atomValue : N).
Compound	CompoundT	set of MoleculeT.
Stoichiometric	StoichiometricT	tuple of (coefficient : N , Compound : CompoundT)
one reaction side	ChemicalEqT	set of StoichiometricT
chemical reaction	ReactionT	sequence [2] of ChemicalEq.

The specification of SMMP uses 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, SMMP 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.
Since chemical reaction is central to SMMP, the following notation is introduced in an attempt to simplify the presentation of the MIS.

Chemical reaction:

$$c_1 R[0]_1 + c_2 R[0]_2 = d_1 R[1]_1 + d_2 R[1]_2$$

where $R[0]_1$ is reactant with known mass, c_1 is the coefficient number associated to this reactant , $R[0]_2$ is reactant with unknown mass, c_2 is the coefficient number associated to this reactant , $R[1]_i$ is a product and d_i is he coefficient number associated to this product.

5 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 Module Atomic Mass Module Balancing Chemical Reaction Module Mass Calculation Module
Software Decision	GUI Module

Table 1: Module Hierarchy

6 MIS of Input Module

6.1 Module

Input

6.2 Uses

GUI (Section 10)

6.3 Syntax

6.3.1 Exported Constants

None.

6.3.2 Exported Access Programs

Name	In	Out	Exceptions
$input_1$	elements :ElementT atomsValue :$\mathbb{N}$	-	MismatchedInput
$input_2$	Mass-Value : $\mathbb{R}$	-	NegativeMassException

6.4 Semantics

6.4.1 State Variables

element : ElementT
atomsValue : $\mathbb{N}$
Mass : $\mathbb{R}$

6.4.2 Environment Variables

scr : 13.3-inch (1440 x 900) MacBook Air screen .
bk : keyboard input.

6.4.3 Assumptions

None.

6.4.4 Access Routine Semantics

$input_1$(Chemical-Reaction):

- transition: ReactionT := element , atomsValue

- output: None.
- exception: exc := (element $\notin$ ElementT $\vee$ atomsValue $\notin \mathbb{N} \Rightarrow$ MismatchedInput)

$input_2$(Mass-Value):

- transition: Mass := Mass-Value
- output: None.
- exception: exc := (Mass-Value $\leq 0 \Rightarrow$ NegativeMassException)

6.4.5 Local Functions

None.

7 MIS of Atomic Mass Module

7.1 Module

Atomic Mass

7.2 Uses

Reaction (Section 8)

7.3 Syntax

7.3.1 Exported Constants

None.

7.3.2 Exported Types

None.

7.3.3 Exported Access Programs

Name	In	Out	Exceptions
Atomic-Mass	e: ElementT	AtomicMass: $\mathbb{R}$	-

7.4 Semantics

7.4.1 State Variables

None.

7.4.2 Environment Variables

None.

7.4.3 Assumptions

None.

7.4.4 Access Routine Semantics

Atomic-Mass(e):

- transition: None.
- output: out := AtomicMass(e)

- exception: None.

7.4.5 Local Functions

None

8 MIS of Balancing Chemical Reaction Module

8.1 Template Module

Reaction

8.2 Uses

Input (Section 6)

8.3 Syntax

8.3.1 Exported Constants

None.

8.3.2 Exported Types

ReactionT = ?

8.3.3 Exported Access Programs

Name	In	Out	Exceptions
ReactionT	ElementT, MoleculeT, CompoundT , StoichiometricT ChemicalEqT.	-	-
IsBalanced-Reaction	-	B: Boolean	-
Balanced-Reaction	Reaction: ReactionT	Reaction*: ReactionT	(¬ IsBalanced-Reaction(Reaction*))

8.4 Semantics

8.4.1 State Variables

R : ReactionT

8.4.2 Environment Variables

None.

8.4.3 Assumptions

None.

8.4.4 Access Routine Semantics

IsBalanced-Reaction():

- transition: None.
- output: out := True $\vee$ False
- exception: None.

Balanced-Reaction(Reaction):

- transition: Reaction* := Reaction
- output: None.
- exception: exc := ($\neg$ IsBalanced-Reaction(Reaction*) $\Rightarrow$ not-balanced)

The program will transfer unbalanced reaction to balanced one by counting the number of atoms used for each element in reactants side and in the products side then compare them together. if the total is not the same, it will add an appropriate coefficient which is belong to natural number before each reactant and product to make the number of atoms in each side equal. it will repeat this process for all element involved in the reaction until it end up with a balance reaction(Reaction*). where Reaction* = coefficient1 * Reactant1 + coefficient2 * Reactant2 = coefficient3 * Product1 + coefficient4 * Product2. If the reaction is already balanced no coefficient will be needed.

8.4.5 Local Functions

1. Number-of-Atoms-in-Molecule: MoleculeT X ElementT $\rightarrow \mathbb{N}$
Number-of-Atoms-in-Molecule(m,e) $\equiv$ (m.element = e $\rightarrow$ m.number | m.element $\neq$ e $\rightarrow$ 0)

2. Number-of-Atoms-in-Compound: CompoundT X ElementT $\rightarrow \mathbb{N}$
Number-of-Atoms-in-Compound(C,e) $\equiv$ + (m : MoleculeT| m $\in$ C $\cdot$ Number-of-Atoms-in-Molecule(m,e))

3. Number-of-Atoms-in-Stoichiometric: StoichiometricT $\times$ ElementT $\rightarrow \mathbb{N}$
Number-of-Atoms-in-Stoichiometric(S,e) $\equiv$ (S.coefficient $\cdot$ Number-of-Atoms-in-Compound (S.Compound,e))

4. Number-of-Atoms-in-ChemicalEq: ChemicalEqT X ElementT $\rightarrow \mathbb{N}$
Number-of-Atoms-in-ChemicalEq (Ce,e) $\equiv$ + (S : StoichiometricT | S $\in$ Ce $\cdot$ Number-of-Atoms-in-Stoichiometric(S,e))

5. ElementsInCompoundT : CompoundT $\rightarrow$ set of ElementT.
ElementsInCompound(C) $\equiv \cup$ { m : MoleculeT | m $\in$ C $\cdot$ m.element}

6. ElementsInChemicalEq : ChemicalEqT $\rightarrow$ set of ElementT.
ElementsInChemicalEq(Ce) $\equiv \cup$ { C : CompoundT | C $\in$ Ce $\cdot$ ElementsInCompound(C)}

7. IsBalancedReactionForElement : ReactionT x ElementT $\rightarrow$ B
IsBalancedReactionForElement (R, e) $\equiv$ Number-of-Atoms-in-ChemicalEq(R[0],e) = Number-of-Atoms-in-ChemicalEq (R[1] ,e)

8. IsBalanced-Reaction : ReactionT $\rightarrow$ B
IsBalanced-Reaction(R) $\equiv \forall$ (e : ElementT | e $\in$ ElementsInChemicalEq(R[0]) $\cdot$ IsBalancedReactionForElement(R, e))

9 MIS of Mass Calculation Module

9.1 Module

Mass

9.2 Uses

Input (Section 6) , Atomic Mass (Section 7) , Reaction (Section 8)

9.3 Syntax

9.3.1 Exported Constants

None.

9.3.2 Exported Access Programs

Name	In	Out	Exceptions
Get-Molecular-weight	Number-of-Atoms-in-Compound: $\mathbb{N}$	Molecular-Weight:$\mathbb{R}$	-
Convert-Mass-to-Mole1	Mass1: $\mathbb{R}$ Molecular-Weight1: $\mathbb{R}$	Mole1: $\mathbb{R}$	Mass ≤ 0
Get-Mole-Ratio	Coefficient1 : $\mathbb{N}$ Coefficient2 : $\mathbb{N}$	Mole-Ratio: $\mathbb{N}$	-
Convert-MoleRatio-to-Mole2	Mole1: $\mathbb{R}$ Mole-Ratio: $\mathbb{R}$	Mole2: $\mathbb{R}$	-
Convert-Mole-to-Mass	Mole2: $\mathbb{R}$ Molecular-Weight2: $\mathbb{R}$	Mass2:$\mathbb{R}$	-

9.4 Semantics

9.4.1 State Variables

None

9.4.2 Environment Variables

None

9.4.3 Assumptions

None

9.4.4 Access Routine Semantics

Get-Molecular-weight(Number-of-Atoms-in-Compound):

- transition: None
- output: out := Atomic-Mass × Number-of-Atoms-in-Compound
- exception: None

Convert-Mass-to-Mole(Mass1, Molecular-Weight1):

- transition: None
- output: Mole1 := $\frac{Mass1}{Molecular-Weight1}$
- exception: exc := (Mass1 $\leq 0 \Rightarrow$ NegativeMassException)

Get-Mole-Ratio(Coefficient2, Coefficient1):

- transition: None
- output: out := $\frac{Coefficient2}{Coefficient1}$
- exception: None

Convert-MoleRatio-to-Mole(Mole1, Mole-Ratio):

- transition: None
- output: Mole2 := $\frac{Mole1}{Mole-Ratio}$
- exception: None

Convert-Mole-to-Mass(Mole2, Molecular-Weight2):

- transition: None
- output: Mass2 := $\frac{Mole2}{Molecular-weight2}$
- exception: None

9.4.5 Local Functions

None

10 MIS of GUI Module

10.1 Module

GUI

10.2 Uses

Reaction (Section 8) , Mass (Section 9)

10.3 Syntax

10.3.1 Exported Constants

None.

10.3.2 Exported Access Programs

Name	In	Out	Exceptions
GUI	-	Mass: $\mathbb{R}$ Reaction*: ReactionT	-

10.4 Semantics

Not Applicable.

References

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.