Software Requirements Specification for ChemCode: A program for solving chemistry problems

Samuel Crawford

February 3, 2023

Contents

Revision History

Date	Version	Notes
Jan. 21, 2023	0.0	Start document, fill in title page, and add references
	0.1	Fill in Problem Description and Goal Statements sections, as well as all relevant definitions
Jan. 22, 2023	0.1.1	Replace the notion of "fractional oxidation state" with "non-stoichiometric compound"
	0.1.2	Fix capTemplate reference
	0.1.3	Update Abbreviations and Acronyms
	0.2	Add IM for balancing a chemical reaction (IM3), along with relevant reference information
Jan. 23, 2023	0.2.1	Add IM for determining if a chemical reaction is feasible (IM2), along with relevant reference information
Jan. 25, 2023	0.2.2	Add references between sections for notation and definitions
	0.2.3	Abstract IM2
	0.2.4	Add IM for converting chemical equation to a matrix (IM1), along with relevant reference information
	0.2.5	Rename and add input constraint to IM3
	0.2.6	Add note about IM3 returning the smallest solution
	0.2.7	Add TM for matrix equation (T1)
	0.2.8	Improve referencing, mainly for TMs, and add note about potential TM for Conservation of Mass
	0.2.9	Add check for Conservation of Mass in IM2
Jan. 26, 2023	0.3	Add Likely Changes from Problem Statement, add assumption about difference between number of elements and compounds (A1), and add mole to Table of Units
	0.3.1	Update T1 (and TM function) from feedback from presentation
	0.3.2	Fill in Scope of Requirements, fix A1 and change "infeasible" to "feasible"
Jan. 27, 2023	0.3.3	Improve section referencing in Revision History
	0.4	Fill in Functional Requirements
Jan. 28, 2023	0.4.1	Fill in main Nonfunctional Requirements, along with relevant reference information
	0.4.2	Add TM for the Law of Conservation of Mass (T2)
	0.4.3	Represent input to IM1 as two sequences of chemical formulas, add assumptions about user input, and remove infeasible likely change
Jan. 29, 2023	0.4.4	Add NFRs for understandability (NFR2) and verifiability (NFR6) and improve quality of other NFRs

Date	Version	Notes
Jan. 29, 2023	0.5	Fill in Introduction based on Drasil examples
	0.5.1	Fill in Physical System Description and remove "product" and "reactant" from Terminology and Definitions
Jan. 30, 2023	0.5.2	Remove unnecessary template comments about SRS document and the ones from Specific System Description up to Instance Models, except for Data Types

1 Reference Material

This section records information for easy reference.

1.1 Table of Units

Throughout this document SI (Système International d'Unités) is employed as the unit system. In addition to the basic units, several derived units are used as described below. For each unit, the symbol is given followed by a description of the unit and the SI name.

symbol	unit	SI
mol	amount of substance	mole
m	length	metre
kg	mass	kilogram
s	time	second
°C	temperature	centigrade
J	energy	joule
W	power	watt ($\mathrm{W} = \mathrm{J\,s^{-1}}$)

[Only include the units that your SRS actually uses. —TPLT]

[Derived units, like newtons, pascal, etc, should show their derivation (the units they are derived from) if their constituent units are in the table of units (that is, if the units they are derived from are used in the document). For instance, the derivation of pascals as $\mathrm{Pa} = \mathrm{N\,m^{-2}}$ is shown if newtons and m are both in the table. The derivations of newtons would not be shown if kg and s are not both in the table. —TPLT]

[The symbol for units named after people use capital letters, but the name of the unit itself uses lower case. For instance, pascals use the symbol Pa, watts use the symbol W, teslas use the symbol T, newtons use the symbol N, etc. The one exception to this is degree Celsius. Details on writing metric units can be found on the NIST web-page. —TPLT]

1.2 Table of Symbols

The table that follows summarizes the symbols used in this document along with their units. The choice of symbols was made to be consistent with the heat transfer literature and with existing documentation for solar water heating systems. The symbols are listed in alphabetical order.

symbol	unit	description
A_C	m^2	coil surface area
A_{in}	m^2	surface area over which heat is transferred in

[Use your problems actual symbols. The si package is a good idea to use for units. —TPLT]

1.3 Abbreviations and Acronyms

symbol	description
1D	One-Dimensional
A	Assumption
DD	Data Definition
GD	General Definition
GS	Goal Statement
IM	Instance Model
LC	Likely Change
PS	Physical System Description
R	Requirement
SRS	Software Requirements Specification
T	Theoretical Model
VnV	Verification and Validation

1.4 Mathematical Notation

Matrices are written in bold uppercase letters (for example $\mathbf{A}$), while vectors (1D matrices) are written in bold lowercase letters, (for example $\mathbf{x}$) [1]. The zero matrix is a special type of matrix where each element is zero and is represented as $\mathbf{0}$ [2]. Individual entries in a matrix are written in lowercase letters with their row and column number, in that order, written as subscripts (for example a_{12}) [1]; these subscripts are sometimes separated by a comma for clarity (for example $a_{12,34}$) [3]. Two (or more) matrices with the same number of rows can be joined together to form an "augmented matrix", written as $(\mathbf{A}|\mathbf{B})$ [4].

[If symbols are used to show mathematical operations, these should be summarized here. In some cases the easiest way to summarize the notation is to point to a text or other source that explains the notation. —TPLT]

2 Introduction

Chemical equations are common ways of representing chemical reactions (as described in Physical System Description), but must be balanced for use in other calculations [5]. Therefore, it is useful to have a tool to automatically balance these chemical reactions for efficiency and accuracy.

The following section provides an overview of the Software Requirements Specification (SRS) for ChemCode. This section explains the purpose of this document, the scope of the requirements, the characteristics of the intended reader, and the organization of the document.

2.1 Purpose of Document

The primary purpose of this document is to record the requirements of the program for solving chemistry problems [Is this a good description? —SC]. Goals, assumptions, theoretical models, definitions, and other model derivation information are specified, allowing the reader to fully understand and verify the purpose and scientific basis of ChemCode. With the exception of system constraints, this SRS will remain abstract, describing what problem is being solved, but not how to solve it.

This document will be used as a starting point for subsequent development phases, including writing the design specification and the software Verification and Validation (VnV) Plan. The design document will show how the requirements are to be realized, including decisions on the numerical algorithms and programming environment. The VnV Plan will show the steps that will be used to increase confidence in the software documentation and the implementation. Although the SRS fits in a series of documents that follow the so-called "waterfall model", the actual development process is not constrained in any way. Even when the waterfall model is not followed, as Parnas and Clements point out in [6], the most logical way to present the documentation is still to "fake" a rational design process.

2.2 Scope of Requirements

The scope of the requirements includes determining if a given chemical equation is feasible (see Terminology and Definitions) and, if it is, balancing it using the smallest whole number coefficients possible, regardless of whether or not the chemical formulas in the equation are correct or if the reaction would take place in a real-world setting.

2.3 Characteristics of Intended Reader

Reviewers of this documentation should have an understanding of high-school level chemistry, namely stoichiometry (see Terminology and Definitions), and undergraduate Level 1 linear algebra, namely solving systems of linear equations. The end user of ChemCode, as described in User Characteristics, does not require knowledge of linear algebra, since relevant concepts will be implemented by the code and hidden from the user.

2.4 Organization of Document

The organization of this document is based on a template from [7], which is based on the template for an SRS for scientific computing software proposed by [8], [9], and [10]. The presentation follows the standard pattern of presenting goals, theories, definitions, and assumptions. For readers that would like a more bottom-up approach, they can start reading the Instance Models and trace back to find any additional information they require.

The Goal Statements are refined to the Theoretical Models, which are refined to the Instance Models.

3 General System Description

This section provides general information about the system. It identifies the interfaces between the system and its environment, describes the user characteristics and lists the system constraints. [This text can likely be borrowed verbatim. —TPLT]

[The purpose of this section is to provide general information about the system so the specific requirements in the next section will be easier to understand. The general system description section is designed to be changeable independent of changes to the functional requirements documented in the specific system description. The general system description provides a context for a family of related models. The general description can stay the same, while specific details are changed between family members. —TPLT]

3.1 System Context

[Your system context will include a figure that shows the abstract view of the software. Often in a scientific context, the program can be viewed abstractly following the design pattern of Inputs → Calculations → Outputs. The system context will therefore often follow this pattern. The user provides inputs, the system does the calculations, and then provides the outputs to the user. The figure should not show all of the inputs, just an abstract view of the main categories of inputs (like material properties, geometry, etc.). Likewise, the outputs should be presented from an abstract point of view. In some cases the diagram will show other external entities, besides the user. For instance, when the software product is a library, the user will be another software program, not an actual end user. If there are system constraints that the software must work with external libraries, these libraries can also be shown on the System Context diagram. They should only be named with a specific library name if this is required by the system constraint. —TPLT]

[For each of the entities in the system context diagram its responsibilities should be listed. Whenever possible the system should check for data quality, but for some cases the user will need to assume that responsibility. The list of responsibilites should be about the inputs and outputs only, and they should be abstract. Details should not be presented here. However, the information should not be so abstract as to just say “inputs” and “outputs”. A summarizing phrase can be used to characterize the inputs. For instance, saying “material

Figure 1: System Context

properties" provides some information, but it stays away from the detail of listing every required properties. —TPLT]

- User Responsibilities:
  - –
- ChemCode Responsibilities:
  - Detect data type mismatch, such as a string of characters instead of a floating point number
  - –

3.2 User Characteristics

The end user of ChemCode should have an understanding of high-school level chemistry, namely stoichiometry (see Terminology and Definitions).

3.3 System Constraints

[System constraints differ from other type of requirements because they limit the developers' options in the system design and they identify how the eventual system must fit into the world. This is the only place in the SRS where design decisions can be specified. That is, the quality requirement for abstraction is relaxed here. However, system constraints should only be included if they are truly required. —TPLT]

4 Specific System Description

This section first presents the problem description, which gives a high-level view of the problem to be solved. This is followed by the solution characteristics specification, which presents the assumptions, theories, definitions and finally the instance models.

4.1 Problem Description

ChemCode is intended to balance chemical equations (including ones with nonstoichiometric compounds; see Terminology and Definitions) so they can be useful for other computations [5]. Additionally, since molecules only exist in whole numbers (since dividing a molecule changes its composition into new types of molecules), the coefficients used to balance the equation must be whole numbers, and by convention should be as small as possible [5].

4.1.1 Terminology and Definitions

This subsection provides a list of terms that are used in the subsequent sections and their meaning, with the purpose of reducing ambiguity and making it easier to correctly understand the requirements:

- **Feasible**: (Referring to a chemical equation) able to be balanced [11].
- **Hydrate**: "A compound formed by the chemical combination of water and some other substance in a definite molecular ratio" [12].
- **Nonstoichiometric Compound**: "Any solid chemical compound in which the numbers of atoms of the elements present cannot be expressed as a ratio of small whole numbers" [13].
- **Stoichiometry**: "The calculation of the quantities of reactants or products in a chemical reaction using the relationships found in a balanced chemical equation" [5, p. 337].

4.1.2 Physical System Description

Chemical reactions are interactions between different types of matter that result in new substances being formed [5, p. 286]. These are represented using chemical equations, with the reactant(s)—the substance(s) present at the beginning of the reaction—on the left-hand side and the product(s)—the substance(s) formed by the reaction—on the right. An example of a chemical equation representing the combustion of methane in the presence of oxygen to form carbon dioxide and water vapour is shown below. The subscripts indicate how many of each element is present in the given chemical compound.

$$\mathrm{CH_4} + \mathrm{O_2} \rightarrow \mathrm{CO_2} + \mathrm{H_2O}$$

As mentioned in Problem Description, chemical formulas must be balanced to be useful in other reactions [5]. This means that there must be the same number of each element before and after the reaction takes place to satisfy the Law of Conservation of Mass (T2). Since changing the subscripts in a formula will change the type of molecule (for example, $\mathrm{O_3}$ is ozone, not oxygen), equations are balanced by introducing coefficients before each compound. If no coefficient is present, there is an implicit coefficient of "1". The above equation looks like this in its balanced form:

$$\mathrm{CH_4} + 2\,\mathrm{O_2} \rightarrow \mathrm{CO_2} + 2\,\mathrm{H_2O}$$

[Is this previous paragraph helpful/needed? —SC]

The physical system of ChemCode includes:

PS1: the products of a given chemical reaction;

PS2: the reactants of the same chemical reaction.

4.1.3 Goal Statements

Given a representation of a chemical equation, the goal statements of ChemCode are:

GS1: Determine if a given chemical reaction is feasible (see Terminology and Definitions).

GS2: Balance the inputted chemical equation with the smallest whole number coefficients possible.

4.2 Solution Characteristics Specification

The instance models that govern ChemCode are presented in Subsection 4.2.6. The information to understand the meaning of the instance models and their derivation is also presented, so that the instance models can be verified.

4.2.1 Assumptions

This section simplifies the original problem and helps in developing the theoretical model by filling in the missing information for the physical system. The numbers given in the square brackets refer to the theoretical model [T], general definition [GD], data definition [DD], instance model [IM], or likely change [LC], in which the respective assumption is used.

A1: For all inputted chemical equations, there is at most one more compound than element. (Through empirical analysis, chemical equations without this constraint lead to unexpected results that require a deeper analysis.)

A2: All inputted chemical formulas describe real chemical formulas. [IM1]

A3: All inputted chemical formulas are formatted following some set of agreed-upon conventions. [IM1, LC6]

A4: All inputted chemical formulas are simple; i.e., only consist of atomic symbols and subscripts. [IM1, LC1]

4.2.2 Theoretical Models

This section focuses on the general equations and laws that ChemCode is based on.

Number: T1

Label: Matrix equation

Equation: $\mathbf{Ax} = \mathbf{b}$

Description: The above equation gives the general form of the matrix equation where $\mathbf{A}$ is an $m \times n$ matrix, $\mathbf{b}$ is a vector in $\mathbb{R}^m$, and $\mathbf{x}$ is a vector in $\mathbb{R}^n$ with unknown values to be solved for [14].

Notes: For any $\mathbf{A}$ and $\mathbf{b}$, there is either zero, one, or infinite values for $\mathbf{x}$ [15].

Source: https://textbooks.math.gatech.edu/ila/matrix-equations.html [Is this a good enough source? —SC]

Ref. By: IM3

Derivation for T1: Not Applicable

Number: T2

Label: Law of Conservation of Mass

Equation: Not Applicable

Description: This law states that "matter can neither be created nor destroyed in a chemical reaction . . . but it may change forms to other substances" [5, p. 112].

Notes: None

Source: https://chem.libretexts.org/Courses/Anoka-Ramsey_Community_College/Introduction_to_Chemistry/03%3A_Matter_and_Energy/3.06%3A_Conservation_of_Mass

Ref. By: IM2

Derivation for T2: Not Applicable

4.2.3 General Definitions

There are no general definitions.

4.2.4 Data Definitions

There are no data definitions.

4.2.5 Data Types

[This section is optional. In many scientific computing programs it isn't necessary, since the inputs and outputs are straightforward types, like reals, integers, and sequences of reals and integers. However, for some problems it is very helpful to capture the type information. —TPLT]

[The data types are not derived; they are simply stated and used by other models. —TPLT]

[All data types must be used by at least one of the models. —TPLT]

[For the mathematical notation for expressing types, the recommendation is to use the notation of [16]. —TPLT]

This section collects and defines all the data types needed to document the models. [Modify the examples below for your problem, and add additional definitions as appropriate. —TPLT]

Type Name	Name for Type
Type Def	mathematical definition of the type
Description	description here
Sources	Citation here, if the type is borrowed from another source

4.2.6 Instance Models

This section transforms the problem defined in Problem Description into one which is expressed in mathematical terms. It uses concrete symbols defined in Section 4.2.4 to replace the abstract symbols in the models identified in Sections 4.2.2 and 4.2.3.

The goals GS1 and GS2 are solved by IM2 and IM3, respectively.

Number	IM1
Label	**Matrix representation of elements present in reaction**
Input	Two sequences of representations of chemical formulas; one for the reactants and one for the products.
Output	A matrix such that each entry indicates the amount of a given element in a given compound; each row corresponds to a different element and each column corresponds to a different compound. Positive entries indicate the compound is a reactant, negative entries indicate the compound is a product, and zeroes indicate the corresponding element is not present in the corresponding compound.
Description	Each chemical formula present in the inputted sequences are assumed to be valid (A2), be formatted correctly (A3), and only consist of atomic symbols and subscripts (A4).
Sources	[11]
Ref. By	IM2, IM3

Number	IM2
Label	**Determination of feasibility**
Input	**A** from IM1
Output	F if a row of **A** does not have a positive and a negative entry or if (**A**|**0**) has no solutions and T otherwise
Description	**A** is a matrix representing the amount of each element in each compound in a reaction from IM1. [Is this necessary? —SC] If a row of **A** only has either a positive or negative entry, then there is an element that is either a reactant or a product but not both, which breaks the Law of Conservation of Mass (T2). [Is this a good place for this? —SC] **0** is the zero matrix (see Mathematical Notation).
Sources	[11]
Ref. By	IM3

Number	IM3
Label	**Vector of compound coefficients in balanced equation**
Input	**A** from IM1 The input is constrained so that **A** is feasible (IM2). [Is this correct? —SC]
Output	**x**, such that it is the smallest solution to $\mathbf{Ax} = \mathbf{0}$ from T1 [Is this sufficient? —SC]
Description	**A** is a matrix representing the amount of each element in each compound in a reaction from IM1. [Is this necessary? —SC] **x** is a vector of coefficients for each compound in the reaction (in the order inputted) so that the equation is balanced. **0** is the zero matrix (see Mathematical Notation).
Sources	[11]
Ref. By	N/A

4.2.7 Input Data Constraints

Table 2 shows the data constraints on the input output variables. The column for physical constraints gives the physical limitations on the range of values that can be taken by the variable. The column for software constraints restricts the range of inputs to reasonable values. The software constraints will be helpful in the design stage for picking suitable algorithms. The constraints are conservative, to give the user of the model the flexibility to experiment with unusual situations. The column of typical values is intended to provide a feel for a common scenario. The uncertainty column provides an estimate of the confidence with which the physical quantities can be measured. This information would be part of the input if one were performing an uncertainty quantification exercise.

The specification parameters in Table 2 are listed in Table 3.

Table 2: Input Variables

Var	Physical Constraints	Software Constraints	Typical Value	Uncertainty
L	$L > 0$	$L_{\text{min}} \leq L \leq L_{\text{max}}$	1.5 m	10%

(*) [you might need to add some notes or clarifications —TPLT]

Table 3: Specification Parameter Values

Var	Value
L_{min}	0.1 m

4.2.8 Properties of a Correct Solution

A correct solution must exhibit [fill in the details —TPLT]. [These properties are in addition to the stated requirements. There is no need to repeat the requirements here. These additional properties may not exist for every problem. Examples include conservation laws (like conservation of energy or mass) and known constraints on outputs, which are usually summarized in tabular form. A sample table is shown in Table 4 —TPLT]

[This section is not for test cases or techniques for verification and validation. Those topics will be addressed in the Verification and Validation plan. —TPLT]

Table 4: Output Variables

Var	Physical Constraints
T_W	$T_{\text{init}} \leq T_W \leq T_C$ (by A??)

5 Requirements

This section provides the functional requirements, the business tasks that the software is expected to complete, and the nonfunctional requirements, the qualities that the software is expected to exhibit.

5.1 Functional Requirements

R1: Input a representation of a chemical equation.

R2: Convert the inputted equation to matrix form (from IM1).

R3: Determine if the inputted equation is feasible (see Terminology and Definitions; from IM2).

R4: If the inputted equation is feasible, balance the chemical equation with the smallest whole number coefficients possible (from IM3).

R5: If the inputted equation is infeasible, output a descriptive message. [Should this go before R4? —SC]

R6: If the inputted equation is feasible, output a balanced form of the equation in the same format as the input, using the coefficients from IM3.

5.2 Nonfunctional Requirements

NFR1: **Accuracy:** Computed solutions should have a relative error no greater than 0.1%, since the most finely-calibrated chemistry equipment has a relative error of 0.1-0.2% [17, 18]. This means that computed integers should be exact.

NFR2: **Understandability:** A new intended user (as described by User Characteristics) should be able to learn how to use ChemCode in an acceptable amount of time, as measured by the procedure in Section X of the Verification and Validation (VnV) Plan.

NFR3: **Usability:** An intended user (as described by User Characteristics) should find ChemCode easy to use, as measured by the procedure in Section X of the VnV Plan.

NFR4: **Maintainability:** The traceability between requirements, assumptions, theoretical models, general definitions, data definitions, instance models, likely changes, and unlikely changes is completely recorded in traceability matrices in the SRS.

NFR5: **Portability:** ChemCode should be able to run on systems with the corresponding programming language installed, including systems running on Windows or macOS. The tests from the VnV Plan should pass in these environments.

NFR6: **Verifiability:** ChemCode is tested following the VnV Plan.

6 Likely Changes

LC1: The user will be able to input complex chemical formulas, such as hydrates (see Terminology and Definitions) or those with polymers or isotopes (A4).

LC2: Given the amount of one substance in a reaction (in moles), ChemCode will be able to calculate the amount of every other substance in the reaction (also in moles).[1]

LC3: Given the amount of each reactant (in moles) in a reaction, ChemCode will be able to determine the limiting reactant(s).[1]

LC4: Given the amount of each [Is "each" a requirement? —SC] reactant (in moles) in a reaction, ChemCode will be able to calculate the theoretical yield of each product (also in moles).[1] This is dependent on LC3.

LC5: Given the amount of each [Is "each" a requirement? —SC] reactant (in moles) in a reaction, ChemCode will be able to calculate the amount of excess reactant(s) (also in moles).[1] This is dependent on LC3.

LC6: ChemCode will be able to parse valid but incorrectly-formatted chemical formulas inputted by the user and format them correctly when outputting them (A3).

LC7: For ease of use, the user will be able to enter the amounts required by LC2, LC3, LC4, and LC5 in terms of mass.[1]

LC8: ChemCode will be able to classify a chemical reaction as "combination (or synthesis), decomposition, combustion, single replacement, ... double replacement" [5, p. 301] or some combination of these.[1]

LC9: ChemCode will allow the user to input phase labels.[1]

LC10: ChemCode will be able to identify the phase labels for compounds, which will involve determining solubility.[1] This is dependent on LC9.

LC11: ChemCode will be able to identify when a reaction will not take place.[1] This is dependent on LC10.

[1]These examples of problems related to chemical equations were taken from [5].

7 Unlikely Changes

LC12: [Give the unlikely changes. The design can assume that the changes listed will not occur. —TPLT] [Is this supposed to be "LC" or "UC"? —SC]

8 Traceability Matrices and Graphs

The purpose of the traceability matrices is to provide easy references on what has to be additionally modified if a certain component is changed. Every time a component is changed, the items in the column of that component that are marked with an "X" may have to be modified as well. Table 5 shows the dependencies of theoretical models, general definitions, data definitions, and instance models with each other. Table 6 shows the dependencies of instance models, requirements, and data constraints on each other. Table 7 shows the dependencies of theoretical models, general definitions, data definitions, instance models, and likely changes on the assumptions.

[You will have to modify these tables for your problem. —TPLT]

[The traceability matrix is not generally symmetric. If GD1 uses A1, that means that GD1's derivation or presentation requires invocation of A1. A1 does not use GD1. A1 is "used by" GD1. —TPLT]

[The traceability matrix is challenging to maintain manually. Please do your best. In the future tools (like Drasil) will make this much easier. —TPLT]

	T??	T??	T??	GD??	GD??	DD??	DD??	DD??	DD??	IM??	IM??	IM??	IM
T??													
T??			X										
T??													
GD??													
GD??	X												
DD??				X									
DD??				X									
DD??													
DD??								X					
IM??					X	X	X				X		
IM??					X		X		X	X			X
IM??		X											
IM??		X	X				X	X	X		X		

Table 5: Traceability Matrix Showing the Connections Between Items of Different Sections

	IM??	IM??	IM??	IM??	4.2.7	R??	R??
IM??		X				X	X
IM??	X			X		X	X
IM??						X	X
IM??		X				X	X
R??							
R??						X	
R??					X		
R??	X	X				X	X
R??	X						
R??		X					
R??			X				
R??				X			
R??			X	X			
R??		X					
R??		X					

Table 6: Traceability Matrix Showing the Connections Between Requirements and Instance Models

	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??
T??	X																		
T??																			
T??																			
GD??		X																	
GD??			X	X	X	X													
DD??							X	X	X										
DD??			X	X						X									
DD??																			
DD??																			
IM??											X	X		X	X	X			X
IM??												X	X			X	X	X	
IM??														X					X
IM??													X					X	
LC??				X															
LC??								X											
LC??									X										
LC??											X								
LC??												X							
LC??															X				

Table 7: Traceability Matrix Showing the Connections Between Assumptions and Other Items

The purpose of the traceability graphs is also to provide easy references on what has to be additionally modified if a certain component is changed. The arrows in the graphs represent dependencies. The component at the tail of an arrow is depended on by the component at the head of that arrow. Therefore, if a component is changed, the components that it points to should also be changed. Figure **??** shows the dependencies of theoretical models, general definitions, data definitions, instance models, likely changes, and assumptions on each other. Figure **??** shows the dependencies of instance models, requirements, and data constraints on each other.

9 Development Plan

[This section is optional. It is used to explain the plan for developing the software. In particular, this section gives a list of the order in which the requirements will be implemented. In the context of a course this is where you can indicate which requirements will be implemented as part of the course, and which will be "faked" as future work. This section can be organized as a prioritized list of requirements, or it could should the requirements that will be implemented for "phase 1", "phase 2", etc. —TPLT]

10 Values of Auxiliary Constants

[Show the values of the symbolic parameters introduced in the report. —TPLT]

[The definition of the requirements will likely call for SYMBOLIC_CONSTANTS. Their values are defined in this section for easy maintenance. —TPLT]

[The value of FRACTION, for the Maintainability NFR would be given here. —TPLT]

References

[1] D. O'Sullivan and D. J. Unwin, "Appendix A: Notation, Matrices, and Matrix Mathematics," in *Geographic Information Analysis*, pp. 373–394, Hoboken, NJ, USA: Wiley, 2 ed., Mar. 2010.

[2] E. W. Weisstein, "Zero Matrix," 2023. Publisher: Wolfram Research, Inc.

[3] L. J. Latecki, "Matrices," 2018.

[4] M. Taboga, "Augmented matrix," 2021.

[5] L. Lund and Anoka-Ramsey Community College, *Introduction to Chemistry*. Cambridge and Coon Rapids, MN, USA: LibreTexts, Jan. 2023.

[6] D. L. Parnas and P. Clements, "A rational design process: How and why to fake it," *IEEE Transactions on Software Engineering*, vol. 12, no. 2, pp. 251–257, February 1986.

[7] W. S. Smith, "capTemplate," Sept. 2022.

[8] W. S. Smith and L. Lai, "A new requirements template for scientific computing," in *Proceedings of the First International Workshop on Situational Requirements Engineering Processes – Methods, Techniques and Tools to Support Situation-Specific Requirements Engineering Processes, SREP'05* (J. Ralyté, P. Ågerfalk, and N. Kraiem, eds.), (Paris, France), pp. 107–121, In conjunction with 13th IEEE International Requirements Engineering Conference, 2005.

[9] W. S. Smith, L. Lai, and R. Khedri, "Requirements analysis for engineering computation: A systematic approach for improving software reliability," *Reliable Computing, Special Issue on Reliable Engineering Computation*, vol. 13, pp. 83–107, February 2007.

[10] N. Koothoor, *A Document Driven Approach to Certifying Scientific Computing Software*. MASc thesis, McMaster University, Hamilton, ON, Canada, May 2013.

[11] I. Hamid, "Balancing Chemical Equations by Systems of Linear Equations," *Applied Mathematics*, vol. 10, pp. 521–526, July 2019.

[12] HarperCollins Publishers, "hydrate."

[13] The Editors of Encyclopaedia Britannica, "nonstoichiometric compound," Oct. 2010.

[14] D. Margalit and J. Rabinoff, "Matrix Equations," in *Interactive Linear Algebra*, pp. 44–52, Atlanta, GA, USA: Georgia Institute of Technology, June 2019.

[15] D. Zwick, "Math 2270 - Lecture 16: The Complete Solution to Ax = b," 2012.

[16] D. M. Hoffman and P. A. Strooper, *Software Design, Automated Testing, and Maintenance: A Practical Approach*. New York, NY, USA: International Thomson Computer Press, 1995.

[17] J. P. Walker, Advanced Instructional Systems, Inc. a, and North Carolina State University, "Volumetric Glassware," 2012.

[18] D. Godambe, "Measuring Volume."

[The following is not part of the template, just some things to consider when filing in the template. —TPLT]
[Grammar, flow and LaTeXadvice:

- For Mac users `*.DS_Store` should be in `.gitignore`
- LaTeX and formatting rules
  - Variables are italic, everything else not, includes subscripts (link to document)
    - Conventions
    - Watch out for implied multiplication
  - Use BibTeX
  - Use cross-referencing
- Grammar and writing rules
  - Acronyms expanded on first usage (not just in table of acronyms)
  - "In order to" should be "to"

—TPLT]
[Advice on using the template:

- Difference between physical and software constraints
- Properties of a correct solution means *additional* properties, not a restating of the requirements (may be "not applicable" for your problem). If you have a table of output constraints, then these are properties of a correct solution.
- Assumptions have to be invoked somewhere
- "Referenced by" implies that there is an explicit reference
- Think of traceability matrix, list of assumption invocations and list of reference by fields as automatically generatable
- If you say the format of the output (plot, table etc), then your requirement could be more abstract

—TPLT]

Appendix — Reflection

The information in this section will be used to evaluate the team members on the graduate attribute of Lifelong Learning. Please answer the following questions:

1. What knowledge and skills will the team collectively need to acquire to successfully complete this capstone project? Examples of possible knowledge to acquire include domain specific knowledge from the domain of your application, or software engineering knowledge, mechatronics knowledge or computer science knowledge. Skills may be related to technology, or writing, or presentation, or team management, etc. You should look to identify at least one item for each team member.

2. For each of the knowledge areas and skills identified in the previous question, what are at least two approaches to acquiring the knowledge or mastering the skill? Of the identified approaches, which will each team member pursue, and why did they make this choice?