Companion Cube Calculator

Geneva Smith

October 2, 2017

Contents

Table 1: **Revision History**

Date	Version	Notes
September 26, 2017	1.1	Completed the first draft of the General System Description (Section 3)
September 25, 2017	1.0	Completed the first draft of the Introduction (Section 2)

1 Reference Material

This section records information for easy reference.

1.1 Table of Units

The tasks performed by the C^3 tool are unitless.

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
b^x	-	Base number b raised to the power of variable x
$D(x)$	-	Domain of x
$f(X)$	-	A function of a variable set X
$R(f(X))$	-	Range of $f(X)$
$\mathbb{R}$	-	The set of real numbers
x^n	-	Variable x raised to the power of constant n

1.3 Abbreviations and Acronyms

Text	Description
A	Assumption
CRPG	Computer Role-Playing Game
DD	Data Definition
GD	General Definition
GS	Goal Statement
GUI	Graphical User Interface
IM	Instance Model
LC	Likely Change
NPC	Non-Player Character
PS	Physical System Description
R	Requirement
SRS	Software Requirements Specification
C^3	Companion Cube Calculator
T	Theoretical Model

2   Introduction

This document is an SRS for the Companion Cube Calculator (C^3), a mathematical tool which determines the range of a user-specified function given the domains of the function's variables. This tool is being developed to aid in the specification and refinement of GLaDOS, an emotion engine for Non-Player Characters (NPCs) in Computer Role-Playing Games (CRPG) as described by Smith (2017).

2.1   Purpose of Document

This document outlines the requirements identified for the development of the C^3 tool, including the product goals, product scope, and the mathematical models driving the design. It also describes the mathematical assumptions, theories, and models used to create the tool. The purpose of documenting this information is to aid in future use, maintenance, and development of the C^3 tool.

This document is intended for two reader types – those who wish to use the tool and those who wish to expand the tool. Even though the Companion Cube Calculator was created to aid in the development of a specific system (GLaDOS), the tested functions do not have any specific units. This means that it can be used for any function that exists in the domain of real numbers ($\mathbb{R}$). Therefore, this document can be used by a user who wishes to use the C^3 tool to determine the range of a specified mathematical function. Since the initial development of the C^3 tool will be limited to arithmetic operators, this document includes information that will be useful to a developer looking to expand the abilities of the C^3 with additional mathematical models such as trigonometry to suit their project.

2.2   Scope of Requirements

The Companion Cube Calculator calculates the mathematical range of a user-defined function using the defined variable domains for that function. In the cases where a variable domain is not provided, it will be assumed that the range is $(-\infty, \infty)$. For the initial version of this product, the mathematical operations allowed in an function will be limited to the four arithmetic operators ($+$, $-$, $\times$, $\div$, exponents). Future iterations can expand this list to include trigonometric functions (sin, cos, tan, arcsin, arccos, arctan), partial functions, and other useful operations and functions.

The purpose of this product is to aid in the design and tuning of the GLaDOS architecture, a specialized game engine enables game designers to create Non-Player Characters (NPCs) that react to their environment by using models of emotion from psychology. One module in the architecture converts information from the environment into an internal representation that directs an NPC's decision-making. This task requires the specification of numerical functions with multiple variables which must be normalized to a range of $[-1, 1]$. In order to normalize the engine's calculation, the range of the function must be known. Currently, the encoded functions are not well-informed by observation or scientific research and must be subjected to an iterative process in order to address this problem. An automated method of calculating the range of a proposed function will make this process faster and less error-prone.

Although this product is being designed for a specific project, the concepts involved can be used in similar projects because it does not contain any project-specific concepts or models.

2.3   Characteristics of Intended Reader

The intended reader of this document must understand elementary algebra, especially interval arithmetic, in the domain of real numbers ($\mathbb{R}$). They must also have an understanding of domain and range with respect to a function in order to understand the outputs of the product and how it relates to its inputs.

2.4   Organization of Document

This document begins by describing the general description of the Companion Cube Calculator tool (Section 3), which includes the system context, constraints, and the intended users' characteristics. This is followed by a description of the problem to be solved, including the models and assumptions that are used to solve it (Section 4). This description is followed by the functional and non-functional requirements of the C^3 tool (Section 5) and the potential changes that will be made to them in future iterations (Section 6). The last section in this document, traceability matrices and graphs, visually describes the dependencies between different document components (Section 7). This information should be used by making changes to the decisions made for this version of the tool.

3   General System Description

This section identifies the interfaces between the system and its environment, describes the user characteristics and lists the system constraints.

3.1   System Context

The Companion Cube Calculator tool is a stand-alone application for calculating the range of a user-provided function given the known domains of the function's component variables. Therefore it is independent and self-contained with respect to external organizations, products, and technologies.

The user interface of the C^3 tool facilitates communication between the product and the user, and must contain the ability to exchange user inputs and system outputs. In general, the user is responsible for ensuring that they have provided a semantically correct function and that their inputs do not contain mathematical operations that are not implemented in the tool. The Companion Cube Calculator is responsible for providing an information exchange interface between itself and the user, performing mathematical calculations, and detecting syntactic errors in the system inputs.

- User Responsibilities:

- – Provide an function from the domain of real numbers ($\mathbb{R}$) that only contains mathematical operations that are recognized by the system
- – Provide the mathematical domains of the function's variables if they are known
- – Ensure that the function given to the tool does not contain semantic errors
- – Ensure that the products outputs are semantically correct with respect to the target application

- • Companion Cube Calculator Responsibilities:

  - – Allow the user to enter their function and known variable domains as system inputs
  - – For variables with unknown domains, provide default values to be used in calculations
  - – Detect data type mismatch, such as a string of characters instead of a floating point number
  - – Detect bracket mismatches
  - – Calculate and output the mathematical range corresponding to the user's inputs; if a result cannot be calculated, communicate this to the user

3.2 User Characteristics

The end user of Companion Cube Calculator is also the intended reader (Section 2.3) of this document.

3.3 System Constraints

The use of a Graphical User Interface (GUI) is useful for the C^3 tool because it can help the user visualize their function during design time and debugging. This has implications on the types of languages that this tool will be developed in and target platform choices. If a GUI is included in the C^3 design, only Windows platforms will be supported in the initial release.

4 Specific System Description

This section first presents the problem description, which gives a high-level view of the motivation behind the Companion Cube Calculator. This is followed by the solution characteristics specification, presenting the assumptions, theories, definitions, and instance models, which are grounded in interval arithmetic.

4.1 Problem Description

The GLaDOS architecture is a specialized game engine that enables game designers to create Non-Player Characters (NPCs) that react to their environment by using models of emotion from

psychology. A key component of the architecture is the primary appraisal module, where inputs from the environment are converted into emotion values. The functions currently used for this task are not well informed by any scientific research, and will be incrementally revised as more information is collected. These new functions are required to conform to the same mathematical range as the currently implemented functions. Ensuring that this requirement is maintained over consecutive iterations of the functions will become tedious and time consuming, distracting from the main task of developing more rigorous and explainable emotion functions.

The Companion Cube Calculator is a tool for calculating the range of a mathematical function based on its input variables, which will help make the function design process faster and less error-prone when compared to doing it by hand.

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:

- Closed interval: A bounded set that includes its end points; this is expressed $[a, b]$

- Closed interval method: A mathematical method for determining the local maximum and minimum values of a function $f(x)$ using a closed domain interval $[a, b]$

- Connected set: a set of values that does not contain any disjoint members

- Domain ($D(x)$): the set of values for a variable x that are valid in $f(X)$

- Extended real interval: the set of all real numbers that also contains $\pm\infty$

- Open interval: A bounded set of values that does not contain its end points; this is expressed (a, b)

- Range ($R(f(X))$): the set of values that are produced by a function $f(X)$ with a set of input variables X

- Real interval: a closed, connected set of real numbers

4.1.2 Physical System Description

The Companion Cube Calculator tool does not have a physical system component because it exists independently of the context of $f(X)$.

4.1.3 Goal Statements

Given an function and a set of variables, the goal statements are:

GS1: Calculate $R(f(X))$.

4.2 Solution Characteristics Specification

The instance models that govern Companion Cube Calculator are presented in Subsection 4.2.5. 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: $D(x)$ is a closed, real interval.

A2: $R(f(X))$ is a closed, real interval.

A3: The mathematical operators used in $f(X)$ can be found in the set $\{+, -, \times, \div, b^x, x^n\}$.

4.2.2 Theoretical Models

This section focuses on the general equations and laws from interval arithmetic that the Companion Cube Calculator is based on. The models presented here are for real intervals, which can be constrained to closed, real intervals using DD1 to satisfy A1 and A2. Additional models exist for interval arithmetic, but only the models named in A3 are mentioned.

Number	T1
Label	**Addition on Real Intervals**
Equation	$X + Y = \{x + y \mid x \in X \wedge y \in Y\}$
Description	The summation interval of two real intervals X and Y is equal to the set of sums for each pairwise element x and y.
Source	Hickey et al. (2001)
Ref. By	IM1

5

Number	T2
Label	**Subtraction on Real Intervals**
Equation	$X - Y = \{x - y \mid x \in X \wedge y \in Y\}$
Description	The difference interval of two real intervals X and Y is equal to the set of differences for each pairwise element x and y.
Source	Hickey et al. (2001)
Ref. By	IM2

Number	T3
Label	**Multiplication on Real Intervals**
Equation	$X \times Y = \{x \times y \mid x \in X \wedge y \in Y\}$
Description	The product interval of two real intervals X and Y is equal to the set of products for each pairwise element x and y.
Source	Hickey et al. (2001)
Ref. By	IM3

Number	T4
Label	**Division on Real Intervals**
Equation	$X \div Y = \{z \mid \exists x \in X, y \in Y \wedge y \neq 0, z = x \div y\}$
Description	The quotient interval of two real intervals X and Y is equal to the set of quotients for each pairwise element x and y where $y \neq 0$. If $0 \in Y$, the quotient is not a real interval which contradicts A2.
Source	Hickey et al. (2001)
Ref. By	IM4, IM5

Number	T5
Label	**Real Interval Exponents on a Constant Base Number**
Equation	$b^X = \{b^x | x \in X \land b > 1\}$
Description	The product interval of a constant number $b > 1$ raised to the power of a real interval X is equal to the set of products b^x for each interval element x.
Source	https://en.wikipedia.org/wiki/Interval_arithmetic
Ref. By	DD2, IM6

Number	T6
Label	**Constant Exponents on a Real Interval Base Number**
Equations	$$X^n = \begin{cases} \{x^n | x \in X \land n \in \mathbb{N}\} & n \text{ is odd} \\ \{[x_1^n, ..., x_z^n] | x \in X \land n \in \mathbb{N}\} & n \text{ is even} \land x_z \geq x_{z+1} \land x_1 > 0 \\ \{[x_z^n, ..., x_1^n] | x \in X \land n \in \mathbb{N}\} & n \text{ is even} \land x_z < x_{z+1} \land x_Z < 0 \\ \{[0, ..., max\{x_1^n, x_z^n\}] | x \in X \land n \in \mathbb{N}\} & \text{ELSE} \end{cases}$$
Description	The product interval of a real interval X raised to the power of a constant, natural number n depends on the properties of X and n: • When n is an odd number, the product interval is equal to the set of products x^n for each element x • When n is even, the function is monotonically decreasing, and $x_1 > 0$, the product interval is equal to the set of products x^n for each element x • When n is even, the function is monotonically increasing, and $x_Z < 0$, the product interval is equal to the set of products x^n for each element x and ordered in the opposite direction of the original interval X • Otherwise, the product interval is bounded by 0 and the maximum product of the first and last elements in the interval set
Source	https://en.wikipedia.org/wiki/Interval_arithmetic
Ref. By	DD3, IM7

4.2.3 General Definitions

No additional information is required to build the data definitions.

4.2.4 Data Definitions

This section collects and defines all the data needed to build the instance models. No dimensions are required because the tasks performed by Companion Cube Calculator are unitless. These definitions are used to constrain the theoretical models to such that thay can be translated into instance models on closed, real intervals to satisfy A1 and A2.

Number	DD1
Label	**Representation of a Closed, Real Interval**
Symbol	$[x, y]$
SI Units	-
Equation	-
Description	The definition $[x, y]$ represents a closed, real interval (A1, A2) with endpoints x and y.
Sources	-
Ref. By	IM1, IM2, IM3, IM4, IM5, IM6, IM7

Number	DD2
Label	**Representation of a Base Number**
Symbol	b
SI Units	-
Equation	-
Description	A constant real number ($\mathbb{R}$) such that $b > 1$ (T6).
Sources	-
Ref. By	IM6

Number	DD3
Label	**Representation of a Power**
Symbol	n
SI Units	-
Equation	-
Description	A constant natural number ($\mathbb{N}$) (T7).
Sources	-
Ref. By	IM7

4.2.5   Instance Models

This section transforms the problem defined in Section 4.1 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 goal GS1 is solved by recursively combining instance models IM1, IM2, IM3, IM4, IM5, IM6, and IM7. For example, if the user provided an equation of the form $a + b \times c$, it is decomposed into $a + B$ where $B = b \times c$. This approach is identical to a hand-written evaluation because a mechanized approach must explicitly state the intermediary steps, which are commonly omitted by humans, in order to complete the computational task.

These models can also be used as a specification for the closed interval method because of the closed, real interval assumption on $D(x)$ (A1).

Number	IM1
Label	**Addition of closed, real intervals**
Input	$[x_1, y_1]$, $[x_2, y_2]$ from DD1
Output	$[x_{sum}, y_{sum}]$ such that $x_{sum} = x_1 + x_2$, $y_{sum} = y_1 + y_2$, $[x_{sum}, y_{sum}]$ is a closed, real interval (A2)
Description	$[x_1, y_1]$ and $[x_2, y_2]$ are the domains of two variables in the user's input function $f(X)$. The combined boundary values x_{sum} and y_{sum} are determined using basic arithmetic addition. The result, $[x_{sum}, y_{sum}]$, is guaranteed to be a closed, real interval (A2) due to the closed, real interval constraint on $[x_1, y_1]$ and $[x_2, y_2]$ (A1).
Sources	Hickey et al. (2001)
Ref. By	-

Number	IM2
Label	**Subtraction of closed, real intervals**
Input	$[x_1, y_1]$, $[x_2, y_2]$ from DD1
Output	$[x_{sub}, y_{sub}]$ such that $x_{sub} = x_1 - x_2$, $y_{sub} = y_1 - y_2$, $[x_{sub}, y_{sub}]$ is a closed, real interval (A2)
Description	$[x_1, y_1]$ and $[x_2, y_2]$ are the domains of two variables in the user's input function $f(X)$. The combined boundary values x_{sub} and y_{sub} are determined using basic arithmetic subtraction. The result, $[x_{sub}, y_{sub}]$, is guaranteed to be a closed, real interval (A2) due to the closed, real interval constraint on $[x_1, y_1]$ and $[x_2, y_2]$ (A1).
Sources	Hickey et al. (2001)
Ref. By	-

Number	IM3
Label	**Multiplication of closed, real intervals**
Input	$[x_1, y_1]$, $[x_2, y_2]$ from DD1
Output	$[x_{mul}, y_{mul}]$ such that $x_{mul} = \min(M)$, $y_{mul} = \max(M)$, where $M = \{x_1 \times x_2, x_1 \times y_2, y_1 \times x_2, y_1 \times y_2\}$ $[x_{mul}, y_{mul}]$ is a closed, real interval (A2)
Description	$[x_1, y_1]$ and $[x_2, y_2]$ are the domains of two variables in the user's input function $f(X)$. The combined boundary value x_{mul} is determined by taking the minimum of all possible products calculated from the sets $[x_1, y_1]$ and $[x_2, y_2]$, where $[x_1, y_1]$ is the multiplier set and $[x_2, y_2]$ is the multiplicand set The combined boundary value y_{mul} is determined by taking the maximum of all possible products calculated from the sets $[x_1, y_1]$ and $[x_2, y_2]$, where $[x_1, y_1]$ is the multiplier set and $[x_2, y_2]$ is the multiplicand set The result, $[x_{mul}, y_{mul}]$, is guaranteed to be a closed, real interval (A2) due to the closed, real interval constraint on $[x_1, y_1]$ and $[x_2, y_2]$ (A1).
Sources	Hickey et al. (2001)
Ref. By	-

Number	IM4
Label	**Division of closed, real intervals (Divisor is a positive interval)**
Input	$[x_1, y_1]$, $[x_2, y_2]$ from DD1 where $0 \le x_2 \le y_2$ and $x_2 \ne 0$
Output	$[x_{div}, y_{div}]$ such that $x_{div} = x_1 \div y_2, \quad y_{div} = y_1 \div x_2 \quad 0 < x_1 \le y_1$ $x_{div} = 0, \qquad\quad y_{div} = y_1 \div x_2 \quad x_1 = 0 \wedge x_1 \le y_1$ $x_{div} = x_1 \div x_2, \quad y_{div} = y_1 \div x_2 \quad x_1 < 0 < y_1$ $x_{div} = x_1 \div x_2, \quad y_{div} = 0 \qquad\quad y_1 = 0, x_1 \le y_1$ $x_{div} = x_1 \div x_2, \quad y_{div} = y_1 \div y_2 \quad x_1 \le y_1 < 0$ $[x_{div}, y_{div}]$ is a closed, real interval (A2)
Description	$[x_1, y_1]$ and $[x_2, y_2]$ are the domains of two variables in the user's input function $f(X)$. The boundary values x_{div} and y_{div} are determined by the monotonicity of the interval represented by $[x_1, y_1]$ and whether or not that interval contains 0. The result, $[x_{div}, y_{div}]$, is guaranteed to be a closed, real interval (A2) due to the closed, real interval constraint on $[x_1, y_1]$ and $[x_2, y_2]$ (A1).
Sources	Hickey et al. (2001)
Ref. By	-

Number	IM5
Label	**Division of closed, real intervals (Divisor is a negative interval)**
Input	$[x_1, y_1]$, $[x_2, y_2]$ from DD1 where $x_2 \leq y_2 \leq 0$ and $y_2 \neq 0$
Output	$[x_{div}, y_{div}]$ such that $\quad x_{div} = y_1 \div y_2, \quad y_{div} = x_1 \div x_2 \quad 0 < x_1 \leq y_1$ $\quad x_{div} = y_1 \div y_2, \quad y_{div} = 0 \quad\quad\quad x_1 = 0 \wedge x_1 \leq y_1$ $\quad x_{div} = y_1 \div y_2, \quad y_{div} = x_1 \div y_2 \quad x_1 < 0 < y_1$ $\quad x_{div} = 0, \quad\quad\quad y_{div} = x_1 \div y_2 \quad y_1 = 0, x_1 \leq y_1$ $\quad x_{div} = y_1 \div x_2, \quad y_{div} = x_1 \div y_2 \quad x_1 \leq y_1 < 0$ $[x_{div}, y_{div}]$ is a closed, real interval (A2)
Description	$[x_1, y_1]$ and $[x_2, y_2]$ are the domains of two variables in the user's input function $f(X)$. The boundary values x_{div} and y_{div} are determined by the monotonicity of the interval represented by $[x_1, y_1]$ and whether or not that interval contains 0. The result, $[x_{div}, y_{div}]$, is guaranteed to be a closed, real interval (A2) due to the closed, real interval constraint on $[x_1, y_1]$ and $[x_2, y_2]$ (A1).
Sources	Hickey et al. (2001)
Ref. By	-

Number	IM6
Label	**Closed, real intervals as Exponents**
Input	$[x, y]$ from DD1, b from DD2
Output	$[x_{exp}, y_{exp}]$ such that $x_{exp} = b^x$, $y_{exp} = b^y$ $[x_{exp}, y_{exp}]$ is a closed, real interval (A2)
Description	$[x, y]$ is the domain of one variable in the user's input function $f(X)$ which is used as an exponent on a constant base number b, which is subject to the data constraints of DD2. The combined boundary values x_{exp} and y_{exp} are determined using basic arithmetic exponents. The result, $[x_{exp}, y_{exp}]$, is guaranteed to be a closed, real interval (A2) due to the closed, real interval constraint on $[x, y]$ and the data constraint on b (A1).
Sources	https://en.wikipedia.org/wiki/Interval_arithmetic
Ref. By	-

Number	IM7
Label	**Powers of closed, real intervals**
Input	$[x, y]$ from DD1, n from DD3
Output	$[x_{pow}, y_{pow}]$ such that $\quad x_{pow} = x^n, \quad y_{pow} = y^n \qquad\qquad n$ is odd $\quad x_{pow} = x^n, \quad y_{pow} = y^n \qquad\qquad n$ is even $\wedge (0 \le x < y)$ $\quad x_{pow} = y^n, \quad y_{pow} = x^n \qquad\qquad n$ is even $\wedge (0 \ge x > y)$ $\quad x_{pow} = 0, \quad\;\; y_{pow} = \max(x^n, y^n) \quad$ ELSE $[x_{pow}, y_{pow}]$ is a closed, real interval (A2)
Description	$[x, y]$ is the domain of one variable in the user's input function $f(X)$ raised to a constant power n, which is subject to the data constraints of DD3. The combined boundary values x_{pow} and y_{pow} are determined by the divisibility of n by two and the monotonicity of $[x, y]$: • For odd values of n and any interval $[x, y]$, and even values of n when $[x, y]$ is monotonically increasing, $[x_{pow}, y_{pow}]$ are determined by simply raising each of x and y to the power of n • For even values of n and $[x, y]$ is monotonically decreasing, the boundary values of x and y are reversed before being raised to the power of n • For all other cases, the interval is bounded by 0 and the maximum of the input bounds raised to the power of n The result, $[x_{exp}, y_{exp}]$, is guaranteed to be a closed, real interval (A2) due to the closed, real interval constraint on $[x, y]$ and the data constraint on n (A1).
Sources	https://en.wikipedia.org/wiki/Interval_arithmetic
Ref. By	-

4.2.6 Data Constraints

Tables 2 and 4 show the data constraints on the input and output variables, respectively. 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 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	$L > 0$	$L_{\min} \leq L \leq L_{\max}$	1.5 m	10%

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

Table 3: Specification Parameter Values

Var	Value
$L_{\min}$	0.1 m

Table 4: Output Variables

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

4.2.7 Properties of a Correct Solution

A correct solution must exhibit [fill in the details —SS]

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: [Requirements for the inputs that are supplied by the user. This information has to be explicit. —SS]

R2: [It isn't always required, but often echoing the inputs as part of the output is a good idea. —SS]

R3: [Calculation related requirements. —SS]

R4: [Verification related requirements. —SS]

R5: [Output related requirements. —SS]

5.2 Nonfunctional Requirements

[List your nonfunctional requirements. You may consider using a fit criterion to make them verifiable. —SS]

6 Likely Changes

LC1: Open, real intervals are supported, which will also remove all restrictions on division (A1, A2)

LC2: Determine $D(x)$ for any $x \in X$ if it is not part of the input set (A1, A2)

LC3: Addition of the trigonometric functions $\sin(x)$ and $\cos(x)$ (A3)

LC4: Addition of operators and trigonometric functions whose range includes $\pm\infty$ (A1, A2, A3)

7 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 7 shows the dependencies of theoretical models, data definitions, and instance models with each other. Table 6 shows the dependencies of instance models, requirements, and data constraints on each other. Table 5 shows the dependencies of theoretical models, data definitions, instance models, and likely changes on the assumptions. Since there is only one goal statement (GS1) in this specification, it can be assumed that changing the goal will impact every item contained within this document.

	A1	A2	A3
T1			
T2			
T3			
T4			
T5			
T6			
DD1	X	X	
DD2			
DD3			
IM1	X	X	
IM2	X	X	
IM3	X	X	
IM4	X	X	
IM5	X	X	
IM6	X	X	
IM7	X	X	
LC1	X	X	
LC2	X	X	
LC3	X		X
LC4	X	X	X

Table 5: Traceability Matrix Showing the Connections Between Assumptions and Other Items

	IM1	IM2	IM3	IM4	IM5	IM6	IM7	4.2.6	R??	R??
IM1										
IM2										
IM3										
IM4										
IM5										
IM6										
IM7										
R2										
R??										
R??										
R??										
R??										
R4										
R??										
R??										

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

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.

	T1	T2	T3	T4	T5	T6	DD1	DD2	DD3	IM1	IM2	IM3	IM4	IM5	IM6	IM7
T1																
T2																
T3																
T4																
T5																
T6																
DD1																
DD2					X											
DD3						X										
IM1	X						X									
IM2		X					X									
IM3			X				X									
IM4				X			X									
IM5				X			X									
IM6					X		X	X								
IM7						X	X		X							

Table 7: Traceability Matrix Showing the Connections Between Items of Different Sections

References

T. Hickey, Q. Ju, and M. H. Van Emden. Interval arithmetic: From principles to implementation. *J. ACM*, 48(5):1038–1068, September 2001. URL http://doi.acm.org/10.1145/502102.502106.

Geneva Smith. GLaDOS: Integrating Emotion-Based Behaviours into Non-Player Characters in Computer Role-Playing Games. M.A.Sc. (Software Engineering) Thesis, McMaster University, April 2017. URL http://hdl.handle.net/11375/21369.

8 Appendix

[Your report may require an appendix. For instance, this is a good point to show the values of the symbolic parameters introduced in the report. —SS]

8.1 Symbolic Parameters

$\mathbb{R}$ Domain of real numbers