Software Requirements Specification for Game of continuous life: An advanced version of the Conway's Game of Life

Baptiste Pignier

January 31, 2025

Contents

Revision History

Date	Version	Notes
Date 1	1.0	Notes
Date 2	1.1	Notes

[This template is intended for use by CAS 741. For CAS 741 the template should be used exactly as given, except the Reflection Appendix can be deleted. For the capstone course it is a source of ideas, but shouldn't be followed exactly. The exception is the reflection appendix. All capstone SRS documents should have a reflection appendix. —TPLT]

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
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	$\mathrm{m^2}$	coil surface area
A_{in}	$\mathrm{m^2}$	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
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
Game of continuous life	
TM	Theoretical Model

[This SRS template is based on Smith and Lai (2005); Smith et al. (2007); Smith and Koothoor (2016). It will get you started. You should not modify the section headings, without first discussing the change with the course instructor. Modification means you are not following the template, which loses some of the advantage of a template, especially standardization. Although the bits shown below do not include type information, you may need to add this information for your problem. If you are unsure, please can ask the instructor. —TPLT]

[Feel free to change the appearance of the report by modifying the LaTeX commands. —TPLT]

[This template document assumes that a single program is being documented. If you are documenting a family of models, you should start with a commonality analysis. A separate template is provided for this. For program families you should look at Smith (2006); Smith et al. (2017). Single family member programs are often programs based on a single physical model. General purpose tools are usually documented as a family. Families of physical models also come up. —TPLT]

[The SRS is not generally written, or read, sequentially. The SRS is a reference document. It is generally read in an ad hoc order, as the need arises. For writing an SRS, and for reading one for the first time, the suggested order of sections is:

- Goal Statement
- Instance Models
- Requirements
- Introduction
- Specific System Description

—TPLT]

[Guiding principles for the SRS document:

- Do not repeat the same information at the same abstraction level. If information is repeated, the repetition should be at a different abstraction level. For instance, there will be overlap between the scope section and the assumptions, but the scope section will not go into as much detail as the assumptions section.

—TPLT]

[The template description comments should be disabled before submitting this document for grading. —TPLT]

[You can borrow any wording from the text given in the template. It is part of the template, and not considered an instance of academic integrity. Of course, you need to cite the source of the template. —TPLT]

[When the documentation is done, it should be possible to trace back to the source of every piece of information. Some information will come from external sources, like terminology. Other information will be derived, like General Definitions. —TPLT]

[An SRS document should have the following qualities: unambiguous, consistent, complete, validatable, abstract and traceable. —TPLT]

[The overall goal of the SRS is that someone that meets the Characteristics of the Intended Reader (Section 2.3) can learn, understand and verify the captured domain knowledge. They should not have to trust the authors of the SRS on any statements. They should be able to independently verify/derive every statement made. —TPLT]

2 Introduction

Conway's Game of Life is the result of applying simple rules of evolution. It allows the observation of an emerging phenomenon called cellular automaton. These cellular automata can be made more complex by making the rules of evolution more complex to observe more complex structure.

This section will develop the characteristics of the SRS document, more than those of the project itself.

2.1 Purpose of Document

The purpose of this Software Requirements Specification (SRS) document is to define the functional and non-functional requirements of the project in a clear and structured manner. This document serves as a communication tool between stakeholders, including clients, developers, and project managers, to ensure a shared understanding of the project's scope and objectives.

2.2 Scope of Requirements

This project aims to develop a simulation in a simplistic artificial environment. As such, the simulated elements will not be governed by any other physical law than those implemented by the environment. No assumptions will be made compared to a classic physical simulation model. The presentation of assumptions is therefore not applicable to this project.

2.3 Characteristics of Intended Reader

Reviewers of this document should have undergraduate mathematical knowledge as well as a good understanding of the physical approximations that will be implemented. He will also need to be aware of the different libraries used for this project.

2.4 Organization of Document

This document is structured as follows. Section 3 covers general information about the project. Section 4 explains this information in detail. Section 5 covers the project require-

ments. Section 6 lists the likely changes and Section 7 lists the unlikely changes. Section 8 presents the requirements traceability. Section 9 presents the project development plan.

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.

3.1 System Context

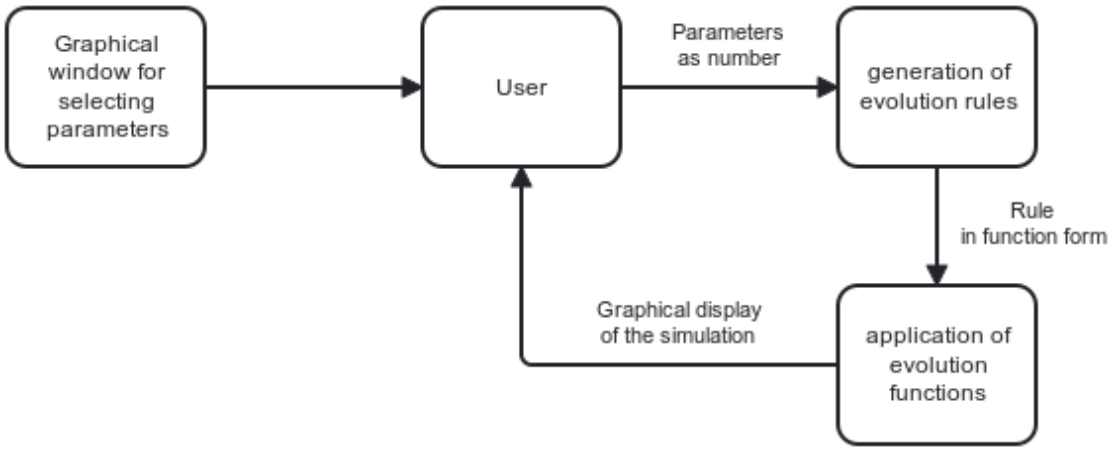

Figure 1: System Context

- User Responsibilities:
  - The user will have no responsibility for the data he transmits to the system since they will be included in a range of values that the system will provide him.
- Game of continuous life Responsibilities:
  - Provide the user with a simulation that faithfully matches the data the user has given to the system.

This software is intended for educational and/or artistic use. As such, its components must not be used for uses requiring security or critical requirements.

3.2 User Characteristics

The user does not need any special knowledge to use the program. All useful information will be provided by the graphical window for selecting parameters.

3.3 System Constraints

No system constraints are identified for this project, as long as its requirements are met.

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

Game of continuous life aims to demonstrate the emergence of macroscopic behavior and structure from microscopic generation and evolution rules.

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:

- A cell : A portion of a display area that contains one or more values
- The grid : Grid cell size as small as possible. The dimensions of this grid are as large as desired.
- Neighborhood : Area around a cell, consisting of the cells in that area. This area is delimited by a neighborhood function
- Neighborhood function: Function that takes a cell as input and returns the cells in its neighborhood.
- Growth function: Function that takes a cell as input and modifies the value(s) contained in the cell based on its neighborhood.
- Structure: Finite set of cells in prolonged interaction

4.1.2 Physical System Description

The physical system of Game of continuous life, as shown in Figure ?, includes the following elements:

PS1 : Cells

PS2 : Growth function

PS3 : Neighborhood function

4.1.3 Goal Statements

Given the input paramters, the goal statements are:

GS1: Create the Growth function

GS2: Create the Neighborhood function

GS3: Apply iterative evolution

GS4: Define interesting evolution

GS5: Find some interesting evolution

4.2 Solution Characteristics Specification

The instance models that govern Game of continuous life 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

Since the project does not aim to model a physical system by simplifying it, no assumptions of physical simplification will be made. Therefore, this section is not relevant.

4.2.2 Theoretical Models

This section focuses on the general equations and laws that Game of continuous life is based on.

RefName: **TM:CTH**

Label: Convolution theorem

Equation: $(f * g)[n] = \sum_{m=-\inf}^{\inf} f[m]g[n-m]$

Description: This formula will be used for applying the neighborhood formula on the grid.

Notes: None.

Source: https://en.wikipedia.org/wiki/Convolution#Discrete_convolution

Ref. By: None

Preconditions for TM:CTH: None

Derivation for TM:CTH: Not Applicable

RefName: **TM:GFT**

Label: Gauss function

Equation: $G(x) = A \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$

Description: This formula will be used for applying the growth and neighborhood function on a cell

Notes: None.

Source: https://en.wikipedia.org/wiki/Gaussian_function

Ref. By: None

Preconditions for TM:GFT: None

Derivation for TM:GFT: Not Applicable

RefName: **TM:RDE**

Label: Reaction-Diffusion equation

Equation: $\frac{\partial u}{\partial t} = D_u \nabla^2 u + f(u, v)$

Description: This theory will be used to model the interactions that cells will have with their environment.

Notes: None.

Source: https://en.wikipedia.org/wiki/Reaction-diffusion_system#Two-component_reaction-diffusion_equations

Ref. By: None

Preconditions for TM:RDE: None

Derivation for TM:RDE: Not Applicable

4.2.3 General Definitions

This section collects the laws and equations that will be used in building the instance models.

Number	GD1
Label	**Reaction Diffusion**
SI Units	N/A
Equation	$\frac{\partial u}{\partial t} = D_u \nabla^2 u - uv^2 + F(1-u)$ $\frac{\partial v}{\partial t} = D_v \nabla^2 v + uv^2 - (F+k)v$
Description	We will use the Gray-Scott model to implement the diffusion reaction. This model models the interactions between two concentration values. It will be used only if the simulation includes multiple channels.
Source	https://www.karlsims.com/rd.html
Ref. By	None

Number	GD2
Label	**Growth function**
SI Units	N/A
Equation	$G(x) = -1 + 2\exp\left(-(\frac{(x-\mu}{\sigma})^2\right)$
Description	We'll use a Gaussian function to start. We want an amplitude between -1 and 1 to simplify the calculations.
Source	None
Ref. By	None

4.2.4 Data Definitions

This project does not use any physical data or quantities. This section is not relevant.

4.2.5 Instance Models

[The motivation for this section is to reduce the problem defined in "Physical System Description" (Section 4.1.2) to one expressed in mathematical terms. The IMs are built by refining the TMs and/or GDs. This section should remain abstract. The SRS should specify the requirements without considering the implementation. —TPLT]

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 goals [reference your goals —TPLT] are solved by [reference your instance models —TPLT]. [other details, with cross-references where appropriate. —TPLT] [Modify the examples below for your problem, and add additional models as appropriate. —TPLT]

Number	IM1
Label	**Energy balance on water to find T_W**
Input	m_W, C_W, h_C, A_C, h_P, A_P, t_{final}, T_C, T_{init}, $T_P(t)$ from IM**??** The input is constrained so that $T_{\text{init}} \leq T_C$ (A**??**)
Output	$T_W(t)$, $0 \leq t \leq t_{\text{final}}$, such that $\frac{dT_W}{dt} = \frac{1}{\tau_W}[(T_C - T_W(t)) + \eta(T_P(t) - T_W(t))]$, $T_W(0) = T_P(0) = T_{\text{init}}$ (A**??**) and $T_P(t)$ from IM**??**
Description	T_W is the water temperature (°C). T_P is the PCM temperature (°C). T_C is the coil temperature (°C). $\tau_W = \frac{m_W C_W}{h_C A_C}$ is a constant (s). $\eta = \frac{h_P A_P}{h_C A_C}$ is a constant (dimensionless). The above equation applies as long as the water is in liquid form, $0 < T_W < 100^o$C, where 0^oC and 100^oC are the melting and boiling points of water, respectively (A**??**, A**??**).
Sources	Citation here
Ref. By	IM**??**

4.2.6 Input Data Constraints

TODO

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 4.

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 4: Specification Parameter Values

Var	Value
L_{min}	0.1 m

4.2.7 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 6 —TPLT]

Table 6: Output Variables

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

[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]

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 data will be given by the user before the simulation starts and must be modifiable during the simulation.

R2: The program must calculate step by step the results of applying the growth function to each cell.

R3: The output should allow the user to correctly distinguish between concentrations of value.

5.2 Nonfunctional Requirements

[List your nonfunctional requirements. You may consider using a fit criterion to make them verifiable. —TPLT] [The goal is for the nonfunctional requirements to be unambiguous, abstract and verifiable. This isn't easy to show succinctly, so a good strategy may be to give a "high level" view of the requirement, but allow for the details to be covered in the Verification and Validation document. —TPLT] [An absolute requirement on a quality of the system is rarely needed. For instance, an accuracy of 0.0101 % is likely fine, even if the requirement is for 0.01 % accuracy. Therefore, the emphasis will often be more on describing now well the quality is achieved, through experimentation, and possibly theory, rather than meeting some bar that was defined a priori. —TPLT] [You do not need an entry for correctness in your NFRs. The purpose of the SRS is to record the requirements that need to be satisfied for correctness. Any statement of correctness would just be redundant. Rather than discuss correctness, you can characterize how far away from the correct (true) solution you are allowed to be. This is discussed under accuracy. —TPLT]

NFR1: **Accuracy** The accuracy of the computed solutions should meet the level needed for correct visualization

NFR2: This software must be able to be used without any special skills or knowledge.

NFR3: **Maintainability** The effort required to make any of the likely changes listed for Game of continuous life should be less than ϵ of the original development time.

NFR4: **Portability** This program should run on all operating systems supported by the libraries used.

5.3 Rationale

It is clear that significant mathematical and physics skills are required to simulate a biologically accurate cellular automaton development environment. Not having such skills, I limit myself to a basic but correct and useful implementation of a cellular automaton simulation.

6 Likely Changes

LC1: Implementing multiple value channels

LC2: Stochastic equation implementation for the evolution rules

7 Unlikely Changes

None

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 9 shows the dependencies of theoretical models, general definitions, data definitions, and instance models with each other. Table 10 shows the dependencies of instance models, requirements, and data constraints on each other. Table 8 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]

	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??
TM??	X																		
TM??																			
TM??																			
GD2		X																	
GD??			X	X	X	X													
DD??							X	X	X										
DD??			X	X						X									
DD??																			
DD??																			
IM1											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 8: Traceability Matrix Showing the Connections Between Assumptions and Other Items

	TM??	TM??	TM??	GD2	GD??	DD??	DD??	DD??	DD??	IM1	IM??	IM??
TM??												
TM??			X									
TM??												
GD2												
GD??	X											
DD??				X								
DD??				X								
DD??												
DD??								X				
IM1					X	X	X				X	
IM??					X		X		X	X		
IM??		X										
IM??		X	X				X	X	X		X	

Table 9: Traceability Matrix Showing the Connections Between Items of Different Sections

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 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]

	IM1	IM??	IM??	IM??	4.2.6	R??	R??
IM1		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 10: Traceability Matrix Showing the Connections Between Requirements and Instance Models

References

W. Spencer Smith. Systematic development of requirements documentation for general purpose scientific computing software. In *Proceedings of the 14th IEEE International Requirements Engineering Conference, RE 2006*, pages 209–218, Minneapolis / St. Paul, Minnesota, 2006. URL http://www.ifi.unizh.ch/req/events/RE06/.

W. Spencer Smith and Nirmitha Koothoor. A document-driven method for certifying scientific computing software for use in nuclear safety analysis. *Nuclear Engineering and Technology*, 48(2):404–418, April 2016. ISSN 1738-5733. doi: http://dx.doi.org/10.1016/j.net.2015.11.008. URL http://www.sciencedirect.com/science/article/pii/S1738573315002582.

W. Spencer Smith and Lei Lai. A new requirements template for scientific computing. In J. Ralyté, P. Ågerfalk, and N. Kraiem, editors, *Proceedings of the First International Workshop on Situational Requirements Engineering Processes – Methods, Techniques and Tools to Support Situation-Specific Requirements Engineering Processes, SREP'05*, pages 107–121, Paris, France, 2005. In conjunction with 13th IEEE International Requirements Engineering Conference.

W. Spencer Smith, Lei Lai, and Ridha Khedri. Requirements analysis for engineering computation: A systematic approach for improving software reliability. *Reliable Computing, Special Issue on Reliable Engineering Computation*, 13(1):83–107, February 2007.

W. Spencer Smith, John McCutchan, and Jacques Carette. Commonality analysis for a family of material models. Technical Report CAS-17-01-SS, McMaster University, Department of Computing and Software, 2017.