Software Requirements Specification for a Library of ODE Solvers (LODES)

Paul Aoanan

September 22, 2017

Contents

1 Reference Material

This section records information for easy reference.

1.1 Table of Units

There are no physical units used in this SRS.

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. —SS]

Table 1: **Revision History**

Date	Version	Notes
Date 1	1.0	Notes
Date 2	1.1	Notes

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
LODES	Library of ODE Solvers
T	Theoretical Model

[Add any other abbreviations or acronyms that you add —SS]

2 Introduction

In physical sciences, mathematical models are derived from scientific models to represent a real world phenomenon through formal mathematical constructs.

Scientific models in the study of radioactivity, carbon decay, and Newton's Law of Cooling involve the use of ordinary differential equations (ODEs).

Known elementary techniques of solving ODEs in the discrete domain use the linear approximation method wherein the solution is based upon assuming or "approximating" the slope of the tangent line from one reference point to the next until the target point has been reached.

The following section provides an overview of the Software Requirements Specification (SRS) for a program family of ODE solvers. The developed program will be called Library of ODE Solvers (LODES). This section explains the purpose of this document, the scope of the system, and the characteristics of the intended readers and the organization of the document.

2.1 Purpose of Document

The main purpose of this document is to formally describe the known well-known methods of solving ODEs. The goals and mathematical models used in the LODES code are provided with an emphasis on explicitly identifying assumptions, constraints, and unambiguous definitions.

This document contains the required functionality of the LODES software library as well the non-functional requirements that the software has to meet. This document governs the subsequent software development activities, including writing the design specification, code, and the software verification and validation plan and execution.

2.2 Scope of Requirements

The scope of requirements is limited to the analysis of the library of ODE solvers. Given the appropriate inputs, each program in LODES is intended to find the solution to an ODE problem.

2.3 Characteristics of Intended Reader

Reviewers of this document should have an elementary understanding of ordinary differential equations and numerical methods, as typically covered in first and second year Calculus courses. The users of LODES can have a lower level expertise, as explained in Section 3.2.

2.4 Organization of Document

The organization of this document follows the template for an SRS for scientific computing software proposed by !!!INSERT CITATION HERE!!!. 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 in Section 4.2.5 and trace back to find any additional information they require. The instance models provide the methods to solve Ordinary Differential Equations (ODEs).

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 Potential System Contexts

Figure ?? shows the system context. A circle represents an external entity outside the software, the user in this case. A rectangle represents the software system itself (LODES). Arrows are used to show the data flow between the system and its environment.

Programs in LODES are used inside a wrapper program. The external interaction is through program calls. The solution to the ODE is the output of the function. The responsibilities of the user and the system are as follows:

- User Responsibilities:
  - Provide the correct program call, while adhering to conventions of the function's prototype
  - Provide the input details of the ODE to be solved, ensuring no errors in data entry
  - Declaration of the ODE method to be used in solving the ODE
- LODES Responsibilities:
  - Detect an improper input, such as invalid characters in the ODE statement and incomplete input arguments
  - Detect a data type mismatch where applicable, such as a string of characters in a floating point argument
  - Calculate the solution to the ODE problem

3.2 User Characteristics

The end user of LODES should have an understanding of undergraduate Level 1 Calculus.

3.3 System Constraints

There are no system constraints.

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

LODES is a software library developed to be provide a means to analyze and compare different well-known ODE solvers. It can be used to investigate and determine which of the solvers is the most accurate method (the method which produces the least error in their scientific computing implementation).

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:

- Initial condition: A "starting point" of known value that exists in the domain of the solution
- Step size: The measure of arbitrary positive value from the point n to n+1 of the domain
- Derivative:
-
- Numerical Analysis: An branch of mathematics and computer science wherein the solutions are numerical approximatations taking into account the errors involved in the process.

4.1.2 Physical System Description

This section does not apply.

4.1.3 Goal Statements

Given the ODE and initial values, the goal statements are:

GS1: Find the solution to an ordinary differential equation initial value problem.

4.2 Solution Characteristics Specification

The instance models that govern LODES 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 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: Higher order ODEs can be represented by a system of first order ODEs.

A2: The only form of ODE that is relevant to the application of the program is a first order ODE.

A3: The user will be responsible for deconstructing higher order ODEs into a system of first order ODEs.

A4: The only ODE considered is a linear ODE.

4.2.2 Theoretical Models

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

Number	T1
Label	**Ordinary Differential Equation**
Equation	$y^{(n)} = F(x, y, y', ..., y^{(n-1)}); y(x_0) = y_0$
Description	The above model gives the definition of an ordinary differential equation. A differential equation is an equation, where the unknown is a function and both the function and its derivatives may appear in the equation.
Source	http://users.math.msu.edu/users/gnagy/teaching/ode.pdf
Ref. By	GD??

4.2.3 General Definitions

This section collects the laws and equations that will be used in deriving the data definitions, which in turn are used to build the instance models. [Some projects may not have any content for this section, but the section heading should be kept. —SS] [Modify the examples below for your problem, and add additional definitions as appropriate. —SS]

Number	GD1
Label	**Newton's law of cooling**
SI Units	$\mathrm{W\,m^{-2}}$
Equation	$q(t) = h\Delta T(t)$
Description	Newton's law of cooling describes convective cooling from a surface. The law is stated as: the rate of heat loss from a body is proportional to the difference in temperatures between the body and its surroundings. $q(t)$ is the thermal flux ($\mathrm{W\,m^{-2}}$). h is the heat transfer coefficient, assumed independent of T (A**??**) ($\mathrm{W\,m^{-2}\,{}^{\circ}C^{-1}}$). $\Delta T(t) = T(t) - T_{\mathrm{env}}(t)$ is the time-dependent thermal gradient between the environment and the object (°C).
Source	(**?**, p. 8)
Ref. By	DD1, DD**??**

Detailed derivation of simplified rate of change of temperature

[This may be necessary when the necessary information does not fit in the description field. —SS]

4.2.4 Data Definitions

This section collects and defines all the data needed to build the instance models. The dimension of each quantity is also given. [Modify the examples below for your problem, and add additional definitions as appropriate. —SS]

Number	DD1
Label	**Heat flux out of coil**
Symbol	q_C
SI Units	$\mathrm{W\,m^{-2}}$
Equation	$q_C(t) = h_C(T_C - T_W(t))$, over area A_C
Description	T_C is the temperature of the coil (°C). T_W is the temperature of the water (°C). The heat flux out of the coil, q_C ($\mathrm{W\,m^{-2}}$), is found by assuming that Newton's Law of Cooling applies (A**??**). This law (GD1) is used on the surface of the coil, which has area A_C ($\mathrm{m^2}$) and heat transfer coefficient h_C ($\mathrm{W\,m^{-2}\,°C^{-1}}$). This equation assumes that the temperature of the coil is constant over time (A**??**) and that it does not vary along the length of the coil (A**??**).
Sources	**?**
Ref. By	IM1

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

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	?
Ref. By	IM??

Derivation of ...

[May be necessary to include this subsection in some cases. —SS]

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.

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

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%

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: [Give the likely changes, with a reference to the related assumption (aref), as appropriate. —SS]

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 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. —SS]

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	T??	T??	GD1	GD??	DD1	DD??	DD??	DD??	IM1	IM??	IM??	IM??
T1													
T??			X										
T??													
GD1													
GD??	X												
DD1				X									
DD??				X									
DD??													
DD??								X					
IM1					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

	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		
R2	X	X				X	X
R??	X						
R??		X					
R??			X				
R??				X			
R4			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??
T1	X																		
T??																			
T??																			
GD1		X																	
GD??			X	X	X	X													
DD1							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 7: Traceability Matrix Showing the Connections Between Assumptions and Other Items

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

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