Module Interface Specification for Damped Harmonic Oscillator

Muhammad Waqar Ul Hassan Awan

March 18, 2024

1 Revision History

Date	Version	Notes
18 Mar, 2024	1.0	Initial MIS documentation

2 Symbols, Abbreviations and Acronyms

See SRS Documentation at SRS

Contents

3 Introduction

The following document details the Module Interface Specifications for a software project aimed at modeling and analyzing harmonic oscillators, providing a tool for educational, research, and industrial applications to observe the effect of damping on oscillating bodies.

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

4 Notation

The structure of the MIS for modules comes from Software Design, Automated Testing, and Maintenance: A Practical Approach, with the addition that template modules have been adapted from Fundamentals of Software Engineering. The mathematical notation comes from Chapter 3 of Software Design, Automated Testing, and Maintenance: A Practical Approach. For instance, the symbol := is used for a multiple assignment statement and conditional rules follow the form $(c_1 \Rightarrow r_1|c_2 \Rightarrow r_2|...|c_n \Rightarrow r_n)$.

The following table summarizes the primitive data types used by Damped Harmonic Oscillator.

Data Type	**Notation**	**Description**
character	char	a single symbol or digit
real	$\mathbb{R}$	any number in (-∞, ∞)

The specification of Damped Harmonic Oscillator uses some derived data types: sequences, strings, and tuples. Sequences are lists filled with elements of the same data type. Strings are sequences of characters. Tuples contain a list of values, potentially of different types. In addition, Damped Harmonic Oscillator uses functions, which are defined by the data types of their inputs and outputs. Local functions are described by giving their type signature followed by their specification.

5 Module Decomposition

The following table is taken directly from the Module Guide document for this project.

Level 1	**Level 2**
Hardware-Hiding	Hardware-Hiding Module
Behaviour-Hiding	Data Input Module Parameter Input Module Calculation Engine Module Output Module User Interface Module
Software Decision	Plotting and Visualization Module ODE Solver Module

Table 1: Module Hierarchy

6 MIS of Data Input

Data input module Manages the input of raw data into the system, potentially including data validation and preprocessing.

6.1 Module

DataInput

6.2 Uses

N/A

6.3 Syntax

6.3.1 Exported Constants

None specified.

6.3.2 Exported Access Programs

Name	In	Out	Exceptions
receiveInputs	parameterValues: Map	Boolean	InvalidInputException
validateInputs	parameterValues: Map	Boolean	ValidationError

6.4 Semantics

6.4.1 State Variables

m: $\mathbb{R}$
k: $\mathbb{R}$
c: $\mathbb{R}$
x_0: $\mathbb{R}$
v_0: $\mathbb{R}$
g: $\mathbb{R}$
F(t): t: $\mathbb{R}$ $\rightarrow$ res: $\mathbb{R}$
x_i: $\mathbb{R}$ v_i: $\mathbb{R}$ a_i: $\mathbb{R}$

6.4.2 Environment Variables

keyboard: Captures input from the system's keyboard device.

6.4.3 Assumptions

- Inputs are provided in a structured format that the module can parse and validate.
- All necessary parameters (mass m, spring constant k, damping coefficient c, initial displacement x0, and initial velocity v0) are included in the input data.

6.4.4 Access Routine Semantics

receiveInputs(parameterValues):

- transition: Validates the provided parameterValues and, if valid, updates inputData.
- output: Returns true if inputs are valid and successfully stored; otherwise, false.
- exception: InvalidInputException: Triggered if parameterValues contains undefined or non-numeric values.

validateInputs(parameterValues):

- transition: N/A
- output: Returns true if all values in parameterValues meet the physical and software constraints; otherwise, false.
- exception: ValidationError: Triggered if any value in parameterValues does not adhere to the constraints defined in Section 4.2.6 of the SRS document.

6.4.5 Local Functions

- isNumeric: Checks if a given input is a numeric value.
- meetsConstraints: Verifies if a given value adheres to specified constraints for each parameter (as defined in SRS Section 4.2.6).

7 MIS of Parameter Input Module

Parameter input module Processes and validates the parameters specific to the damped harmonic oscillator simulation, such as mass, spring constant, damping coefficients, etc.

7.1 Module

ParameterInput

7.2 Uses

Data Input Module

7.3 Syntax

7.3.1 Exported Constants

None specified.

7.3.2 Exported Access Programs

Name	In	Out	Exceptions
setParameter	paramName:String,value: Real	Boolean	InvalidParameterException
getParameter	paramName:String	Real	ParameterNotFoundException

7.4 Semantics

7.4.1 State Variables

m: $\mathbb{R}$
k: $\mathbb{R}$
c: $\mathbb{R}$
x_0: $\mathbb{R}$
v_0: $\mathbb{R}$
F(t): t: $\mathbb{R} \rightarrow$ res: $\mathbb{R}$

7.4.2 Environment Variables

N/A

7.4.3 Assumptions

- The module is called after all necessary parameters are initialized to default values.
- Parameters passed to setParameter are within the valid range as defined in the SRS and are the correct type (Real numbers for most parameters).

7.4.4 Access Routine Semantics

setParameter(paramName, value):

- transition: Updates the state variable corresponding to paramName with value.
- output: Returns true if the parameter is successfully updated; otherwise, false.
- exception: InvalidParameterException: Triggered if value is out of the acceptable range or paramName does not correspond to a valid parameter name.

getParameter(paramName):

- transition: N/A
- output: Returns the value of the state variable corresponding to paramName.
- exception: ParameterNotFoundException: Triggered if paramName does not correspond to a valid parameter name.

7.4.5 Local Functions

- validateParameter: Ensures that a given parameter value is within the acceptable range and type for the parameter.

8 MIS of Calculation Engine Module

Calculation engine module Performs the core calculations based on input parameters and models, generating the dynamics of the oscillator over time.

8.1 Module

CalcEngine

8.2 Uses

Data Input, Parameter Input modules

8.3 Syntax

8.3.1 Exported Constants

None specified.

8.3.2 Exported Access Programs

Name	In	Out	Exceptions
calculateMotion	time: Real	MotionResults	CalculationException
getResult	paramName: String	Real	ResultNotFoundException

8.4 Semantics

8.4.1 State Variables

timeSeries: Array$< \mathbb{R} >$
displacementSeries: Array$< \mathbb{R} >$

velocitySeries: Array< $\mathbb{R}$ >
accelerationSeries: Array< $\mathbb{R}$ >

8.4.2 Environment Variables

N/A

8.4.3 Assumptions

- Parameters required for the calculations (m, k, c, x0, v0) have been correctly set and validated before invoking calculateMotion.
- The time parameter for the calculateMotion method is within the simulation time range defined by the user.

8.4.4 Access Routine Semantics

calculateMotion(time):

- transition: Uses the current set parameters (m, k, c, x0, v0) and external forces, if any, to calculate the motion of the damped harmonic oscillator at the specified time point. Updates timeSeries, displacementSeries, velocitySeries, and accelerationSeries with the calculated values.
- output: MotionResults, a structure containing time, displacement, velocity, and acceleration at the specified time.
- exception: CalculationException: Triggered if the calculation fails due to invalid parameters or numerical errors.

getResult(paramName):

- transition: N/A
- output: Returns the series (Array< $\mathbb{R}$ >) corresponding to the requested parameter name ("displacement", "velocity", or "acceleration").
- exception: ResultNotFoundException: Triggered if paramName does not correspond to any calculated result series.

8.4.5 Local Functions

- solveDifferentialEquation: Applies the appropriate numerical method to solve the differential equation governing the damped harmonic oscillator's motion based on the parameters and external forces.

9 MIS of Output Module

Output module transforms calculation results into various output formats (e.g., graphical, tex- tual) for user interpretation.

9.1 Module

Output

9.2 Uses

Calculation Engine, User Interface modules

9.3 Syntax

9.3.1 Exported Constants

None specified.

9.3.2 Exported Access Programs

Name	In	Out	Exceptions
displayResults	results:MotionResults,outputType: String	boolean	DisplayException

9.4 Semantics

9.4.1 State Variables

N/A

9.4.2 Environment Variables

- **screen:** A device or interface for displaying results to the user.

9.4.3 Assumptions

- The results object contains valid motion data (displacement, velocity, acceleration) for the damped harmonic oscillator.
- The outputType specifies the format in which the user prefers to view the results (e.g., ”graph”, ”table”).

9.4.4 Access Routine Semantics

displayResults(results, outputType):

- transition: N/A
- output: Displays the motion results in the specified format (outputType) on the screen.
- exception: DisplayException: Triggered if the specified outputType is not supported or if there is an error in rendering the results.

9.4.5 Local Functions

- **formatResults:** Formats the motion results into the specified output type (e.g., converting data into a graphical representation or a table format for display purposes).

10 MIS of User Interface Module

User Interface module provides the graphical or command-line interface through which users interact with the software, including inputting parameters and viewing results.

10.1 Module

UserInterface

10.2 Uses

N/A

10.3 Syntax

10.3.1 Exported Constants

None specified.

10.3.2 Exported Access Programs

Name	In	Out	Exceptions
displayMainMenu	None	None	UIException
getInput	prompt: String	String	InputFormatException
displayResults	results: MotionResults	None	DisplayException
displayError	message: String	None	None
confirmAction	confirmationMessage: String	Boolean	None

10.4 Semantics

10.4.1 State Variables

N/A

10.4.2 Environment Variables

- **keyboard:** Interface for capturing user input.
- **screen:** Interface for displaying information to the user.

10.4.3 Assumptions

- Users interact with the software through a graphical or command-line interface that provides clear prompts and displays.
- Input errors and exceptions are handled gracefully, informing the user of the issue without causing crashes or unhandled errors.

10.4.4 Access Routine Semantics

displayMainMenu():

- transition: N/A
- output: Displays the main menu options to the screen, allowing users to choose actions like parameter entry, simulation start, or results viewing.
- exception: UIException: Triggered if there is an issue displaying the main menu.

getInput(prompt):

- transition: N/A
- output: Returns the user input collected in response to the displayed prompt.
- exception: InputFormatException: Triggered if the input provided by the user does not match the expected format for the prompt.

displayResults(results):

- transition: N/A
- output: Formats and displays the simulation results contained in results to the screen.
- exception: DisplayException: Triggered if there is an issue displaying the results.

displayError(message):

- transition: N/A
- output: Displays an error message to the screen, informing the user of any issues encountered during operation.
- exception: N/A

confirmAction(confirmationMessage):

- transition: N/A
- output: Displays a confirmationMessage to the user and returns true if the user confirms the action; otherwise, returns false.
- exception: N/A

10.4.5 Local Functions

N/A

11 MIS of Plotting and Visualization Module

Plotting and Visualization module creates charts, graphs, and animations to visually depict the behavior of the damped harmonic oscillator based on the simulation outcomes.

11.1 Module

PlotVis

11.2 Uses

Output Module

11.3 Syntax

11.3.1 Exported Constants

None specified.

11.3.2 Exported Access Programs

Name	In	Out	Exceptions
plotResults	results:MotionResults, plotType: String	Boolean	PlottingException

11.4 Semantics

11.4.1 State Variables

N/A

11.4.2 Environment Variables

- **displayDevice:** A device or interface for visual output, such as a monitor for displaying plots.

11.4.3 Assumptions

- The results provided contain valid and complete data necessary for plotting and report generation.
- The plotType specifies the desired type of plot (e.g., "displacement vs. time", "velocity vs. time").

11.4.4 Access Routine Semantics

plotResults(results, plotType):

- transition: Generates and displays a plot based on the results and the specified plotType.
- output: Returns true if the plot is successfully generated and displayed; otherwise, false.
- exception: PlottingException: Triggered if there is an error in generating or displaying the plot, such as an unsupported plotType or a plotting library error.

11.4.5 Local Functions

- **formatResultsForPlotting:** Formats the motion results into a structure suitable for plotting based on the selected plotType.

12 MIS of ODE Solver Module

ODE solver module provides the computational backbone for accurately solving the equations of mo- tion under various conditions and damping models.

12.1 Module

ODESolver

12.2 Uses

Calculation Engine module

12.3 Syntax

12.3.1 Exported Constants

None specified.

12.3.2 Exported Access Programs

Name	In	Out	Exceptions
solveODE	equation: ODE, initialConditions: Conditions, timeSpan: TimeSpan	Solution	ODESolverException

12.4 Semantics

12.4.1 State Variables

- **equation:** ODE - The ordinary differential equation to be solved, representing the motion of the damped harmonic oscillator.
- **initialConditions:** Conditions - Initial state of the system, including initial displacement and velocity.
- **timeSpan:** TimeSpan - The range of time over which to solve the ODE.

12.4.2 Environment Variables

N/A

12.4.3 Assumptions

- The ODE provided correctly models the damped harmonic oscillator based on parameters set in the ParameterInput module.
- Initial conditions and time span are appropriate for the problem being solved and have been validated before calling solveODE.

12.4.4 Access Routine Semantics

solveODE(equation, initialConditions, timeSpan):

- transition: Uses numerical methods to solve the provided equation over the specified timeSpan starting from initialConditions.

- output: Solution, an object containing the time series and corresponding solution values (displacement and velocity) for the ODE.
- exception: ODESolverException: Triggered if the solver fails to converge to a solution or encounters numerical instabilities.

12.4.5 Local Functions

- **numericalIntegrationMethod:** A numerical solver that approximates the solution of the ODE based on the input equation, initial conditions, and time span.

13 Appendix

Message ID	Error Message
FileNotFound	Error: The specified file could not be found.
InvalidParameter	Error: One or more parameters provided are invalid or out of range.
ParameterNotFound	Error: The requested parameter does not exist.
CalculationError	Error: An error occurred during the calculation process.
DisplayError	Error: Failed to display the requested information.
SaveError	Error: The results could not be saved to the specified location.
InputFormatException	Error: The provided input does not match the expected format.
PlottingError	Error: An error occurred while generating the plot.
ReportGenerationError	Error: Failed to generate the report in the specified format.
ODESolverError	Error: Failed to solve the ODE due to numerical instabilities or convergence issues.
UIError	Error: An unexpected error occurred in the User Interface.

Table 2: Possible Exceptions