# Module Interface Specification for Solar Water Heating Systems Incorporating Phase Change Material

Brooks MacLachlan

July 29, 2016

## Contents

# 1 Introduction

The following document details the Module Interface Specifications for the implemented modules in a program simulation Solar Water Heating System with Phase Change Material. It is intended to ease navigation through the program for design and maintenance purposes. Complementary documents include the System Requirement Specifications and Module Guide.

# 2 Notation

The following table summarizes the primitive data types used by SWHS. SWHSalso uses some derived data types: arrays ,strings, and structures. Arrays are lists filled with elements of the same data type. Strings are arrays of characters. Structures contain pairs of keys and values, where keys are unique variable names used to identify their corresponding value, and values can be any data type.

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

# 3 Module Decomposition

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

| **Level 1** | **Level 2** |
|---|---|
| Hardware-Hiding Module | |
| Behaviour-Hiding Module | Input Format Module<br>Input Parameters Module<br>Input Verification Module<br>Output Format Module<br>Output Verification Module<br>Temperature ODEs Module<br>Energy Equations Module<br>Control Module |
| Software Decision Module | Sequence Data Structure Module<br>ODE Solver Module<br>Plotting Module |

Table 1: Module Hierarchy

# 4 MIS of Control Module

## 4.1 Module

main

## 4.2 Uses

parameters (5), load_params (6), verify_params (7), temperature (8), energy (9), verify_output (10), plot (11), output (12)

## 4.3 Syntax

### 4.3.1 Exported Access Programs

| Name | In | Out | Exceptions |
|---|---|---|---|
| main | string | - | - |

## 4.4 Semantics

### 4.4.1 State Variables

$filename$: string<br>
$time$: array of reals<br>
$tempW$: array of reals<br>
$tempP$: array of reals<br>
$eW$: array of reals<br>
$eP$: array of reals<br>
$eTot$: array of reals

### 4.4.2 Environment Variables

$win$: 2D array of pixels displayed on the screen

### 4.4.3 Access Routine Semantics

main($s$):
- transition: Fills the $time$, $tempW$, $tempP$, $eW$, $eP$, and $eTot$ lists with the simulation results. Modifies the screen environment.
- exception: none

# 5 MIS of Input Parameters Module

## 5.1 Module

parameters

## 5.2 Uses

N/A

## 5.3 Syntax

### 5.3.1 Exported Data Types

parameters := structure

### 5.3.2 Exported Access Programs

N/A

## 5.4 Semantics

### 5.4.1 State Variables

$L$: real
$diam$: real
$Vp$: real
$Ap$: real
$rho\_p$: real
$Tmelt$: real
$C\_ps$: real
$C\_pl$: real
$Hf$: real
$Ac$: real
$Tc$: real
$rho_w$: real
$C\_w$: real
$hc$: real
$hp$: real
$Tinit$: real
$tstep$: real
$tfinal$: real
$AbsTol$: real
$RelTol$: real
$ConsTol$: real
$Vt$: real
$Mw$: real
$tau\_w$: real
$eta$: real
$Mp$: real
$tau\_ps$: real
$tau\_pl$: real
$Epmelt\_init$: real
$Ep\_melt3$: real

$Mw\_noPCM$: real
$tau\_w\_no\_PCM$: real

### 5.4.2 Access Routine Semantics

N/A

# 6 MIS of Input Format Module

## 6.1 Module

load_params

## 6.2 Uses

parameters (5)

## 6.3 Syntax

## 6.4 Exported Access Programs

| Name | In | Out | Exceptions |
|---|---|---|---|
| load_params | string | parameters | - |

## 6.5 Semantics

### 6.5.1 State Variables

$filename$: string
$params$: parameters

### 6.5.2 Assumptions

The input string corresponds to an existing filename in the current directory. The input file is formatted correctly.

### 6.5.3 Access Routine Semantics

load_params($s$):
transition: Fills the parameters structure with the input parameters specified in the input file, and with other parameters calculated from the input parameters.
exception: none

# 7 MIS of Input Verification Module

## 7.1 Module

verify_params

## 7.2 Uses

parameters (5)

## 7.3 Syntax

### 7.3.1 Exported Access Programs

| Name | In | Out | Exceptions |
|---|---|---|---|
| verify_valid | parameters | - | badLength, badDiam, badPCMVolume, badPCMAndTankVol, badPCMArea, badPCMDensity, badMeltTemp, badCoilAndInitTemp, badCoilTemp, badPCMHeatCapSolid, badPCMHeatCapLiquid, badHeatFusion, badCoilArea, badWaterDensity, badWaterHeatCap, badCoilCoeff, badPCMCoeff, badInitTemp, badFinalTime, badInitAndMeltTemp |
| verify_recommended | parameters | - | - |

## 7.4 Semantics

### 7.4.1 Environment Variables

*win*: 2D array of pixels displayed on the screen.

### 7.4.2 Assumptions

The load_params function has been called on *params*, so the variables have all been assigned a value.

#### 7.4.3 Access Routine Semantics

verify_valid(*params*):

transition: Modifies *win* by displaying an error message when appropriate.

exceptions: Exceptions occur if any of the input parameters lie outside of boundaries determined by physical law. Error messages corresponding to each exception are shown in the Appendix (13).

verify_recommended(*params*):

transition: Modifies *win* by displaying warning messages.

exception: none

# 8 MIS of Temperature ODEs Module

## 8.1 Module

temperature

## 8.2 Uses

parameters (5)

## 8.3 Syntax

### 8.3.1 Exported Access Programs

| Name | In | Out | Exceptions |
|---|---|---|---|
| temperature1 | array of reals, array of reals, array of reals, parameters | real, real | - |
| temperature2 | array of reals, array of reals, array of reals, parameters | real, real, real | - |
| temperature3 | array of reals, array of reals, array of reals, parameters | real, real | - |
| event1 | array of reals, array of reals, array of reals, parameters | real | - |
| event2 | array of reals, array of reals, array of reals, parameters | real | - |

## 8.4 Semantics

### 8.4.1 State Variables

$time$: array of reals  
$tempW$: array of reals  
$tempP$: array of reals  
$latHeat$: array of reals  
$params$: parameters

### 8.4.2 Assumptions

The load_params function has been called on $params$, so the variables have all been assigned a value. The verify_valid function has been called on $params$, so no exceptions occur due to physically impossible values.

### 8.4.3 Access Routine Semantics

temperature1($t$, $Tw$, $Tp$, $params$):
output: Returns values for water and PCM temperature for the case where PCM has not started melting.
exception: none

temperature2($t$, $Tw$, $Tp$, $params$):
output: Returns values for water and PCM temperature and latent heat for the case where PCM is in the process of melting.
exception: none

temperature3($t$, $Tw$, $Tp$, $params$):
output: Returns values for water and PCM temperature for the case where PCM has finished melting.
exception: none

event1($t$, $Tw$, $Tp$, $params$):
output: Returns a value that signals the ODE solver to either continue generating the solution for the next time point, or stop solving the ODE system at the current time point.
exception: none

event2($t$, $Tw$, $Tp$, $params$):
output: Returns a value that signals the ODE solver to either continue generating the solution for the next time point, or stop solving the ODE system at the current time point.
exception: none

# 9 MIS of Energy Module

## 9.1 Module

energy

## 9.2 Uses

parameters (5)

## 9.3 Syntax

### 9.3.1 External Access Programs

| **Name** | **In** | **Out** | **Exceptions** |
|---|---|---|---|
| energy1Wat | array of reals, parameters | array of reals | - |
| energy1PCM | array of reals, parameters | array of reals | - |
| energy2Wat | array of reals, parameters | array of reals | - |
| energy2PCM | array of reals, parameters | array of reals | - |
| energy3Wat | array of reals, parameters | array of reals | - |
| energy3PCM | array of reals, parameters | array of reals | - |

## 9.4 Semantics

### 9.4.1 State Variables

$tempW$: array of reals
$tempP$: array of reals
$latHeat$: array of reals
$eW$: array of reals
$eP$: array of reals
$params$: parameters

### 9.4.2 Assumptions

The load_params function has been called on $params$, so all variables have been assigned a value. The verify_params function has been called on $params$, so there are no exceptions due to physically impossible values.

### 9.4.3 Access Routine Semantics

energy1Wat($Tw$, $params$):
output: energy1Wat outputs an array of reals representing the energy profile of the water while the PCM has not started melting.
exception: none

energy1PCM($Tp$, $params$):
output: energy1PCM outputs an array of reals representing the energy profile of the PCM while the PCM has not started melting.
exception: none

energy2Wat($Tw$, $params$):
output: energy2Wat outputs an array of reals representing the energy profile of the water while the PCM is melting.
exception: none

energy2PCM($Qp$, $params$):
output: energy2PCM outputs an array of reals representing the energy profile of the PCM while the PCM is melting.
exception: none

energy3Wat($Tw$, $params$):
output: energy3Wat outputs an array of reals representing the energy profile of the water after the PCM has finished melting.
exception: none

energy3PCM($Tp$, $params$):
output: energy3PCM outputs an array of reals representing the energy profile of the PCM after the PCM has finished melting.
exception: none

# 10 MIS of Output Verification Module

## 10.1 Module

verify_output

## 10.2 Uses

parameters (5)

## 10.3 Syntax

### 10.3.1 Exported Access Programs

| Name | In | Out | Exceptions |
|---|---|---|---|
| verify_output | array of reals, array of reals, array of reals, array of reals, array of reals, parameters | - | - |

## 10.4 Semantics

### 10.4.1 State Variables

*time*: array of reals
*tempW*: array of reals
*tempP*: array of reals
*eW*: array of reals
*eP*: array of reals
*params*: parameters

### 10.4.2 Environment Variables

*win*: 2D array of pixels displayed on the screen

### 10.4.3 Local Variables

$errorWater$: real
$errorPCM$: real

### 10.4.4 Assumptions

The load_params function has been called on *params*, so every variable in the structure has a value. The verify_valid function has been called on *params*, so there are no exceptions due to physically impossible values. The temperature and energy arrays have been filled by the ODE solver and energy functions, so there is no divide by zero exception.

### 10.4.5 Access Routine Semantics

verify_output($t$, $Tw$, $Tp$, $Ew$, $Ep$, $params$): transition: Modifies *win* with a warning if $errorWater$ or $errorPCM$ is greater than $ConsTol$.
exception: none

# 11 MIS of Plotting Module

## 11.1 Module

plot

## 11.2 Uses

N/A

## 11.3 Syntax

### 11.3.1 Exported Access Programs

| Name | In | Out | Exceptions |
|---|---|---|---|
| plot | array of reals, array of reals, array of reals, array of reals, array of reals, parameters, string | - | - |

## 11.4 Semantics

### 11.4.1 State Variables

*time*: array of reals<br>
*tempW*: array of reals<br>
*tempP*: array of reals<br>
*eW*: array of reals<br>
*eP*: array of reals<br>
*params*: parameters<br>
*filename*: string

### 11.4.2 Environment Variables

*directory*: The current directory of files from which the program is run.

### 11.4.3 Access Routine Semantics

plot($t$, $Tw$, $Tp$, $Ew$, $Ep$, $params$, $filename$):

transition: Modifies *directory* by writing to it a .png file containing the graphs of the simulation results.

exception: none

# 12 MIS of Output Module

## 12.1 Module

output

## 12.2 Uses

parameters (5)

## 12.3 Syntax

### 12.3.1 Exported Access Program

| Name | In | Out | Exceptions |
|---|---|---|---|
| output | string, array of reals, array of reals, array of reals, array of reals, array of reals, array of reals, array of reals, parameters | - | - |

## 12.4 Semantics

### 12.4.1 State Variables

*params*: parameters
*time*: array of reals
*tempW*: array of reals
*tempP*: array of reals
*eW*: array of reals
*eP*: array of reals
*eTot*: array of reals
*filename*: string

### 12.4.2 Environment Variables

*directory*: The current directory of files from which the program is run.

### 12.4.3 Assumptions

The load_params function was called on *params*, so all the variables have been assigned a value. The ODE solver and energy functions have filled *time*, *tempW*, *tempP*, *eW*, and *eP* with results.

### 12.4.4 Access Routine Semantics

output($params$, $t$, $Tw$, $Tp$, $Ew$, $Ep$, $filename$):

transition: Modifies *directory* by writing to it a .txt file containing the input parameters, calculated parameters, and results of the simulation.

exception: none

# 13 Appendix

| Message ID | Error Message |
|---|---|
| badLength | Error: Tank length must be > 0 |
| badDiam | Error: Tank diameter must be > 0 |
| badPCMVolume | Error: PCM volume must be > 0 |
| badPCMAndTankVol | Error: PCM volume must be < tank volume |
| badPCMArea | Error: PCM area must be > 0 |
| badPCMDensity | Error: rho_p must be > 0 |
| badMeltTemp | Error: Tmelt must be > 0 and < $Tc$ |
| badCoilAndInitTemp | Error: Tc must be > Tinit |
| badCoilTemp | Error: Tc must be > 0 and < 100 |
| badPCMHeatCapSolid | Error: C_ps must be > 0 |
| badPCMHeatCapLiquid | Error: C_pl must be > 0 |
| badHeatFusion | Error: Hf must be > 0 |
| badCoilArea | Error: Ac must be > 0 |
| badWaterDensity | Error: rho_w must be > 0 |
| badWaterHeatCap | Error: C_w must be > 0 |
| badCoilCoeff | Error: hc must be > 0 |
| badPCMCoeff | Error: hp must be > 0 |
| badInitTemp | Error: Tinit must be > 0 and < 100 |
| badFinalTime | Error: tfinal must be > 0 |
| badInitAndMeltTemp | Error: Tinit must be < Tmelt |