Module Interface Specification for Data Center Minimization Cost Analysis

Ning Wang

May 4, 2023

Contents

1 Revision History **iv**

2 Symbols, Abbreviations and Acronyms **1**

3 Introduction **1**

4 Notation **1**

5 Module Decomposition **1**

6 MIS of Control Module **2**
- 6.1 Module 2
- 6.2 Uses 2
- 6.3 Syntax 2
  - 6.3.1 Exported Access Programs 2
- 6.4 Semantics 2
  - 6.4.1 State Variables 2
  - 6.4.2 Access Routine Semantics 2

7 MIS of Input Parameters Module **4**
- 7.1 Module 4
- 7.2 Uses 4
- 7.3 Syntax 4
- 7.4 Semantics 4
  - 7.4.1 Environment Variables 4
  - 7.4.2 State Variables 4
  - 7.4.3 Assumptions 5
  - 7.4.4 Access Routine Semantics 5
- 7.5 Considerations 7

8 MIS of Input Verification Module **8**
- 8.1 Module 8
- 8.2 Uses 8
- 8.3 Syntax 8
  - 8.3.1 Exported Access Programs 8
- 8.4 Semantics 8
  - 8.4.1 Environment Variables 8
  - 8.4.2 Assumptions 8
  - 8.4.3 Access Routine Semantics 8
- 8.5 Considerations 9

9 MIS of Optimization Module **10**
9.1 Module . . . 10
9.2 Uses . . . 10
9.3 Syntax . . . 10
9.3.1 Exported Access Programs . . . 10
9.4 Semantics . . . 10
9.4.1 State Variables . . . 10
9.4.2 Assumptions . . . 10
9.4.3 Access Routine Semantics . . . 11

10 MIS of Output Verification Module **12**
10.1 Module . . . 12
10.2 Uses . . . 12
10.3 Syntax . . . 12
10.3.1 Exported Constant . . . 12
10.4 Semantics . . . 12
10.4.1 State Variables . . . 12
10.4.2 Assumptions . . . 12
10.4.3 Access Routine Semantics . . . 12

11 MIS of Plotting Module **13**
11.1 Module . . . 13
11.2 Uses . . . 13
11.3 Syntax . . . 13
11.3.1 Exported Access Programs . . . 13
11.4 Semantics . . . 13
11.4.1 State Variables . . . 13
11.4.2 Environment Variables . . . 13
11.4.3 Assumptions . . . 13
11.4.4 Access Routine Semantics . . . 13

12 MIS of Output Module **14**
12.1 Module . . . 14
12.2 Uses . . . 14
12.3 Syntax . . . 14
12.3.1 Exported Constants . . . 14
12.3.2 Exported Access Program . . . 14
12.4 Semantics . . . 14
12.4.1 State Variables . . . 14
12.4.2 Environment Variables . . . 14
12.4.3 Access Routine Semantics . . . 14

13 MIS of Specification Parameters **15**
13.1 Module 15
13.2 Uses 15
13.3 Syntax 15
13.3.1 Exported Constants 15
13.4 Semantics 15

14 Appendix **16**

1 Revision History

Date	Version	Notes
March 12, 2023	1.0	First Draft
May 2, 2023	1.1	Revision 1

2 Symbols, Abbreviations and Acronyms

See SRS Documentation at https://github.com/KarenKarenWang/cas741_project1/tree/main/docs/SRS

3 Introduction

The following document details the Module Interface Specifications for the implemented modules in a program simulating a Minimization 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. The full documentation and implementation can be found at https://github.com/KarenKarenWang/cas741_project1/tree/main/docs/SRS.

4 Notation

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

Table 1 summarizes the primitive data types used by SWHS.

Data Type	Notation	Description
character	char	a single symbol or digit
integer	$\mathbb{Z}$	a number without a fractional component in (-∞, ∞)
natural number	$\mathbb{N}$	a number without a fractional component in [1, ∞)
real	$\mathbb{R}$	any number in (-∞, ∞)

The specification of SWHS 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, SWHS 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 Module	
Behaviour-Hiding Module	Input Parameters Module Input Verification Module Output Format Module Output Verification Module Control Module Specification Parameters Module
Software Decision Module	Optimization Module Sequence Data Structure Module Plotting Module

Table 1: Module Hierarchy

6 MIS of Control Module

6.1 Module

main

6.2 Uses

Parameter (Section 7), Optimization (Section 9), verify_output (Section 10), plot (Section 11), output (Section 12)

6.3 Syntax

6.3.1 Exported Access Programs

Name	In	Out	Exceptions
main	-	-	-

6.4 Semantics

6.4.1 State Variables

None

6.4.2 Access Routine Semantics

main():

- transition: Modify the state of grid power consumption and the renewable power consumption for the Plot and Output modules by following these steps

Get (filenameIn: string) and (filenameOut: string) from user

load_params(filenameIn)

#Find minimization cost function (C_{total}, C_R, C_G, C_g, C_r, L_{total}, R_{total}, P_{total}), *and transmission loss* (T_L) *and distances of data centers with power station* (d_i)

$A_{eq} \cdot x = b_{eq}$:= solve(linp_optimization, L_{total}, P_{total}, R_{total}, C_{total}, C_R, C_G,)

L_{total} := solve(linp_optimization, R_i^N, P_i^N)

#find transmission loss along distances

$$T_L = (d_i \cdot 0.03 \cdot P_i)^N$$

#Power distribution

$$L_i = R_i + P_i$$

#Output calculated values to a file and to a plot. Verify the sum of calculated values as less than total power consumption.

verify_output(L_{total}, R_{total}, P_{total})

plot(L_i, d_i)

output(filenameOut, L_i, d_i, R_i, P_i, C_{total})

7 MIS of Input Parameters Module

The secrets of this module are the input datas, how the values are input and how the values are verified. The load and verify secrets are isolated to their own access programs.

7.1 Module

Param

7.2 Uses

SpecParam (Section 13)

7.3 Syntax

Name	In	Out	Exceptions
load_params	string	-	FileTypeError
verify_params	-	-	badLength, valueError, outofboundary, negativevalue, badTotalPower, badRenewableRate, badGridPowerRate, badDistances
d_i	-	$\mathbb{R}$	
C_R	-	$\mathbb{R}$	
C_P	-	$\mathbb{R}$	
L_{total}	-	$\mathbb{R}$	
...	...	...	
L_i	-	$\mathbb{R}$	

7.4 Semantics

7.4.1 Environment Variables

inputFile: sequence of string *#f[i] is the ith string in the text file f*

7.4.2 State Variables

From T1

R_i: $\mathbb{R}$ (renewable energy consumption for each data center)

P_i: $\mathbb{R}$ (grid energy consumption for each data center)

C_{total}: $\mathbb{R}$ (total cost)

C_r: $\mathbb{R}$ (cost for renewable consumption)

C_p : $\mathbb{R}$ (cost for grid consumption)
From T2
T_L: $\mathbb{R}$ (total power loss)
P_i: $\mathbb{R}$ (grid energy consumption for each data center)
d_i: $\mathbb{R}$ (distances between power station and each data center)
From T3
L_{total}: $\mathbb{R}$ (total load)
L_i: $\mathbb{R}$ (load for each data center)
R_i: $\mathbb{R}$ (renewable energy consumption for each data center)
P_i: $\mathbb{R}$ (grid energy consumption for each data center

To Support IM1
L_{total}: $\mathbb{R}$ (total load
L_i: $\mathbb{R}$ (load for each data center)

7.4.3 Assumptions

- readtable(filename) will be called before the values of any state variables will be accessed.
- The file contains the string equivalents of the numeric values for each input parameter in order, each on a new line. The order is the same as in the table in R1 of the SRS. Any comments in the input file should be denoted with a '#' symbol.

7.4.4 Access Routine Semantics

Param.R_i:

- output: $out := R_i$
- exception: none

Param.P_i:

- output: $out := P_i$
- exception: none

...

Param.T_L:

- output: $out := T_L$
- exception: none

Param.L_i:

- output: $out := L_i$
- exception: none

Param.C_{total}:

- output: $out := C_T$
- exception: none

load_params(s):
Param.C_r:

- output: $out := C_r$
- exception: none

Param.C_p:

- output: $out := C_p$
- exception: none

Param.L_{total}:

- output: $out := L_{\text{total}}$
- exception: none

Param.L_max:

- output: $out := L_{\max}$
- exception: none

Param.d_i:

- output: $out := d_i$
- exception: none

$$
\begin{array}{ll}
\neg(d_i > 2000) & \Rightarrow \text{warnLength} \\
\neg(C_p < 0) & \Rightarrow \text{badValue} \\
\neg(0 \le L_T \le L_{\max}) & \Rightarrow \text{warnLength} \\
\neg(d_i < 0) & \Rightarrow \text{badValue} \\
\neg(L_T < 0) & \Rightarrow \text{badValue} \\
\neg(L_{\max} < 0) & \Rightarrow \text{badValue}
\end{array}
$$

etc. See Appendix (Section 14) for the complete list of exceptions and associated error messages.

7.5 Considerations

The value of each state variable can be accessed through its name (getter). An access program is available for each state variable. There are no setters for the state variables, since the values will be set and checked by load params and not changed for the life of the program.

8 MIS of Input Verification Module

8.1 Module

verify_params

8.2 Uses

Param (Section 7)

8.3 Syntax

8.3.1 Exported Access Programs

Name	In	Out	Exceptions
verify_valid	-	-	badLength, valueError, outofboundary, negative-value, badTotalPower, badRenewableRate, bad-GridPowerRate, badDistances
verify_recommend	-	-	-

8.4 Semantics

8.4.1 Environment Variables

Not applicable

8.4.2 Assumptions

All of the fields Param have been assigned values before any of the access routines for this module are called.

8.4.3 Access Routine Semantics

verify_valid():

- transition: Reading user inputs: Reading the CSV file and extracting values: Setting up the optimization problem: Solving the optimization problem: Calculating and displaying results:

- exceptions: exc := (
  Param.get$d_i() \leq 0 \Rightarrow$ badLength |
  Params.get$C_r() \leq 0 \Rightarrow$ badValue |
  Params.get$L_T() \leq L_{\max} \Rightarrow$ badValue|
  Params.get$C_p() \leq 0 \Rightarrow$ badValue |

8.5 Considerations

See Appendix (Section 14) for the complete list of exceptions and associated error messages.

9 MIS of Optimization Module

9.1 Module

Minimize total cost

9.2 Uses

Param (Section 7)

9.3 Syntax

9.3.1 Exported Access Programs

Name	In	Out	Exceptions
linprog	–	$(\mathbb{R} \rightarrow \mathbb{R})$	distribution of consumptions
optimoptions	–	$(\mathbb{R} \rightarrow \mathbb{R})$	minimization cost
transloss	–	$(\mathbb{R} \rightarrow \mathbb{R})$	real number

9.4 Semantics

9.4.1 State Variables

- Renewable energy power consumption
- Grid energy power consumption
- Actual grid energy power consumption after considering transmission loss
- Total cost of energy
- Optimal power consumption for each data center

9.4.2 Assumptions

none

9.4.3 Access Routine Semantics

linprog():

- output: $A_{eq} \cdot x = b_{eq} :=$ solve(L_{total}, P_{total}, R_{total}, C_{total}, C_R, C_G,)

- exception: none

optimoptions():

- output: $L_{\text{total}} :=$ solve(linp_optimization, R_i^N, P_i^N)

- exception: none

10 MIS of Output Verification Module

10.1 Module

verify_output

10.2 Uses

Param (Section 7)

10.3 Syntax

10.3.1 Exported Constant

None

10.4 Semantics

10.4.1 State Variables

None

10.4.2 Assumptions

All of the fields of the input parameters structure have been assigned a value.

10.4.3 Access Routine Semantics

verify_output(L_i(scalar), d_i(row vector), R_i(row vector), P_i(row vector), C_{total}(scalar), L_{total}(scalar), L_{max}(scalar)):

- verification := ($L_{\text{total}} = \sum_{i=1}^{n} \mathrm{L}_i$)

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	$d_i : \mathbb{R} \rightarrow \mathbb{R}, L_i : \mathbb{R} \rightarrow \mathbb{R},$	-	-

11.4 Semantics

11.4.1 State Variables

None

11.4.2 Environment Variables

will display on MATLAB within it own graph

11.4.3 Assumptions

None

11.4.4 Access Routine Semantics

plot(d_i(distances between power station and data centers), L_i(load distributed for each data centers):

- transition: To display a plot where the vertical axis The power consumption distribution and the horizontal axis is the distance between data centers and power stations.
- exception: none

12 MIS of Output Module

12.1 Module

output

12.2 Uses

Param (Section 7)

12.3 Syntax

12.3.1 Exported Constants

totalcost: integer

12.3.2 Exported Access Program

Name	In	Out	Exceptions
output	filename: string, $C_T : \mathbb{R} \rightarrow \mathbb{R}, L_i : \mathbb{R} \rightarrow \mathbb{R}, P_i : \mathbb{R} \rightarrow \mathbb{R}, R_i : \mathbb{R} \rightarrow \mathbb{R}$, $L_T : \mathbb{R}$	-	-

12.4 Semantics

12.4.1 State Variables

None

12.4.2 Environment Variables

file: A text file

12.4.3 Access Routine Semantics

output(filename, C_T, L_i, L_T, R_i, P_i):

- transition: Write and export the result into a file the following: the input parameters from Param, and the calculated values C_T, L_i, R_i, P_i. The functions will be output as sequences in this file. The spacing between points in the sequence should be selected so that the heating behaviour is captured in the data.
- exception: none

13 MIS of Specification Parameters

The secrets of this module is the value of the specification parameters.

13.1 Module

SpecParam

13.2 Uses

N/A

13.3 Syntax

13.3.1 Exported Constants

Some Default Value
$L_{\text{max}} := 6$ (max load for each data center is 6MW)
$L_{\text{total}} := 50$ (total load for all of data centers)
$C_r := 0.041$ (price for renewable resources)
$C_p := 0.009$ (price for grid power)
$N := 5$(5 of data centers)

13.4 Semantics

N/A

14 Appendix

Table 2: Possible Exceptions

Message ID	**Error Message**
badValue	Error: Input Value must be >0
linprog	Error: no feasible region
Error: Wrong input type	