Module Interface Specification for Data Center Minimization Cost Analysis

Ning Wang

March 18, 2023

Contents

1 Symbols, Abbreviations and Acronyms

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

2 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.

The specification is given in terms of functions, rather than sequences. For instance, the power loss along transmission is given as a function of distances ($\mathbb{R} \rightarrow \mathbb{R}$), not as a sequence ($\mathbb{R}^n$). This approach is more straightforward for the specification, but in the implementation stage, it will likely be necessary to introduce a sequence, assuming that a numerical solver is used for the system of linear programming.

3 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 $(c_1 \Rightarrow r_1 | c_2 \Rightarrow r_2 | ... | c_n \Rightarrow r_n)$.

The following table 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.

4 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 Power loss Calculation Module Control Module Specification Parameters Module
Software Decision Module	Optimization Module Sequence Data Structure Module Plotting Module

Table 1: Module Hierarchy

5 MIS of Control Module

5.1 Module

main

5.2 Uses

Parameter (Section 6), Temperature (Section 8), Optimization (Section **??**), Energy (Section **??**), verify_output (Section 9), plot (Section 10), output (Section 11)

5.3 Syntax

5.3.1 Exported Access Programs

Name	In	Out	Exceptions
main	-	-	-

5.4 Semantics

5.4.1 State Variables

None

5.4.2 Access Routine Semantics

main():

- transition: Modify the state of Param module and the environment variables 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_T, C_R, C_G, C_g, C_r, L_T, R_T, P_T), *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_T, P_T, R_T, C_T, C_R, C_G,)

L_T := 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_T, R_T, P_T)

plot(L_i, d_i)

output(filenameOut, L_i, d_i, R_i, P_i, C_T)

6 MIS of Input Parameters Module

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

6.1 Module

Param

6.2 Uses

SpecParam (Section 12)

6.3 Syntax

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

6.4 Semantics

6.4.1 Environment Variables

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

6.4.2 State Variables

\# From T1
R_i: $\mathbb{R}$
P_i: $\mathbb{R}$
C_T: $\mathbb{R}$

C_r: $\mathbb{R}$
C_p : $\mathbb{R}$

From T2
T_L: $\mathbb{R}$
P_i: $\mathbb{R}$
d_i: $\mathbb{R}$

From T3
L_T: $\mathbb{R}$
L_i: $\mathbb{R}$
R_i: $\mathbb{R}$
P_i: $\mathbb{R}$

To Support IM1
L_T: $\mathbb{R}$
L_i: $\mathbb{R}$

6.4.3 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.

6.4.4 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_T:

- 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_T:

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

Param.L_max:

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

Param.d_i:

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

$\neg(C_r < 0)$	$\Rightarrow$ badValue
$\neg(d_i > 2000)$	$\Rightarrow$ warnLength
$\neg(C_p < 0)$	$\Rightarrow$ badValue
$\neg(0 \leq L_T \leq L_{\max})$	$\Rightarrow$ warnLength
$\neg(d_i < 0)$	$\Rightarrow$ badValue
$\neg(L_T < 0)$	$\Rightarrow$ badValue
$\neg(L_{\max} < 0)$	$\Rightarrow$ badValue

etc. See Appendix (Section 13) for the complete list of exceptions and associated error messages.

6.5 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.

7 MIS of Input Verification Module

7.1 Module

verify_params

7.2 Uses

Param (Section 6)

7.3 Syntax

7.3.1 Exported Access Programs

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

7.4 Semantics

7.4.1 Environment Variables

Distances upper boundary

7.4.2 Assumptions

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

7.4.3 Access Routine Semantics

verify_valid():

- transition: none
- exceptions: exc := (
  Param.getd_i() $\leq 0 \Rightarrow$ badLength |
  Params.getC_r() $\leq 0 \Rightarrow$ badValue |
  Params.getL_T() $\leq L_{\max} \Rightarrow$ badValue|
  Params.getC_p() $\leq 0 \Rightarrow$ badValue |
  Params.get$L_{\max}$() $\leq 0 \Rightarrow$ badValue |

7.5 Considerations

See Appendix (Section 13) for the complete list of exceptions and associated error messages.

8 MIS of Optimization Module

8.1 Module

Minimize total cost

8.2 Uses

Param (Section 6)

8.3 Syntax

8.3.1 Exported Access Programs

Name	In	Out	Exceptions
linprog	–	$(\mathbb{R} \rightarrow \mathbb{R})$	-
optimoptions	–	$(\mathbb{R} \rightarrow \mathbb{R})$	-
transloss	–	$(\mathbb{R} \rightarrow \mathbb{R})$	-

8.4 Semantics

8.4.1 State Variables

none

8.4.2 Assumptions

none

8.4.3 Access Routine Semantics

linprog():

- output: $A_{eq} \cdot x = b_{eq}$:= solve(linp_optimization, L_T, P_T, R_T, C_T, C_R, C_G,)

- exception: none

optimoptions():

- output: L_T := solve(linp_optimization, R_i^N, P_i^N)

- exception: none

transloss():

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

- exception: none

9 MIS of Output Verification Module

9.1 Module

verify_output

9.2 Uses

Param (Section 6)

9.3 Syntax

9.3.1 Exported Constant

None

9.4 Semantics

9.4.1 State Variables

None

9.4.2 Assumptions

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

9.4.3 Access Routine Semantics

verify_output(L_i, d_i, R_i, P_i, C_T, L_T, $L_{\max}$):

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

10 MIS of Plotting Module

10.1 Module

plot

10.2 Uses

N/A

10.3 Syntax

10.3.1 Exported Access Programs

Name	In	Out	Exceptions
plot	$d_i : \mathbb{R} \to \mathbb{R}, L_i : \mathbb{R} \to \mathbb{R},$	-	-

10.4 Semantics

10.4.1 State Variables

None

10.4.2 Environment Variables

will display on MATLAB within it own graph

10.4.3 Assumptions

None

10.4.4 Access Routine Semantics

plot(d_i, L_i):

- 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

11 MIS of Output Module

11.1 Module

output

11.2 Uses

Param (Section 6)

11.3 Syntax

11.3.1 Exported Constants

totalcost: integer

11.3.2 Exported Access Program

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

11.4 Semantics

11.4.1 State Variables

None

11.4.2 Environment Variables

file: A text file

11.4.3 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

12 MIS of Specification Parameters

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

12.1 Module

SpecParam

12.2 Uses

N/A

12.3 Syntax

12.3.1 Exported Constants

\# Some Default Value

$L_{\text{max}} := 6$

$L_T := 10$

$C_r := 0.041$

$C_p := 0.009$

$N := 5$

$d_1 := 10$

$d_2 := 20$

$d_3 := 30$

$d_4 := 40$

$d_5 := 50$

12.4 Semantics

N/A

13 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	