Module Interface Specification for SPDFM

S. Shayan Mousavi M.

December 18, 2020

1 Revision History

Date	**Version**	**Notes**
Nov 23, 2020	1.0	Initial Design
Dec 18, 2020	1.1	Revised Design

2 Symbols, Abbreviations and Acronyms

See SRS Documentation at `https://github.com/shmouses/SPDFM/blob/master/docs/SRS/SRS.pdf`.

Contents

3 Introduction

The following document details the Module Interface Specifications for SPDFM program. SPDFM is a software for simulating surface plasmon enhanced electric field and current density in meshed geometry.

Complementary documents include the System Requirement Specifications and Module Guide. The full documentation and implementation can be found at SPDFM repository on github.

4 Notation

The structure of the MIS for modules comes from Hoffman and Strooper (1995), with the addition that template modules have been adapted from Ghezzi et al. (2003). The mathematical notation comes from Chapter 3 of Hoffman and Strooper (1995). 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 SPDFM.

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 (-∞, ∞)
imaginary	$\mathbb{I}$	any number of form $i \times \mathbb{R}$ where i is $\sqrt{-1}$

The specification of SPDFM 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, SPDFM 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	SPDFM Control Module (M2, Section 6) Input Parameter Module (M5, Section 7) Constant Parameters Module (M4, Section 8) Mesh Input Module (M6, Section 9) Frequency Domain FEM Solver Module (M7, Section 10) Output Module (M8, Section 12)
Software Decision Module	Data Structure Module (M3, Section 11)

Table 1: Module Hierarchy

6 MIS of SPDFM Control Module

6.1 Module

main

6.2 Uses

- Data Structure (Section 11)
- Input Parameter Module (Section 7)
- Mesh Input Module (Section 9)
- Frequency Domain FEM Solver Module (Section 10)
- Output Module (Section 12)

6.3 Syntax

6.3.1 Exported Constants

None.

6.3.2 Exported Access Programs

Name	In	Out	Exceptions
main	-	-	-

6.4 Semantics

SPDFM Control Module is designed to control the process flow in the software with respect to the SRS document. In this regard, current module organizes all other modules to satisfy the requirements that are specified in SRS. This module also help maintainability and expandability of SPDFM by classifying different parts of the code.

6.4.1 State Variables

None

6.4.2 Environment Variables

None

6.4.3 Assumptions

None

6.4.4 Access Routine Semantics

main():

- transition: Control the order of execution of different modules:

  Initiate the SPData object (M3, Section 11): doing so provides an empty framework that other modules can use to call or export data.

  ParamLoad (M5, Section 7): This module inputs all the parameters and stores them in the Data object.

  MshInput (M6, Section 9): This module loads and prepares the mesh geometry for finite element calculations.

  FreqSolver (M7, Section 10): Controls process flow and interactions with the FEniCS FEM solver. FEniCS is an external library used in this study for solving non-local hydrodynamic PDEs (see IM 2 in the SRS document). Interested readers can find more information about FEniCS at Alnæs et al. (2015); Logg et al. (2012).

  SPDoutput (M8, Section 12): Exports the final results of the simulation into .vtk file.

- output: None
- exception: None

6.4.5 Local Functions

None

7 MIS of Input Parameter Module

7.1 Module

ParamLoad

7.2 Uses

- Conatant Parameters Module (Section 8)
- Data Structure Module (Section 11)

7.3 Syntax

7.3.1 Exported Constants

None

7.3.2 Exported Access Programs

Name	In	Out	Exceptions
ParamLoad	string	-	PolarizationRangeError, DirectionNormalityError, LightOrthogonalityError, WavelengthRangeError, GammaRangeError

7.4 Semantics

7.4.1 State Variables

SPData object: stores input light source and material properties data

7.4.2 Environment Variables

ParamFile: A file containing sequence of strings that provides data related to the light source and material properties.

7.4.3 Assumptions

- ParamLoad 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 of the input data is as below (further explanation about input parameters can be found in):

  # Data in ParamLSfile:

line 1: p_x, p_y, p_z (light polarization)

line 2: d_x, d_y, d_z (direction of light propagation)

line 3: WL_{min} (the shortest wavelength of the light source)

line 4: WL_{max} (the longest wavelength of the light source)

line 5: NumLS (number of wavelengths/light sources between min and max wavelength)

line 6: eps0 (environment permittivity)

line 7: mu0 (environment permeability)

line 8: gamma (plasmon damping parameter)

line 9: PlasmaFreq (plasma frequency)

line 10: Beta (Fermi velocity proportionality)

7.4.4 Access Routine Semantics

ParamLoad is a function to load, verify, and store input data (R1 and R2 from SRS).

ParamLoad(pathLS, pathMP):

- transition: pathLS (light source data) and pathMP (material properties) are the file paths for the input files which user provides. The following procedure is performed:

  – Verify the format of the files to be .txt.

  – From ParamLSfile (located at pathLS) below parameters are obtained (As PDE equation (IM 2 SRS document) is being solve in the frequency domain light source should have a minimum and a maximum wavelength to specify the interval of the frequency domain. In this regard, number of frequencies in the intervals should be input as NumLS):

    Polarization of the incident light($\mathbb{R}^3$ vector): data.pol := p := $[p_x, p_y, p_z]$

    Direction of the incident light ($\mathbb{R}^3$ vector): data.dir := d := $[d_x, d_y, d_z]$

    Minimum wavelength of the light source ($\mathbb{R}$): data.WLmin := WL$_{min}$

    Maximum wavelength of the light source ($\mathbb{R}$): data.WLmax := WL$_{max}$

    Number of different wavelengths in the interval ($\mathbb{N}$): data.NumLS := NumLS

    Environment permittivity ($\mathbb{R}$): data.eps0 := eps0

    Environment permeability ($\mathbb{R}$): data.mu0 := mu0

    Plamson damping parameter ($\mathbb{R}$): data.damp := gamma

    Plasma frequency ($\mathbb{R}$): data.Pfreq := PlasmaFreq

Fermi velocity proportionality ($\mathbb{R}$): data.beta := Beta

  - VerifyPol
  - VerifyDir
  - VerifyWL
  - VerifyGamma

- output: None
- exception: exc :=

If the file addressed by pathLS or path MP doesn't exist	=> badFilePath
If the file format is not .csv	=> badFileFormat

7.4.5 Local Functions

verifyPol:

- output: None
- exception: exc :=

$\neg(data.PminLmt < \|p\| < data.PmaxLmt)$ => PolarizationRangeError

verifyDir:

- output: None
- exception: exc :=

$\|d\| \neq 1$	=> DirectionRangeError
$d.p! = 0$	=> LightOrthogonalityError

verifyWL:

- output: None
- exception: exc :=

$\neg(data.WLminLmt < WL_{min} < WL_{max} < data.WLmaxLmt)$ =>WavelengthRangeError

verifyGamma:

- output: None
- exception: exc :=

  $\neg(data.dampMinLmt < gamma < data.dampMaxLmt)$ => GammaRangeError

8 MIS of Constant Parameters Module

8.1 Module

Template Module

8.2 Uses

- Hardware Hiding Module

8.3 Syntax

8.3.1 Exported Constants

None

8.3.2 Exported Access Programs

Name	In	Out	Exceptions
ConstParam	-	-	-

8.4 Semantics

8.4.1 State Variables

ConstParam: An object containing real values.

- ConstParam.PminLmt $\in \mathbb{R}$
- ConstParam.PmaxLmt $\in \mathbb{R}$
- ConstParam.RadiusMinLmt $\in \mathbb{R}$
- ConstParam.RadiusMaxLmt $\in \mathbb{R}$
- ConstParam.WLminLmt $\in \mathbb{R}$
- ConstParam.WLmaxLmt $\in \mathbb{R}$
- ConstParam.dampMinLmt $\in \mathbb{R}$
- ConstParam.dampMaxLmt $\in \mathbb{R}$

8.4.2 Environment Variables

N/A

8.4.3 Assumptions

None

8.4.4 Access Routine Semantics

Store constant variables as indicated in Table 2 of the SRS (aligned with R2 of the SRS).

ConstParam:

- transition: Initiation of ConstParam object and stroing constant values obtained from Table 2 of the SRS in it:

  ConstParam.PminLmt := -100

  ConstParam.PmaxLmt := 100

  ConstParam.RadiusMinLmt := 10

  ConstParam.RadiusMaxLmt := 100

  ConstParam.WLminLmt := 200

  ConstParam.WLmaxLmt := 1000

  ConstParam.dampMinLmt:= 0.01

  ConstParam.dampMaxLmt:= 1

- output: None.
- exception: None.

8.4.5 Local Functions

None

9 MIS of Mesh Input Module

9.1 Module

GmshInput

9.2 Uses

- Data Structure (Section 11)

9.3 Syntax

9.3.1 Exported Constants

None.

9.3.2 Exported Access Programs

Name	**In**	**Out**	**Exceptions**
GmshInput	string	-	FileError

9.4 Semantics

9.4.1 State Variables

SPData object to store the mesh data.

9.4.2 Environment Variables

inputMesh: .xml files containing the data related to the meshed geometry (nodes, physical regions, boundaries).

9.4.3 Assumptions

None

9.4.4 Access Routine Semantics

This module satisfies R1 and R2 requirements from the SRS document.

gmshInput(pathMESH):

- transition: pathMESH is the file path for the input mesh file. The following procedure is performed:
    - Load mesh data from the input file.
    - Geometry is stored in the data structure: SPData.mesh := XMLmesh

- output: None
- exception: None

9.4.5 Local Functions

None

10 MIS of Frequency Domain FEM Solver Module:

10.1 Module

FreqSolver

10.2 Uses

- Data Structure Modules (Section 11)

10.3 Syntax

10.3.1 Exported Constants

None.

10.3.2 Exported Access Programs

Name	In	Out	Exceptions
FreqSolver	-	-	-

10.4 Semantics

10.4.1 State Variables

SPData: the object containing all the data required for FEM solver

10.4.2 Environment Variables

None

10.4.3 Assumptions

None

10.4.4 Access Routine Semantics

This module specifically feeds the frequency domain PDE equations to the FEniCS finite element PDE solver. FEniCS toolbox is used for all finite element setups in SPDFM. To better understand the FEniCS implementation and data structure, readers are encouraged to look up Alnæs et al. (2015) and Logg et al. (2012). The FEniCS-related implementations which include defining function space, element space, trial function, test function, and inputting variational form of the system of equations are not discussed in this document as these descriptions are part of FEniCS toolbox and are discussed in details in references mentioned above.

FreqSolver():

- transition: FEniCS toolbox will receive nonlocal hydrodynamic equation system (IM2 SRS document) to calculate the electric field and electric current at different given frequencies and the mesh.

  – Setting up FEniCS environment.

  – importing below equations to the FEniCS, all the parameters in the equation are defined in IM2 and also the table of symbols in the SRS document:

$$-\int_{\Omega}\beta^2(\nabla.\psi)(\nabla.\mathbf{J}_{HD})dV+\omega(\omega+i\gamma)\int_{\Omega}\psi.\mathbf{J}_{H}DdV-\ i\omega\omega_p^2\int_{\Omega}\psi.\varepsilon_0\mathbf{E}dV=0$$

$$\int_{\Omega}((\nabla\times\phi).(\mu_0^{-1}\nabla\times\mathbf{E})-\omega^2\phi.\epsilon_{local}\mathbf{E})dV+\ \int_{\partial\Omega}\phi.(\mathbf{n}\times(\mu_0^{-1}\nabla\times\mathbf{E}))dA=i\omega\int_{\Omega}\phi.\mathbf{J}_{HD}dV$$

$$Boundary\ Conditions:$$

$$\int_{\Omega}((\nabla\times\phi).(\mu_0^{-1}\nabla\times\mathbf{E})-\omega^2\phi\epsilon_{local}\mathbf{E}_i)dV+\int_{\partial\Omega}\phi.DtN(\mathbf{E})dA\quad-i\omega\int_{\Omega}\phi.\mathbf{J}_{HD}dV=$$
$$-\int_{\partial\Omega}\phi.(n\times(\mu_0^{-1}\nabla\times\mathbf{E}_i))dA+\int_{\partial\Omega}\phi.DtN(\mathbf{E}_i)dA$$

$$n.\mathbf{J}_{HD}=0\ \ on\ \partial\Omega$$

# For importing the data to FEniCS, as complex numbers are not defined in this toolbox parameters and equations should be separated into real and imaginary parts.

$$\mathbf{J}_{HD}=\mathbf{J}_{HD}^{real}+i\mathbf{J}_{HD}^{img}$$
$$\mathbf{E}=\mathbf{E}^{real}+i\mathbf{E}^{img}$$

–Store the result for each frequency in the SPData.FEMresult.

$$SPData.FEMresult=\begin{pmatrix}\mathbf{E}^{real}\\ \mathbf{E}^{img}\\ \mathbf{J}_{HD}^{real}\\ \mathbf{J}_{HD}^{img}\end{pmatrix}$$

11 MIS of Data Structure Module

11.1 Module

Template Module

11.2 Uses

- Hardware Hiding Module

11.3 Syntax

11.3.1 Exported Constants

None.

11.3.2 Exported Access Programs

Name	In	Out	Exceptions
init	-	-	-

11.4 Semantics

11.4.1 State Variables

data:object

- data.pol $\in \mathbb{R}^3$
- data.dir $\in \mathbb{R}^3$
- data.PminLmt $\in \mathbb{R}$
- data.PmaxLmt $\in \mathbb{R}$
- data.WLmin $\in \mathbb{R}$
- data.WLmax $\in \mathbb{R}$
- data.WLminLmt $\in \mathbb{R}$
- data.WLmaxLmt $\in \mathbb{R}$
- data.NumLS $\in \mathbb{N}$
- data.eps0 $\in \mathbb{R}$
- data.mu0 $\in \mathbb{R}$

- data.beta $\in \mathbb{R}$
- data.damp $\in \mathbb{R}$
- data.dampMinLmt $\in \mathbb{R}$
- data.dampMaxLmt $\in \mathbb{R}$
- data.Pfreq $\in \mathbb{R}$
- data.XDFMmesh : Mesh object
- data.FreqRes $\in \mathbb{R}^4$

11.4.2 Environment Variables

N/A

11.4.3 Assumptions

None

11.4.4 Access Routine Semantics

N/A

11.4.5 Local Functions

None

12 MIS of Output Module

12.1 Module

Output

12.2 Uses

- Hardware Hiding Module
- Data Structure Module (Section 11)

12.3 Syntax

12.3.1 Exported Constants

None.

12.3.2 Exported Access Programs

Name	In	Out	Exceptions
Output	-	string (.pvd and .vtk file)	-

12.4 Semantics

12.4.1 State Variables

None

12.4.2 Environment Variables

None

12.4.3 Assumptions

None

12.4.4 Access Routine Semantics

Output():

- transition: None
- output: export .pvd and .vtk files from SPData.FEMresult
- exception: None

12.4.5 Local Functions

None

References

Martin Alnæs, Jan Blechta, Johan Hake, August Johansson, Benjamin Kehlet, Anders Logg, Chris Richardson, Johannes Ring, Marie E Rognes, and Garth N Wells. The fenics project version 1.5. *Archive of Numerical Software*, 3(100), 2015.

Carlo Ghezzi, Mehdi Jazayeri, and Dino Mandrioli. *Fundamentals of Software Engineering*. Prentice Hall, Upper Saddle River, NJ, USA, 2nd edition, 2003.

Daniel M. Hoffman and Paul A. Strooper. *Software Design, Automated Testing, and Maintenance: A Practical Approach*. International Thomson Computer Press, New York, NY, USA, 1995. URL http://citeseer.ist.psu.edu/428727.html.

Anders Logg, Kent-Andre Mardal, and Garth N Wells. Finite element assembly. In *Automated Solution of Differential Equations by the Finite Element Method*, pages 141–146. Springer, 2012.