A Python software framework for the design of photonic integrated circuits

Emmanuel Lambert – Martin Fiers

Department of Information Technology
Photonics Research Group
Sint-Pietersnieuwstraat 41
9000 Ghent
Belgium
http://photonics.intec.ugent.be
Photonics Research

Setting the scene…
UNIFORM PYTHON SCRIPTING
PCELL ENGINE (MASK DESIGN)

PICAZZO
Toolkit for design of photonic components

VIRTUAL FABRICATION
Generate a material geometry

ELECTROMAGNETIC SIMULATION FRAMEWORK

FDTD
Eigenmode solvers
Beam propagation

GDS2
Photonic Development API
OpenAccess (e.g. Cadence)

Interfacing to external design tools

UNIFORM PYTHON SCRIPTING

USER

DEVELOPER
• Architecture

• Technical implementation
  • Virtual fabrication
  • Interface with FDTD simulator
Main architectural concept:

separation of concerns through **Mixins**
What is a mixin? Let’s illustrate it ...

class calculating some scientific data

Two of them mixed in as base class when module is imported

Mixin-class: visualize 2D with Matplotlib

Several alternative implementations for visualization

Mixin-class: visualize 2D with Gnuplot

Mixin-class: visualize 3D with Mayavi

Mixin-class: visualize 3D with Gnuplot
How we do it in Python...

```python
class MyCalculation(MixinBowl):
    # core functionality only
    
    def calculate(self):
        ....

    def get_data(self):
        ...
        return (X, Y, Z)
```

```
class __Visualize__(object):
    def plot_2d(self):
        raise NotImplementedError("Abstract class")

    def plot_3d(self):
        raise NotImplementedError("Abstract class")
```

```python
class Visualize2DGnuplot(object):
    def plot_2d(self):
        ...

    def plot_3d(self):
        ...
```

```python
class Visualize3DMayavi(object):
    def plot_2d(self):
        ...

    def plot_3d(self):
        ...
```

```python
class Visualize2DMatplotlib(object):
    def plot_2d(self):
        ...
```

```python
class Visualize3DGnuplot(object):
    def plot_3d(self):
        ...
```

```python
in the __init__.py file of the package:
MyCalculation.mixin(Visualize2DGnuplot)
MyCalculation.mixin(Visualize3DMayavi)
```
emmanuel@emmanuel-desktop:~$ python
Python 2.6.5 (r265:79063, Apr 16 2010, 13:09:56)
[GCC 4.4.3] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> from mycalculation import *
>>> c = MyCalculation()
>>> c.plot_3d()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
AttributeError: 'MyCalculation' object has no attribute 'plot 3d'
>>> from visualization import *
>>> c.plot_3d()
Applied to our framework ...

“PCell engine”

LAYOUT

VISUALIZATION 2D / 3D

SYMBOLIC REPRESENTATION

SCHEMATIC (NETLIST) REPRESENTATION

OpenAccess compliance
Implementation of the virtual fabrication

VIRTUAL FABRICATION
Generate a material geometry

UNIFORM PYTHON SCRIPTING

PICAZZO
Toolkit for design of photonic components
Mask layout =
a collection of shapes
on different layers

Virtual fabrication =
Can be expressed as
an algorithm with logical
operations on subsets of
the shapes
(AND, OR, XOR, NOT)
Challenge: transform a mask layout into a materials geometry simulating the physical fabrication processes.
- Geometrical Python packages **Shapely** and **Descartes**
- Transform the shapes of the mask layout into **Shapely polygons** (per layer)

  ```python
  shapely.geometry.polygon.Polygon
  shapely.geometry.multipolygon.MultiPolygon
  ```

- Apply the **logical operations at polygon level** through Shapely functions:

  - **AND**
  - **OR**
  - **NOT**
  - **XOR**

  Shapely: intersection  
  Shapely: union  
  Shapely: difference with the canvas  
  Shapely: symmetric_difference
Advantages of implementation with Shapely vs more brute-force approaches:

- **High accuracy**:  
  - “analytical” information about the geometry is maintained throughout the algorithm  
  - Allows interfacing with other tools (such as simulators) without loss of detail

- **Great performance**:  
  - Very fast  
  - Consumes very little memory
• **Descartes essential for 2D visualization with Matplotlib**
  (direct plotting of Shapely polygons)

• **3D visualisation with Mayavi surface plot (to be improved)**
Interfacing to the Meep FDTD simulator

**UNIFORM PYTHON SCRIPTING**

- **PICAZZO**
  - Toolkit for design of photonic components

- **VIRTUAL FABRICATION**
  - Generate a material geometry

- **ELECTROMAGNETIC SIMULATION FRAMEWORK**
  - FDTD
  - Eigenmode solvers
  - Beam Propagation
• Meep is a popular open source EM FDTD simulator developed by MIT
• It allows scripting through Scheme and C++

Challenge : seamless integration

Material geometry from virtual fabrication

FDTD simulation

Python

Scheme / C++
Step towards this goal:

- **Python bindings for Meep**, based on SWIG
- Released as open source on Launchpad in December 2009 ("Python-Meep")

---

**Callback:** Given a \((x,y,z)\) coordinate: What is the material?

(Meep core engine)

* `get material(x,y,z)`

*(Python) script describing the simulation*

* `return material(x,y,z)`

---

**Millions of times**

> potential performance issue
Approaches for interfacing the material data with the Meep callback:

Meep core engine

Callback:
Given a (x,y,z) coordinate:
What is the material?

PURE PYTHON CALLBACK

Python-MEEP BINDINGS

SIMULATION SCRIPT

C++

Strategy 1:
Accumulation of Python callbacks is a bottleneck

Python
Approaches for interfacing the material data with the Meep callback:

**Callback:**
Given a (x,y,z) coordinate: What is the material?

**Meep core engine**

**PYTHON-MEEP BINDINGS**

**Numpy matrix**
with a discretised representation of the materials

**SIMULATION SCRIPT**

**Strategy 2**:
- Inaccurate (discretization)
- (very) high memory usage

**C++**
Approaches for interfacing the material data with the Meep callback:

**Callback:**
Given a (x,y,z) coordinate: What is the material?

**Python-MEEP Bindings**

**Polygon Representation** of the materials. Given a point (x,y,z), which polygon is the point in?
- winding number OR
- ray-casting algorithm

**Strategy 3:**
- very accurate
- low memory requirements
Software framework for the design of photonic integrated circuits

Shapely & Descartes
Matplotlib
Numpy, Scipy
Mayavi2
SWIG

Python