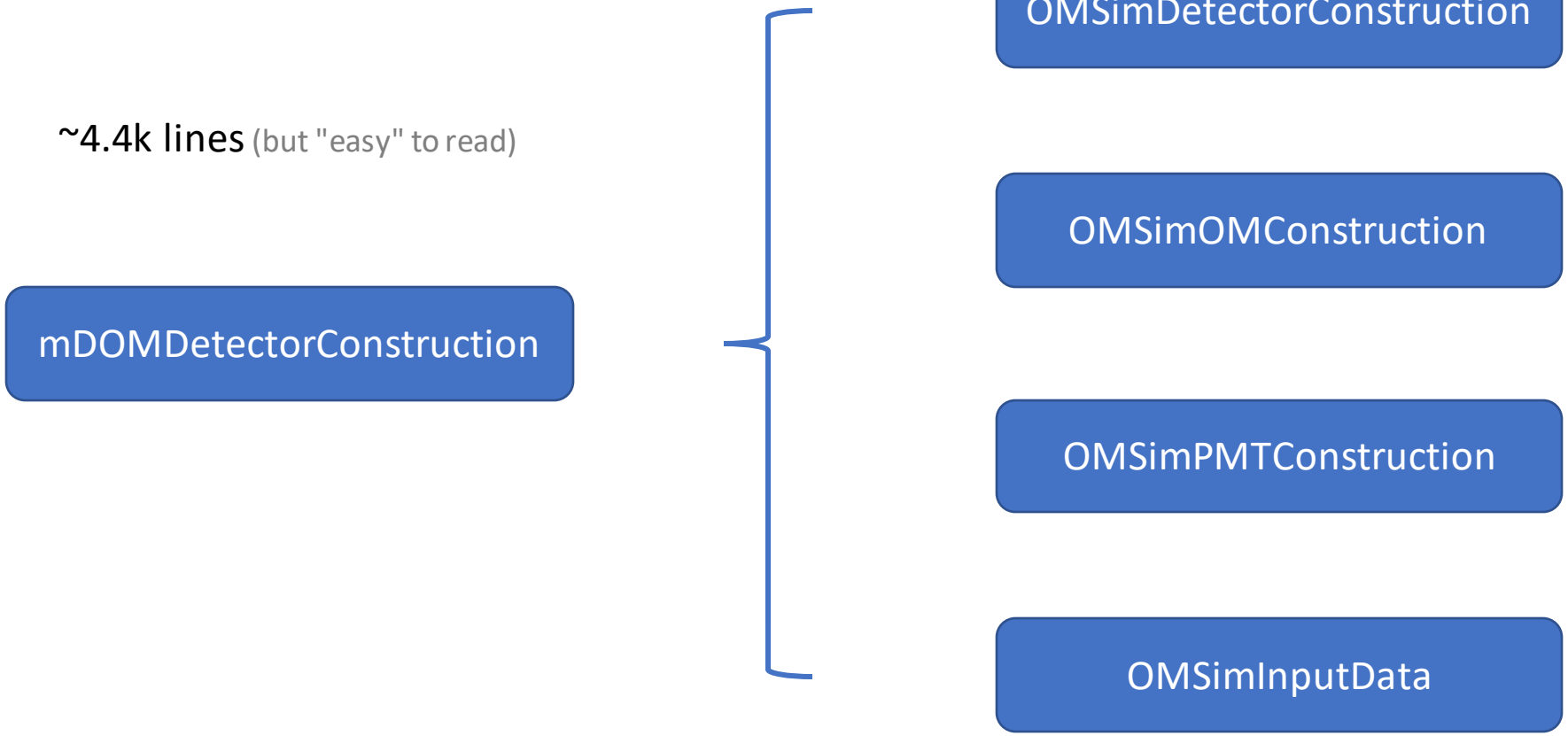
Geant4 Geometry Classes

Martin Unland

26.03.2021

~1.4k lines (may be a little bit convoluted...)

~4.4k lines (but "easy" to read)



General things

- Geant4 10.7! We should keep updating with new releases
- Code in GitHub and updated only when code is of general interest
- Material and PMT/OM data in JSON files

- Documentation in Doxygen style
- Code in CamelCase and following nomenclature:

```
OMSimClass::ClassMethod(type pInput){ // input parameters start with p
        int lLocalInteger = 5;                  // local parameters start with l
        mMemberInteger = 5;                     //member parameters (defined in header) with m
        int lFromFile = OMSimInputData->Get("jMyVariable"); //input from files with j

}
```

JSON files are easy to make!

```python
import json
pythonDictionary = {}
pythonDictionary["jBool"] = True
pythonDictionary["jArray1"] = [1,2,3,4]
pythonDictionary["jDictionary"] = {"name": ["Martin", "Lew"], "BeardPoints": [0, 100]}

with open("file.dat", 'w') as outfile:
    json.dump(pythonDictionary, outfile,indent=3)
```

```
{
   "jBool": true,
   "jArray1": [
      1,
      2,
      3,
      4
   ],
   "jDictionary": {
      "name": [
         "Martin",
         "Lew"
      ],
      "BeardPoints": [
         0,
         100
      ]
   }
}
```

OMSimDetectorConstruction (1 mDOM)

```
void OMSimDetectorConstruction::ConstructWorld(OMSimInputData* pData){
    mWorldSolid = new G4Orb("World",gworldsize*m);
    mWorldLogical = new G4LogicalVolume(mWorldSolid, pData->GetMaterial("Ri_Air"), "World logical", 0, 0, 0);
    mWorldPhysical = new G4PVPlacement (0, G4ThreeVector(0.,0.,0.), mWorldLogical, "World_phys", 0, false, 0);
    mWorldLogical->SetVisAttributes(G4VisAttributes::GetInvisible());
}

G4VPhysicalVolume* OMSimDetectorConstruction::Construct() {
    
    OMSimInputData* lData = new OMSimInputData();
    lData->SearchFolders();

    ConstructWorld(lData);

    OMSimOMConstruction* lOpticalModule = new OMSimOMConstruction(lData);
    lOpticalModule->SelectModule("mDOM");
    lOpticalModule->PlaceIt(G4ThreeVector(0,0,0), new G4RotationMatrix(), mWorldLogical, false, "MyFirstmDOM" );

    return mWorldPhysical;
}
```

OMSimDetectorConstruction (1 mDOM)

Creating the world has its own function

```
void OMSimDetectorConstruction::ConstructWorld(OMSimInputData* pData){
    mWorldSolid = new G4Orb("World",gworldsize*m);
    mWorldLogical = new G4LogicalVolume(mWorldSolid, pData->GetMaterial("Ri_Air"), "World logical", 0, 0, 0);
    mWorldPhysical = new G4PVPlacement (0, G4ThreeVector(0.,0.,0.), mWorldLogical, "World_phys", 0, false, 0);
    mWorldLogical->SetVisAttributes(G4VisAttributes::GetInvisible());
}

G4VPhysicalVolume* OMSimDetectorConstruction::Construct() {

    OMSimInputData* lData = new OMSimInputData();
    lData->SearchFolders();

    ConstructWorld(lData);

    OMSimOMConstruction* lOpticalModule = new OMSimOMConstruction(lData);
    lOpticalModule->SelectModule("mDOM");
    lOpticalModule->PlaceIt(G4ThreeVector(0,0,0), new G4RotationMatrix(), mWorldLogical, false, "MyFirstmDOM" );

    return mWorldPhysical;
}
```

OMSimDetectorConstruction (1 mDOM)

```
void OMSimDetectorConstruction::ConstructWorld(OMSimInputData* pData){
    mWorldSolid = new G4Orb("World",gworldsize*m);
    mWorldLogical = new G4LogicalVolume(mWorldSolid, pData->GetMaterial("Ri_Air"), "World logical", 0, 0, 0);
    mWorldPhysical = new G4PVPlacement (0, G4ThreeVector(0.,0.,0.), mWorldLogical, "World_phys", 0, false, 0);
    mWorldLogical->SetVisAttributes(G4VisAttributes::GetInvisible());
}

G4VPhysicalVolume* OMSimDetectorConstruction::Construct() {

    OMSimInputData* lData = new OMSimInputData();
    lData->SearchFolders();

    ConstructWorld(lData);

    OMSimOMConstruction* lOpticalModule = new OMSimOMConstruction(lData);
    lOpticalModule->SelectModule("mDOM");
    lOpticalModule->PlaceIt(G4ThreeVector(0,0,0), new G4RotationMatrix(), mWorldLogical, false, "MyFirstmDOM" );

    return mWorldPhysical;
}
```

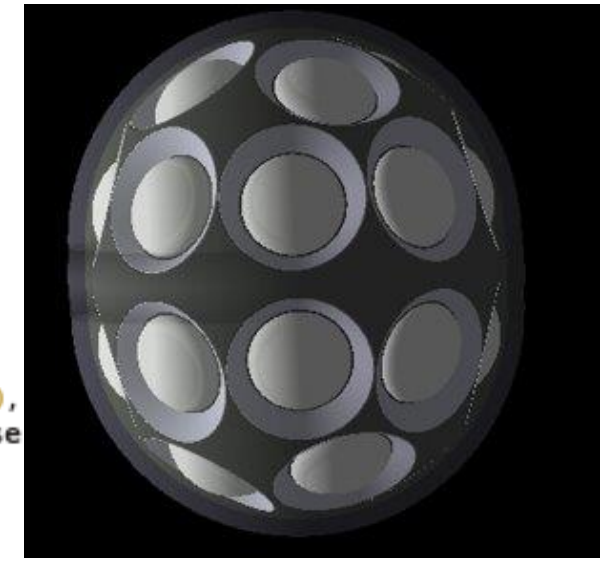
Create data-instance and search input files

Instance passed to other classes or functions

Instance provides data from input files

OMSimDetectorConstruction (1 mDOM)

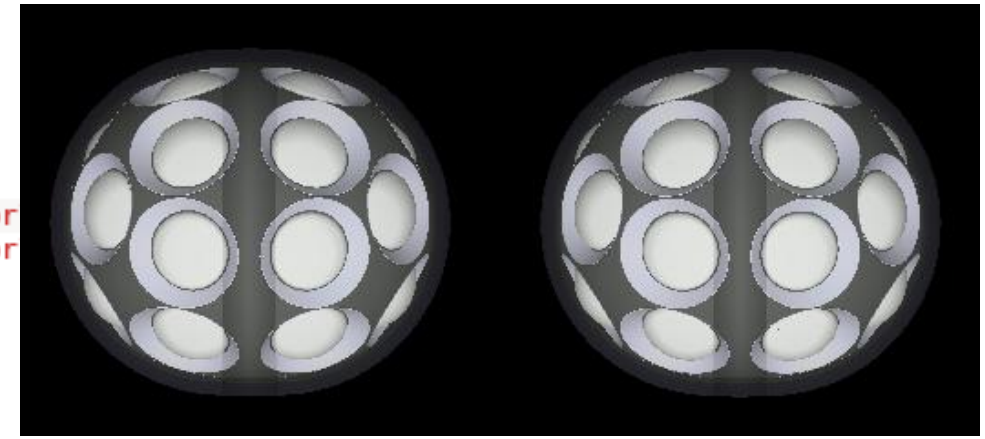
Create OM-class instance, select optical module and place it somewhere in the world



```
void OMSimDetectorConstruction::ConstructWorld(OMSimInputData* pData){
    mWorldSolid = new G4Orb("World",gworldsize*m);
    mWorldLogical = new G4LogicalVolume(mWorldSolid, pData->GetMaterial("Ri_Air"), "World logical", 0, 0,
    mWorldPhysical = new G4PVPlacement (0, G4ThreeVector(0.,0.,0.), mWorldLogical, "World_phys", 0, false
    mWorldLogical->SetVisAttributes(G4VisAttributes::GetInvisible());
}

G4VPhysicalVolume* OMSimDetectorConstruction::Construct() {
    OMSimInputData* lData = new OMSimInputData();
    lData->SearchFolders();

    ConstructWorld(lData);

    OMSimOMConstruction* lOpticalModule = new OMSimOMConstruction(lData);
    lOpticalModule->SelectModule("mDOM");
    lOpticalModule->PlaceIt(G4ThreeVector(0,0,0), new G4RotationMatrix(), mWorldLogical, false, "MyFirstmDOM" );

    return mWorldPhysical;
}
```

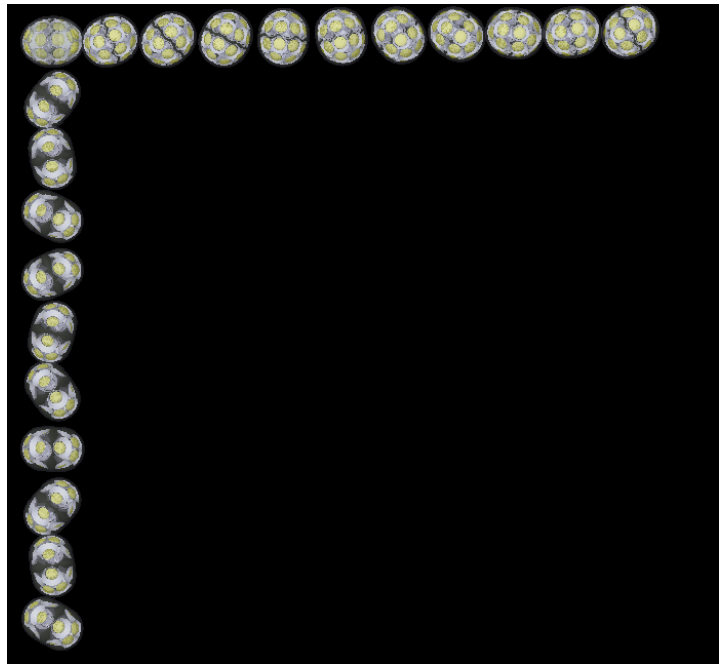
OMSimDetectorConstruction (2 mDOMs)

More modules by calling PlaceIt() again



```
void OMSimDetectorConstruction::ConstructWorld(OMSimInputData* pData){
    mWorldSolid = new G4Orb("World",gworldsize*m);
    mWorldLogical = new G4LogicalVolume(mWorldSolid, pData->GetMaterial("Ri_Air"), "Wor
    mWorldPhysical = new G4PVPlacement (0, G4ThreeVector(0.,0.,0.), mWorldLogical, "Wor
    mWorldLogical->SetVisAttributes(G4VisAttributes::GetInvisible());
}

G4VPhysicalVolume* OMSimDetectorConstruction::Construct() {

    OMSimInputData* lData = new OMSimInputData();
    lData->SearchFolders();

    ConstructWorld(lData);

    OMSimOMConstruction* lOpticalModule = new OMSimOMConstruction(lData);
    lOpticalModule->SelectModule("mDOM");
    lOpticalModule->PlaceIt(G4ThreeVector(0,0,0), new G4RotationMatrix(), mWorldLogical, false, "MyFirstmDOM" );
    lOpticalModule->PlaceIt(G4ThreeVector(0,0,-50*cm), new G4RotationMatrix(), mWorldLogical, false, "MySecondmDOM" );

    return mWorldPhysical;
}
```

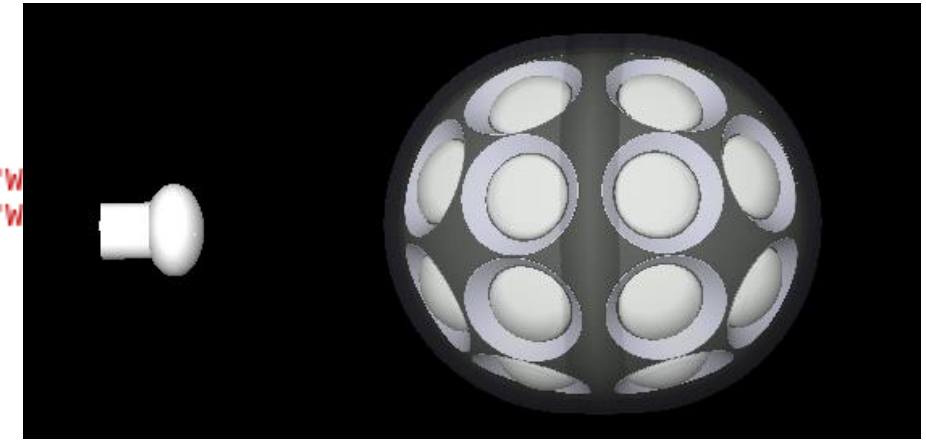
OMSimDetectorConstruction (you can get creative...)



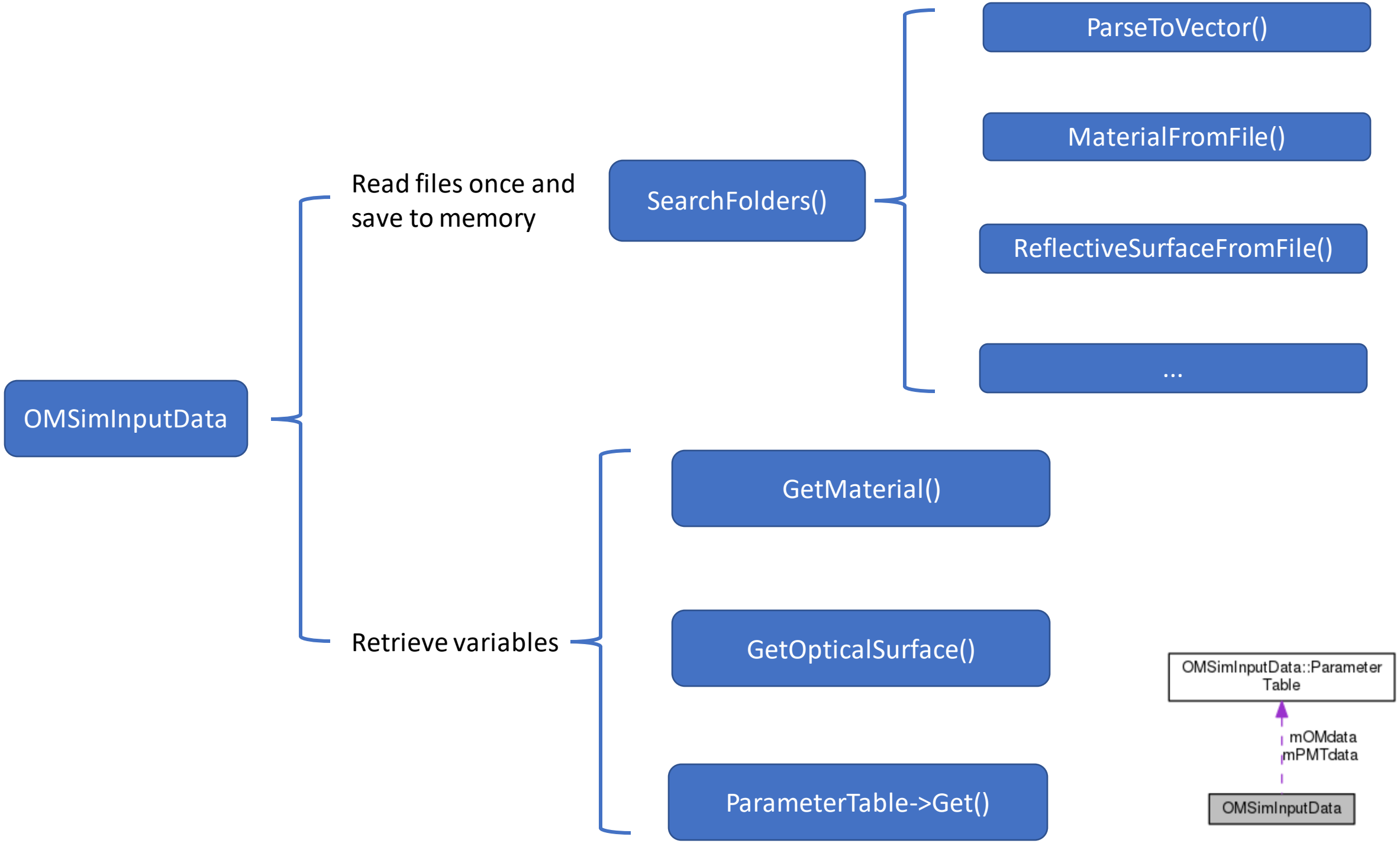
```
60      ConstructWorld(lData);
61
62      OMSimOMConstruction* lOpticalModule = new OMSimOMConstruction(lData);
63      lOpticalModule->SelectModule("mDOM");
64
65      std::stringstream lConverter;
66      for (int k = 0; k <=10; k++) {
67          lConverter.str("mDOM");
68          lConverter << k ;
69          G4RotationMatrix* lRot = new G4RotationMatrix();
70          lRot->rotateX(21*k*deg);
71          lOpticalModule->PlaceIt(G4ThreeVector(0,0,k*40*cm), lRot, mWorldLogical, false, lConverter.str());
72      }
73
74      lOpticalModule->SelectModule("LOM");
75      for (int k = 0; k <=10; k++) {
76          lConverter.str("LOM");
77          lConverter << k ;
78          G4RotationMatrix* lRot = new G4RotationMatrix();
79          lRot->rotateX(51*k*deg);
80          lOpticalModule->PlaceIt(G4ThreeVector(0,-k*40*cm,0), lRot, mWorldLogical, false, lConverter.str());
81      }
82
```

OMSimDetectorConstruction

same structure for PMTs...



```
void OMSimDetectorConstruction::ConstructWorld(OMSimInputData* pData){
    mWorldSolid = new G4Orb("World",gworldsize*m);
    mWorldLogical = new G4LogicalVolume(mWorldSolid, pData->GetMaterial("Ri_Air"), "W
    mWorldPhysical = new G4PVPlacement (0, G4ThreeVector(0.,0.,0.), mWorldLogical, "W
    mWorldLogical->SetVisAttributes(G4VisAttributes::GetInvisible());
}

G4VPhysicalVolume* OMSimDetectorConstruction::Construct() {

    OMSimInputData* lData = new OMSimInputData();
    lData->SearchFolders();

    ConstructWorld(lData);

    OMSimOMConstruction* lOpticalModule = new OMSimOMConstruction(lData);
    lOpticalModule->SelectModule("mDOM");
    lOpticalModule->PlaceIt(G4ThreeVector(0,0,40*cm), new G4RotationMatrix(), mWorldLogical, false, "MyFirstmDOM");

    OMSimPMTConstruction* lPMT = new OMSimPMTConstruction(lData);
    lPMT->SelectPMT("pmt_ETEL_9320KFL-KFB");
    lPMT->PlaceIt(G4ThreeVector(0,0,0), new G4RotationMatrix(), mWorldLogical,  "MyFirstPMT" );


    return mWorldPhysical;
}
```


OMSimInputData
Read files once and save to memory
SearchFolders()
ParseToVector()
MaterialFromFile()
ReflectiveSurfaceFromFile()
...
Retrieve variables
GetMaterial()
GetOpticalSurface()
ParameterTable->Get()
OMSimInputData::Parameter Table
mOMdata
mPMTdata
OMSimInputData

Input materials file nomenclature

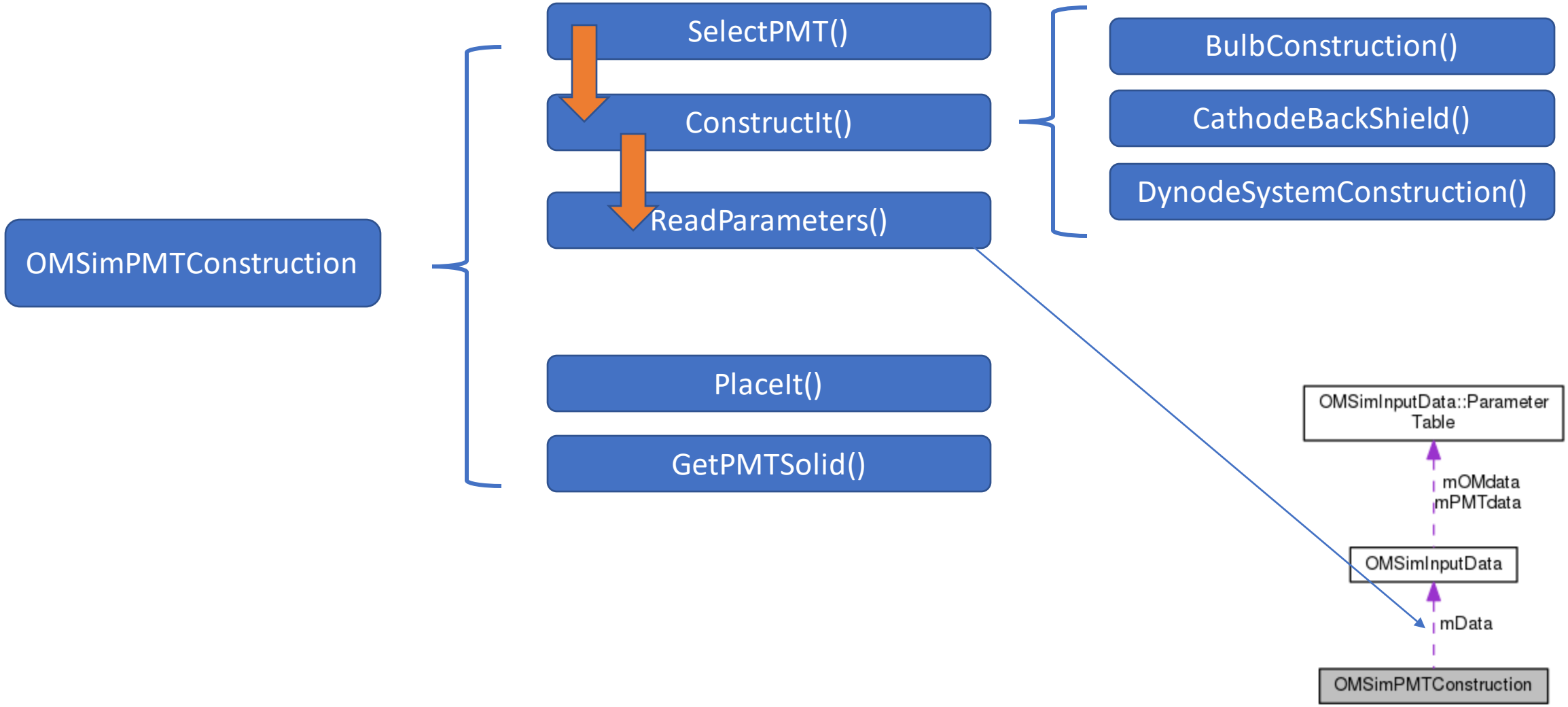
- *"Ri_"*: Material only with refractive index
- *"RiAbs_"*: Material with refractive index and absorption length
- *"Refl_"*: Optical surface
- *"NoOptic_"*: Material without optical properties

- In *"IceCubeICE.dat"* data for SPICE and ICE from IceCube

PMT and OM files nomenclature

- PMT files start with *"pmt_"* and are in folder *"data/PMTs"*

- OM files start with *"om_"* and are in folder *"data/SegmentedModules"*

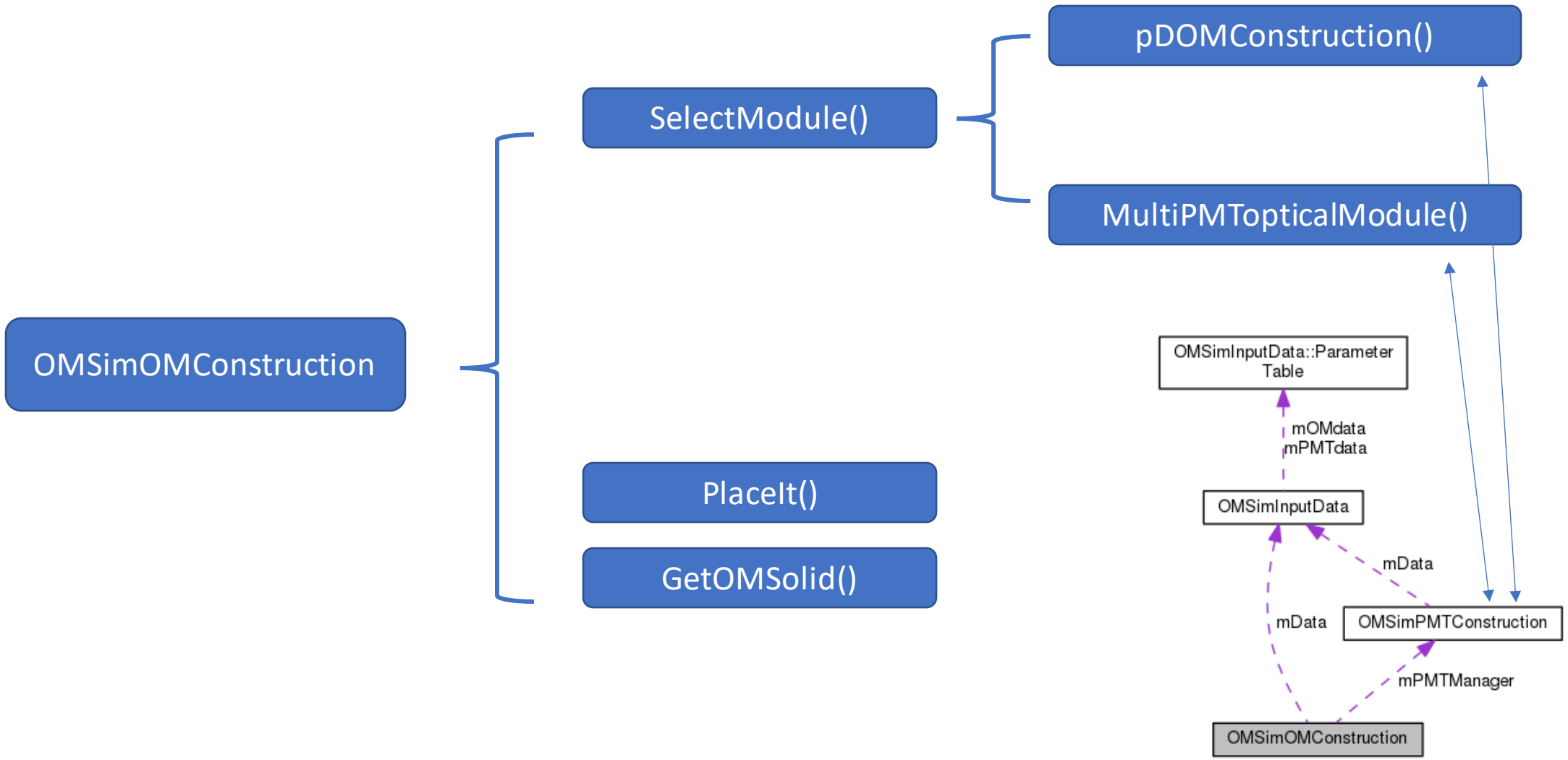

OMSimPMTConstruction
SelectPMT()
ConstructIt()
ReadParameters()
PlaceIt()
GetPMTSolid()
BulbConstruction()
CathodeBackShield()
DynodeSystemConstruction()
OMSimInputData::Parameter Table
mOMdata
mPMTdata
OMSimInputData
mData
OMSimPMTConstruction

PMT files

- Data in PMT files are fits from photocathode shape + tube length
- Until now three shape types:
  - Circle+Ellipse
  - Ellipse+Ellipse

  - Circle+Ellipse for front
    linear function and circle for mirrored back parts (for internal reflections)

- Jupyter Notebooks with fits will be in GitHub for reference


OMSimOMConstruction
SelectModule()
PlaceIt()
GetOMSolid()
pDOMConstruction()
MultiPMTopticalModule()
OMSimInputData::Parameter Table
mOMdata
mPMTdata
OMSimInputData
mData
mData
OMSimPMTConstruction
mPMTManager
OMSimOMConstruction

Argument Parameters

- With "select"-functions you can get an specific thing,
  e.g. *GetMaterial("RiAbs_Glass_Vitrovex")* for Vitrovex glass,
  but they can also look for argument parameters (given through the terminal when running the programm)

- OMSimInputData::**GetMaterial**()
  - GetMaterial(*"argVesselGlass"*)
  - GetMaterial(*"argGel"*)
  - GetMaterial(*"argWorld"*)
- OMSimInputData::**GetOpticalSurface**(*"argReflector"*)
- OMSimPMTConstruction::**SelectPMT**(*"argPMT"*)
- OMSimOMConstruction::**SelectOM**(*"argOM"*)

Doxygen

- Doxygen style is clean, and can be then compiled for standard documentation (html and pdf)
- Read here for more info

```
/**
 * Get G4OpticalSurface defined in files. In order to get custom built materials, method SearchFolders() should have already been called.
 *
 * @param pName name of the optical surface
 * @return G4OpticalSurface
 */
G4OpticalSurface* OMSimInputData::GetOpticalSurface(G4String pName){
```

~1.4k lines (may be a little bit convoluted...)

What you need to change...

~4.4k lines (but "easy" to read)

mDOMDetectorConstruction

OMSimDetectorConstruction

Change this depending on geometry of your study!!

OMSimOMConstruction

OMSimPMTConstruction

Change these if there is a completely new OM/PMT geometry (e.g. WOM, D-Egg)… Otherwise just make new "pmt_" or "om_" data files

OMSimInputData

This probably does not have to be changed much... only if you need a new data category saved in files