

Refactoring- Exercise 2

Maria Grazia Pia

INFN Genova, Italy

Maria.Grazia.Pia@cern.ch

<http://www.ge.infn.it/geant4/training/APC2014/>

Simplified version of the refactoring actually done in an ongoing project of **Geant4 physics validation**

The full set of **validation results enabled by the refactoring** will be documented in a forthcoming **journal publication**

Exercise 2

A real-life physics scenario

Working with legacy code

Refactoring techniques

Guaranteed fun...

Photoelectric effect in Geant4 standard electromagnetic package

Photoelectric cross section based on an empirical formula:

$$\mu_{ij} = \frac{A_{ij1}}{E} + \frac{A_{ij2}}{E^2} + \frac{A_{ij3}}{E^3} + \frac{A_{ij4}}{E^4}$$

F. Biggs and R. Lighthill, Analytical Approximations for X-Ray CrossSections III, Sandia Report SAND87-0070, 1988

For some elements and intervals

Geant4 uses A_{ij} parameters improved over the original A_{ij}

We want to **validate** Geant4 standard photoelectric cross section w.r.t. experimental data

How much has the accuracy **improved**?

```

processes_management:G4VProcess
# aProcessManager :G4ProcessManager* (readOnly)
# pParticleChange :G4VParticleChange*
# aParticleChange :G4ParticleChange*
# theNumberOfInteractionLengthLeft :G4double
# currentInteractionLength :G4double
# theInitialNumberOfInteractionLength :G4double
# theProcessName :G4String
# thePhysicsTableName :G4String
# theProcessType :G4ProcessType
# theProcessSubType :G4int
# thePILfactor :G4double
# enableAtRestDolt :G4bool
# enableAlongStepDolt :G4bool
# enablePostStepDolt :G4bool
# verboseLevel :G4int
# masterProcessShadow :G4VProcess*

- operator=() :G4VProcess &
+ G4VProcess()
+ G4VProcess()
- ~G4VProcess()
+ operator=(G4int query)
+ operator=(G4int query)
+ PostStepDolt() :G4VParticleChange*
+ AlongStepDolt() :G4VParticleChange*
+ AtRestDolt() :G4VParticleChange*
+ AlongStepGetPhysicalInteractionLength() :G4double
+ AtRestGetPhysicalInteractionLength() :G4double
+ PostStepGetPhysicalInteractionLength() :G4double
+ GetCurrentInteractionLength() :G4double (query)
+ SetPILfactor() :void
+ GetPILfactor() :G4double (query)
+ AlongStepGPIL() :G4double
+ AtRestGPIL() :G4double
+ PostStepGPIL() :G4double
+ IsApplicable() :G4bool
+ BuildPhysicsTable() :void
+ PreparePhysicsTable() :void
+ StorePhysicsTable() :G4bool
+ RetrievePhysicsTable() :G4bool
+ GetPhysicsTableName() :G4String &
+ GetProcessName() :G4String & (query)
+ GetProcessType() :G4ProcessType (query)
+ SetProcessType() :void
+ GetProcessSubType() :G4int (query)
+ SetProcessSubType() :void
+ GetProcessTypeName() :G4String &
+ StartTracking() :void
+ EndTracking() :void
+ SetProcessManager() :void
+ GetProcessManager() :G4ProcessManager*
+ ResetNumberOfInteractionLengthLeft() :void
+ GetNumberOfInteractionLengthLeft() :G4double (query)
+ GetTotalNumberOfInteractionLengthTraversed() :G4double (query)
+ SubtractNumberOfInteractionLengthLeft() :void
+ ClearNumberOfInteractionLengthLeft() :void
+ isAtRestDoltsEnabled() :G4bool (query)
+ isAlongStepDoltsEnabled() :G4bool (query)
+ isPostStepDoltsEnabled() :G4bool (query)
+ DumpInfo() :void (query)
+ SetVerboseLevel() :void
+ GetVerboseLevel() :G4int (query)
+ BuildWorkerProcess() :void
+ GetMasterProcess() :G4VProcess* (query)
+ BuildWorkerPhysicsTable() :void
+ PrepareWorkerPhysicsTable() :void
    
```

```

processes_management:G4VDiscreteProcess
+ G4VDiscreteProcess()
+ G4VDiscreteProcess()
- ~G4VDiscreteProcess()
+ PostStepGetPhysicalInteractionLength() :G4double
+ PostStepDolt() :G4VParticleChange*
+ AlongStepGetPhysicalInteractionLength() :G4double
+ AtRestGetPhysicalInteractionLength() :G4double
+ AtRestDolt() :G4VParticleChange*
+ AlongStepDolt() :G4VParticleChange*
# GetMeanFreePath() :G4double
- G4VDiscreteProcess()
- operator=() :G4VDiscreteProcess &
    
```

```

em_utils:G4EmProcess
- Manager :G4LossTableManager*
- modelManager :G4EmModelManager*
- trackManager :G4EmTrackManager*
- theGamma :G4ParticleDefinition* (readOnly)
- theElectron :G4ParticleDefinition* (readOnly)
- thePositron :G4ParticleDefinition* (readOnly)
- thePhoton :G4ParticleDefinition* (readOnly)
- theNeutrino :G4ParticleDefinition* (readOnly)
- theAntineutrino :G4ParticleDefinition* (readOnly)
- theMuon :G4ParticleDefinition* (readOnly)
- theTau :G4ParticleDefinition* (readOnly)
- thePion :G4ParticleDefinition* (readOnly)
- theKaon :G4ParticleDefinition* (readOnly)
- theProton :G4ParticleDefinition* (readOnly)
- theNeutron :G4ParticleDefinition* (readOnly)
- theLambda :G4ParticleDefinition* (readOnly)
- theSigma :G4ParticleDefinition* (readOnly)
- theXi :G4ParticleDefinition* (readOnly)
- theOmega :G4ParticleDefinition* (readOnly)
- theDelta :G4ParticleDefinition* (readOnly)
- theNucleon :G4ParticleDefinition* (readOnly)
- theHyperon :G4ParticleDefinition* (readOnly)
- theMeson :G4ParticleDefinition* (readOnly)
- theBaryon :G4ParticleDefinition* (readOnly)
- theLepton :G4ParticleDefinition* (readOnly)
- theQuark :G4ParticleDefinition* (readOnly)
- theAntiquark :G4ParticleDefinition* (readOnly)
- theGluon :G4ParticleDefinition* (readOnly)
- thePhoton :G4ParticleDefinition* (readOnly)
- theNeutrino :G4ParticleDefinition* (readOnly)
- theAntineutrino :G4ParticleDefinition* (readOnly)
- theMuon :G4ParticleDefinition* (readOnly)
- theTau :G4ParticleDefinition* (readOnly)
- thePion :G4ParticleDefinition* (readOnly)
- theKaon :G4ParticleDefinition* (readOnly)
- theProton :G4ParticleDefinition* (readOnly)
- theNeutron :G4ParticleDefinition* (readOnly)
- theLambda :G4ParticleDefinition* (readOnly)
- theSigma :G4ParticleDefinition* (readOnly)
- theXi :G4ParticleDefinition* (readOnly)
- theOmega :G4ParticleDefinition* (readOnly)
- theDelta :G4ParticleDefinition* (readOnly)
- theNucleon :G4ParticleDefinition* (readOnly)
- theHyperon :G4ParticleDefinition* (readOnly)
- theMeson :G4ParticleDefinition* (readOnly)
- theBaryon :G4ParticleDefinition* (readOnly)
- theLepton :G4ParticleDefinition* (readOnly)
- theQuark :G4ParticleDefinition* (readOnly)
- theAntiquark :G4ParticleDefinition* (readOnly)
- theGluon :G4ParticleDefinition* (readOnly)
- thePhoton :G4ParticleDefinition* (readOnly)
- theNeutrino :G4ParticleDefinition* (readOnly)
- theAntineutrino :G4ParticleDefinition* (readOnly)
- theMuon :G4ParticleDefinition* (readOnly)
- theTau :G4ParticleDefinition* (readOnly)
- thePion :G4ParticleDefinition* (readOnly)
- theKaon :G4ParticleDefinition* (readOnly)
- theProton :G4ParticleDefinition* (readOnly)
- theNeutron :G4ParticleDefinition* (readOnly)
- theLambda :G4ParticleDefinition* (readOnly)
- theSigma :G4ParticleDefinition* (readOnly)
- theXi :G4ParticleDefinition* (readOnly)
- theOmega :G4ParticleDefinition* (readOnly)
- theDelta :G4ParticleDefinition* (readOnly)
- theNucleon :G4ParticleDefinition* (readOnly)
- theHyperon :G4ParticleDefinition* (readOnly)
- theMeson :G4ParticleDefinition* (readOnly)
- theBaryon :G4ParticleDefinition* (readOnly)
- theLepton :G4ParticleDefinition* (readOnly)
- theQuark :G4ParticleDefinition* (readOnly)
- theAntiquark :G4ParticleDefinition* (readOnly)
- theGluon :G4ParticleDefinition* (readOnly)
    
```

```

G4PhotoElectricEffect
- isInitialised :G4bool
+ G4PhotoElectricEffect()
+ ~G4PhotoElectricEffect()
+ IsApplicable() :G4bool
+ PrintInfo() :void
# InitialiseProcess() :void
    
```

```

em_utils:G4EmModel
+ FlucModel :G4VEmFluctuationModel*
+ angModel :G4VEmAngularDistribution*
+ name :G4String (readOnly)
+ lowLimit :G4double
+ highLimit :G4double
+ emInActive :G4double
+ emMaxActive :G4double
+ polarAngleLimit :G4double
+ secondaryThreshold :G4double
+ thePILflag :G4bool
+ flagDeexcitation :G4bool
+ flagForceBuildTable :G4bool
+ isMaster :G4bool
+ localTable :G4bool
+ localElmSelectors :G4bool
+ useAngularGenerator :G4bool
+ selectors :G4int
+ elmSelectors :std::vector<G4EmElementSelector*>
+ fElementData :G4ElementData*
+ pParticleChange :G4VParticleChange*
+ xSectionTable :G4PhysicsTable*
# theDensityFactor :std::vector<G4Double*> (readOnly)
# theDensityId :std::vector<G4int*> (readOnly)
+ idTable :size_t
+ fManager :G4LossTableManager*
+ fCurrentCouple :G4MaterialCutsCouple* (readOnly)
+ fCurrentElement :G4Element* (readOnly)
+ nsec :G4int
+ xsec :std::vector<G4Double>

+ G4VEmModel()
+ G4VEmModel()
+ Initialise() :void
+ SampleSecondaries() :void
+ InitialiseLocal() :void
+ InitialiseForMaterial() :void
+ InitialiseForElement() :void
+ ComputeEDPPerVolume() :G4double
+ CrossSectionPerVolume() :G4double
+ ComputeCrossSectionPerAtom() :G4double
+ ChangeSquareRatio :G4double
+ GetChangeSquareRatio() :G4double
+ GetParticleChange() :G4double
+ StartTracking() :void
+ CorrectionsAlongStep() :void
+ Value() :G4double
+ MinPrimaryEnergy() :G4double
+ MinEnergyCut() :G4double
+ SetupForMaterial() :void
+ DefineForRegion() :void
+ GetParticleChangeForLoss() :G4double
+ GetParticleChangeForGamma() :G4ParticleChangeForGamma*
+ MaxSecondaryEnergy() :G4double
+ InitialiseElementSelectors() :void
+ GetElementSelectors() :std::vector<G4EmElementSelector*>
+ SetElementSelectors() :void
+ ComputeEDN() :G4double
+ CrossSection() :G4double
+ ComputeMeanFreePath() :G4double
+ ComputeCrossSectionPerAtom() :G4double
+ SelectIsotopeNumber() :G4int
+ SelectRandomAtom() :G4Element*
+ SelectRandomAtom() :G4Element*
+ SelectRandomAtomNumber() :G4int
+ SetParticleChange() :void
+ SetCrossSectionTable() :void
+ GetElementData() :G4ElementData*
+ GetCrossSectionTable() :G4PhysicsTable*
+ GetModelOfFluctuations() :G4VEmFluctuationModel*
+ GetAngularDistribution() :G4VEmAngularDistribution*
+ SetAngularDistribution() :void
+ HighEnergyLimit() :G4double (query)
+ LowEnergyLimit() :G4double (query)
+ HighEnergyActivationLimit() :G4double (query)
+ LowEnergyActivationLimit() :G4double (query)
+ PolarAngleLimit() :G4double (query)
+ SecondaryThreshold() :G4double (query)
+ LPFlag() :G4bool (query)
+ DeexcitationFlag() :G4bool (query)
+ ForceBuildTableFlag() :G4bool (query)
+ UseAngularGeneratorFlag() :G4bool (query)
+ IsAngularGeneratorFlag() :void
+ SetHighEnergyLimit() :void
+ SetLowEnergyLimit() :void
+ SetActivationHighEnergyLimit() :void
+ SetActivationLowEnergyLimit() :void
+ IsActive() :G4bool
+ SetPolarAngleLimit() :void
+ SetSecondaryThreshold() :void
+ SetLPFlag() :void
+ SetDeexcitationFlag() :void
+ SetForceBuildTable() :void
+ SetMasterThread() :void
+ IsMaster() :G4bool (query)
+ MaxSecondaryKinEnergy() :G4double
+ GetName() :G4String & (query)
+ SetCurrentCouple() :void
+ GetCurrentElement() :G4Element* (query)
+ GetCurrentCouple() :G4MaterialCutsCouple* (query)
+ SetCurrentElement() :void
+ operator=() :G4EmModel &
+ G4VEmModel()
    
```

```

G4PEffectFluoModel
- theGamma :G4ParticleDefinition*
- theElectron :G4ParticleDefinition*
- ParticleChange :G4ParticleChangeForGamma*
- AtomDeexcitation :G4VAtomDeexcitation*
- fMinimalEnergy :G4double
- fSandiaCof :std::vector<G4Double>

+ G4PEffectFluoModel()
+ ~G4PEffectFluoModel()
+ Initialise() :void
+ ComputeCrossSectionPerAtom() :G4double
+ CrossSectionPerVolume() :G4double
+ SampleSecondaries() :void
+ operator=() :G4PEffectFluoModel &
+ G4PEffectFluoModel()
    
```

Prepare the material for the exercise

- Download the tarball: `wget`
<http://www.ge.infn.it/geant4/training/APC2014/exercise2/APC2014ex2.tgz>
- Unpack the tarball in your home directory:
`tar -zxf APC2014ex2.tgz`
- In `$HOME/APC2014ex2`:

<code>geant4.10.00.p01</code>	Geant4 code, version 10.0-patch01
<code>work10APC</code>	Geant4 libraries and test executables
<code>parametersAPC</code>	original" and "improved" A_{ij} parameters
<code>dataAPC</code>	experimental data for validation
<code>test</code>	test code for the refactoring process
<code>rwork</code>	R scripts for data analysis
<code>setup10APC.csh, setup10APC.sh</code>	

- Define environment variables for the exercise:
`source setup10APC.csh` (or `setup10APC.sh`)

Solutions

- Solutions corresponding to the various phases of the exercise can be downloaded from <http://www.ge.infn.it/geant4/training/APC2014/exercise2.html>
- Beware: although simplified for the APC school, this is still unpublished material, which is intended to go into a publication
- Two students Min Cheol Han, Han Sung Kim (Hanyang Univ., Seoul) contributed significantly to Geant4 photoelectric effect refactoring and validation

Photoelectric cross section

G4PEEffectFluoModel

\$G4INSTALL/source/processes/electromagnetic/standard/include/G4PEEffectFluoModel.hh

\$G4INSTALL/source/processes/electromagnetic/standard/src/G4PEEffectFluoModel.cc

G4PEEffectFluoModel has a member function

```
G4double ComputeCrossSectionPerAtom(const G4ParticleDefinition*,
                                     G4double kinEnergy,
                                     G4double Z,
                                     G4double A,
                                     G4double, G4double)
```

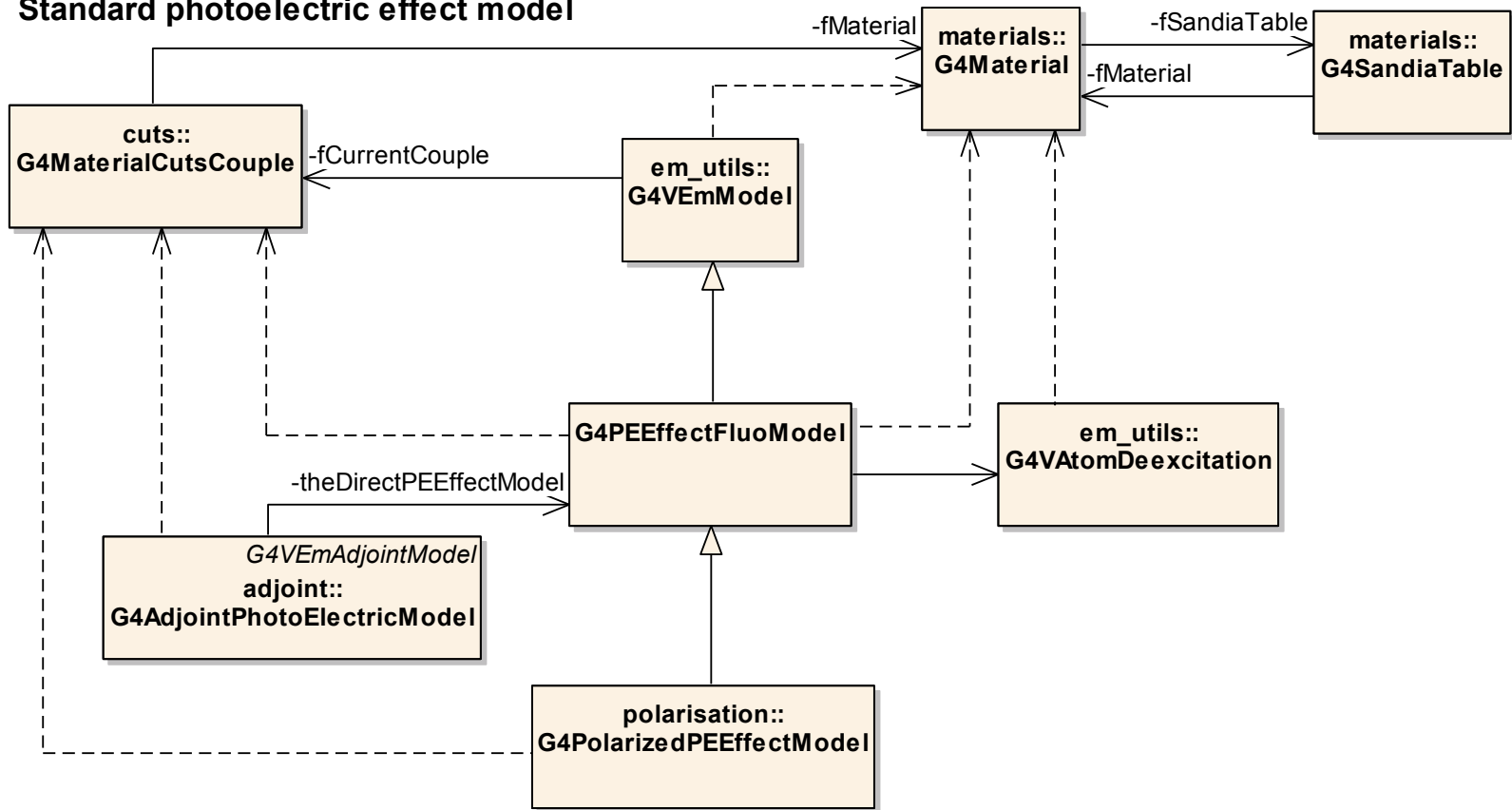
This is what we are looking for indeed!

Testing G4PEEffectFluoModel cross section

- Create a simple *G4PEEffectFluoModelTest.cc*, which
 - instantiates a *G4PEEffectFluoModel* object
 - invokes *ComputeCrossSectionPerAtom*
 - Calculates pre-defined configurations of photon energy and target element
- Place the unit test in \$APCDIR/test
- **Build the test:**
cd \$APCDIR/test
setenv TESTTARGET G4PEEffectFluoModelTest
gmake
- **Run the test:**
\$G4WORKDIR/bin/Linux-g++/G4PEEffectFluoModelTest
- **Translate the result into Bavarian**

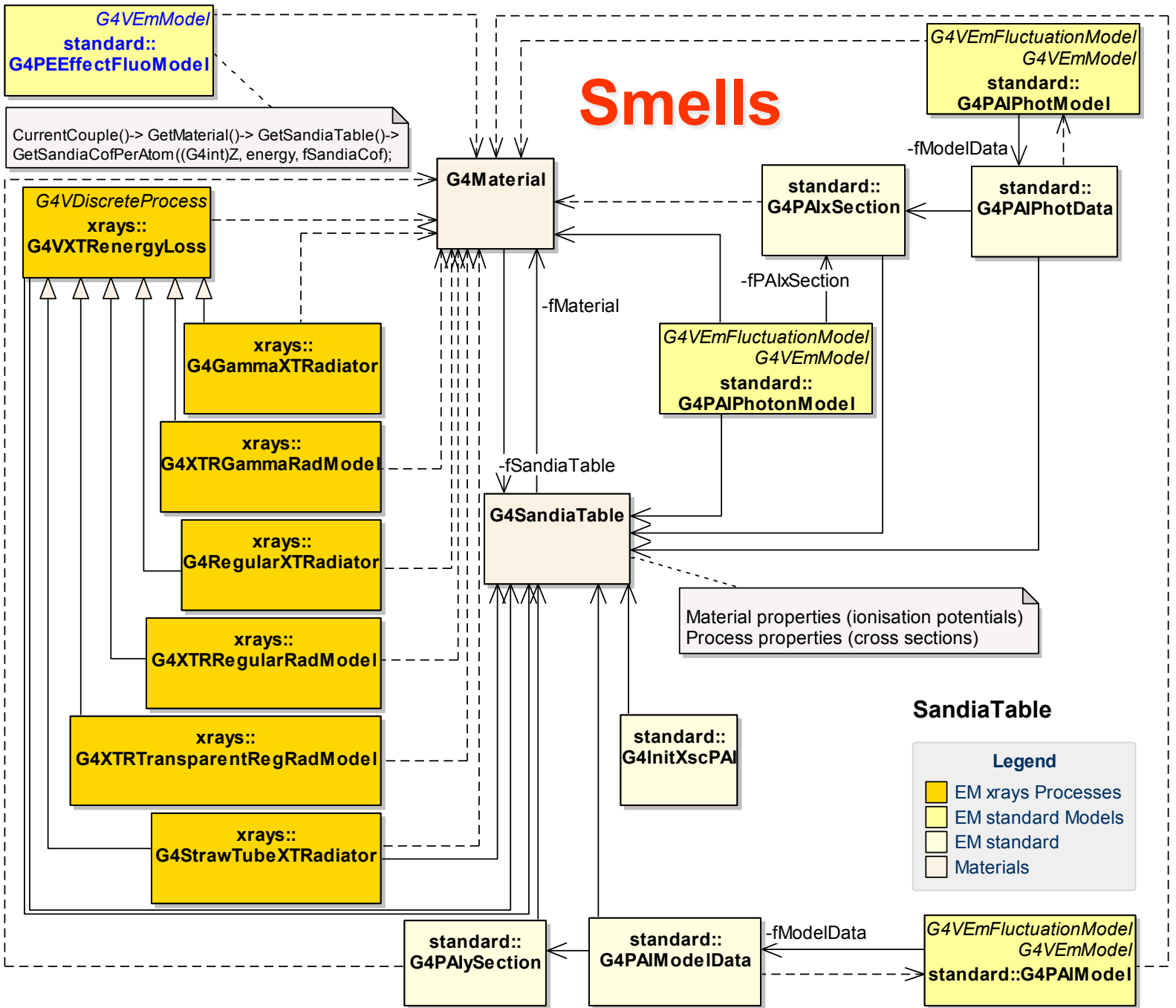
Hidden dependencies

Standard photoelectric effect model



M. Feathers, "Working Effectively with Legacy Code"
Techniques to make the calculation of the photoelectric cross section
testable

Smells



Identify smells

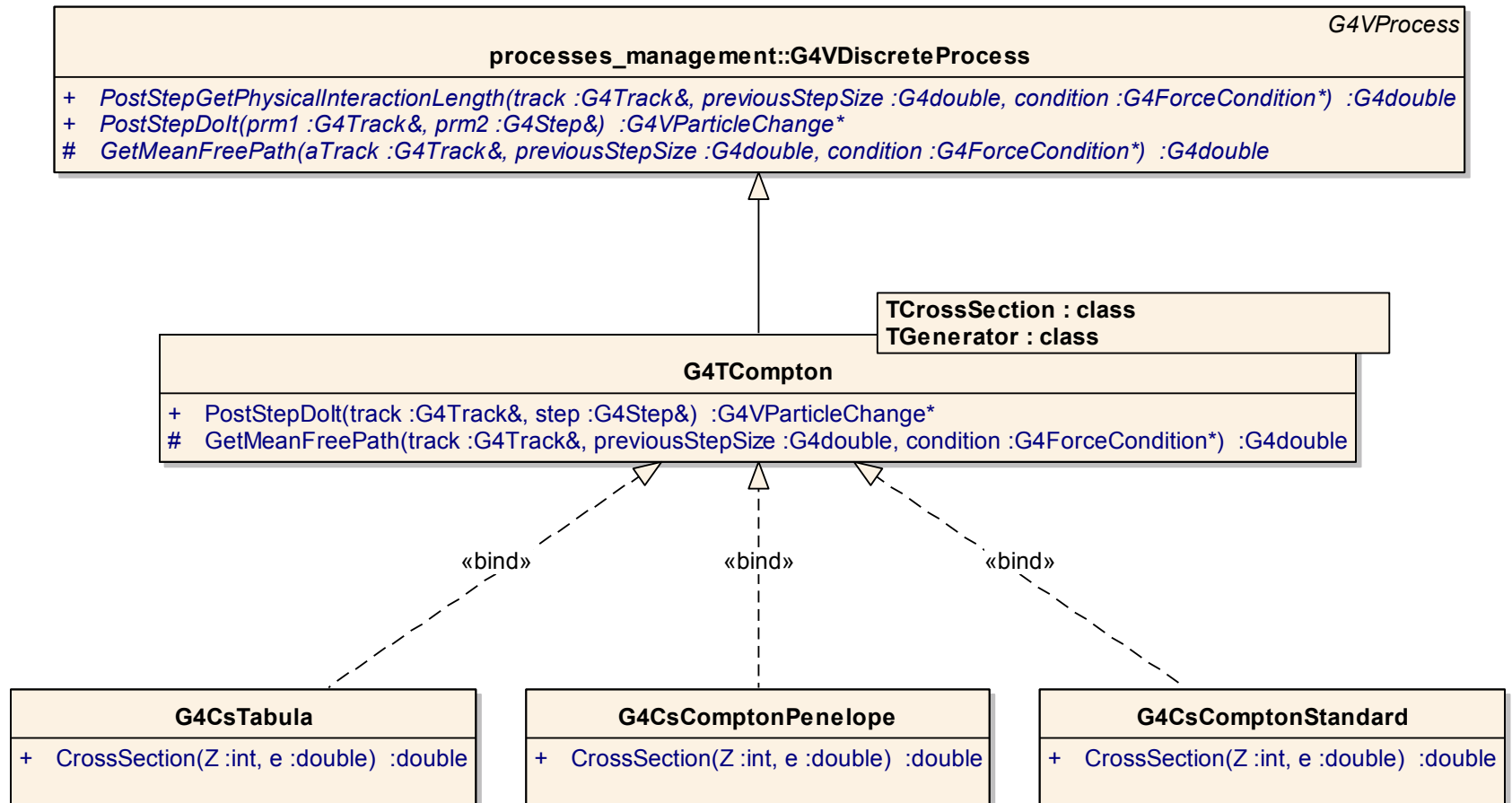
- Long argument list of *ComputeCrossSsectionPerAtom* smells of **Speculative Generality**
- *G4SandiaTable* is a **Large Class** with fuzzy responsibility
 - (what is the relation between calculating photoelectric cross sections and ionization potentials?)...
- *etc.*
- Hunt for more smells!
- List of smells
<http://www.industriallogic.com/wp-content/uploads/2005/09/smellstorefactorings.pdf>

Physicist's attitude

- As physicists, we want to be able to know the cross sections used in our simulation and to compare them with experimental data without creating unnecessary objects that have no relation with a cross section.
- **Refactor** (reengineer) the code and make the photoelectric cross section **testable**
- Apply **Extract Class**
- Create a new *G4CsPhotoelectricAPC* class
- Move the relevant fields and methods from the old *G4SandiaTable*, *G4StaticSandiaData* and *G4PEEffectFluoModel* classes into the new class
- **Hint:** *G4CsComptonAPC* class in `$G4INSTALL/source/processes/electromagnetic/pii/`
 - simplified version of the actual class that will be documented in the forthcoming publication

How could a testable cross section class fit into Geant4

(Simplified) hint about introducing a testable cross section class in Geant4 kernel design



Compiling new or modified classes

- Place new classes in
`$G4INSTALL/source/processes/electromagnetic/pii/`
`pii/include/G4CsPhotoelectricAPC.hh` (header file)
`pii/src/G4CsPhotoelectricAPC.cc` (implementation)
- Whenever you add or modify a class in the pii package, recompile it:
`cd $G4INSTALL/source/processes/electromagnetic/pii/`
`gmake`
- The resulting library will be in `$G4LIB/Linux-g++/libG4empii.a`
- Then build (or rebuild) the corresponding unit test as previously shown

Testing the new class

- Create a test for *G4CsPhotoelectricAPC* class in `$APCDIR/PhotoelectricAPCTest.cc`
- Hint: a test for the similar Compton cross section class in <http://www.ge.infn.it/geant4/training/APC2014/code2/test/ComptonAPCTest.cc>
- Build and run the test:
cd `$APCDIR/test`
setenv TESTTARGET PhotoelectricAPCTest
gmake
`$G4WORKDIR/bin/Linux-g++/PhotoelectricAPCTest`
- How do we verify that *G4CsPhotoelectricAPC* is **correct** and **equivalent** to the cross section calculation in *G4PEEffectFluoModel*?
 - *G4PEEffectFluoModelTest* does not let us test cross sections
 - Compare with the reference plots in the [Sandia87-0070](http://www.sandia.gov/pubs/87-0070)
 - Shortcut for the APC some previously verified reference values in http://www.ge.infn.it/geant4/training/APC2014/exercise2/dataAPC/all_photoelectricAPC.txt
 - ▷ column 6: cross sections with original A_{ij} parameters
 - ▷ column 7: cross sections with improved parameters

Adding functionality to the new class

- How much did the **improved parameters** actually improve the accuracy of photoelectric cross sections?
- We require the capability to calculate cross sections with **original** or **improved** A_{ij} parameters
 - In *G4PEEffectFluoModel* this is not possible, since the parameters are hardcoded in *G4StaticSandiaData*
- Hint: store the parameters in external files and make *G4PhotoelectricAPC* load them when it is instantiated
- For your convenience, files containing the two sets of original and improved A_{ij} parameters are available in $\$APCDIR/parametersAPC/$
- You may pass an argument to the constructor of *G4PhotoelectricAPC* to decide whether to load the original or improved parameters

Comparison with experimental data

Experimental data

\$APCDIR/data/exp_photoelectricAPC.txt

Columns:

1. Published reference identifier
2. Z
3. Energy (keV)
4. Cross section (b)
5. Cross section error (b)

Your test should produce

\$APCDIR/data/all_photoelectricAPC.txt

Columns:

1. Published reference identifier
2. Z
3. Energy (keV)
4. Cross section (b)
5. Cross section error (b)
6. Cross section (b) calculated with original A_{ij} parameters
7. Cross section (b) calculated with improved A_{ij} parameters

Plot photoelectric cross sections with R

- No-frills basic R environment for this simple analysis
 - no GUI, only R core packages, basic graphics etc.
 - you can do much more with a powerful system such as R!
- `$APCDIR/rwork/plotAPC.R` plots experimental and calculated photoelectric cross sections for $Z=10$ (neon) and $Z=18$ (argon)
- Start R:
`cd $APCDIR/rwork`
`R -g Tk &`
- In the R console click on the **File menu** on the top left corner, choose "**Source R code**" and select the `plotAPC.R` script
 - will load the file `$APCDIR/dataAPC/all_photoelectricAPC.txt`
 - will produce the plots in the R graphics window
 - will place a copy in PDF files in the same directory