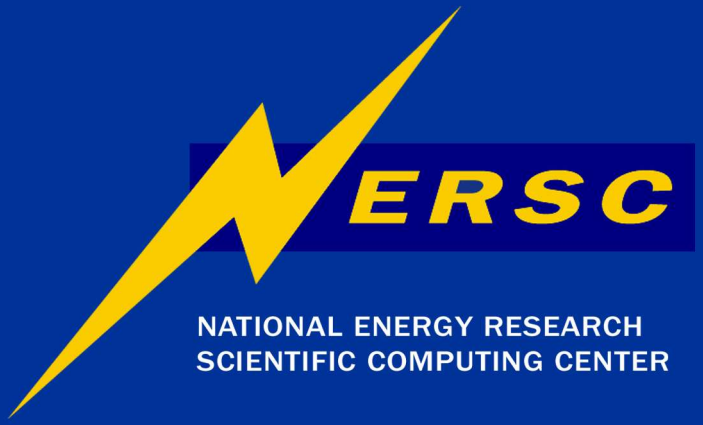


National Energy Research Scientific Computing Center (NERSC)

Physics-Level Job Configuration

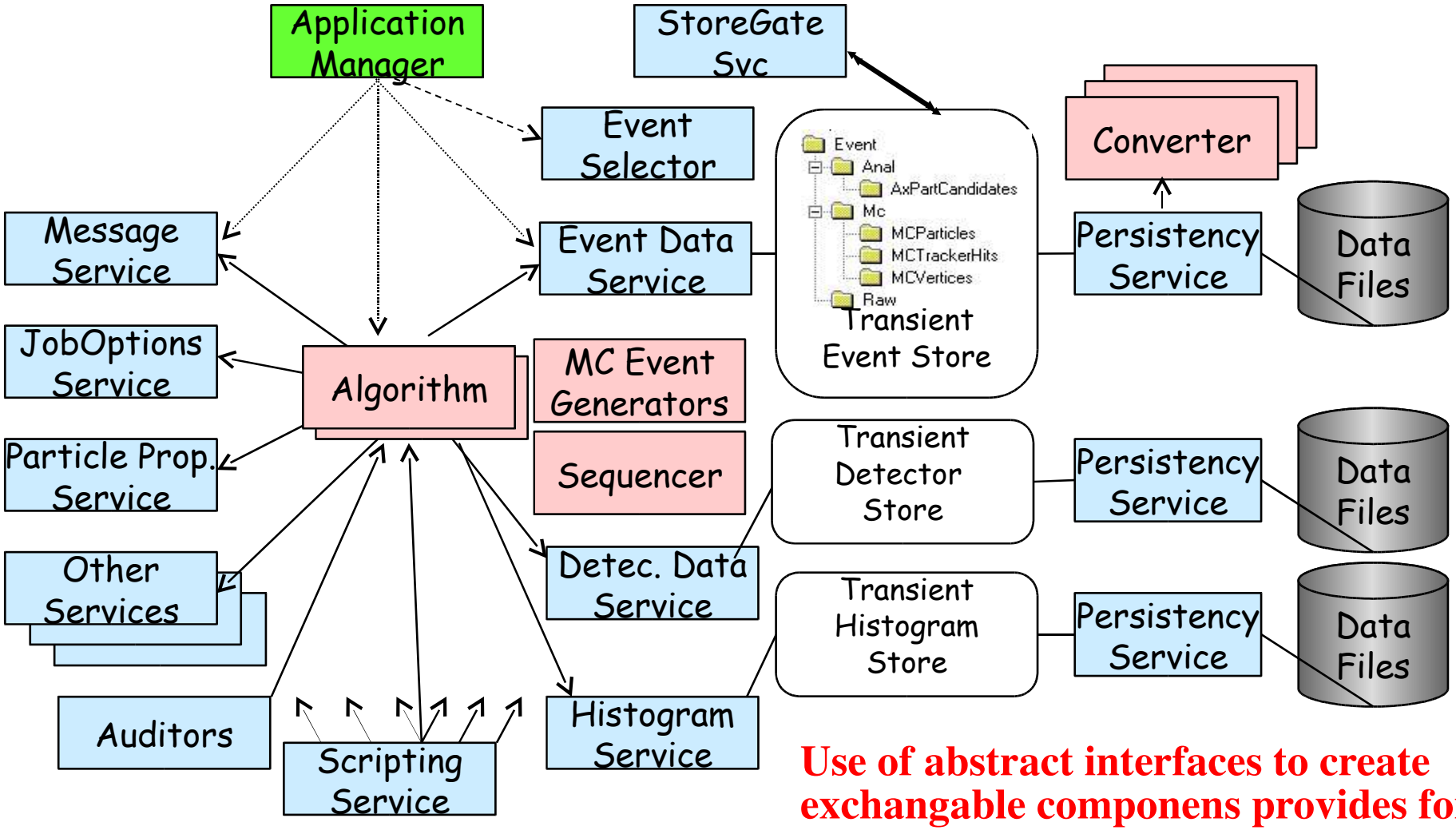
Wim Lavrijsen, Wolfgang Liebig
Paolo Calafiura, Peter Loch,
David Rousseau, Andreas Salzburger

CHEP – Mumbai, February 2006

Upcoming changes in Atlas

- **Software priorities are changing**
  - Development => maintenance
  - More physicist/end-users
    - Writing analysis and running reconstruction
- **Need to be ready for cosmics in 2007**
  - External and core software freeze early
  - Deliver High Level Trigger software
  - Understandability and usability
    - Work environment, configuration, docs

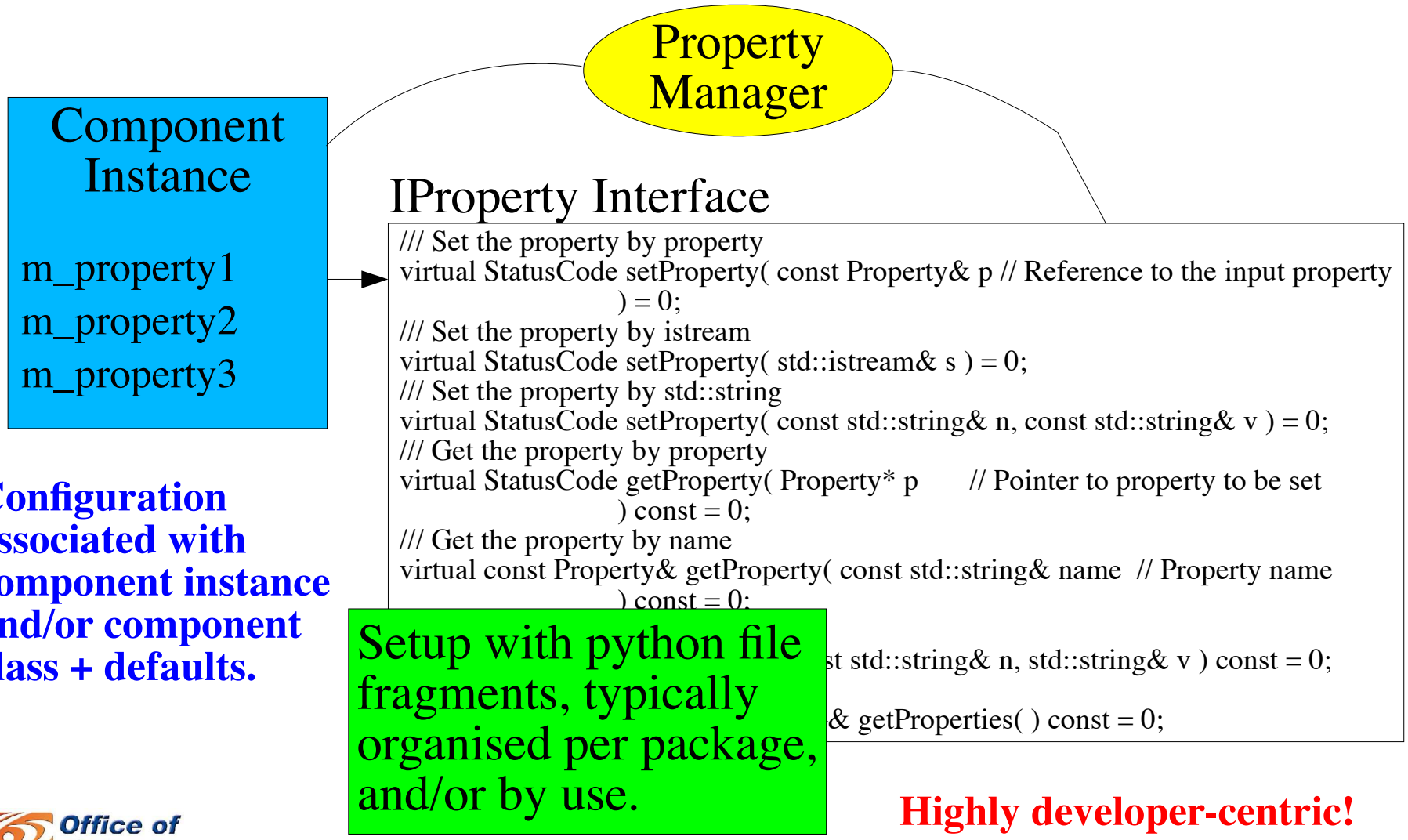
Athena Component Model



Use of abstract interfaces to create exchangable componens provides for flexible development, but also means many components in a single job.

Office of Science
U.S. DEPARTMENT OF ENERGY

Athena Configuration Model



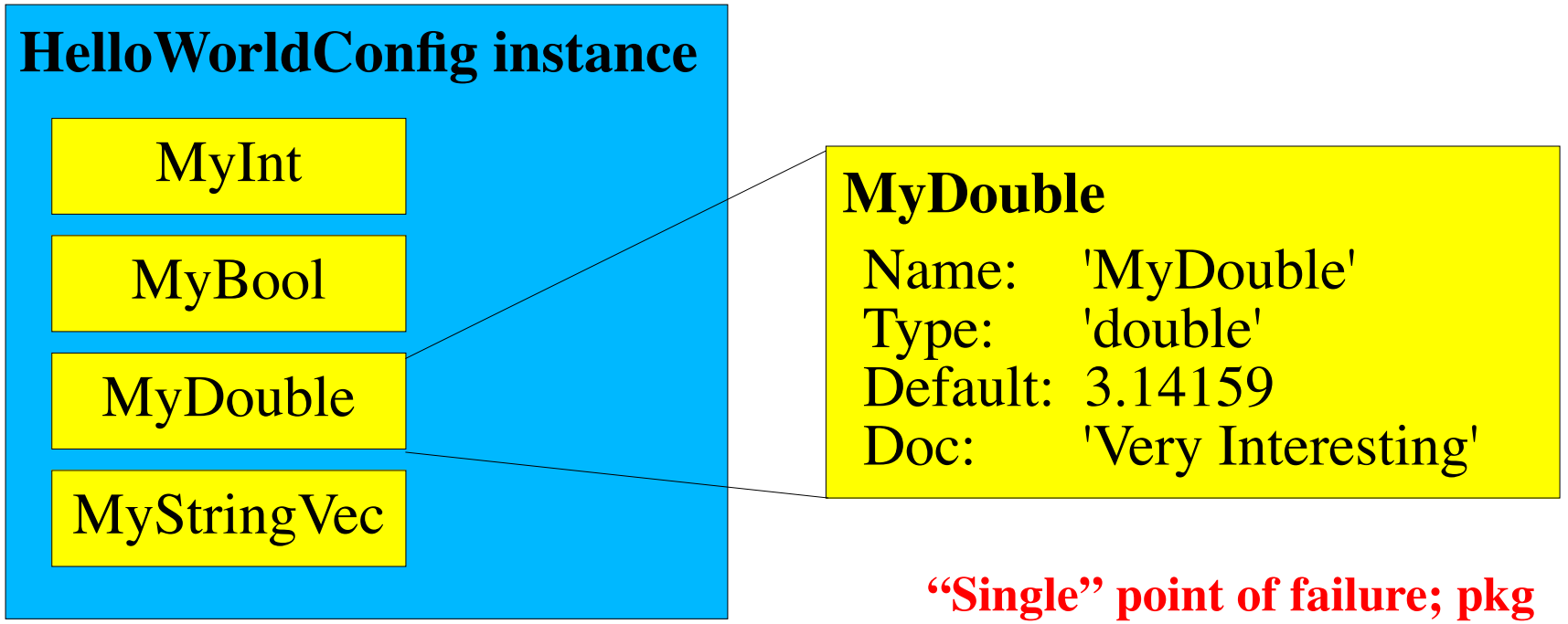
Move to a user-centric model

- **More, specific, file fragments not enough**
  - Reside in developer pkg: wrong granularity
  - Localizing fragments is in itself a problem
    - As are mixing, matching, and multiple inclusions
- **Solution is a three step process:**
  1) *Provide smarter, low-level building blocks*
     - Automatically generated for all components
  2) *Provide structuring support*
     - Removes boiler-plate code from options files
  3) *Build higher level structures*
     - Driven by physics/developer community

1) “Configurables”

```
# one python module per package, one python class per component
from AthExHelloWorld.AthExHelloWorldConfig import HelloWorldConfig

# make a Configurable available only (no Athena-side configuration yet)
HelloWorld = HelloWorldConfig( “HelloWorld” )
HelloWorld.MyDouble = 6.6261      # <- can be immediately verified
```



“Single” point of failure; pkg structure only used for lookup; fully mix&match safe.

1) Automatically generated

- **<MyPackage>Config.py generated with:**

```
$ genconf MyPackage     # single script
```

  - Assumes components in 'libMyPackage.so'
  - Works for Algorithm, Service, & AlgTool

- **Build-time Config.py generation:**

```
$ cat <MyPackage>/cmt/requirements
[...]
apply_pattern genconf_run
[...]
```

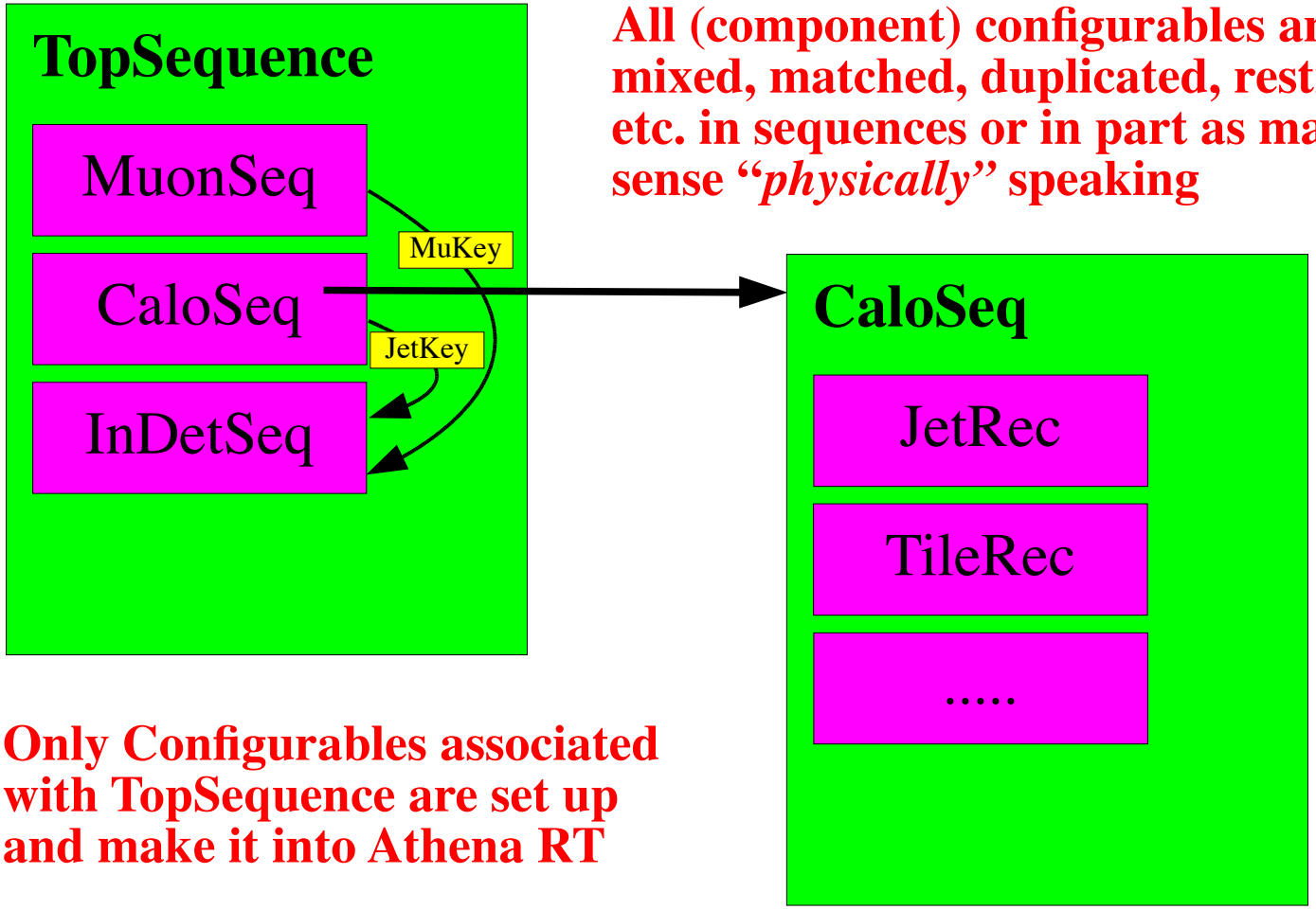
1) Specialisation

- **Derive from any class in <>Config.py**
  - Implement hook for specialization:

```
class MyClassSpecial( MyClassConfig ):
    def setUserDefaults( self, handle ):
        handle.special = specialValue
```

  - Add dependent components in __init__
  - Override other methods such as getDlls()
  - Used like any other Configurable
  - C++ style type checks help you get it right

2) Structure support



All (component) configurables and can be mixed, matched, duplicated, restructured, etc. in sequences or in part as makes sense "*physically*" speaking

Only Configurables associated with TopSequence are set up and make it into Athena RT

2) Pseudo-code: CaloSeq

```
def ConfigureCone4Jets( .., MinimumSignal = 10.*GeV, .. ):
 # setup Cone4Jets with private tools
   Cone4Jets = JetAlgorithmConfig( "Cone4Jets" )
   FinalEtCut = JetSignalSelectorToolConfig( "FinalEtCut" )
   FinalEtCut.UseTransversEnergy = True
   FinalEtCut.MinimumSignal      = MinimumSignal

 # [... other tools ...]
   Cone4Jets += [ ..., FinalEtCut, ... ]
   return Cone4Jets

def ReconstructClusters():
   sequence = AlgSequence( "Clusters" )
 # [... other algorithms ...]
   sequence += ConfigureCone4Jets()
   return sequence


topSeq += ReconstructClusters()
topSeq.Clusters.Cone4Jets.FinalEtCut.MinimumSignal = 20.*GeV
```

Make a tree of Configurables: explorable, modifiable.

Structural placeholder.

End-user overrides always take precedence.

3) Physics-level

- "Physics-level" different meaning ...
  - ... for subdetector developer,
  - ... for reconstruction coordinator,
  - ... for member of heavy-Higgs group, etc.
- Several developer-user relations
  - Each relation is a layer
  - Users now work with logical blocks
    - Provided by developer or "power-user"
    - Guaranteed internal consistency
    - Overrides possible for (temp!) workarounds

Summary & Outlook

- **Atlas software is changing:**
  - development focus => analysis focus
- **Configuration building blocks provided**
  - Auto-generated, checkable, independent
- **Layered structures now possible**
  - With layered builders (functions/classes)
  - End-user modifiable, exploration-safe
- **Opens up possibilities for new tools**
  - Browsers, validators, code generators, etc.

Resources

- **Ongoing documentation:**
  - https://uimon.cern.ch/twiki/bin/view/Atlas/HighLevelJobConfiguration
  - https://uimon.cern.ch/twiki/bin/view/Atlas/PrototypeDataDrivenConfig#Prototype_with_Configurable