Re-thinking the Hadronic Framework

Geant4 Collaboration Workshop
12 September 2012
Dennis Wright (SLAC)

Why Change the Hadronic Framework?

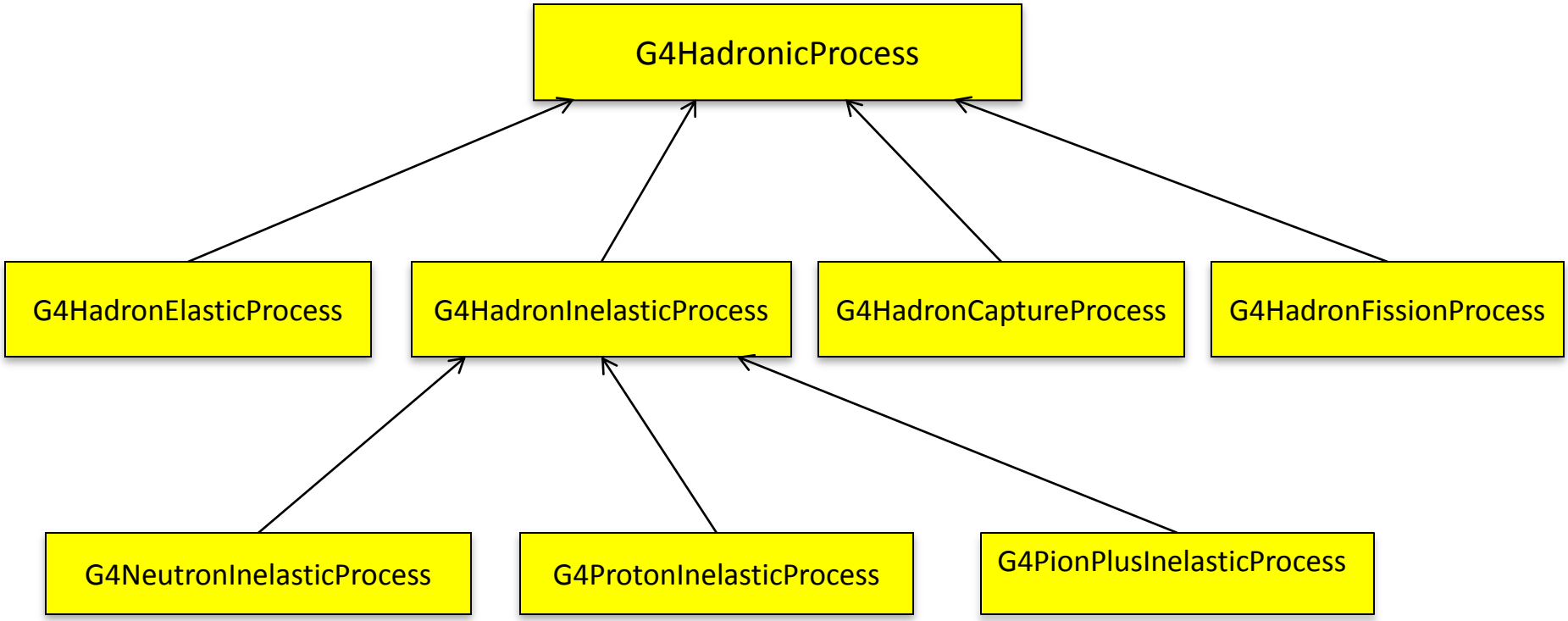
- It may not be flexible enough
  - does not easily accommodate all hadronic processes, models and cross sections
- It may be too deep
  - too many levels of inheritance which complicate and slow down code
- It may give too much (or not enough) control to users
  - should we consider more defaults (other than cross sections)?
- With a major release in the foreseeable future are there changes we want to make in the interfaces?

Removing Inheritance Layers

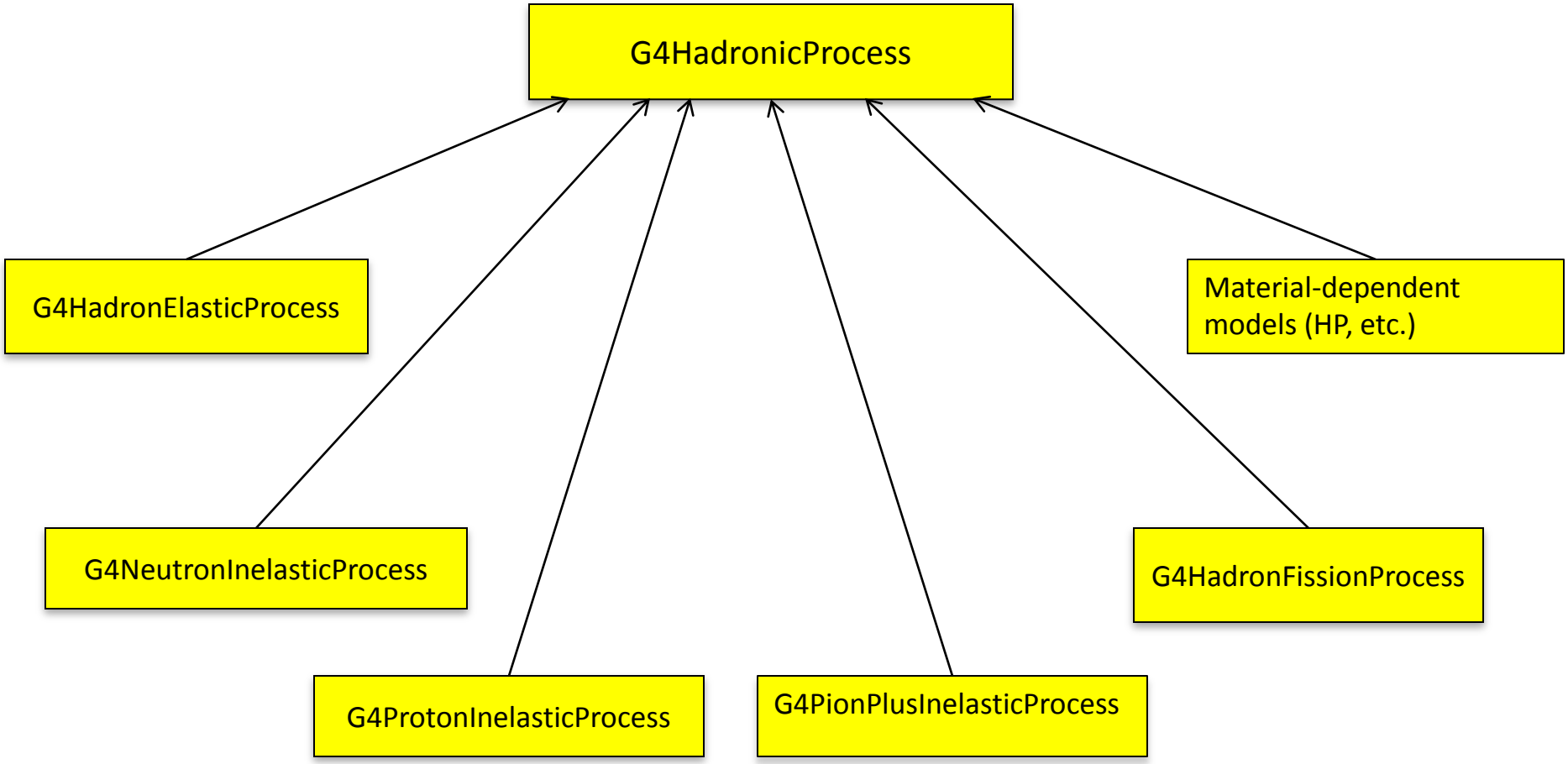
- Suggestion by Makoto that removing inheritance layers could
  - significantly reduce execution time
  - improve multi-threaded, multi-CPU behavior

- Tatsumi proposed removing a layer in hadronic process inheritance
  - remove G4HadronInelasticProcess (doesn't do much)
- Dennis proposed removing layers in model inheritance

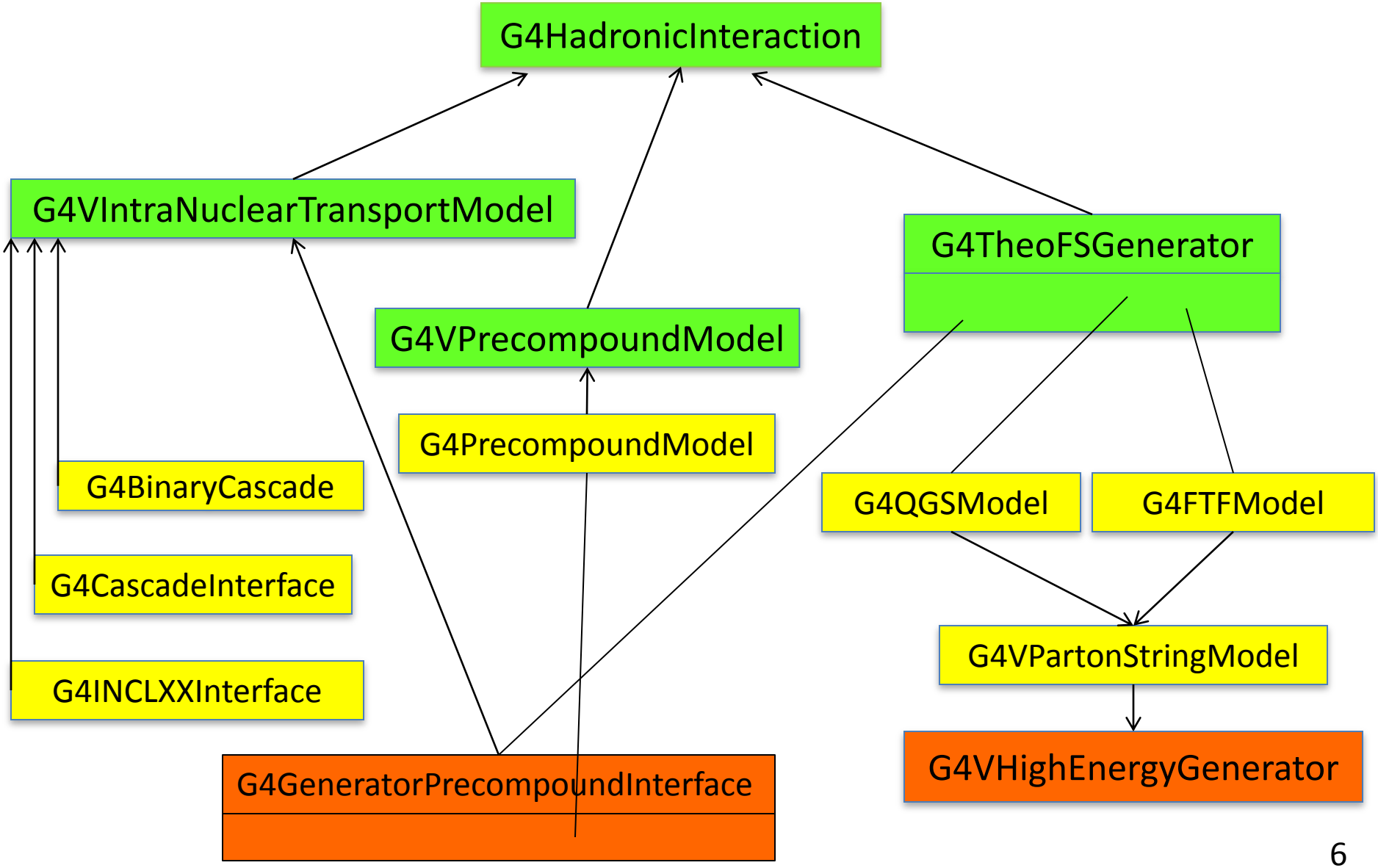
Current Inheritance Scheme for In-flight Hadronic Processes



Proposed Inheritance Scheme for In-flight Hadronic Processes



Model Inheritance



Do We Need Capture and Fission Processes?

- Some models now handle capture automatically
  - -> make capture a part of in-flight process

- Fission is already part of several in-flight models
  - G4LFission (GHEISHA) only active, stand-alone fission model

At-rest Processes

- A consistent treatment of in-flight and stopping processes is desired
  - current stopping models do not derive from G4HadronicProcess because it in turn derives from G4VDiscreteProcess
  - derived directly from G4VRestProcess instead
- Drawbacks:
  - stopping and in-flight processes can never be treated equally
  - stopping processes cannot inherit useful methods from G4HadronicProcess
  - model approach of framework cannot be used
- Solutions:
  - derive all hadronic processes from G4VRestDiscreteProcess, or
  - make all stopping processes in-flight

High Precision Neutrons

- HP and LEND models require material pointers and do their own sampling of isotopes
  - this adds a lot of complication to G4HadronicProcess and to cross section classes
  - such complication is not required for any other model
  - specialized inheritance for HP and LEND
  - possible inheritance diagram on slide 5
- G4MaterialDependentNeutronProcess
  - would have G4Material pointer
  - other processes would not have material pointer
  - would do its own isotope selection

Cross Section Review and Clean-up

- Cross section classes still not handled clearly or consistently
- Re-design completed more than a year ago
  - some planned migrations completed, not all
  - end result not very satisfactory
  - one reason: material dependence of HP neutron models
- Factory-based mechanism to assure a single instantiation of a cross section which may be used by more than one different entity
- General means for smooth blending of one cross section set into another vs. energy

Framework Rules

- Currently have default cross sections but not default models
  - add default models?
- G4HadFinalState
  - Currently must copy into particle change
  - can we modify particle change classes to avoid this copying?
- User hooks into hadronic models – good idea or bad idea?