Geant 4

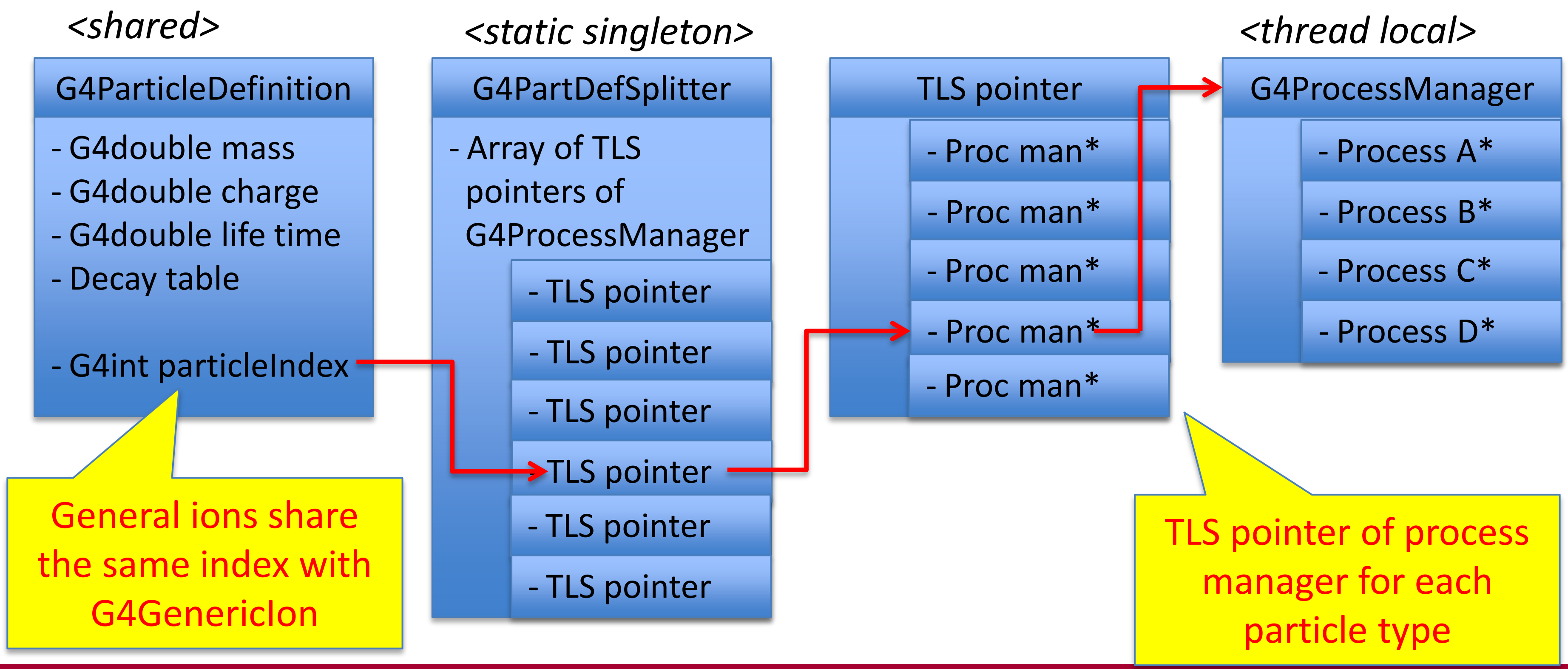
Use of Ions in MT

Makoto Asai

2013 Geant4 Collaboration Meeting @ Seville

Split class – case of particle definition

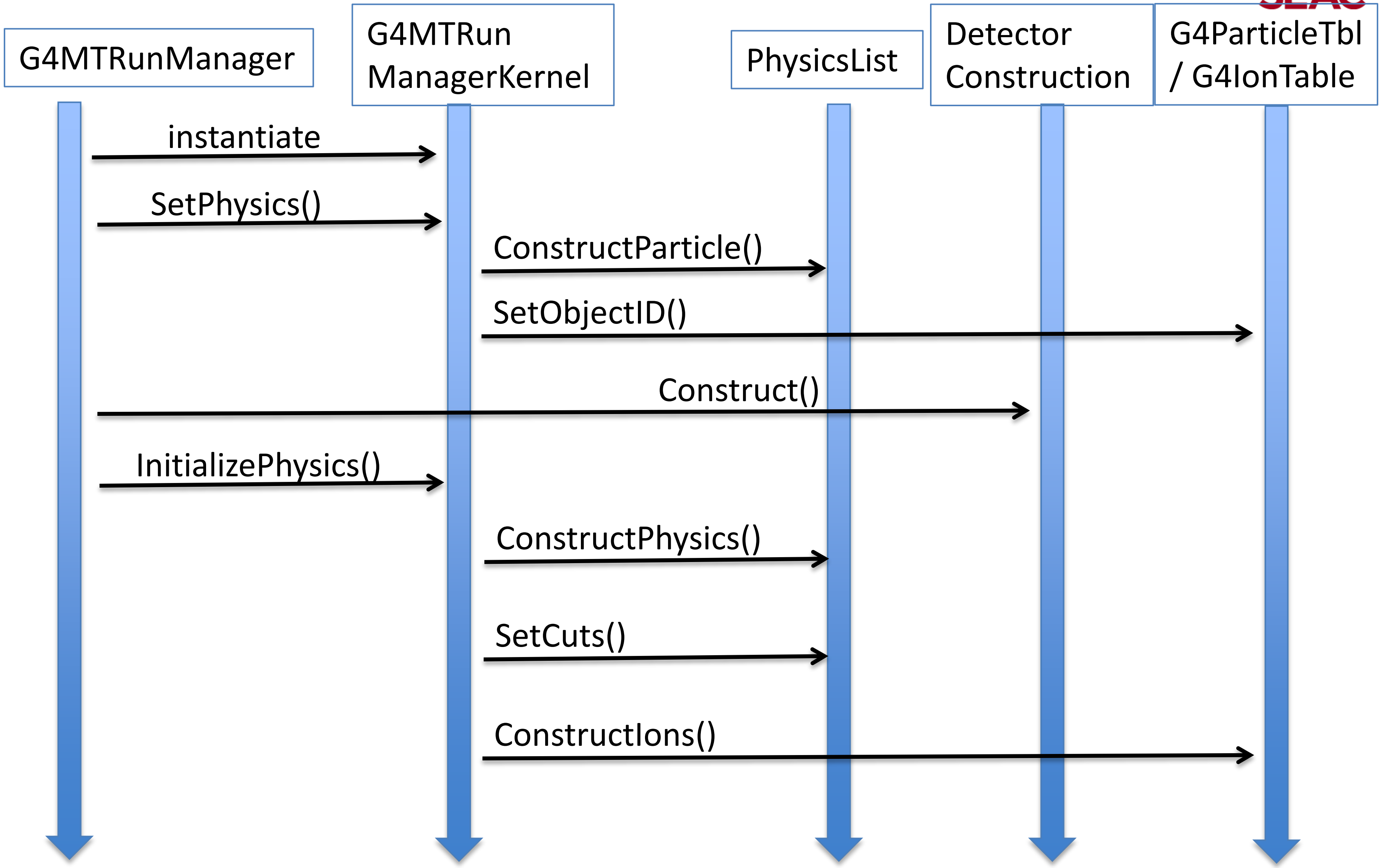
- In Geant4, each particle type has its own dedicated object of G4ParticleDefinition class.
  - Static quantities : mass, charge, life time, decay channels, etc.,
    - To be shared by all threads.
  - Dedicated object of G4ProcessManager : list of physics processes this particular kind of particle undertakes.
    - Physics process object must be thread-local.



Use of Ions in MT

- With the following minor restrictions, ions could be instantiated on the fly.
  - G4GenericIon must be defined in the physics list.
  - Ions to be created on the fly must be "general ions" that are the objects of G4Ions class and share exactly the same G4ProcessManager object with G4GenericIon.
    - So-called light ions are not included in this general ion category. They have to be pre-defined. Actually these light ions are automatically pre-defined if G4GenricIon is defined.
    - G4ParticleDefinition has a new method IsGeneralIon(), that returns true if the particle is general ion that share the G4ProcessManager object with G4GenericIon. G4GenericIon itself is not included.
  - Such ions must be created through G4IonTable::GetIon() methods. No direct instantiation of G4Ions class object is allowed.
  - Creating an ion on the fly costs some performance penalty.
    - Geant4 kernel will consult to G4NuclideTable (new class, Tatsumi and Dennis are working on it) about the creation of "default" ions before the event loop. We expect the number of such "default" ions to be reasonably small so that they won't cause much of the memory consumption problem, but it should be tunable on use-cases (again, Tatsumi and Dennis are working on it).

Before/During run_initialization in master thread

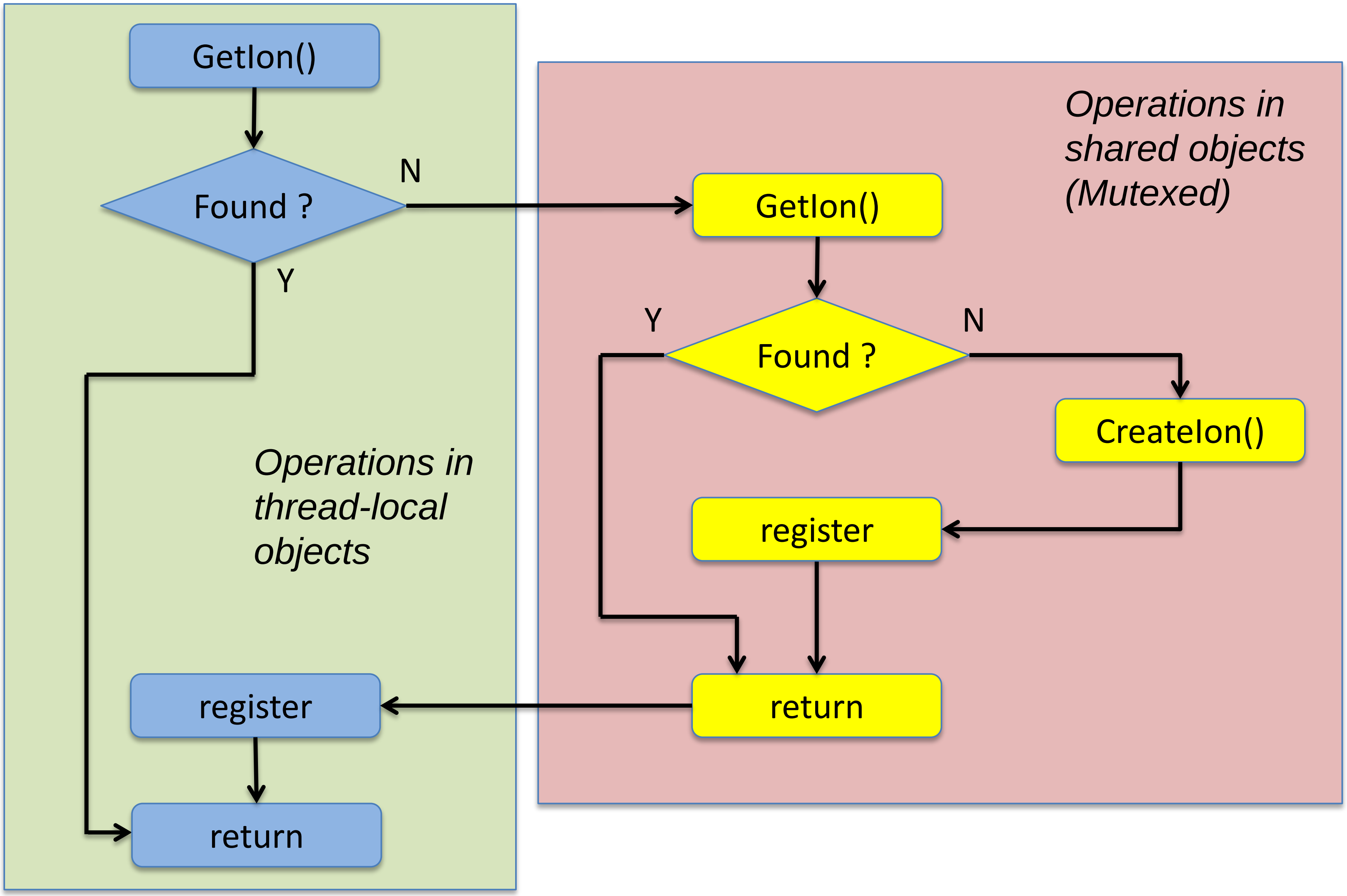


Particle iterator is now intelligent

```
G4ParticleTable::G4PTblDicIterator* theParticleIterator
  = G4ParticleTable::GetParticleTable()->GetIterator();
theParticleIterator->reset();
while( (*theParticleIterator)() ) {
      G4ParticleDefinition* particle = theParticleIterator->value();
```

- If the particle iterator is reset() without an argument, it skips general ions. If it is reset(false), it iterates also with all general ions that are registered to the G4ParticleTable so far.

How G4IonTable::GetIon() works in worker thread



Warning message

```
-------- WWWW ------- G4Exception-START -------- WWWW -------
*** G4Exception : PART11117
      issued by : G4ParticleTable::FindIon()
This method is obsolete and will be dropped from v10.0. Use
G4IonTable::FindIon().
*** This is just a warning message. ***
-------- WWWW -------- G4Exception-END --------- WWWW -------
```

- GetIon() and FindIon() methods in G4ParticleTable are obsolete and should not be used any more.
- Use equivalent (and enhanced) methods in G4IonTable.

```
G4ParticleTable::GetParticleTable() -> GetIon(…);
```

Should be replaced by

```
G4IonTable::GetIonTable() -> GetIon(…);
```

- Currently, this warning message is issued. Soon these methods will be actually removed, so that classes that still use these methods won't compile any more.

A request

- Don’t do

  ionTable -> GetIon(“ion_name”) -> GetMass();

  ionTable -> GetIon(A, Z, lvl) -> GetMass();

- Instead, do

  ionTable -> GetMass(“ion_name”);

  ionTable -> GetMass(A, Z, lvl);

- GetIon() method creates an ion object. If the mass is the only information you need, don’t create the ion.
- For the full list of GetIon(), FindIon(), GetMass() and other methods on ions, please consult to G4IonTable class.