Geant 4

Kernel updates

Makoto Asai

2013 Geant4 Collaboration Meeting @ Seville

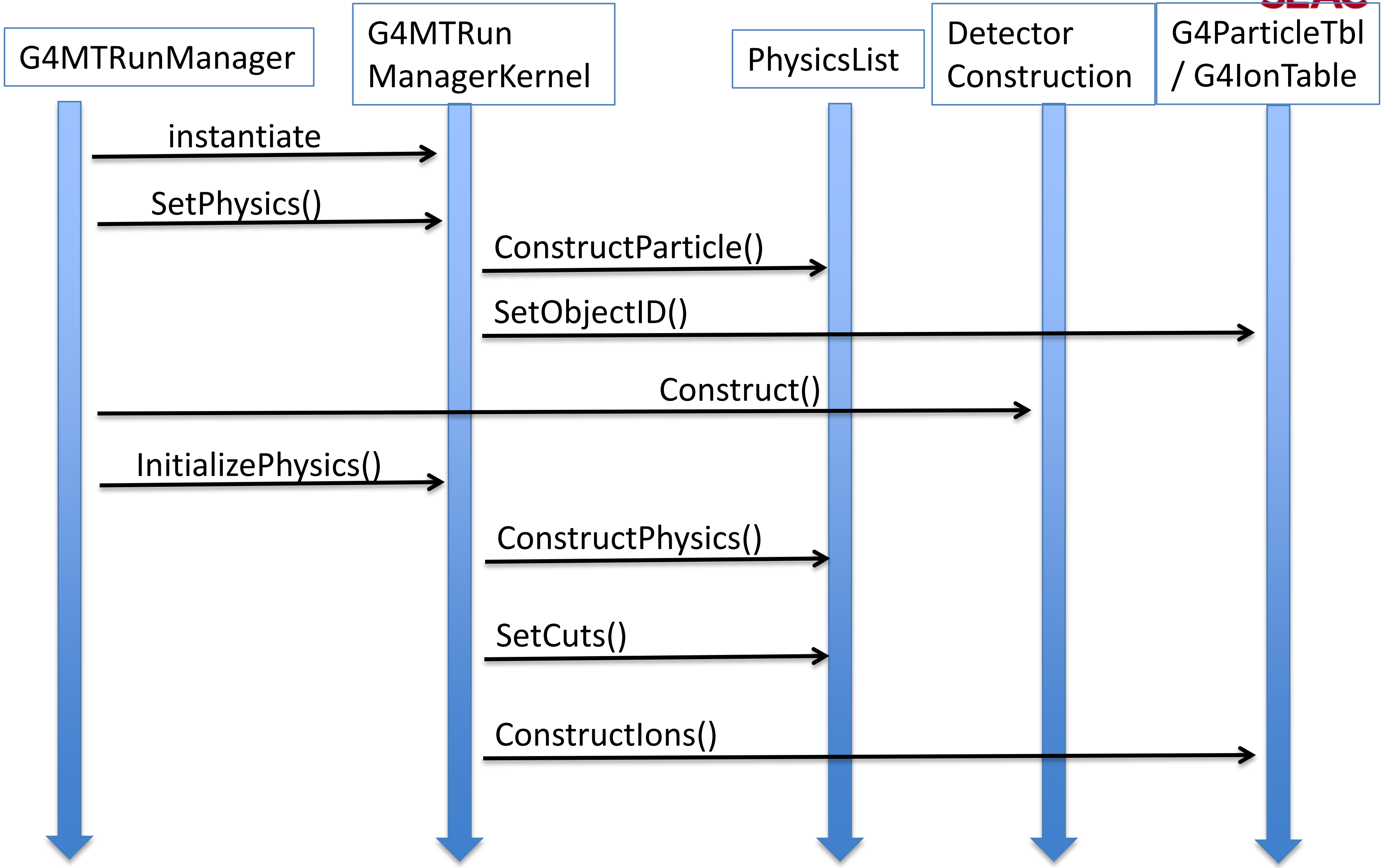
SLAC NATIONAL ACCELERATOR LABORATORY

U.S. DEPARTMENT OF ENERGY
Office of Science

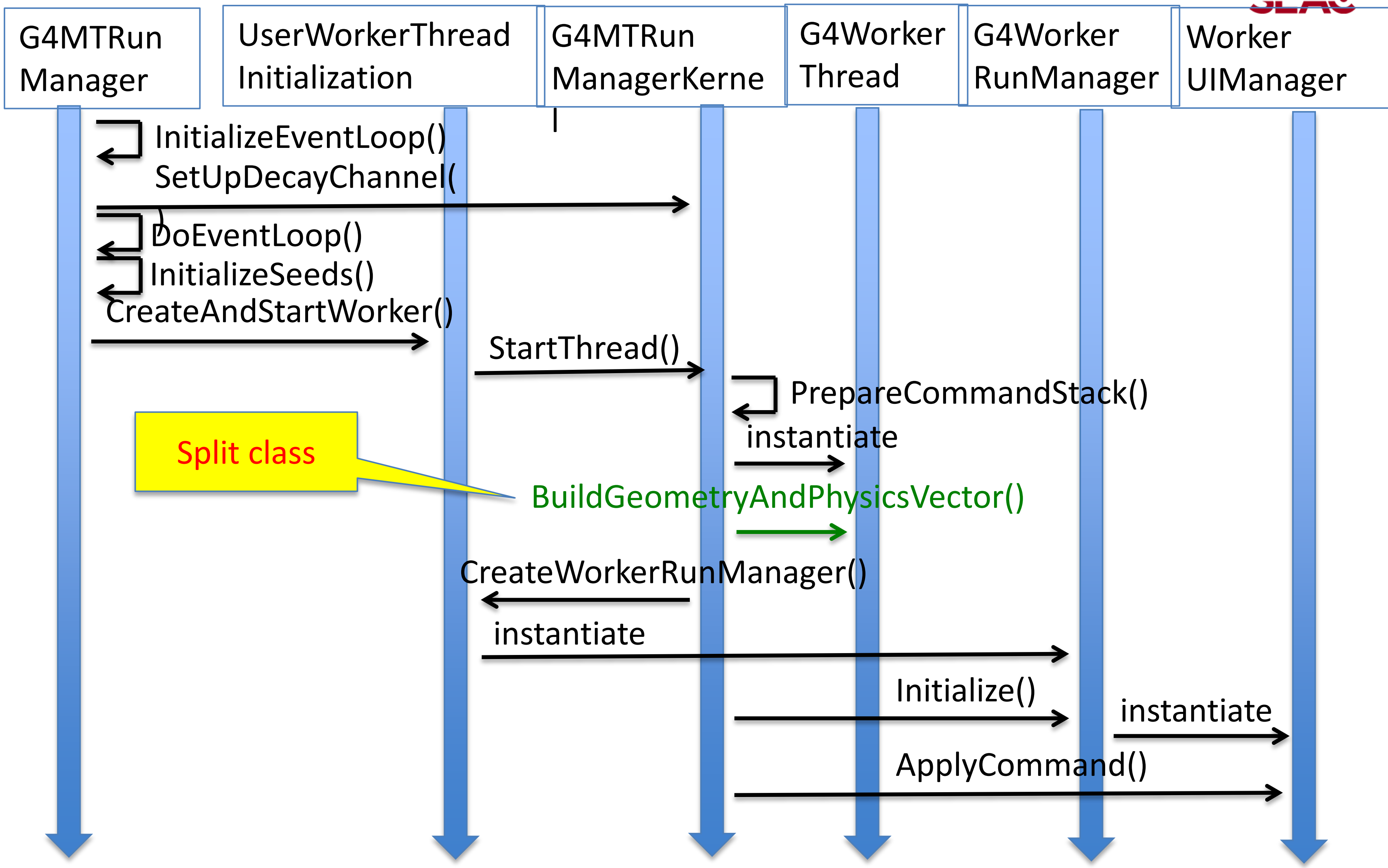
Kernel updates

- There is almost no update in the categories of run, event, digits_hits and particles except for the massive modifications related to multi-threading.
- So, taking this opportunity, let me explain the major code flow taken care by these categories in multi-threaded mode.
  1. Before/During run_initialization in master thread
  2. BeamOn in master thread
  3. During run_initialization in worker thread
  4. BeamOn in worker thread
- Also, let me add a few notes on split class and also on some error messages.

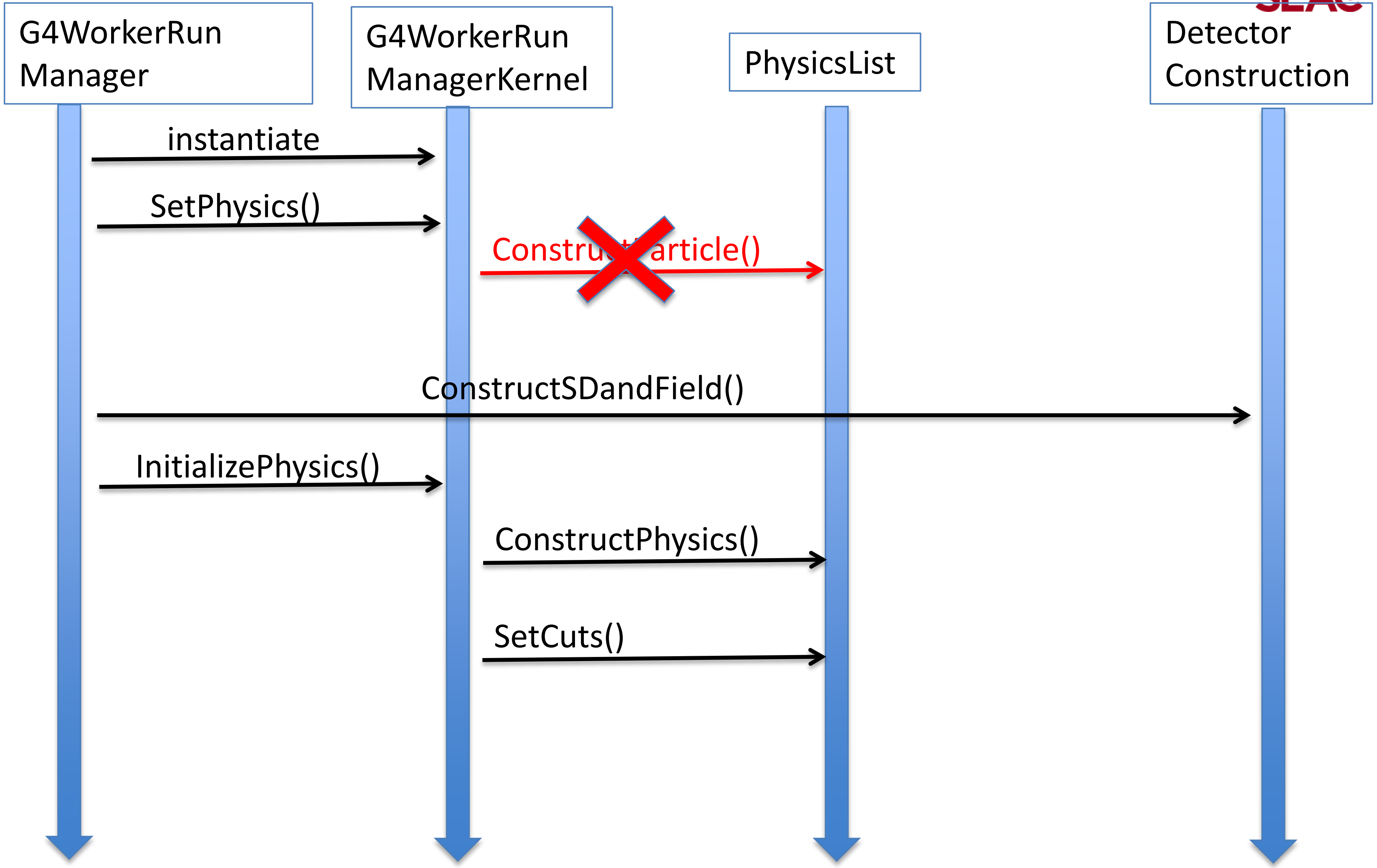
Before/During run_initialization in master thread



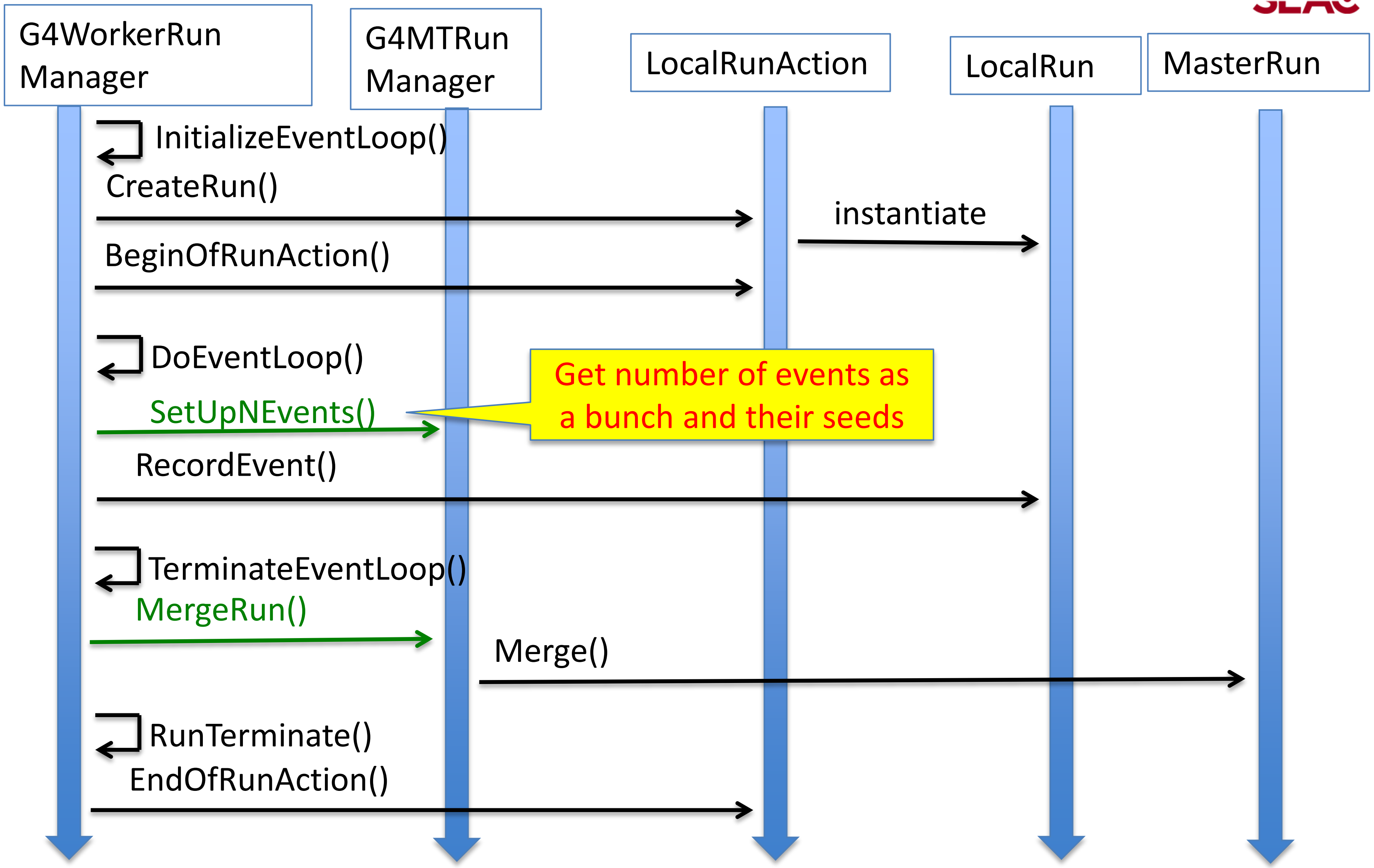
BeamOn in master thread



During run_initialization in worker thread

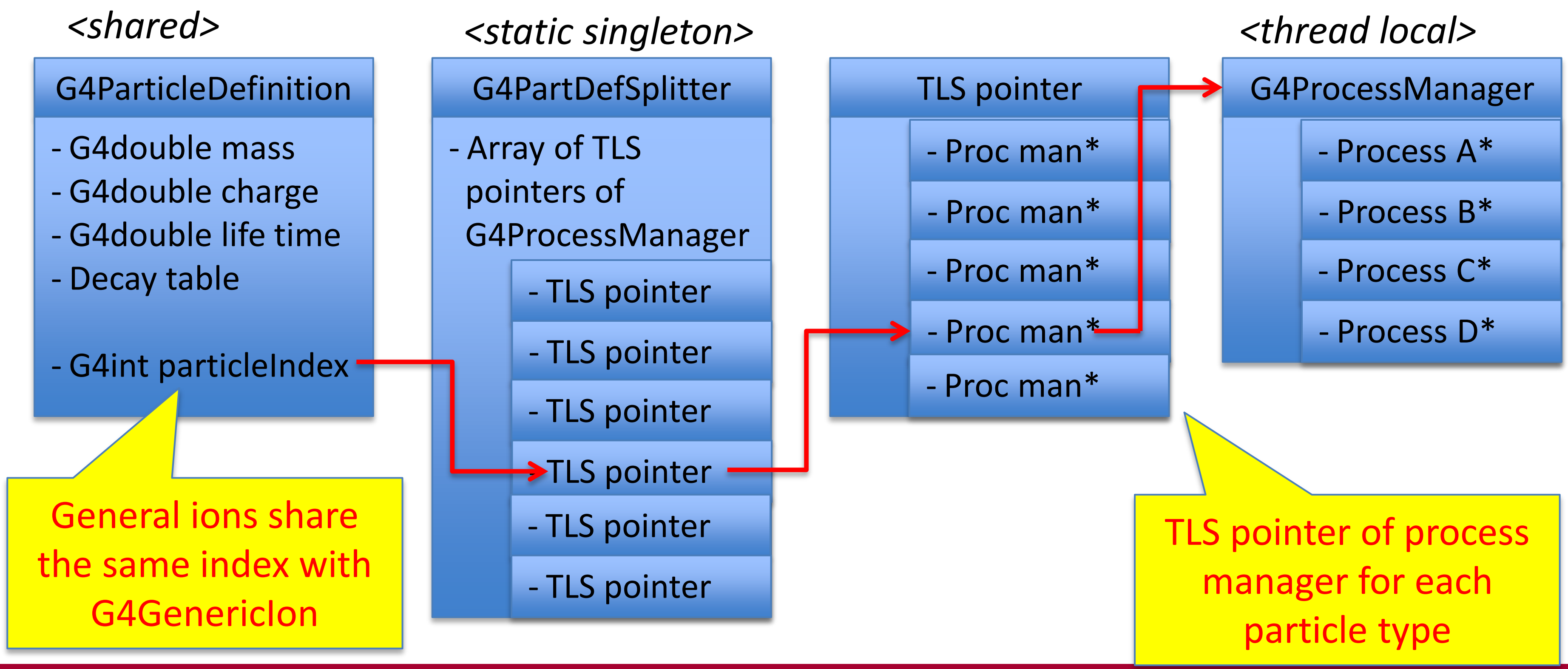


During BeamOn in worker thread



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.



Error message - 1

```
-------- EEEE ------- G4Exception-START -------- EEEE -------
*** G4Exception : Run0035
      issued by : G4RunManagerKernel::G4RunManagerKernel()
Size of G4ProcessVector is inconsistent between master and worker threads
  for the particle <B+>.
size of G4ProcessVector for worker thread is 4 while masther thread is 5.
*** Fatal Exception *** core dump ***
----------------------------------------------------
```

- Check your physics list. Most-likely your ConstructPhysics() method or its granular method has a data member Boolean flag or something similar to protect not to execute this method more than once.
- Physics list is a singleton object, while this ConstructPhysics() method is invoked for each worker thread.

Error message - 2

```
-------- EEEE ------- G4Exception-START -------- EEEE -------
*** G4Exception : PART122
      issued by : G4ParticleTable::Insert()
The particle geantino  has already been registered in the Particle Table
*** Fatal Exception *** core dump ***
-------- EEEE -------- G4Exception-END --------- EEEE -------
```

- Something in the worker thread tries to create a new particle.
  - It is not PhysicsList::ConstructParticle(), that is not invoked in a worker thread.
- This is not allowed except for general ions.

Error message - 3

```
-------- 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 similar (and enhanced) methods in G4IonTable.
- 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.