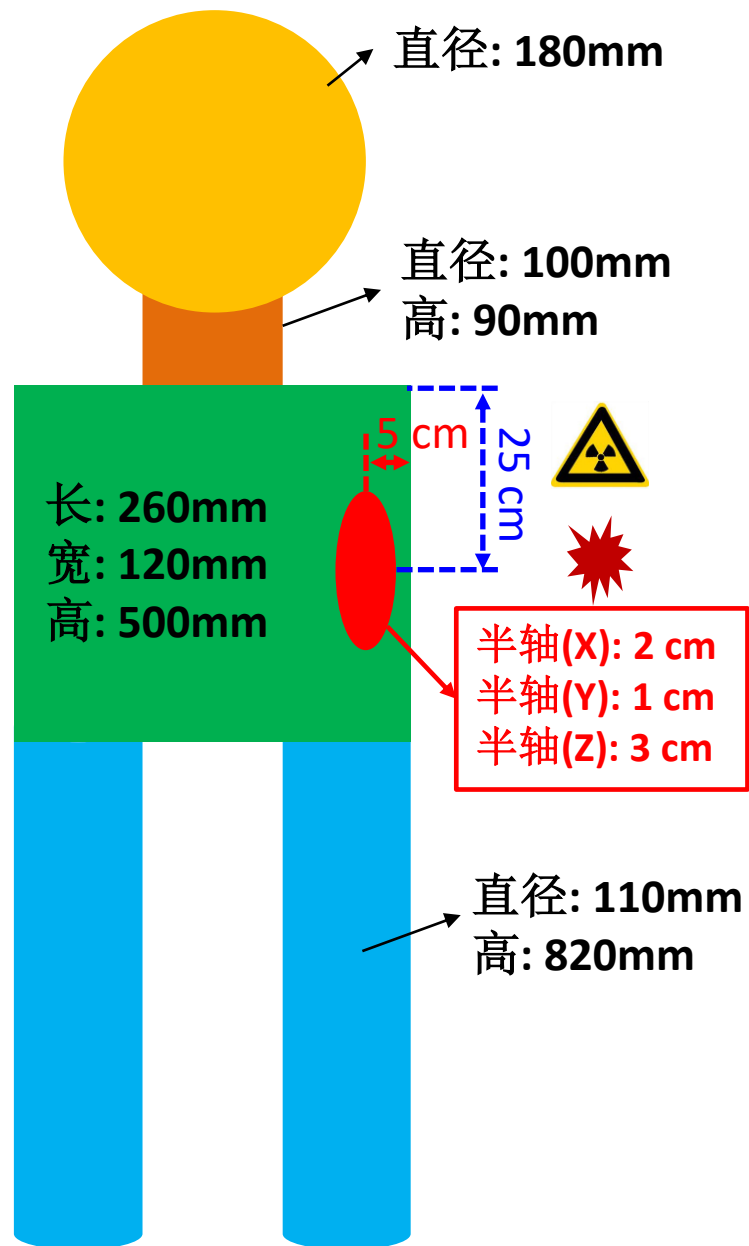


探测器模拟与数据分析

--探测器模拟基础

曹国富

中国科学院高能物理研究所



- ✧ 放疗是肿瘤治疗的一种常用手段，射线种类有伽马/质子/中子/离子等。
- ✧ 请设计并实现一个探测器模拟程序，从能量沉积的大小、尺度和位置几方面，对伽马和质子射线能量进行优化，对比使用这两种射线进行放疗的优劣。
- ✧ 硼中子俘获疗法(BNCT)是一种新的疗法，2020年全球首个BNCT设备在日本获批上市。假如使用**0.5 eV**的热中子束流，请研究预期治疗效果与肿瘤区内¹⁰B原子浓度的关系，以及束流截面大小的影响。最后，与其它射线的治疗效果进行对比。
- ✧ 肿瘤大小和位置如图红色区域所示。

- ❄ **Geant4 provides a number of primitive scorers, each one accumulating one physics quantity (e.g. total dose) for an event.**
- ❄ **G4MultiFunctionalDetector is a concrete class derived from G4VSensitiveDetector**
 - It should be assigned to a logical volume as a kind of sensitive detector
 - It takes an arbitrary number of G4VPrimitiveScorer classes, to define the scoring quantities that you need
 - Each G4VPrimitiveScorer accumulates one physics quantity for each physical volume
 - E.g. G4PSDoseScorer (a concrete class of G4VPrimitiveScorer provided by Geant4) accumulates dose for each cell
 - Each G4VPrimitiveScorer is automatically named as **<MultiFunctionalDetectorName>/<PrimitiveScorerName>**
- ❄ **By using this approach, no need to implement sensitive detector and hit classes!**

```
MyDetectorConstruction::ConstructSDandField()  
{  
    G4MultiFunctionalDetector* myScorer = new  
    G4MultiFunctionalDetector("myCellScorer");  
  
    myCellLog->SetSensitiveDetector(myScorer);  
  
    G4VPrimitiveScorer* totalSurfFlux = new  
        G4PSFlatSurfaceFlux("TotalSurfFlux");  
    myScorer->RegisterPrimitive(totalSurfFlux);  
  
    G4VPrimitiveScorer* totalDose = new  
        G4PSDoseDeposit("TotalDose");  
    myScorer->RegisterPrimitive(totalDose);  
}
```

instantiate multi-functional detector

attach to volume

create a primitive scorer (surface flux) and register it

create a primitive scorer (total dose) and register it

❄ **Concrete Primitive Scorers (→ Application Developers Guide 4.4.5)**

➤ **Track length**

- **G4PSTrackLength, G4PSPassageTrackLength**

➤ **Deposited energy**

- **G4PSEnergyDeposit, G4PSDoseDeposit**

➤ **Current/Flux**

- **G4PSFlatSurfaceCurrent,**
- **G4PSSphereSurfaceCurrent, G4PSPassageCurrent,**
- **G4PSFlatSurfaceFlux, G4PSCellFlux, G4PSPassageCellFlux**

❄ **Others**

- **G4PSMinKinEAtGeneration, G4PSNofSecondary, G4PSNofStep, G4PSCellCharge**

❄ **A G4VSDFilter can be attached to G4VPrimitiveScorer to define which kind of tracks have to be scored (e.g. one wants to know surface flux of **protons** only)**

- **G4SDChargeFilter** (accepts only **charged** particles)
- **G4SDNeutralFilter** (accepts only **neutral** particles)
- **G4SDKineticEnergyFilter** (accepts tracks in a defined range of **kinetic energy**)
- **G4SDParticleFilter** (accepts tracks of a given **particle type**)
- **G4VSDFilter** (base class to create user-customized filters)

```
MyDetectorConstruction::ConstructSDandField()
```

```
{
```

```
    G4VPrimitiveScorer* protonSurfFlux  
    = new G4PSFlatSurfaceFlux("pSurfFlux");
```

} create a primitive
scorer (**surface
flux**), as before

```
    G4VSDFilter* protonFilter = new  
        G4SDParticleFilter("protonFilter");  
    protonFilter->Add("proton");
```

} create a **particle
filter** and add
protons to it

```
    protonSurfFlux->SetFilter(protonFilter);
```

} register the **filter**
to the primitive
scorer

```
    myScorer->RegisterPrimitive(protonSurfFlux);
```

```
}
```

register the **scorer** to the
multifunc detector (as
shown before)

```
//needed only once
G4int collID = G4SDManager::GetSDMpointer()
    ->GetCollectionID("myCellScorer/TotalSurfFlux");
G4HCofThisEvent* HCE = event->GetHCofThisEvent();
G4THitsMap<G4double>* evtMap =
    static_cast<G4THitsMap<G4double>*>
    (HCE->GetHC(collID));
for (auto pair : *(evtMap->GetMap())) {
    G4double flux = *(pair.second);
    G4int copyNb = *(pair.first);
}
```

Get **ID** for the collection (given the name)

Get **all HC** available in this event

Get the HC with the **given ID** (need a cast)

Loop over the **individual entries** of the HC: the key of the map is the copyNb, the other field is the real content


```
//needed only once
G4int collID = G4SDManager::GetSDMpointer()
    ->GetCollectionID("myCellScorer/TotalSurfFlux");
G4HCofThisEvent* HCE = event->GetHCofThisEvent();
G4THitsMap<G4double>* evtMap =
    static_cast<G4THitsMap<G4double>*>
    (HCE->GetHC(collID));
```

Get **ID** for the collection (given the name)

Get **all HC** available in this event

Get the HC with the **given ID** (need a cast)

	HCofThisEvent	
	Scorer 1	Scorer 2
Event#1	(0, 5.32)	<div style="border: 1px dashed black; padding: 5px; display: inline-block;"> (0, 1.43) (2, 7.41) </div>

❄ **UI session**

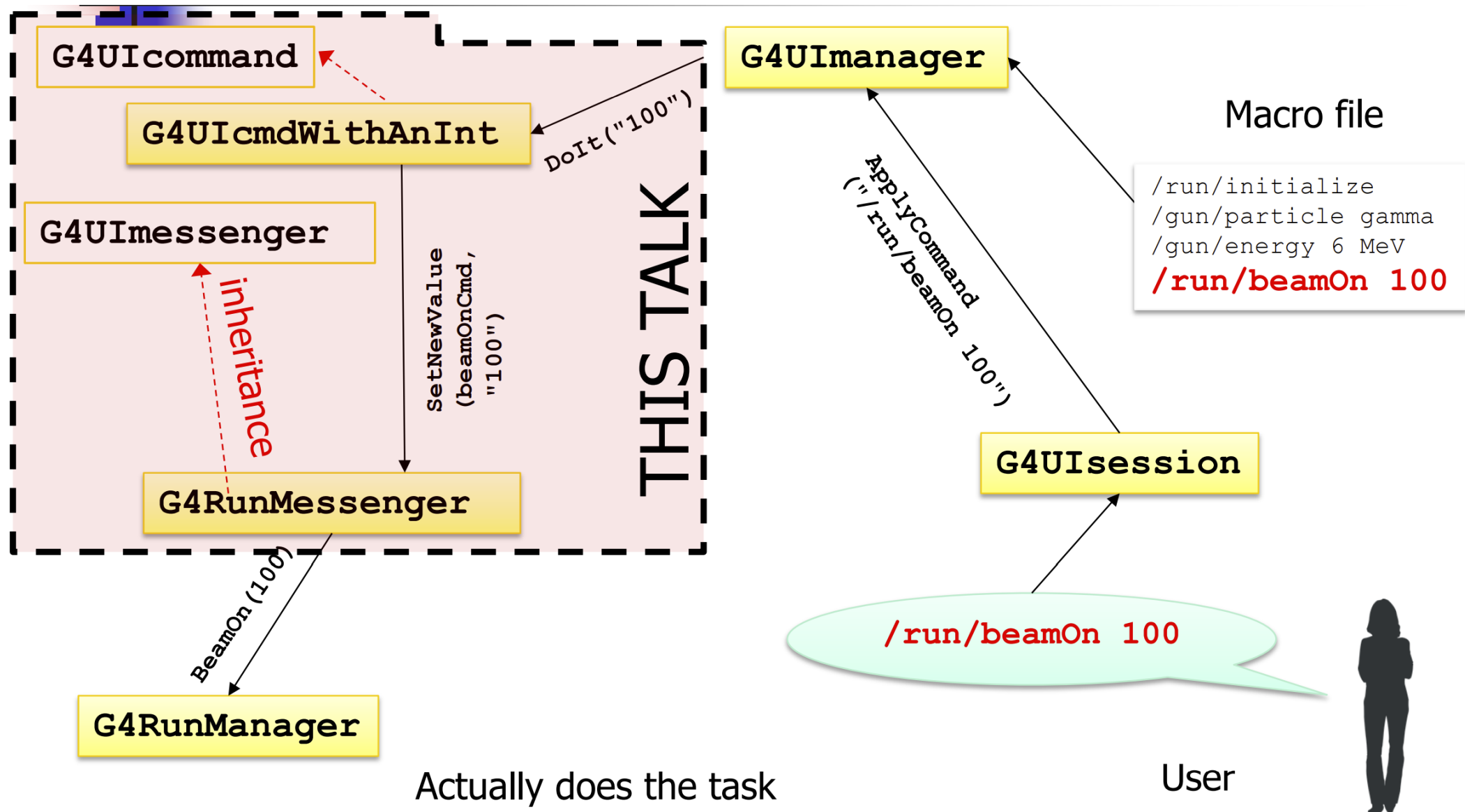
- **G4UIExecutive**
- **G4UIterminal**
- **Batch mode**

❄ **UI commands, can be used**

- **in a UI session**
- **in a macro file**
- **by hard-coded implementation**

❄ **Messenger**

- **If built-in commands are not enough, you define your own command.**



❄ **There are built-in commands roughly organized according to Geant4 categories**

```
Idle> ls
```

```
Command directory path : /
```

```
Sub-directories :
```

```
/control/    UI control commands.  
/units/      Available units.  
/geometry/   Geometry control commands.  
/tracking/   TrackingManager and SteppingManager control commands.  
/event/      EventManager control commands.  
/run/        Run control commands.  
/random/     Random number status control commands.  
/particle/   Particle control commands.  
/process/    Process Table control commands.  
/material/   Commands for Materials  
/vis/        Visualization commands.  
/gun/        Particle Gun control commands.
```

A complete list of built-in commands is available in the Geant4 Application Developers Guide, Chapter 7.1

G4UIcmdWithoutParameter: /run/initialize
G4UIcmdWithAnInteger: /run/beamOn 10
G4UIcmdWithABool: /process/em/auger true*
G4UIcmdWithADouble: /particle/property/decay/br 0.5
G4UIcmdWithADoubleAndUnit: /gps/time 1.5 us
G4UIcmdWithAString: /gps/particle e-
G4UIcmdWith3Vector: /gps/direction 0.707 0.707 0.0
G4UIcmdWith3VectorAndUnit:
/gps/position 0.0 0.0 0.0 m

***Note:** The **true** values are **t**, **true**, **y**, **yes** and **1** (case insensitive); everything else is **false**.

MyClassMessenger.hh

```
#include <G4UImessenger.hh>
#include <G4UIDirectory.hh>
#include <G4UIcmdWithADouble.hh>
#include <G4UIcmdWithoutParameters.hh>
```

Geant4 headers

```
#include "MyClass.hh"
```

Header to the class to manage

```
class MyClassMessenger : public G4UImessenger
{
public:
```

Inherit from G4UImessenger

```
    MyClassMessenger(MyClass* myObject);
```

Commands are defined in constructor

```
    ~MyClassMessenger();
```

```
    void SetNewValue(G4UIcommand* command, G4String newValue) override;
```

```
private:
```

```
    MyClass* fObject;
```

```
    G4UIDirectory* fDirectory;
```

```
    G4UIcmdWithADouble* fSomeParameterCommand;
```

```
    G4UIcmdWithoutParameters* fDoSomethingCommand;
```

```
};
```

Method **reacting** to **all command calls** managed by this messenger

MyClassMessenger.cc

```
#include "MyClassMessenger.hh"
```

Include the header

```
MyClassMessenger::~MyClassMessenger() {  
    delete fSomeParameterCommand;  
    delete fDoSomethingCommand;  
    delete fDirectory;  
}
```

UI command and directory
pointers should be **deleted**!

```
MyClassMessenger::MyClassMessenger(MyClass* myObject)  
    : fObject(myObject) {  
    fDirectory = new G4UIdirectory("/myClass/");  
    fDirectory->SetGuidance("Commands for MyClass");  
  
    fSomeParameterCommand =  
    // ...
```

Assign the pointer to
the **associated**
object

Create the **command
directory** (with
description)

To be continued in the next slide...

MyClassMessenger.cc**Create a new command** with description

```
// ...  
fSomeParameterCommand = new G4UicmdWithADouble("/myClass/setParameter", this);  
fSomeParameterCommand->SetGuidance("Set some parameter");  
fSomeParameterCommand->AvailableForStates(G4State_PreInit, G4State_Idle);  
// ... More details
```

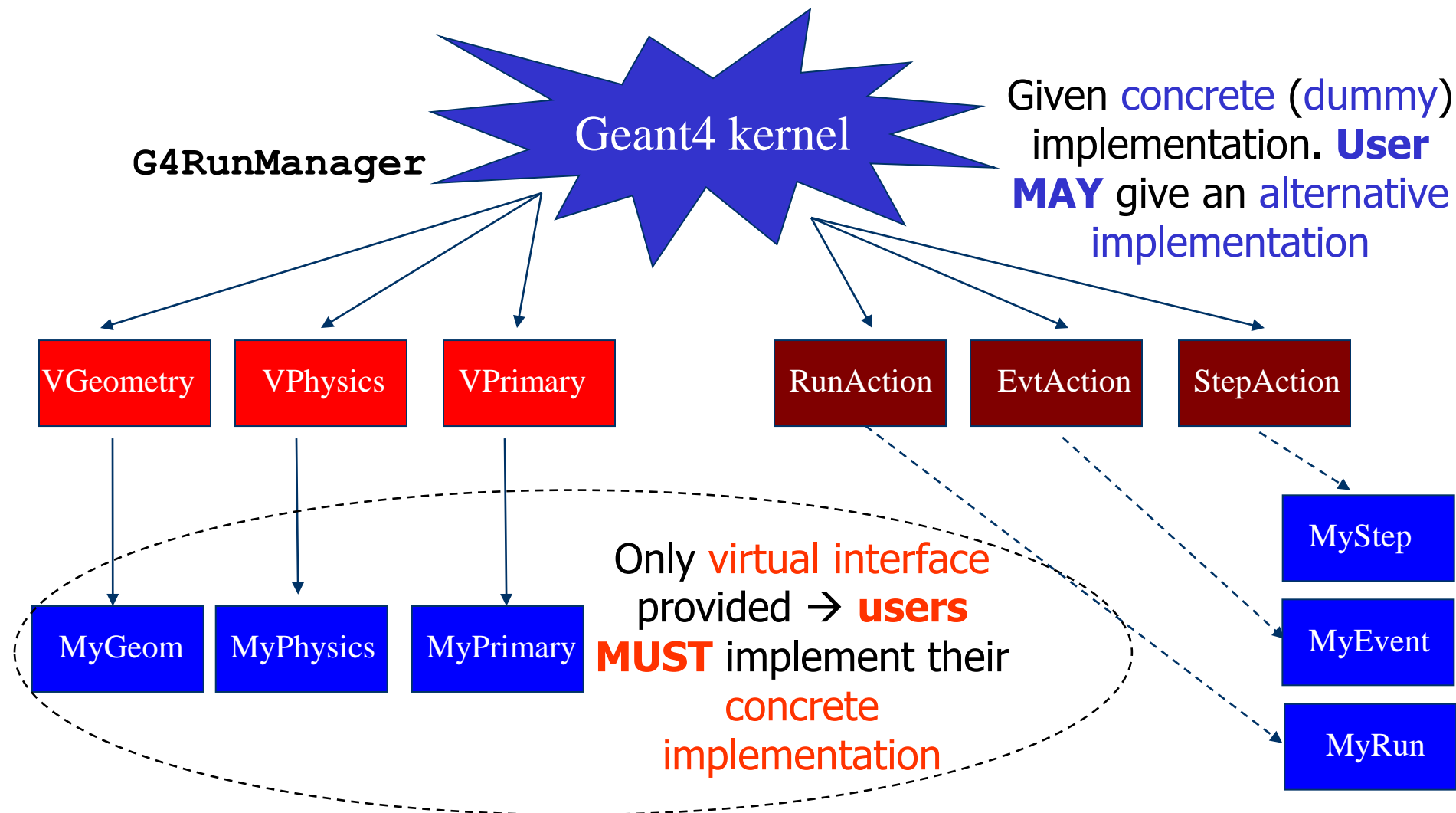
Enable only in certain state(s)

```
fDoSomethingCommand = new G4UicmdWithoutParameters("/myClass/do", this);
```

Method **reacting** to
command callsWhich command
is it?What are the
command's parameters?

```
void MyClassMessenger::SetNewValue(G4Uicommand* cmd, G4String newValue) {  
    if (cmd == fDoSomethingCommand) { fObject->DoSomething(); }  
    else if (cmd == fSomeParameterCommand) {  
        G4double value = fSomeParameterCommand->GetNewDoubleValue(newValue);  
        fObject->SetParameter(newValue);  
    }  
}
```

Simple
reaction**Use** the parsed
valueIt is necessary to **parse**
the parameters **string**!



Initialisation classes

Invoked at the initialization

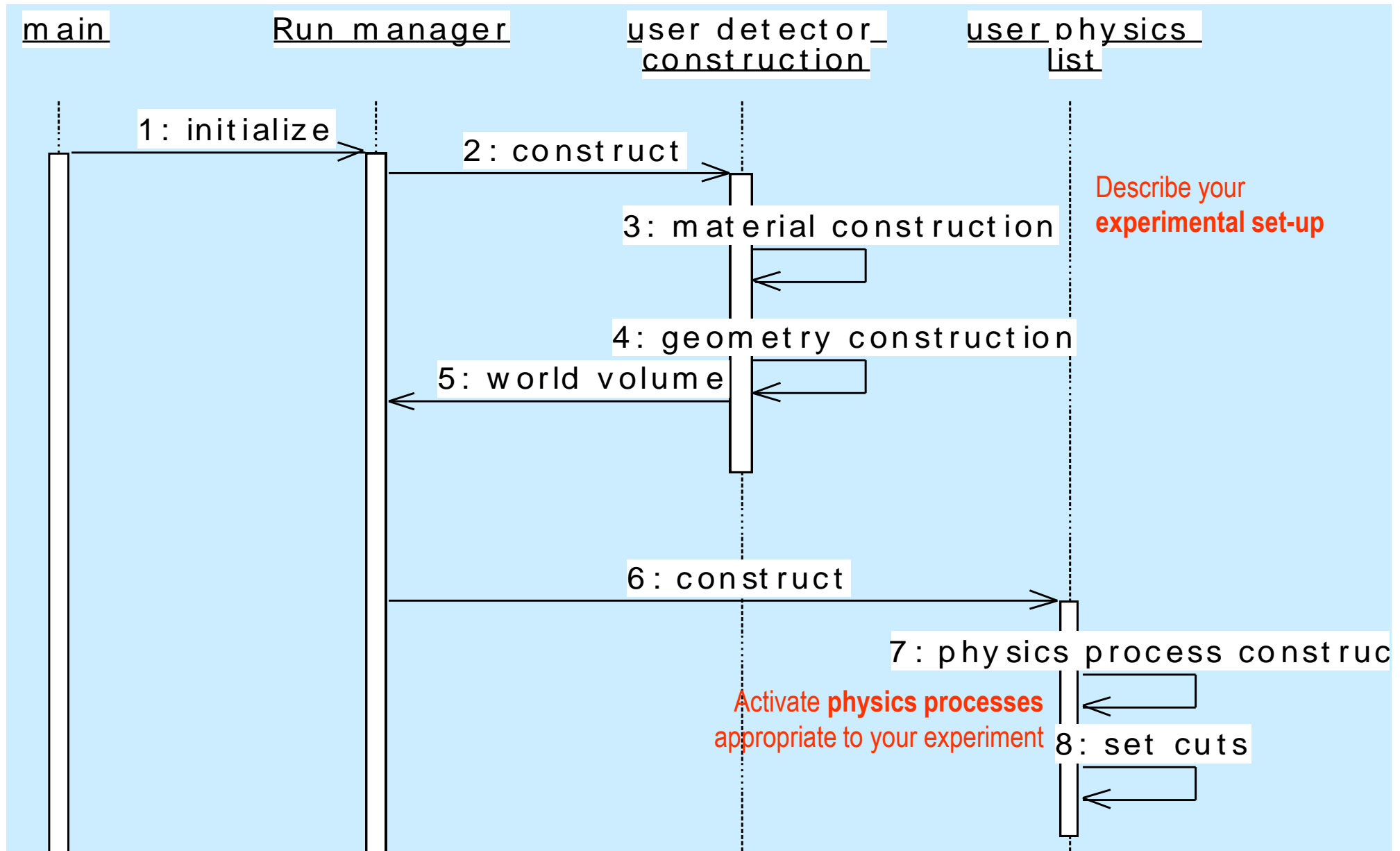
- ❄ **G4VUserDetectorConstruction**
- ❄ **G4VUserPhysicsList**

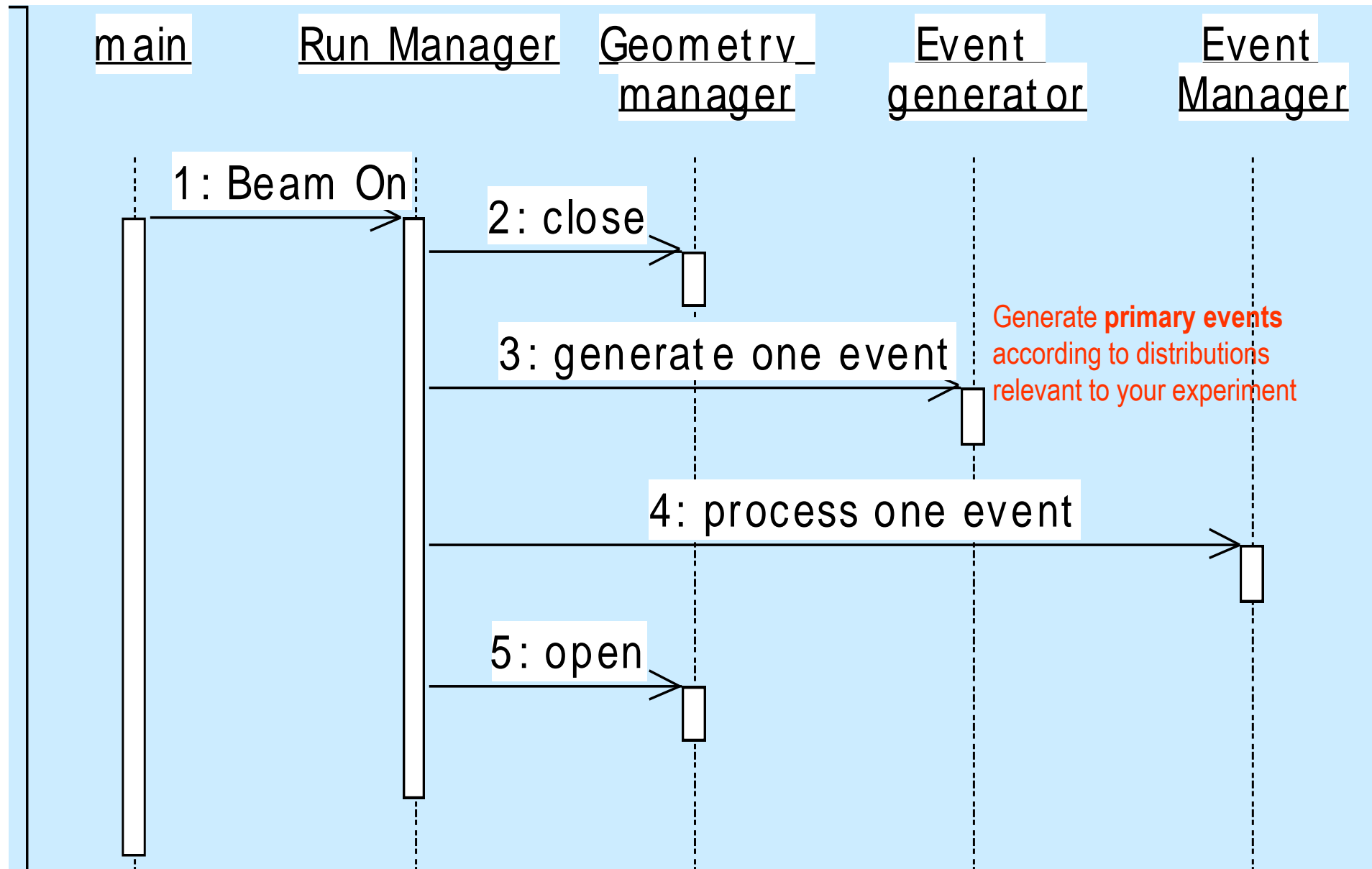
Classes having name starting with **G4V** are **abstract classes** (containing purely virtual methods)

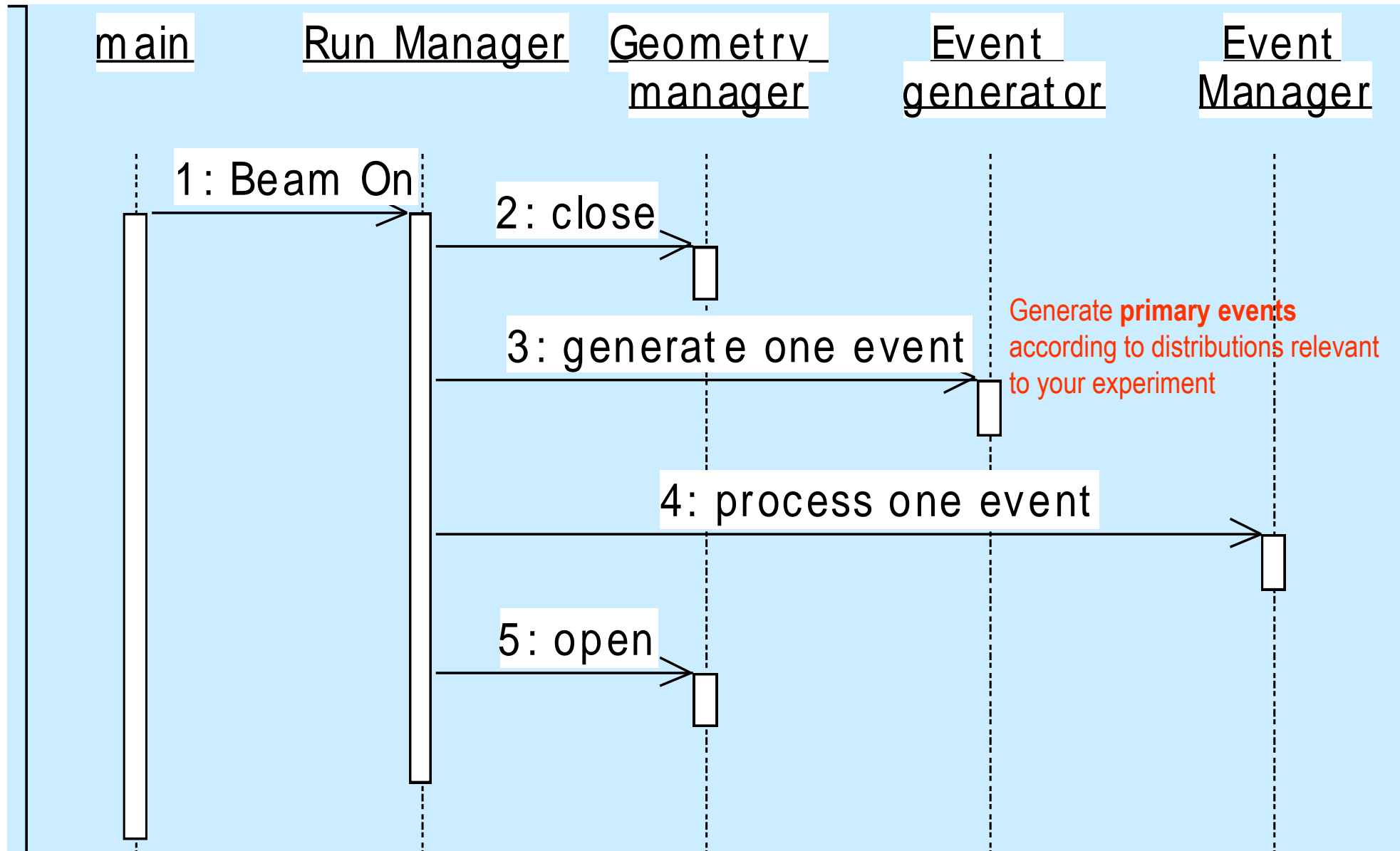
Action classes

Invoked during the execution loop

- **G4VUserPrimaryGeneratorAction**
- **G4UserRunAction**
- **G4UserEventAction**
- **G4UserTrackingAction**
- **G4UserStackingAction**
- **G4UserSteppingAction**

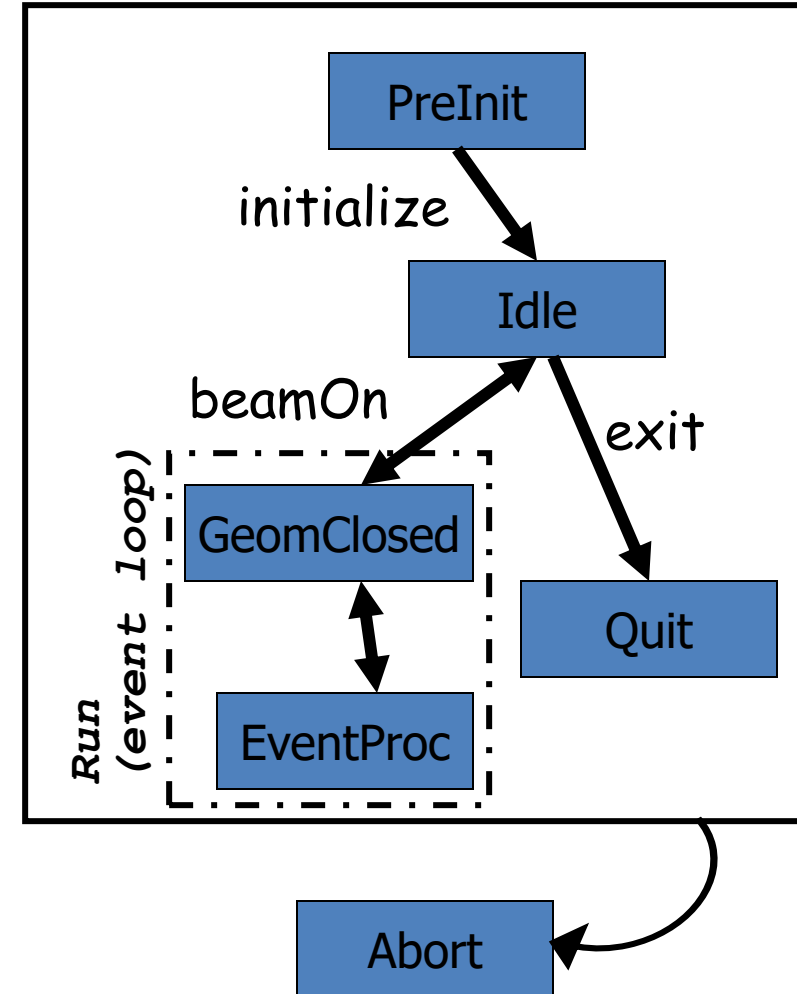






❄ **Geant4 has six application states.**

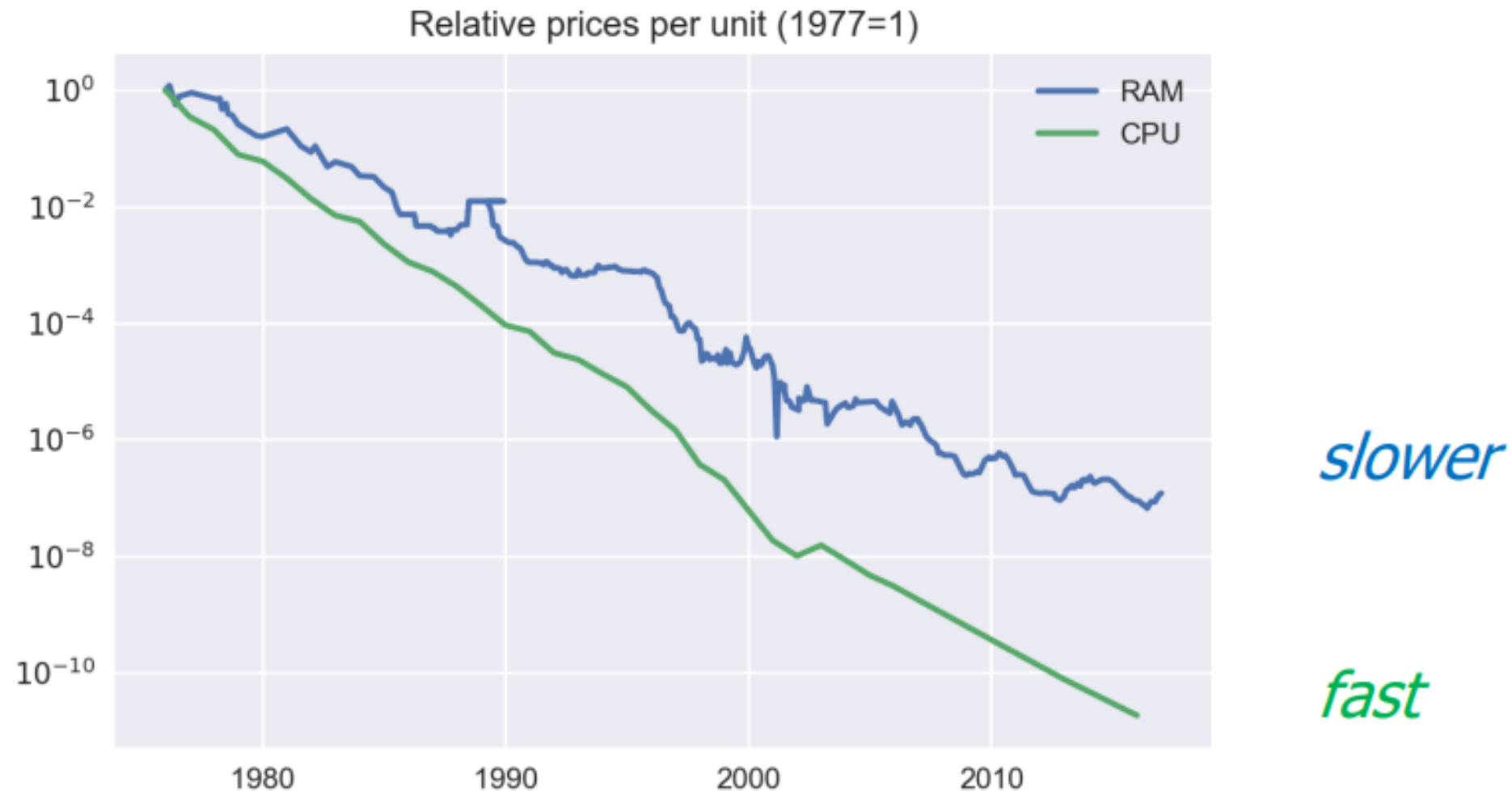
- **G4State_PreInit**
 - **Material, Geometry, Particle and/or Physics Process need to be initialized/defined**
- **G4State_Idle**
 - **Ready to start a run**
- **G4State_GeomClosed**
 - **Geometry is optimized and ready to process an event**
- **G4State_EventProc**
 - **An event is processing**
- **G4State_Quit**
 - **(Normal) termination**
- **G4State_Abort**
 - **A fatal exception occurred and program is aborting**



Multithreading

■ Multi-core CPUs

■ Expensive memory

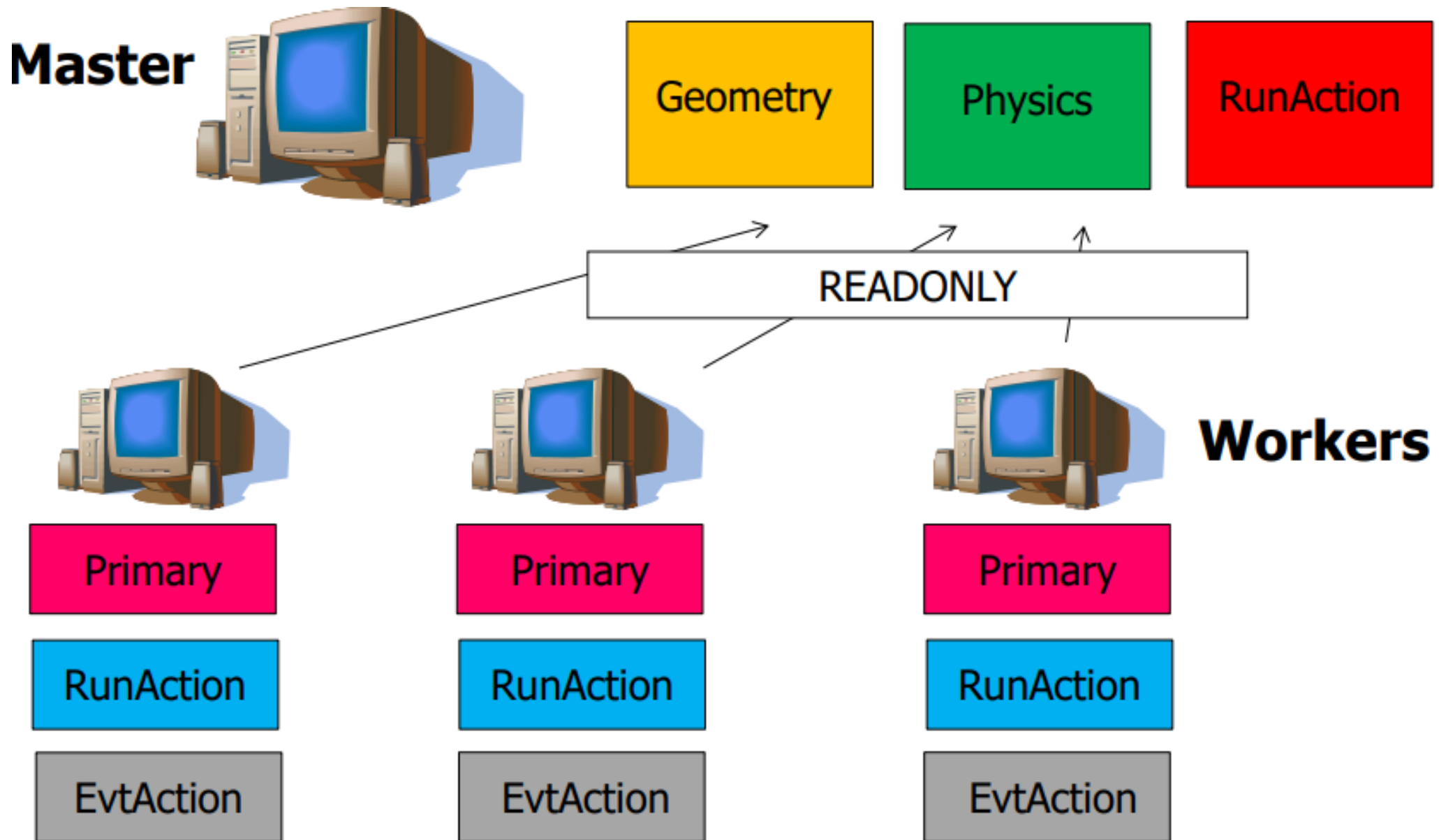


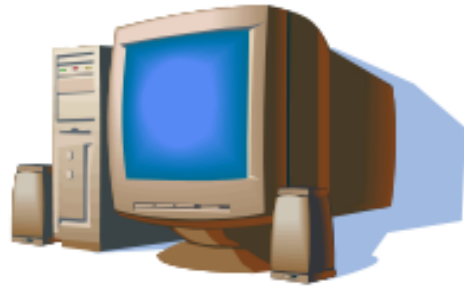
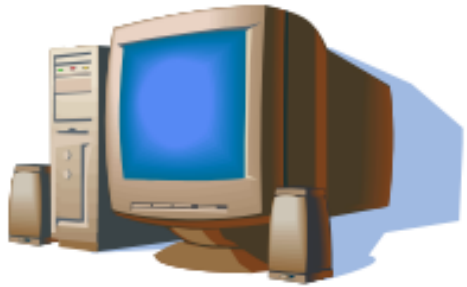
⇒ **Memory optimization** is more and more important!

- MT code integrated into G4
- Public release
- All functionalities ported to MT



- Proof of principle
- Identify objects to be shared
- First testing
- API re-design
- Example migration
- Further testing
- First optimizations
- Further Refinements
- Focus on further performance improvements





Node

Geometry

Geometry

Geometry

☐ Each node hosts a **complete** simulation

Physics

Physics

Physics

☐ **Many copies** of geometry and physics tables

Primary

Primary

Primary

RunAction

RunAction

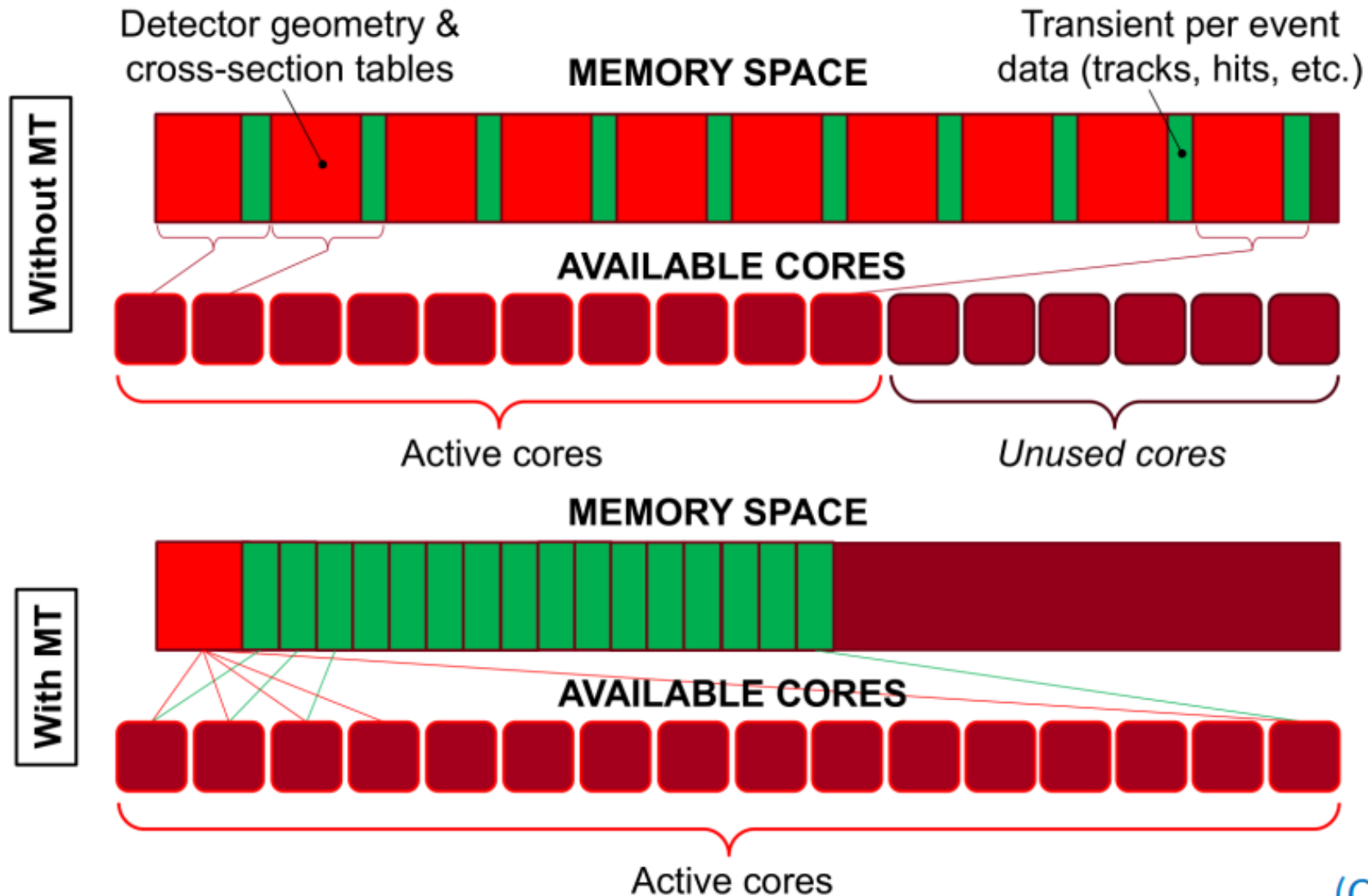
RunAction

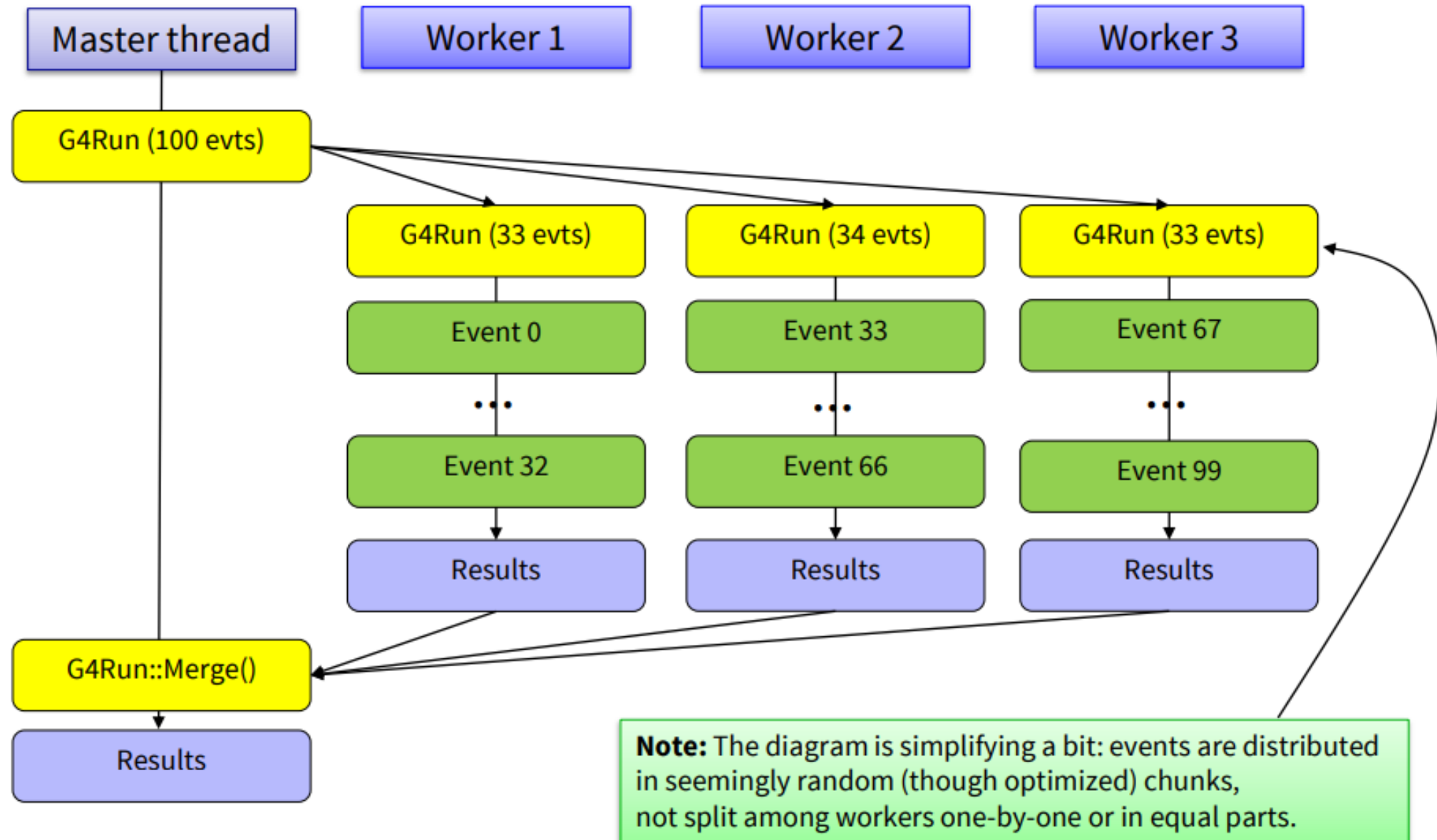
☐ More **memory-thirsty**

EvtAction

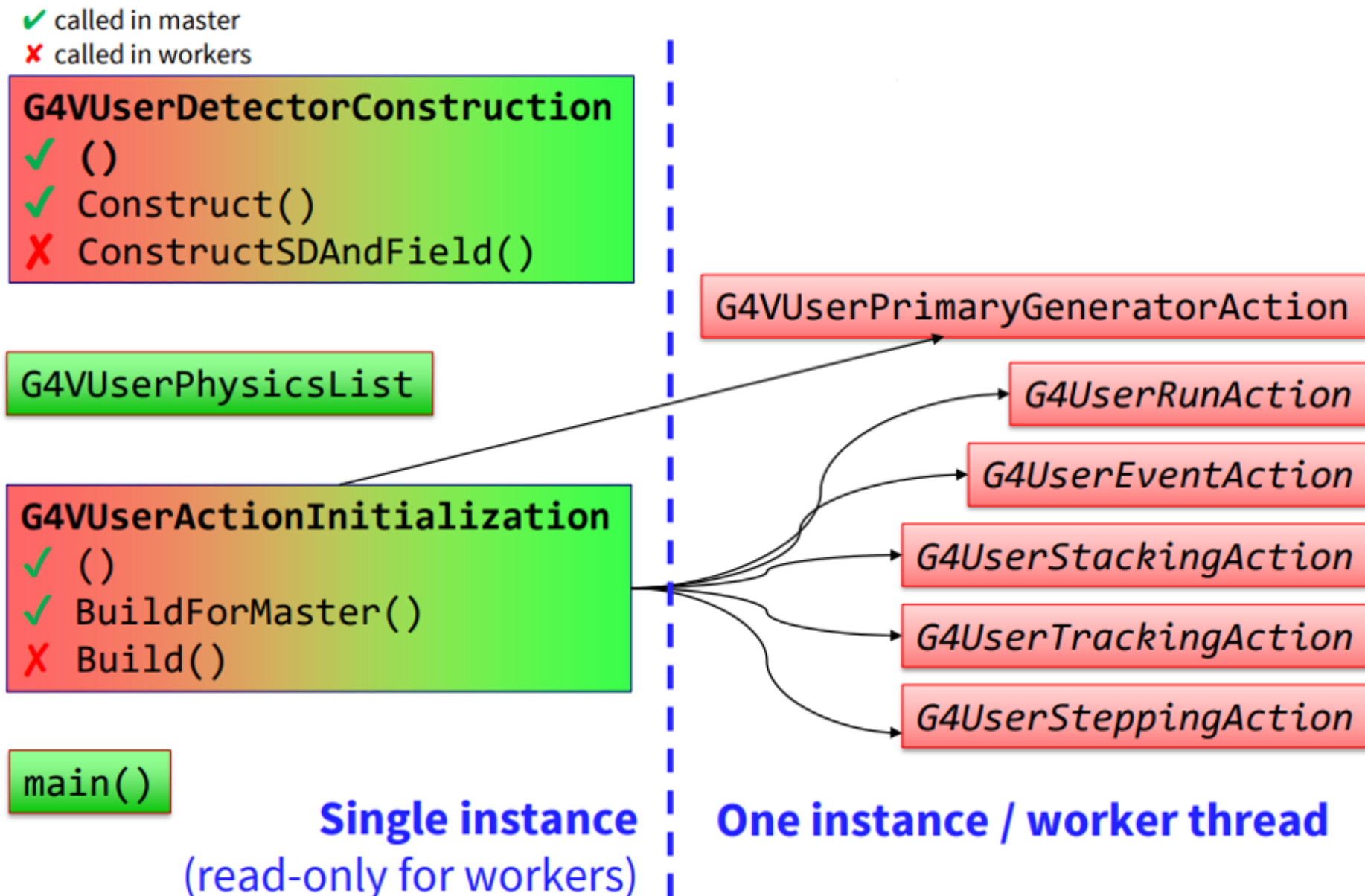
EvtAction

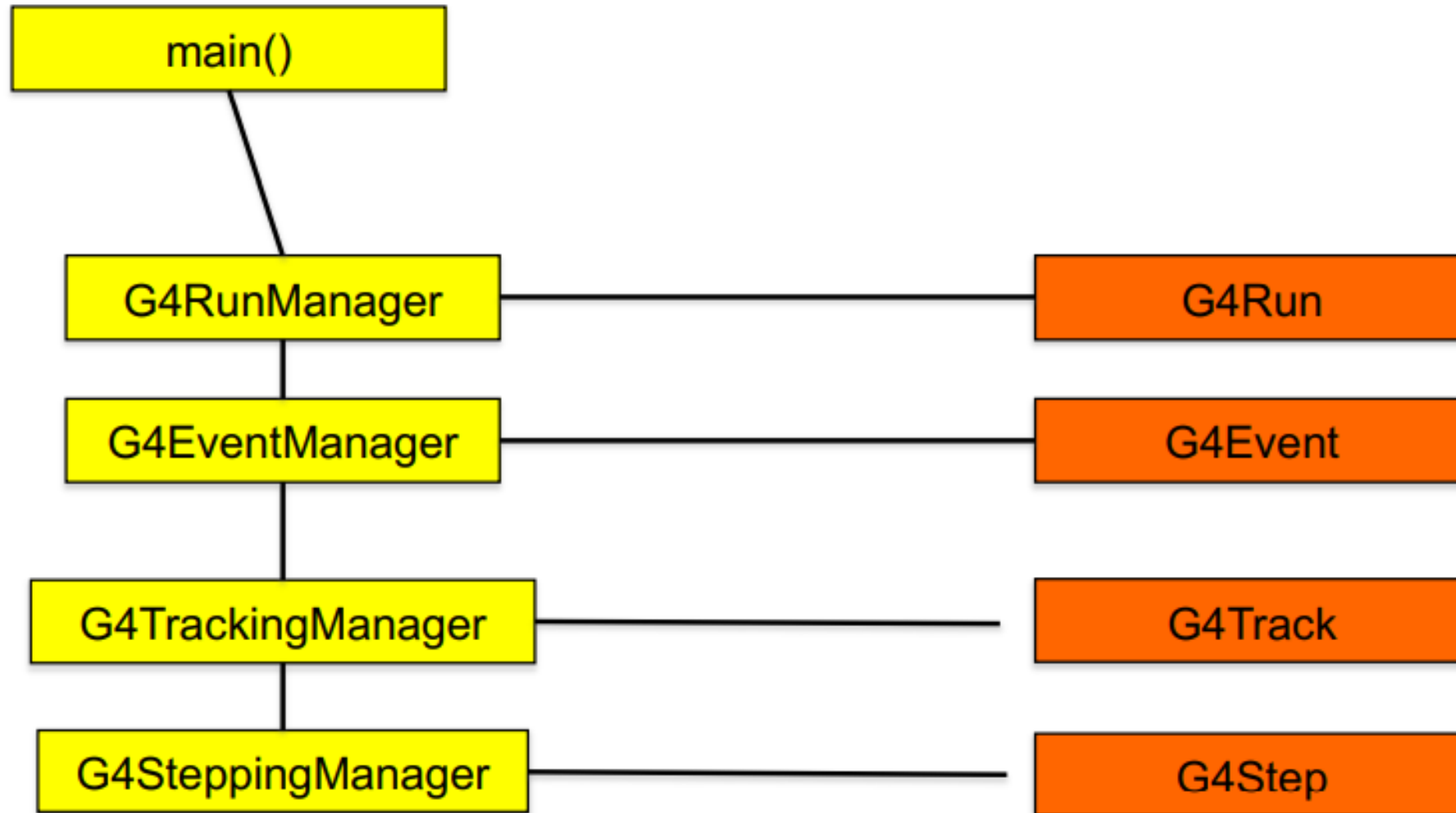
EvtAction

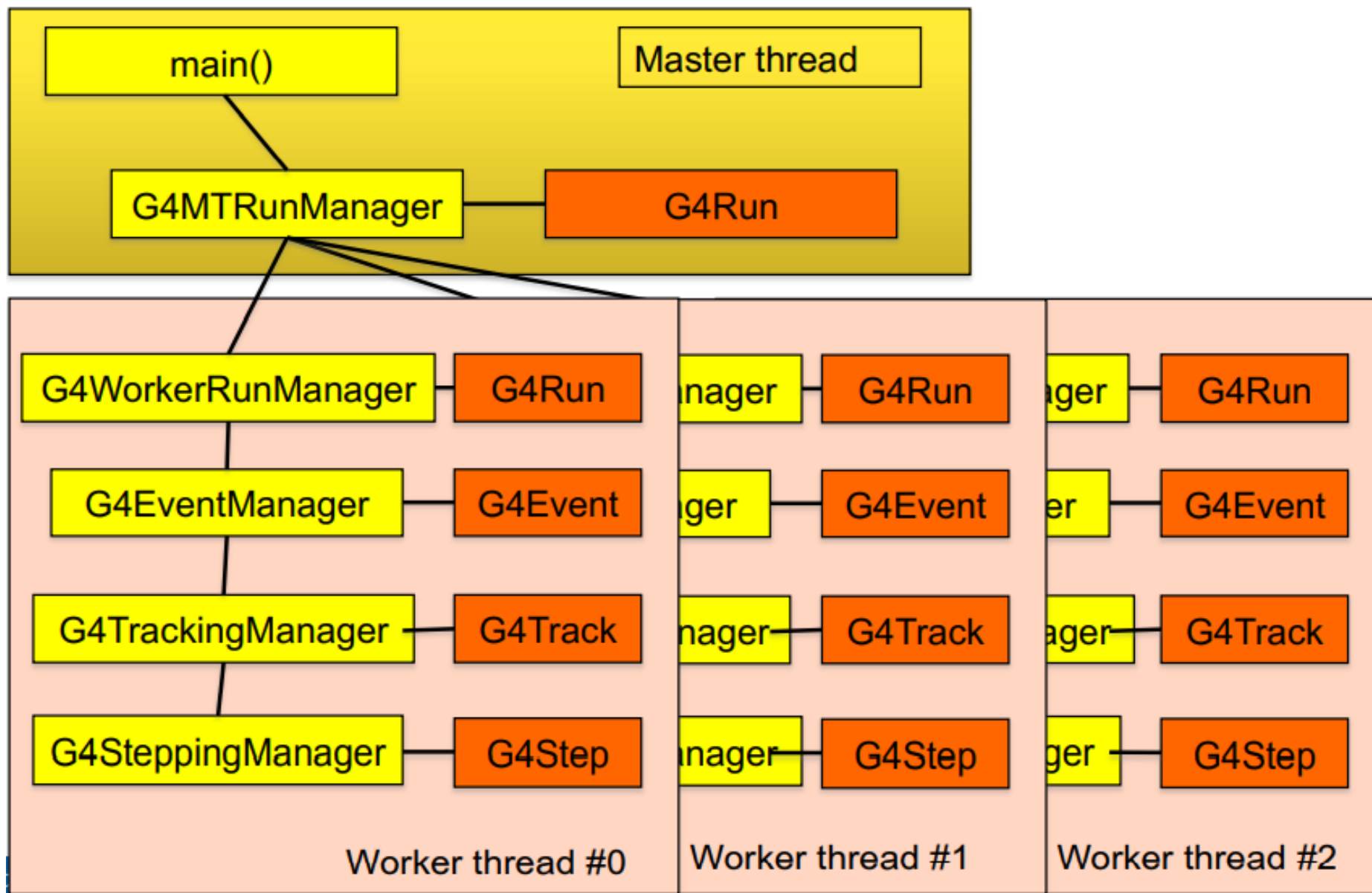


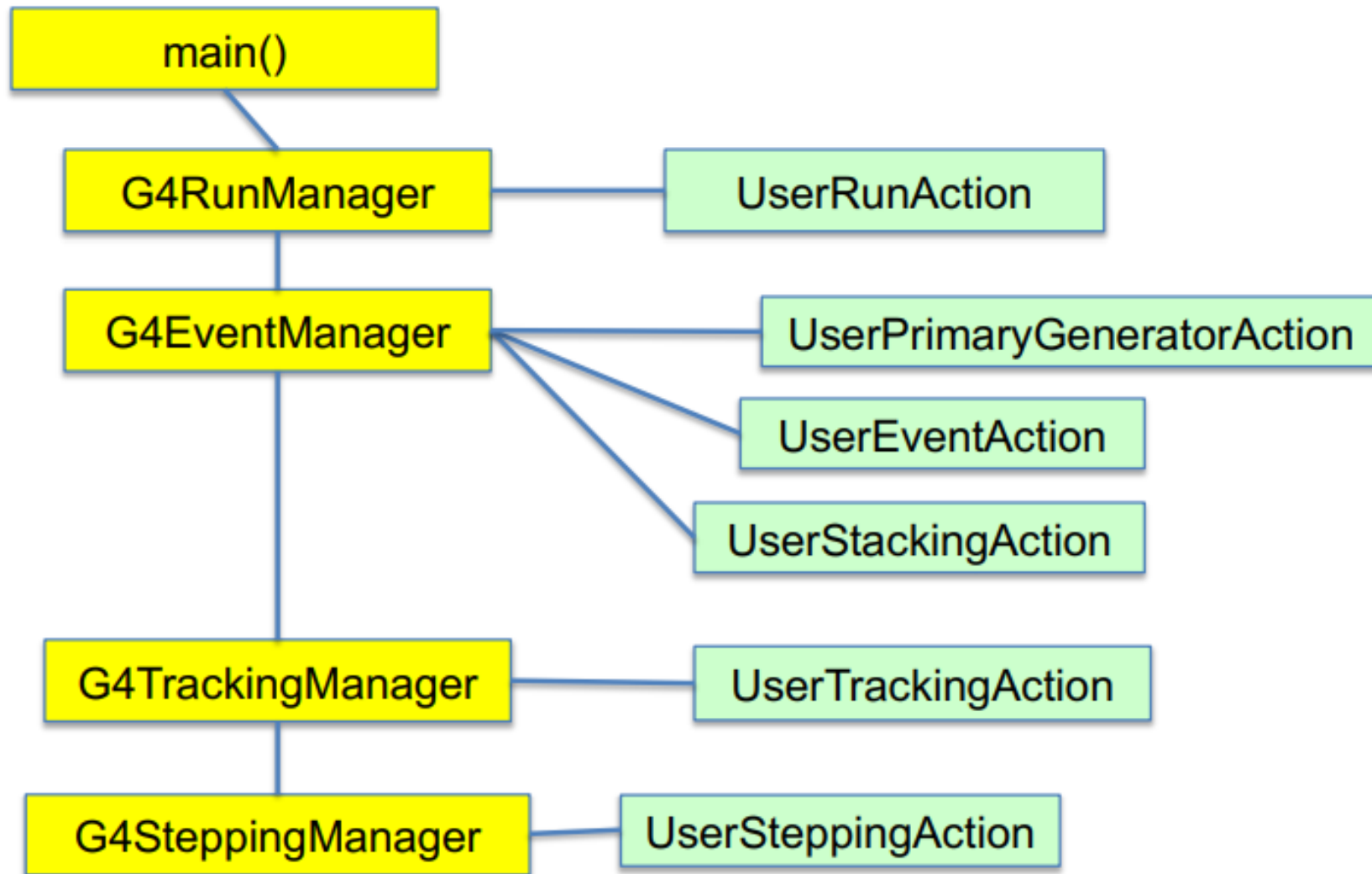


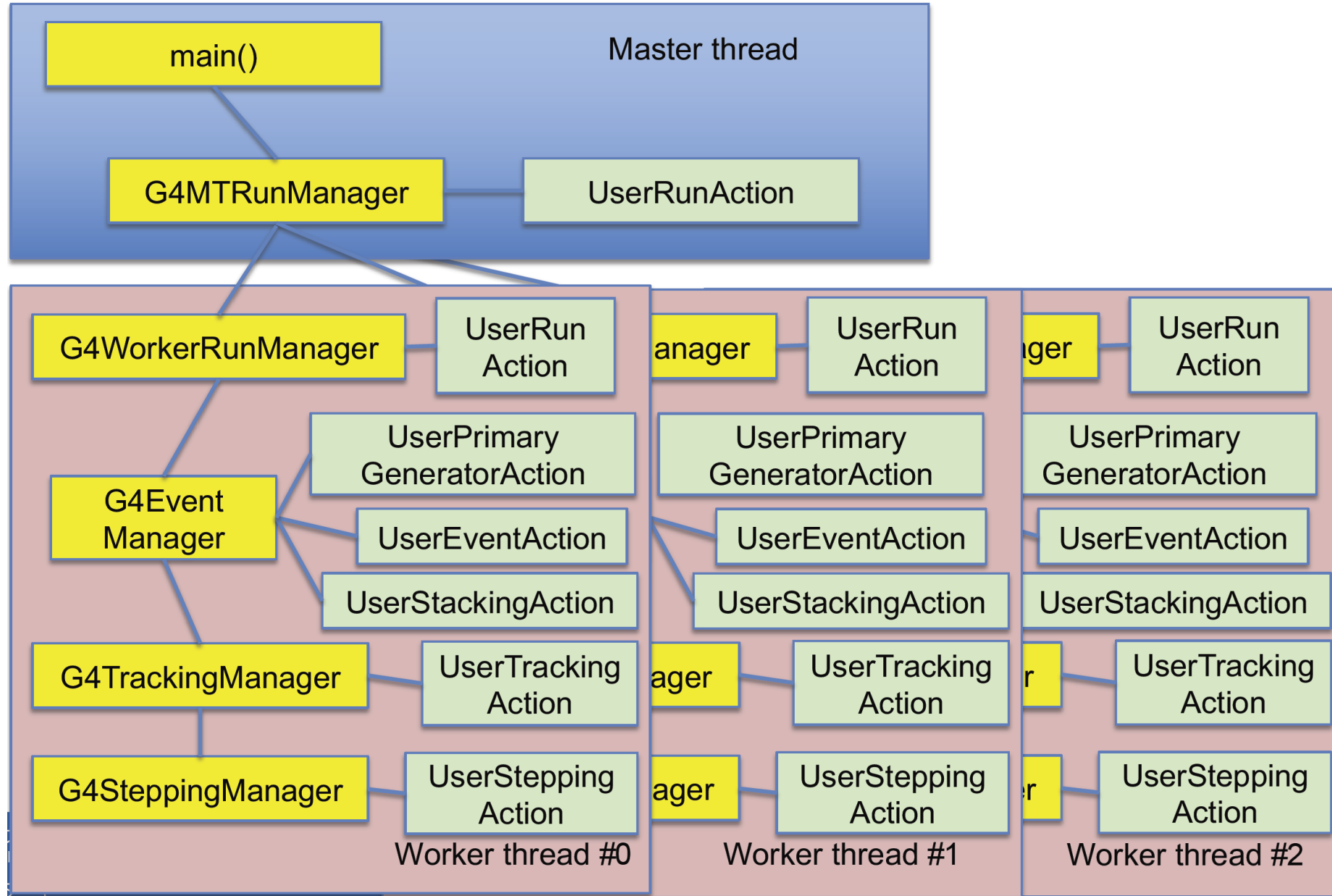
- ❑ The geometry and physics tables **are shared**
- ❑ The event, track, step, trajectory, hits, etc., as well as several Geant4 manager classes (**EventManager, TrackingManager, SteppingManager, TransportationManager, FieldManager, SensitiveDetectorManager**) and Navigator **are thread-local**
- ❑ Among the user classes, the initialization classes (G4VUserDetectorConstruction, G4VUserPhysicsList and the new G4VUserActionInitialization) **are shared**, whereas all user action classes and sensitive detector classes **are thread-local**
 - It is not recommended to access a thread-local object from a shared class object, e.g. from detector construction to stepping action.
 - Please note that thread-local objects are instantiated and initialized at the first *BeamOn*.
- ❑ To avoid potential errors, try to keep in mind which classes are shared and which classes are thread-local











❄ This action (unlike the rest) can apply in both **worker** and **master** threads:

- To distinguish where you are, use **IsMaster()** method
- If you have behavior for master, register the instance in **G4VUserActionInitialization::BuildForMaster()**

```
void MyActionInitialization::Build() const {  
    SetUserAction(new MyRunAction());  
    // ...other actions  
}  
  
void MyActionInitialization::BuildForMaster() const {  
    SetUserAction(new MyRunAction());  
    // Only run action  
}
```

Note: This, in principle, can be a different class

❄ **Default number of threads: 2**

❄ **Change this using:**

➤ **UI command**

- `/run/numberOfThreads 6`
- `/run/useMaximumLogicalCores`

➤ **C++ code**

- `runManager->SetNumberOfThreads(4)`

➤ **Environment variable (highest priority)**

- `G4FORCENUMBEROFTHREADS=4`

❄ **G4Threading::G4GetNumberOfCores() tells the actual number of logical cores**

❄ **Note: Must be done in pre-initialize stage**

- ❑ **G4cout is relatively synchronized and each message is prepended with the thread number (this is not true for std::cout)**

```
### Run 0 starts.  
G4WT1 > EventAction: absorber energy/time scorer ID: 0  
G4WT1 > EventAction: scintillator energy/time scorer ID: 1  
G4WT0 > EventAction: absorber energy/time scorer ID: 0  
G4WT0 > EventAction: scintillator energy/time scorer ID: 1  
Run terminated.  
Run Summary  
  Number of events processed : 10000  
  User=21s Real=11.36s Sys=1.59s
```

- ❑ **To buffer the output from each thread at a time, so that the output of each thread is grouped and printed at the end of the job**
 - **/control/cout/useBuffer true|false**
- ❑ **To limit the output from threads to one selected thread only:**
 - **/control/cout/ignoreThreadsExcept 0**
- ❑ **To redirect the output from threads in a file:**
 - **/control/cout/setCoutFile coutFileName**
 - **/control/cout/setCerrFile cerrFileName**