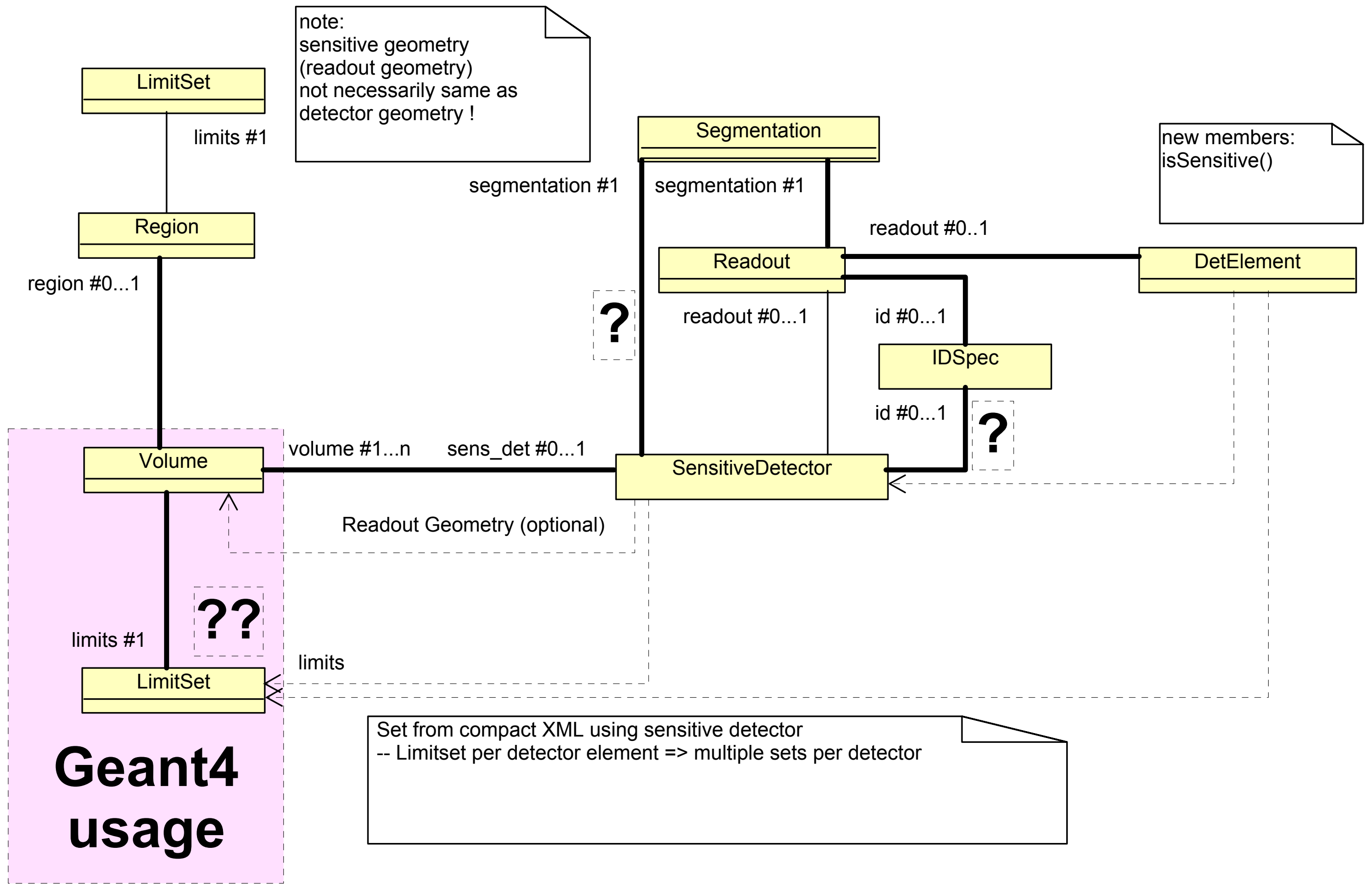
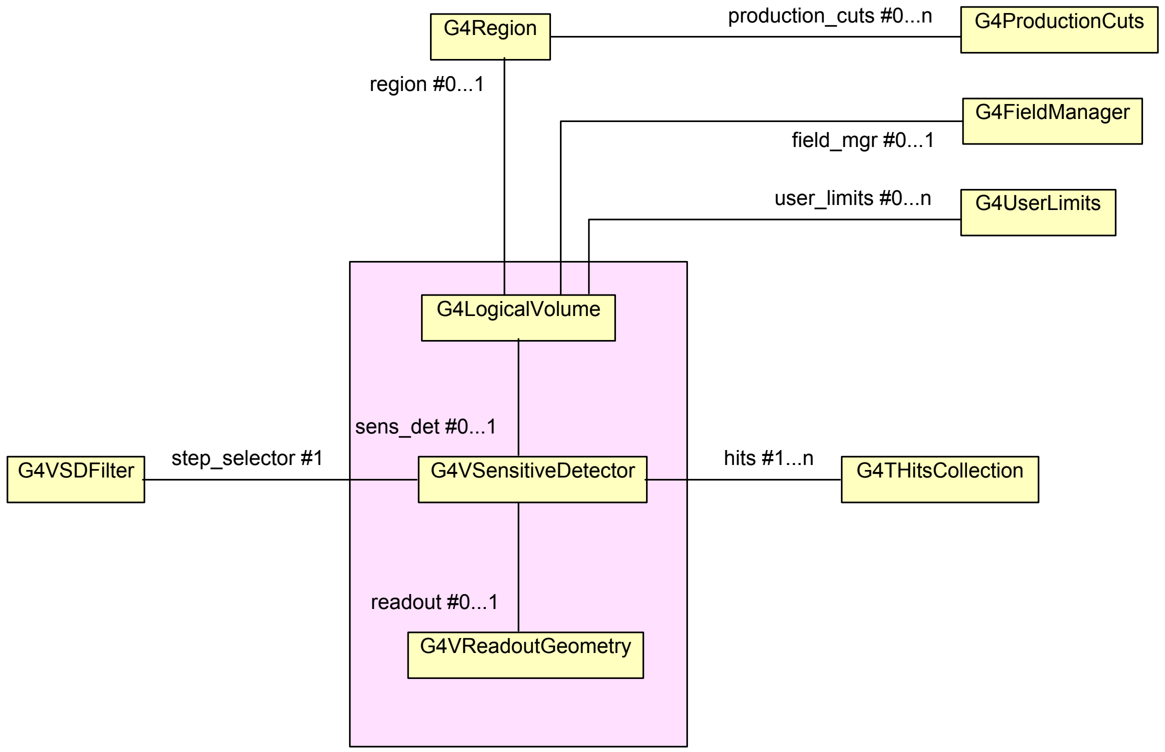
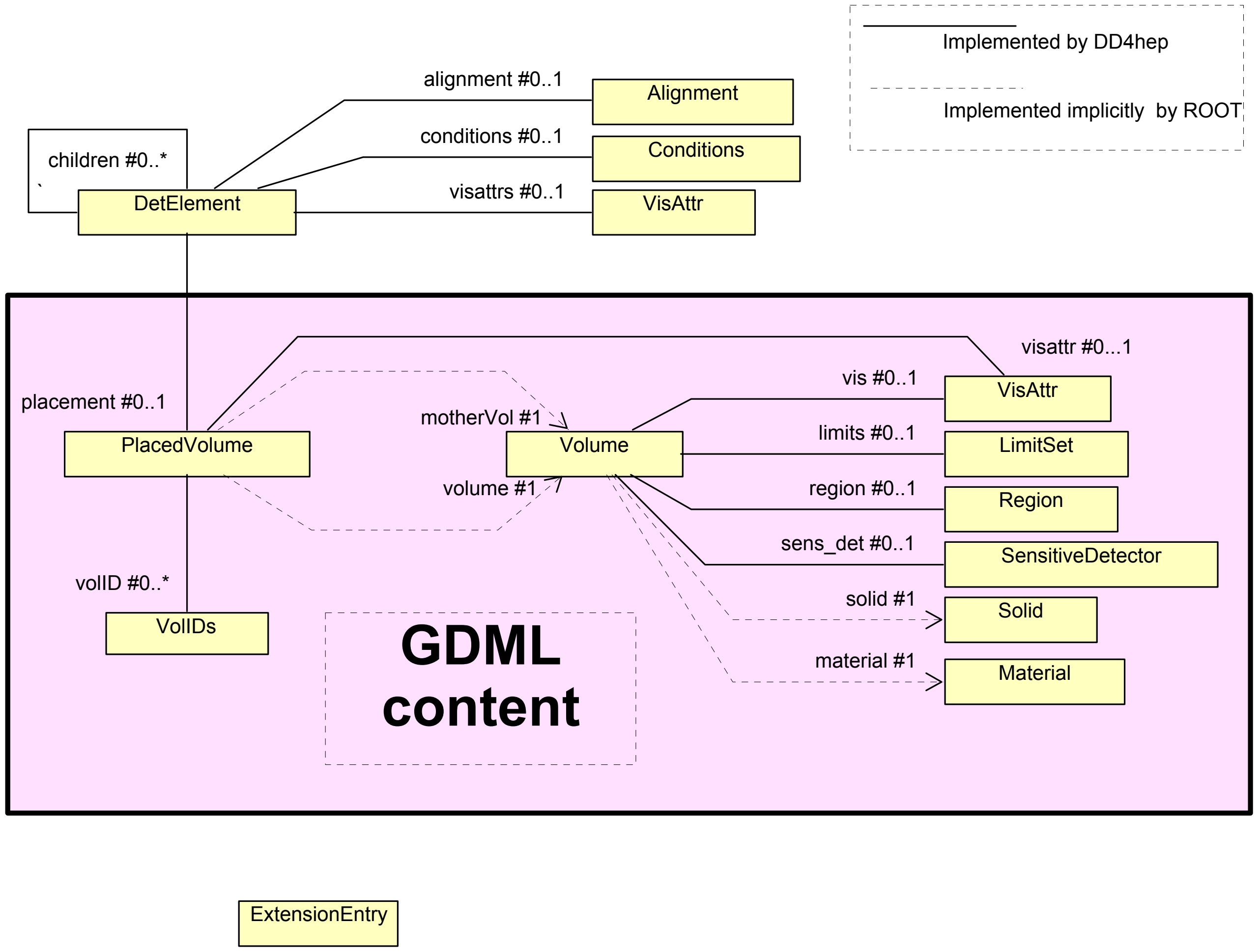
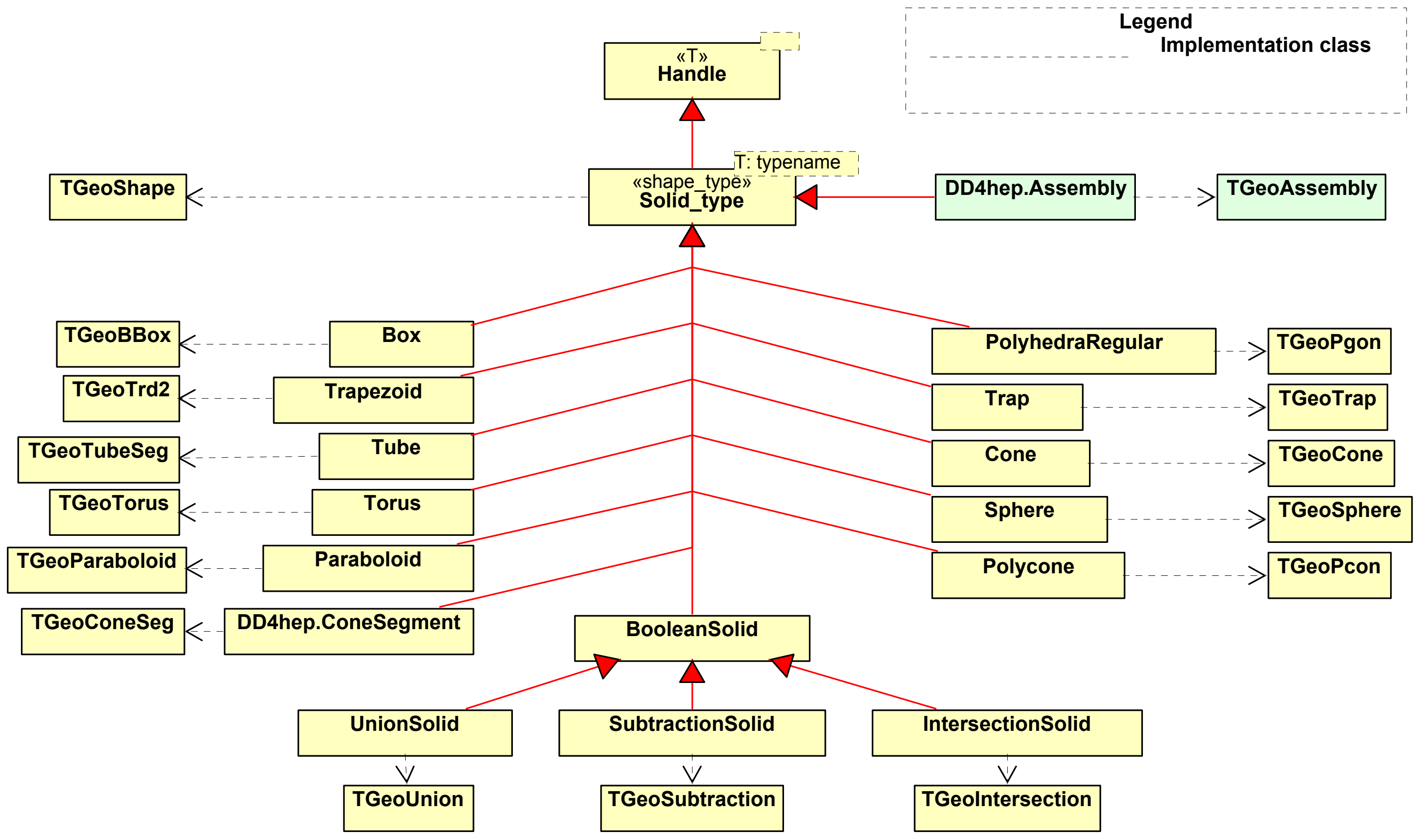

note:
sensitive geometry
(readout geometry)
not necessarily same as
detector geometry !
LimitSet
limits #1
Region
region #0...1
Segmentation
segmentation #1
segmentation #1
Readout
readout #0..1
DetElement
new members:
isSensitive()
readout #0...1
id #0...1
IDSpec
id #0...1
?
?
Volume
volume #1...n
sens_det #0...1
SensitiveDetector
Readout Geometry (optional)
??
limits #1
LimitSet
limits
Set from compact XML using sensitive detector
-- Limitset per detector element => multiple sets per detector
Geant4
usage

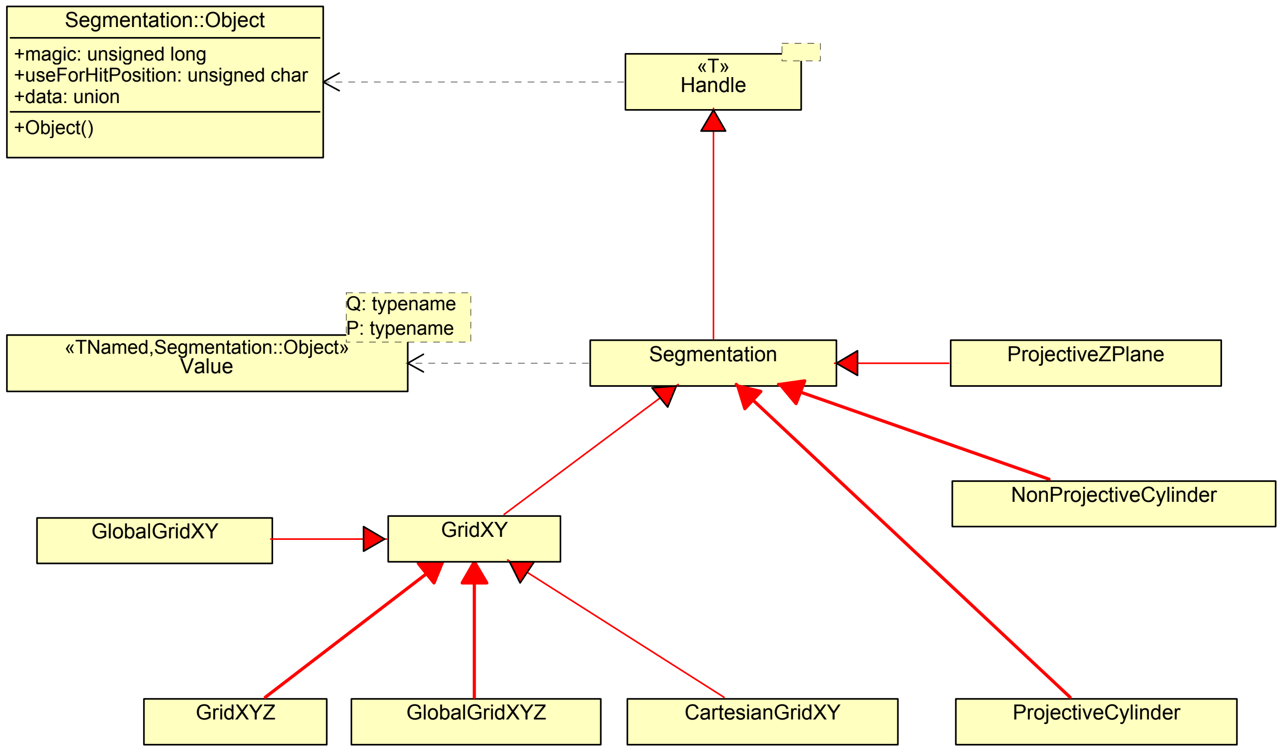


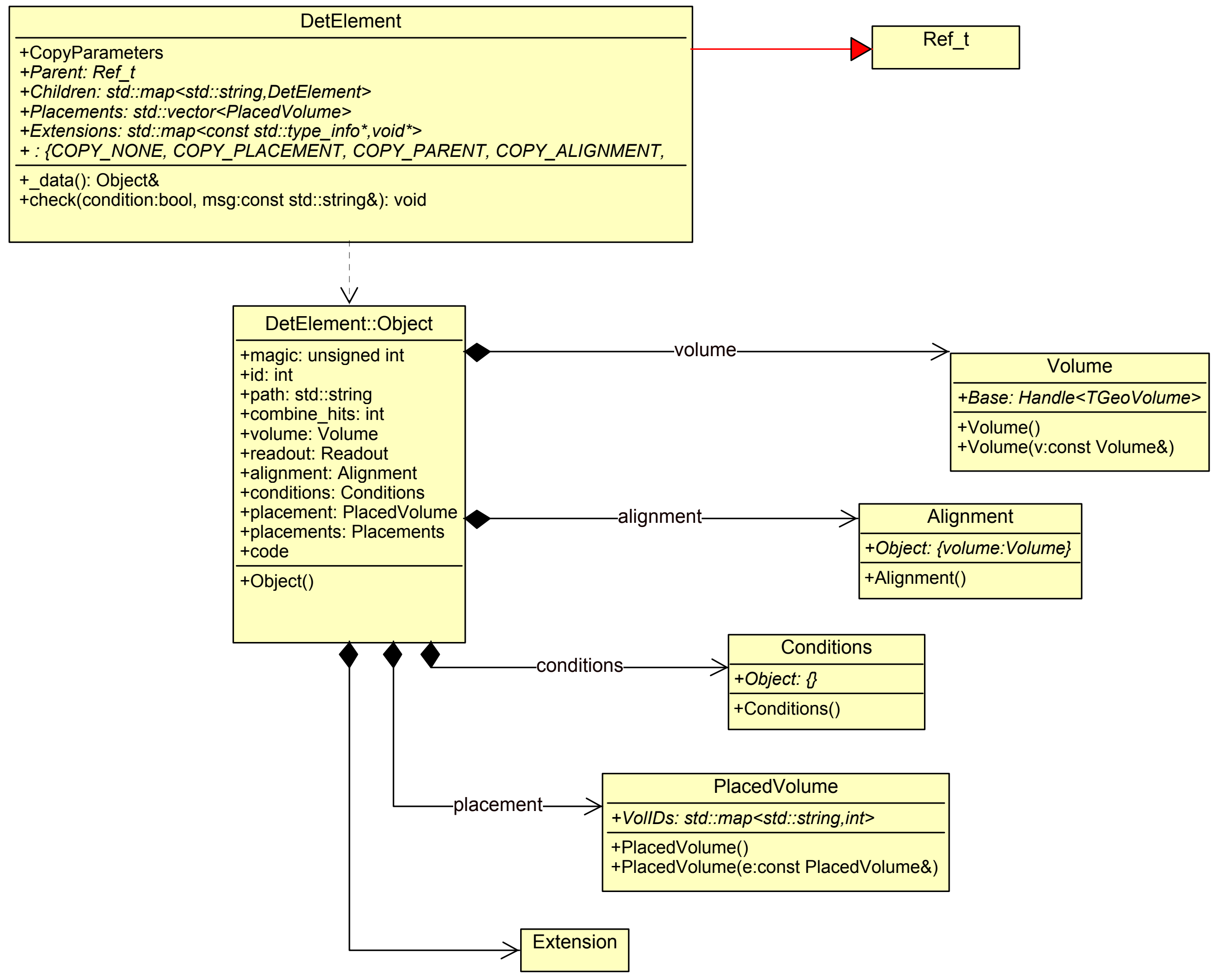
G4Region
production_cuts #0...n
G4ProductionCuts
region #0...1
G4FieldManager
field_mgr #0...1
user_limits #0...n
G4UserLimits
G4LogicalVolume
sens_det #0...1
G4VSDFilter
step_selector #1
G4VSensitiveDetector
hits #1...n
G4THitsCollection
readout #0...1
G4VReadoutGeometry

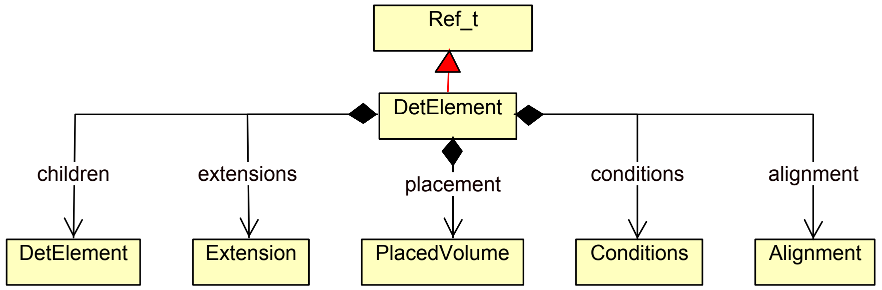


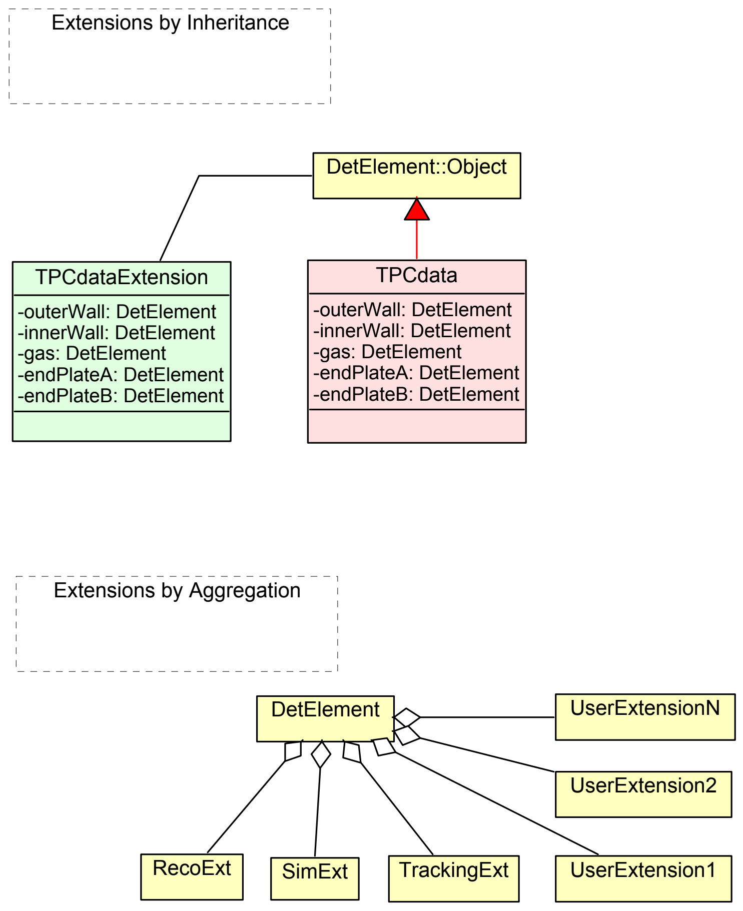
Implemented by DD4hep
Implemented implicitly by ROOT
alignment #0..1
Alignment
conditions #0..1
Conditions
children #0..*
DetElement
visattrs #0..1
VisAttr
visattr #0...1
vis #0..1
placement #0..1
PlacedVolume
motherVol #1
Volume
volume #1
limits #0..1
LimitSet
region #0..1
Region
sens_det #0..1
SensitiveDetector
solid #1
Solid
material #1
Material
volID #0..*
VolIDs
GDML content
ExtensionEntry

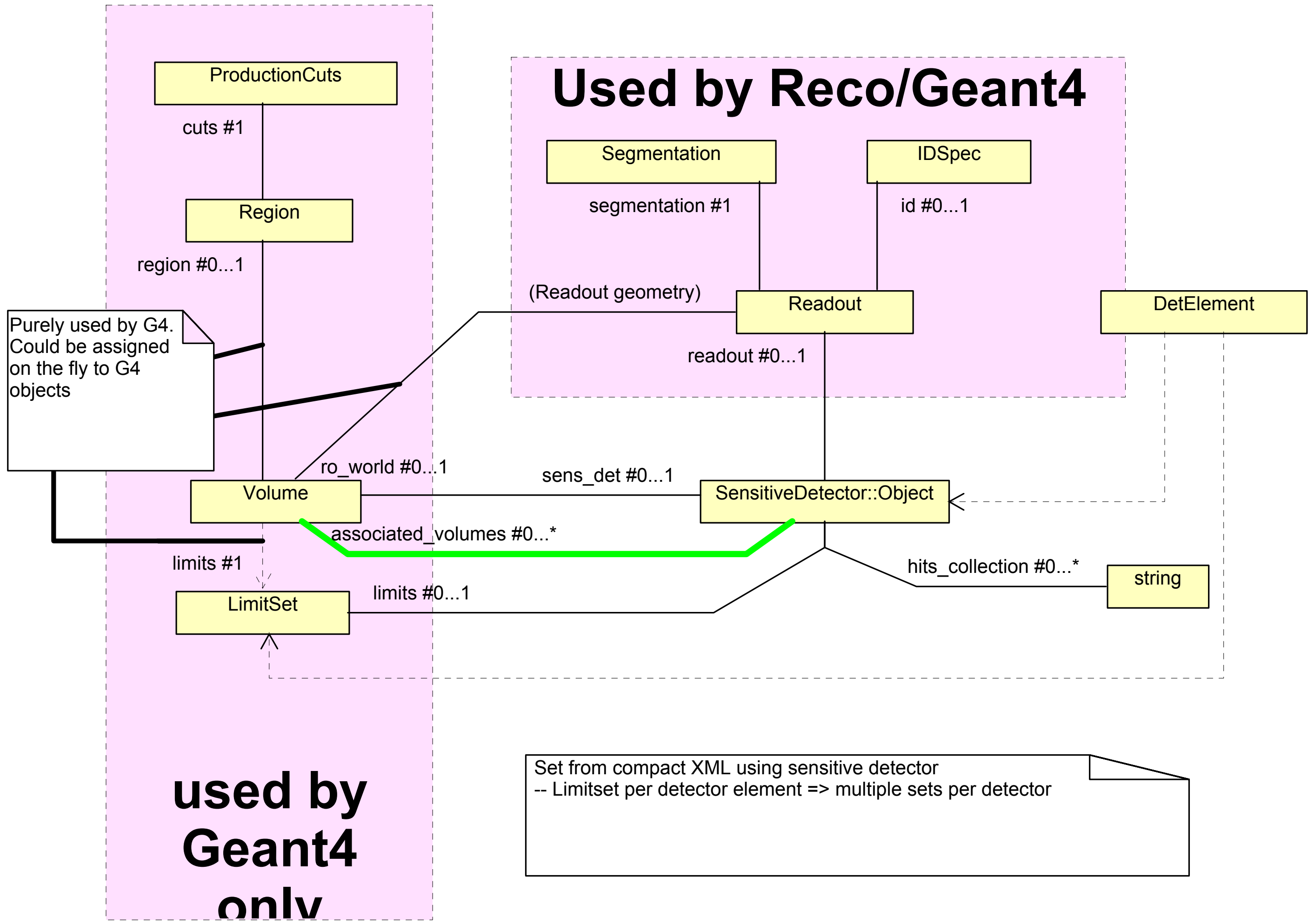

Legend
Implementation class
«T»
Handle
T: typename
«shape_type»
Solid_type
TGeoShape
DD4hep.Assembly
TGeoAssembly
TGeoBBox
Box
TGeoTrd2
Trapezoid
TGeoTubeSeg
Tube
TGeoTorus
Torus
TGeoParaboloid
Paraboloid
TGeoConeSeg
DD4hep.ConeSegment
PolyhedraRegular
TGeoPgon
Trap
TGeoTrap
Cone
TGeoCone
Sphere
TGeoSphere
Polycone
TGeoPcon
BooleanSolid
UnionSolid
SubtractionSolid
IntersectionSolid
TGeoUnion
TGeoSubtraction
TGeoIntersection


Segmentation::Object
+magic: unsigned long
+useForHitPosition: unsigned char
+data: union
+Object()
«T»
Handle
Q: typename
P: typename
«TNamed,Segmentation::Object»
Value
Segmentation
ProjectiveZPlane
NonProjectiveCylinder
GlobalGridXY
GridXY
GridXYZ
GlobalGridXYZ
CartesianGridXY
ProjectiveCylinder

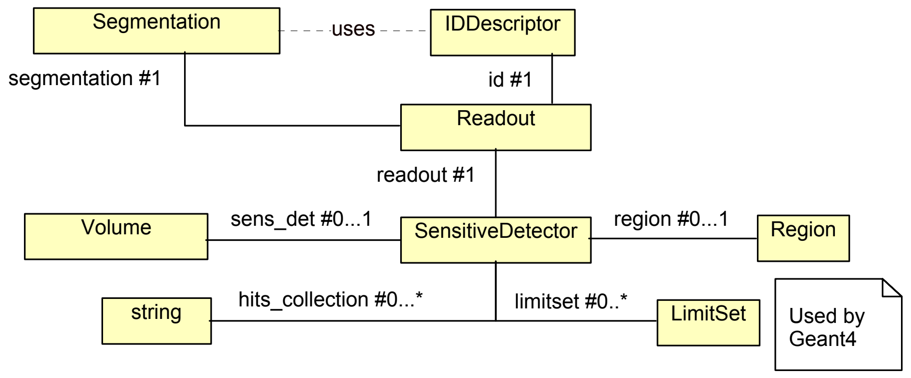

DetElement
+CopyParameters
+Parent: Ref_t
+Children: std::map<std::string,DetElement>
+Placements: std::vector<PlacedVolume>
+Extensions: std::map<const std::type_info*,void*>
+ : {COPY_NONE, COPY_PLACEMENT, COPY_PARENT, COPY_ALIGNMENT,
+_data(): Object&
+check(condition:bool, msg:const std::string&): void
Ref_t
DetElement::Object
+magic: unsigned int
+id: int
+path: std::string
+combine_hits: int
+volume: Volume
+readout: Readout
+alignment: Alignment
+conditions: Conditions
+placement: PlacedVolume
+placements: Placements
+code
+Object()
volume
Volume
+Base: Handle<TGeoVolume>
+Volume()
+Volume(v:const Volume&)
alignment
Alignment
+Object: {volume:Volume}
+Alignment()
conditions
Conditions
+Object: {}
+Conditions()
placement
PlacedVolume
+VolIDs: std::map<std::string,int>
+PlacedVolume()
+PlacedVolume(e:const PlacedVolume&)
Extension

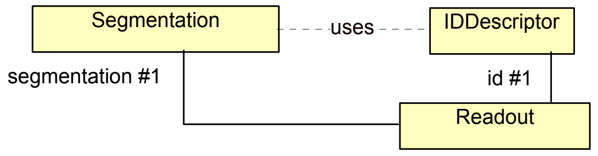

Ref_t
DetElement
children
extensions
placement
conditions
alignment
DetElement
Extension
PlacedVolume
Conditions
Alignment

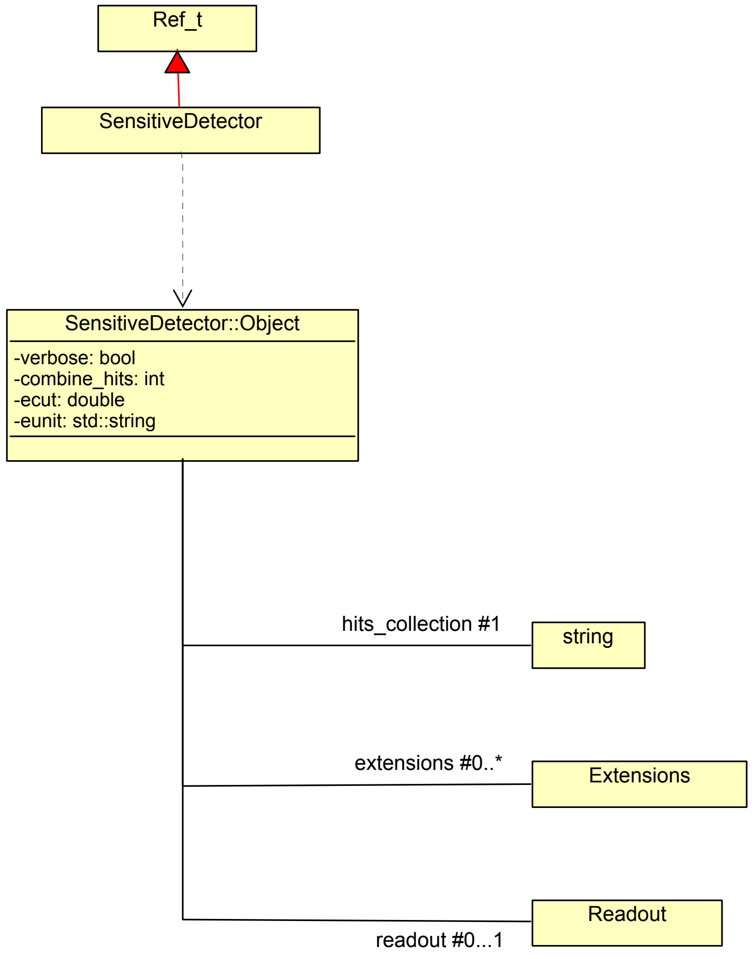

Extensions by Inheritance
DetElement::Object
TPCdataExtension
-outerWall: DetElement
-innerWall: DetElement
-gas: DetElement
-endPlateA: DetElement
-endPlateB: DetElement
TPCdata
-outerWall: DetElement
-innerWall: DetElement
-gas: DetElement
-endPlateA: DetElement
-endPlateB: DetElement
Extensions by Aggregation
DetElement
UserExtensionN
UserExtension2
RecoExt
SimExt
TrackingExt
UserExtension1

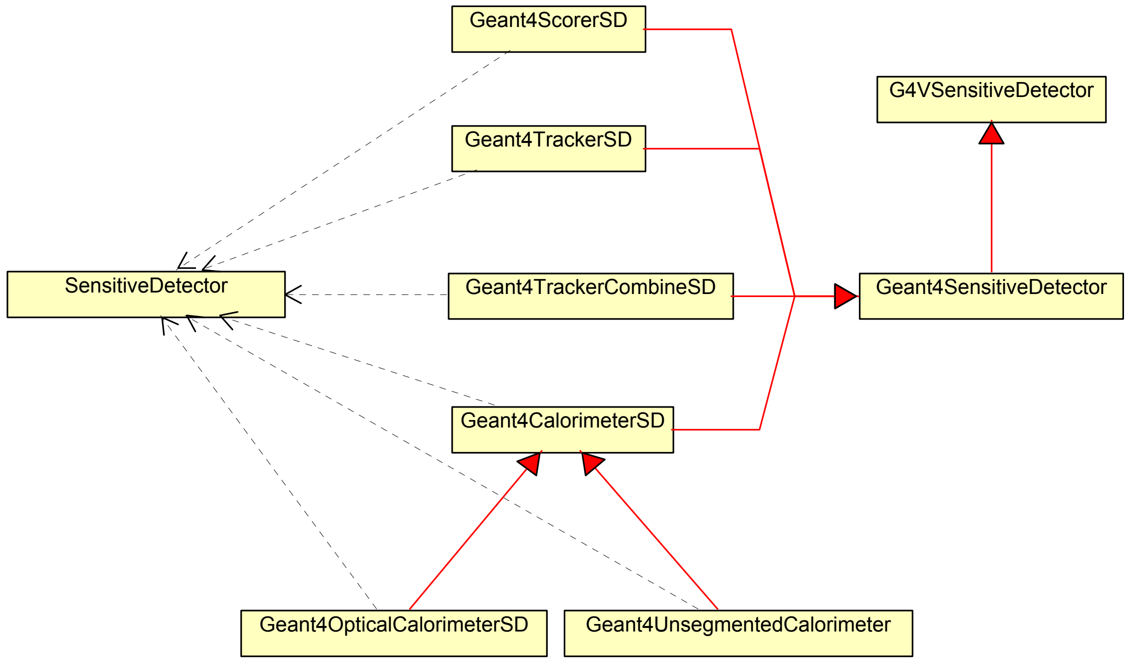


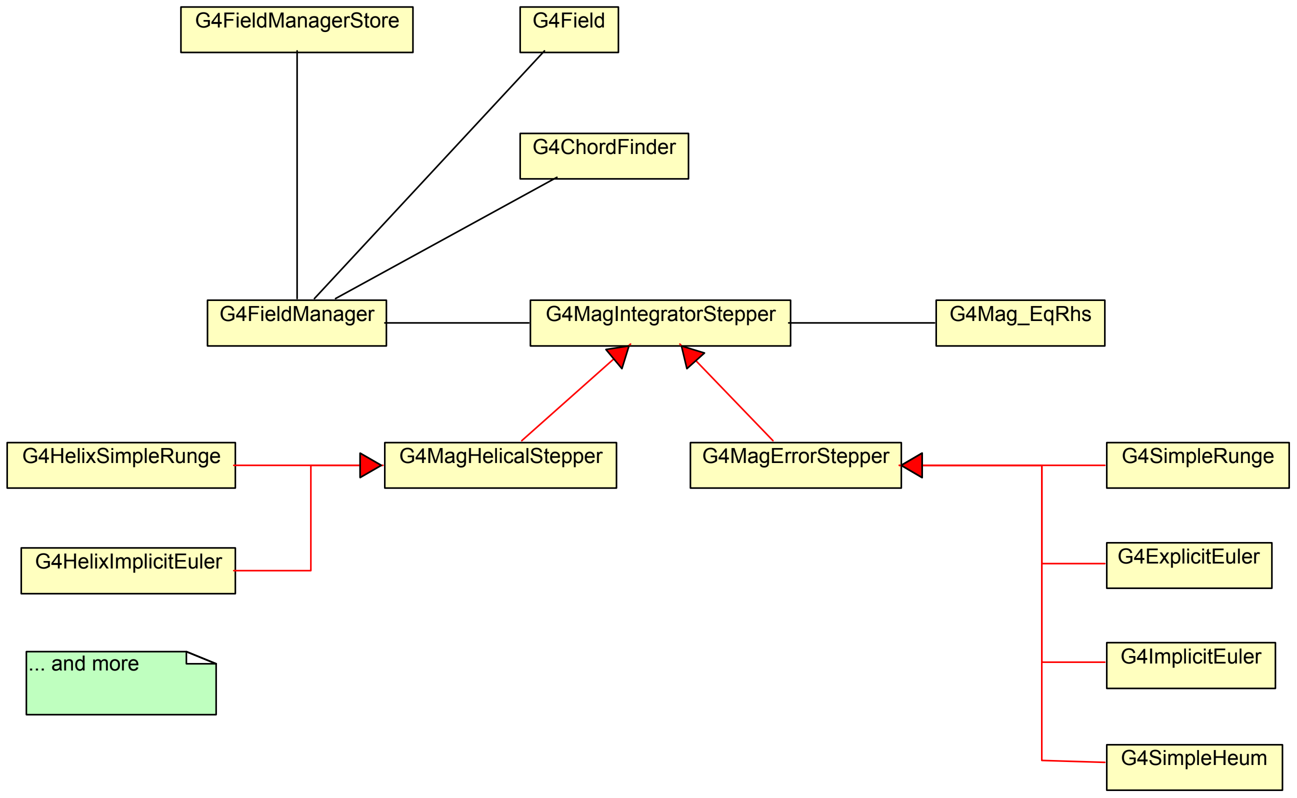
Set from compact XML using sensitive detector
-- Limitset per detector element => multiple sets per detector

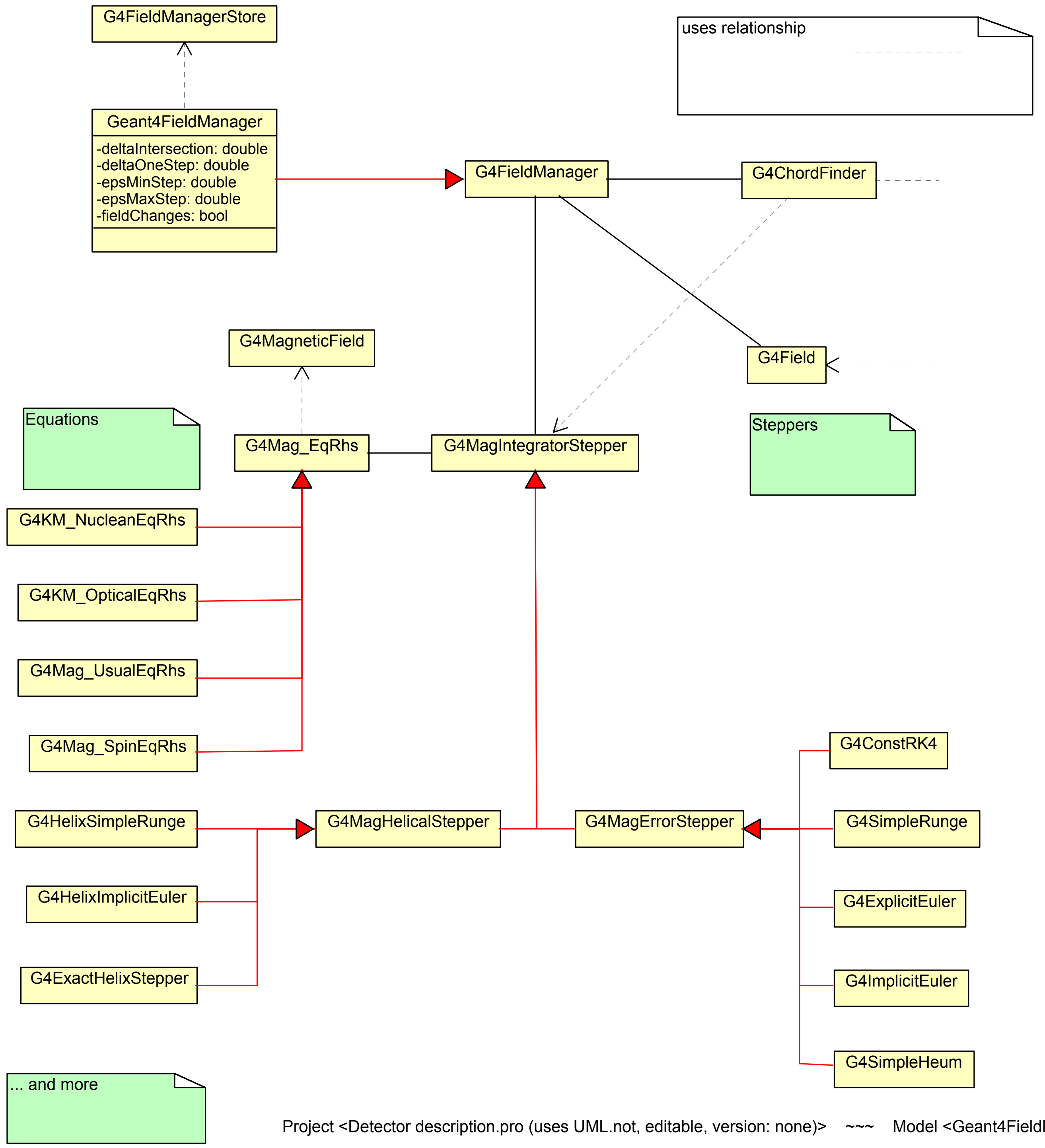

Segmentation
uses
IDDescriptor
segmentation #1
id #1
Readout
readout #1
Volume
sens_det #0...1
SensitiveDetector
region #0...1
Region
string
hits_collection #0...*
limitset #0..*
LimitSet
Used by Geant4

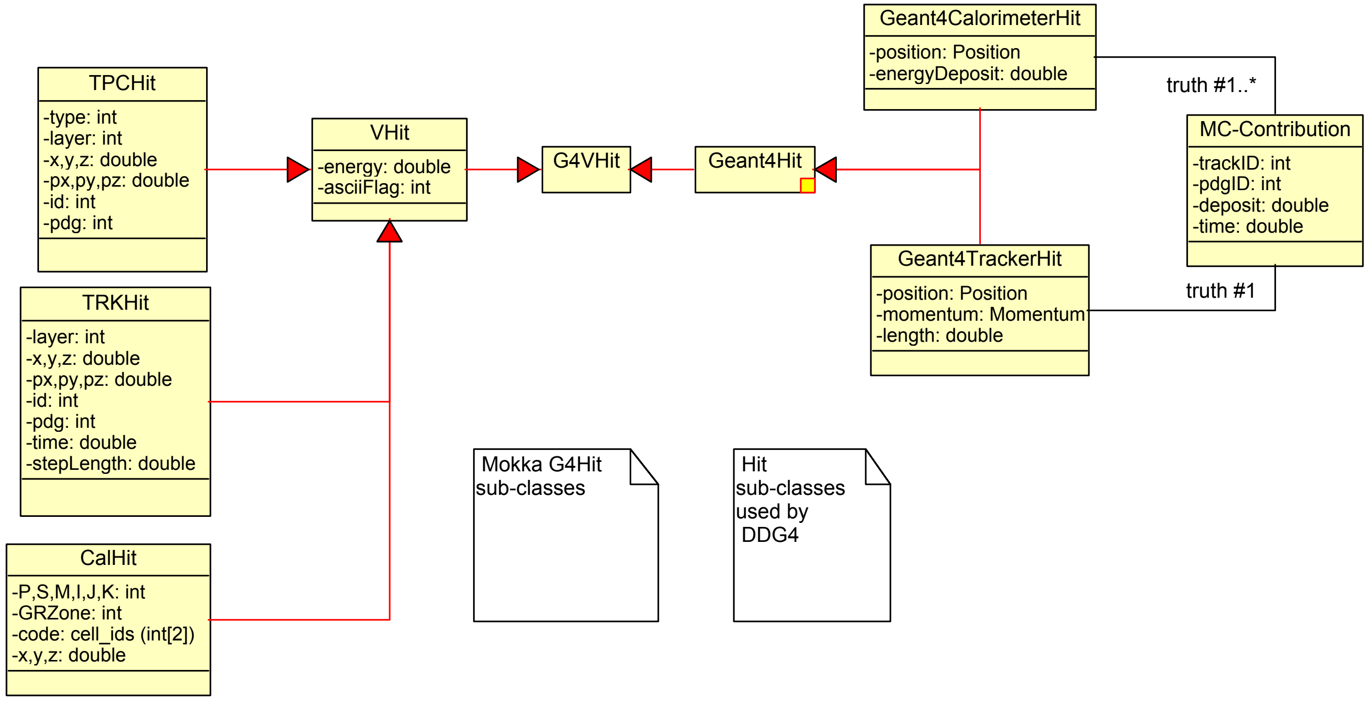

Segmentation
uses
IDDescriptor
segmentation #1
id #1
Readout

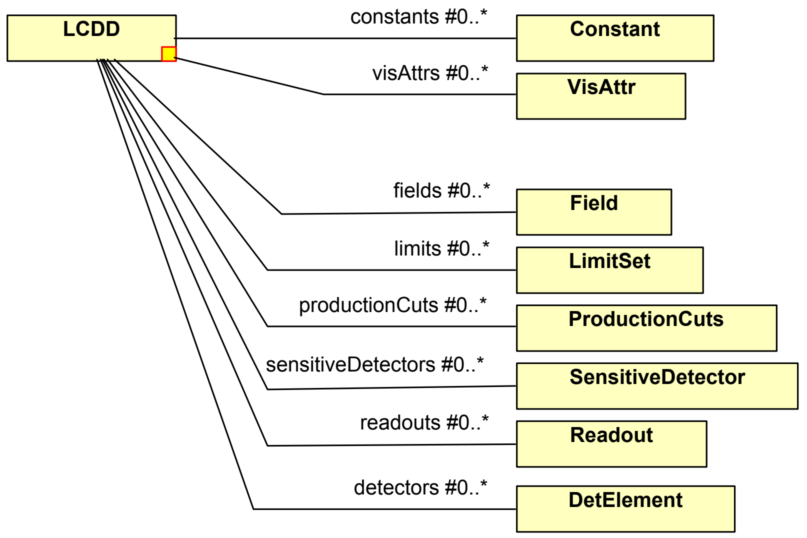

Ref_t
SensitiveDetector
SensitiveDetector::Object
-verbose: bool
-combine_hits: int
-ecut: double
-eunit: std::string
hits_collection #1
string
extensions #0..*
Extensions
Readout
readout #0...1

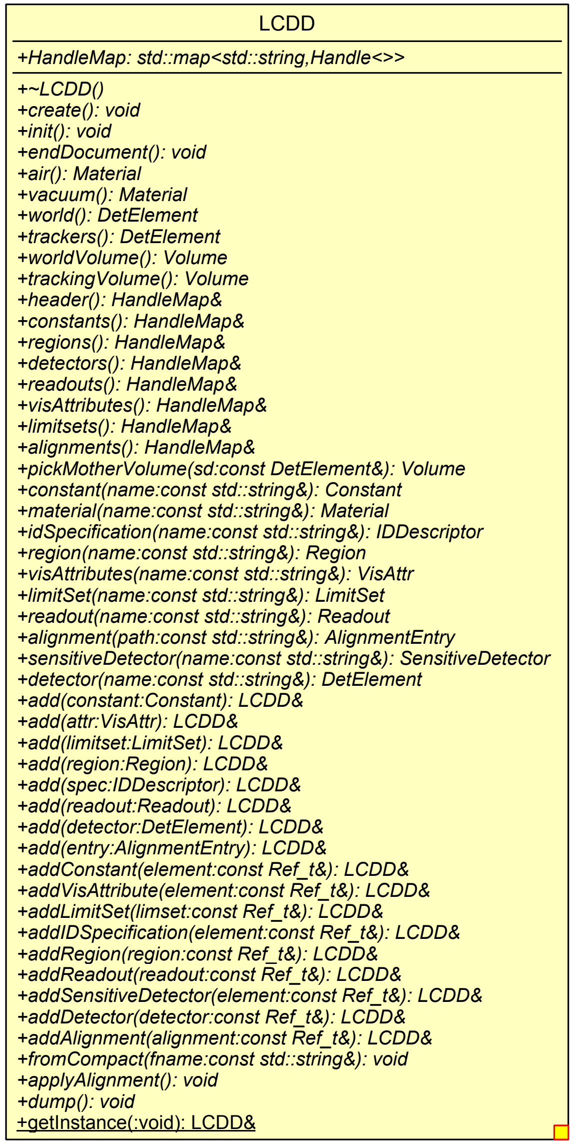

Geant4ScorerSD
Geant4TrackerSD
G4VSensitiveDetector
SensitiveDetector
Geant4TrackerCombineSD
Geant4SensitiveDetector
Geant4CalorimeterSD
Geant4OpticalCalorimeterSD
Geant4UnsegmentedCalorimeter


G4FieldManagerStore
G4Field
G4ChordFinder
G4FieldManager
G4MagIntegratorStepper
G4Mag_EqRhs
G4HelixSimpleRunge
G4MagHelicalStepper
G4MagErrorStepper
G4SimpleRunge
G4HelixImplicitEuler
G4ExplicitEuler
G4ImplicitEuler
G4SimpleHeum
... and more


G4FieldManagerStore
Geant4FieldManager
-deltaIntersection: double
-deltaOneStep: double
-epsMinStep: double
-epsMaxStep: double
-fieldChanges: bool
uses relationship
G4FieldManager
G4ChordFinder
G4MagneticField
G4Field
Equations
G4Mag_EqRhs
G4MagIntegratorStepper
Steppers
G4KM_NucleanEqRhs
G4KM_OpticalEqRhs
G4Mag_UsualEqRhs
G4Mag_SpinEqRhs
G4ConstRK4
G4HelixSimpleRunge
G4MagHelicalStepper
G4MagErrorStepper
G4SimpleRunge
G4HelixImplicitEuler
G4ExplicitEuler
G4ExactHelixStepper
G4ImplicitEuler
G4SimpleHeum
... and more


TPCHit
-type: int
-layer: int
-x,y,z: double
-px,py,pz: double
-id: int
-pdg: int
TRKHit
-layer: int
-x,y,z: double
-px,py,pz: double
-id: int
-pdg: int
-time: double
-stepLength: double
CalHit
-P,S,M,I,J,K: int
-GRZone: int
-code: cell_ids (int[2])
-x,y,z: double
VHit
-energy: double
-asciiFlag: int
G4VHit
Geant4Hit
Geant4CalorimeterHit
-position: Position
-energyDeposit: double
Geant4TrackerHit
-position: Position
-momentum: Momentum
-length: double
MC-Contribution
-trackID: int
-pdgID: int
-deposit: double
-time: double
truth #1..*
truth #1
Mokka G4Hit sub-classes
Hit sub-classes used by DDG4

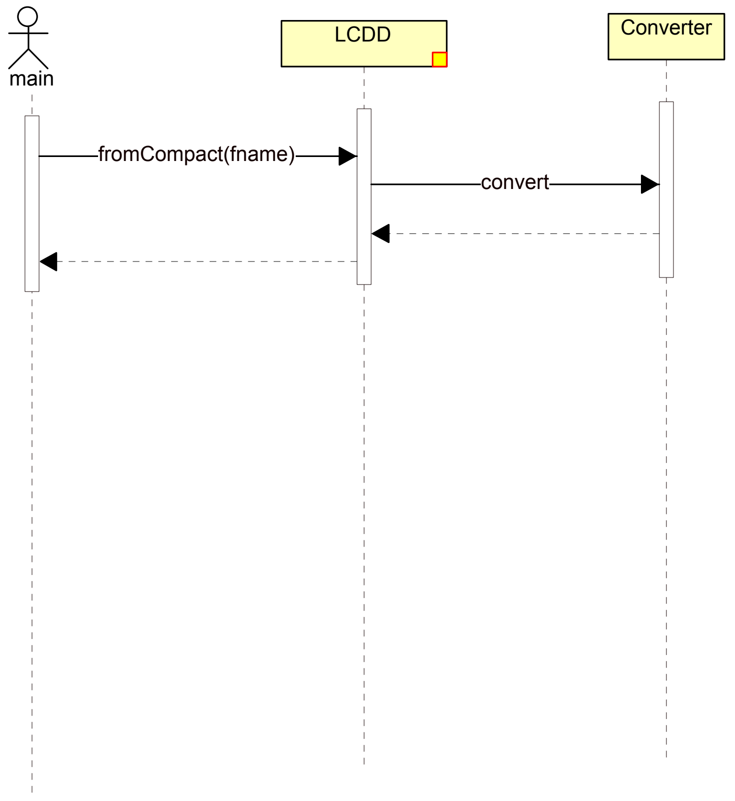

LCDD
constants #0..*
Constant
visAttrs #0..*
VisAttr
fields #0..*
Field
limits #0..*
LimitSet
productionCuts #0..*
ProductionCuts
sensitiveDetectors #0..*
SensitiveDetector
readouts #0..*
Readout
detectors #0..*
DetElement

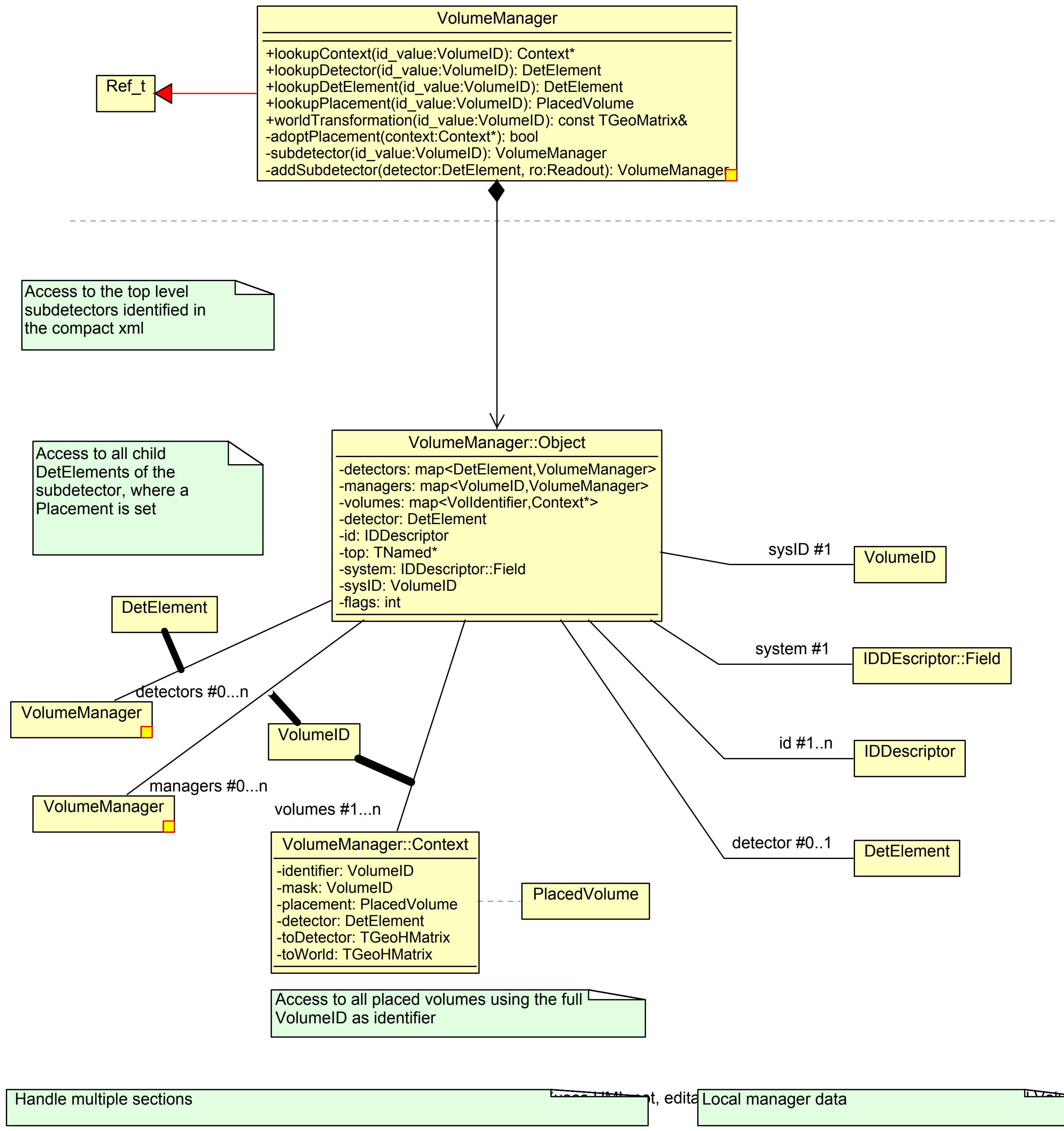


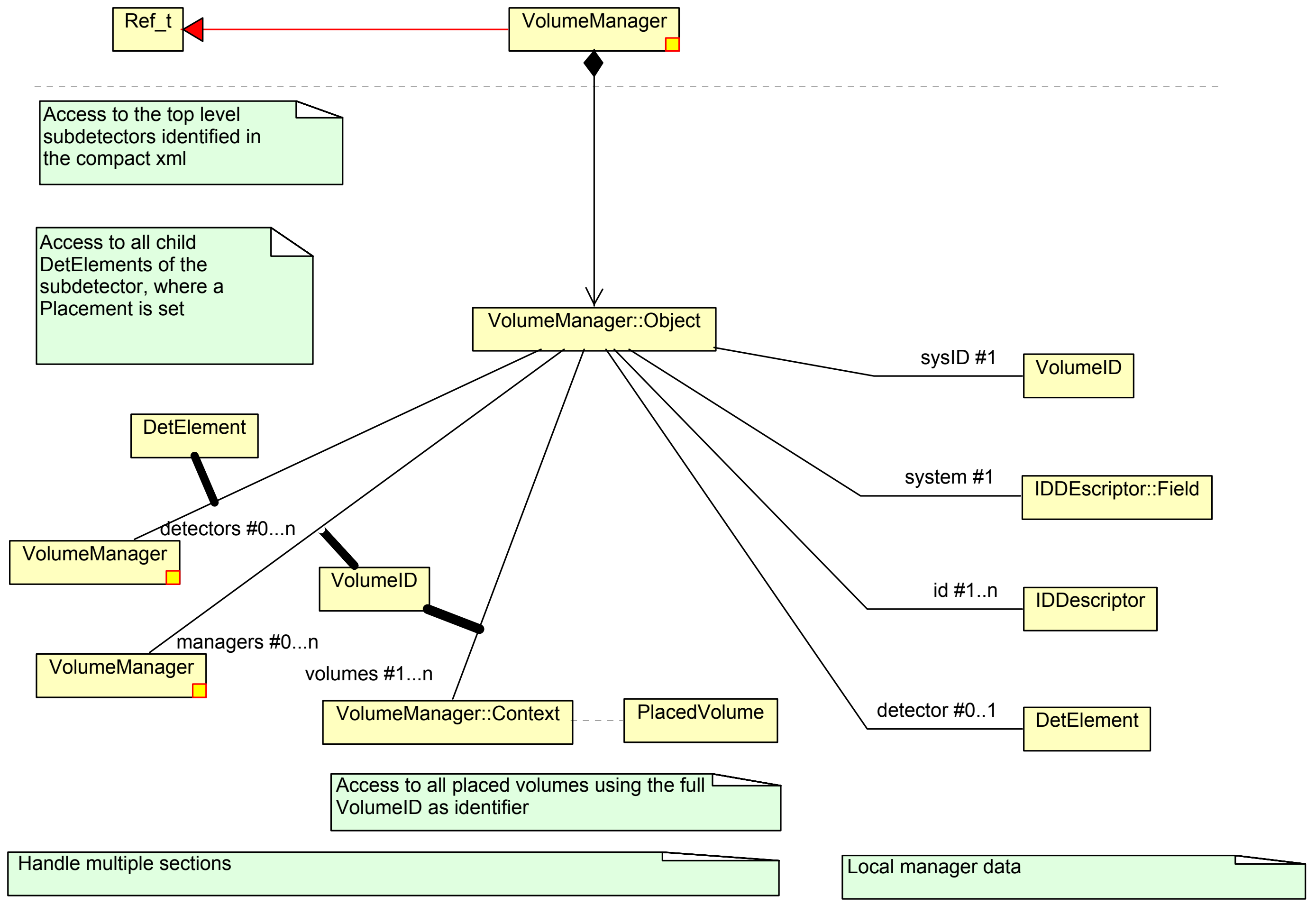
LCDD

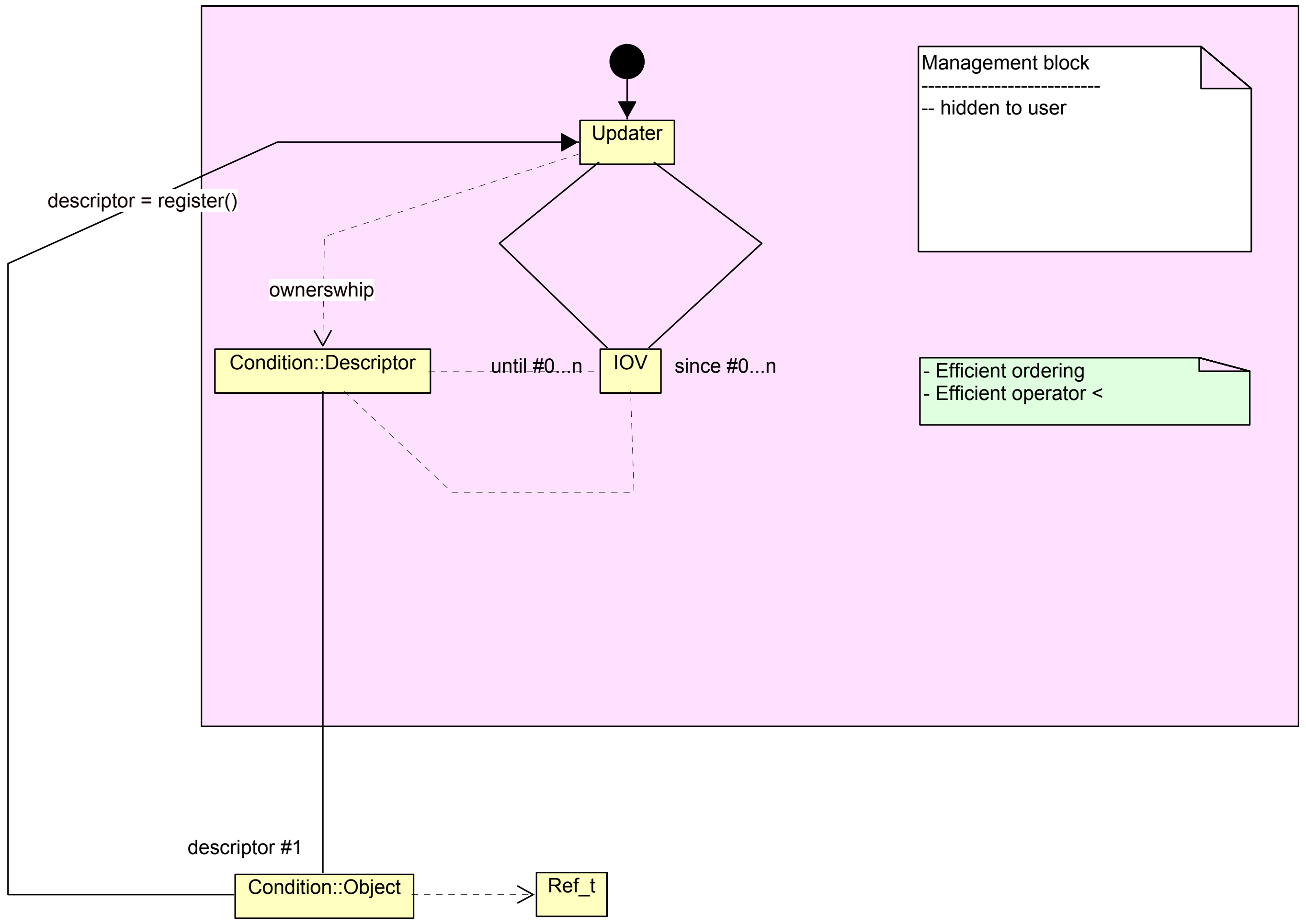
+HandleMap: std::map<std::string,Handle<>>

+~LCDD()
+create(): void
+init(): void
+endDocument(): void
+air(): Material
+vacuum(): Material
+world(): DetElement
+trackers(): DetElement
+worldVolume(): Volume
+trackingVolume(): Volume
+header(): HandleMap&
+constants(): HandleMap&
+regions(): HandleMap&
+detectors(): HandleMap&
+readouts(): HandleMap&
+visAttributes(): HandleMap&
+limitsets(): HandleMap&
+alignments(): HandleMap&
+pickMotherVolume(sd:const DetElement&): Volume
+constant(name:const std::string&): Constant
+material(name:const std::string&): Material
+idSpecification(name:const std::string&): IDDescriptor
+region(name:const std::string&): Region
+visAttributes(name:const std::string&): VisAttr
+limitSet(name:const std::string&): LimitSet
+readout(name:const std::string&): Readout
+alignment(path:const std::string&): AlignmentEntry
+sensitiveDetector(name:const std::string&): SensitiveDetector
+detector(name:const std::string&): DetElement
+add(constant:Constant): LCDD&
+add(attr:VisAttr): LCDD&
+add(limitset:LimitSet): LCDD&
+add(region:Region): LCDD&
+add(spec:IDDescriptor): LCDD&
+add(readout:Readout): LCDD&
+add(detector:DetElement): LCDD&
+add(entry:AlignmentEntry): LCDD&
+addConstant(element:const Ref_t&): LCDD&
+addVisAttribute(element:const Ref_t&): LCDD&
+addLimitSet(limset:const Ref_t&): LCDD&
+addIDSpecification(element:const Ref_t&): LCDD&
+addRegion(region:const Ref_t&): LCDD&
+addReadout(readout:const Ref_t&): LCDD&
+addSensitiveDetector(element:const Ref_t&): LCDD&
+addDetector(detector:const Ref_t&): LCDD&
+addAlignment(alignment:const Ref_t&): LCDD&
+fromCompact(fname:const std::string&): void
+applyAlignment(): void
+dump(): void
+getInstance(:void): LCDD&

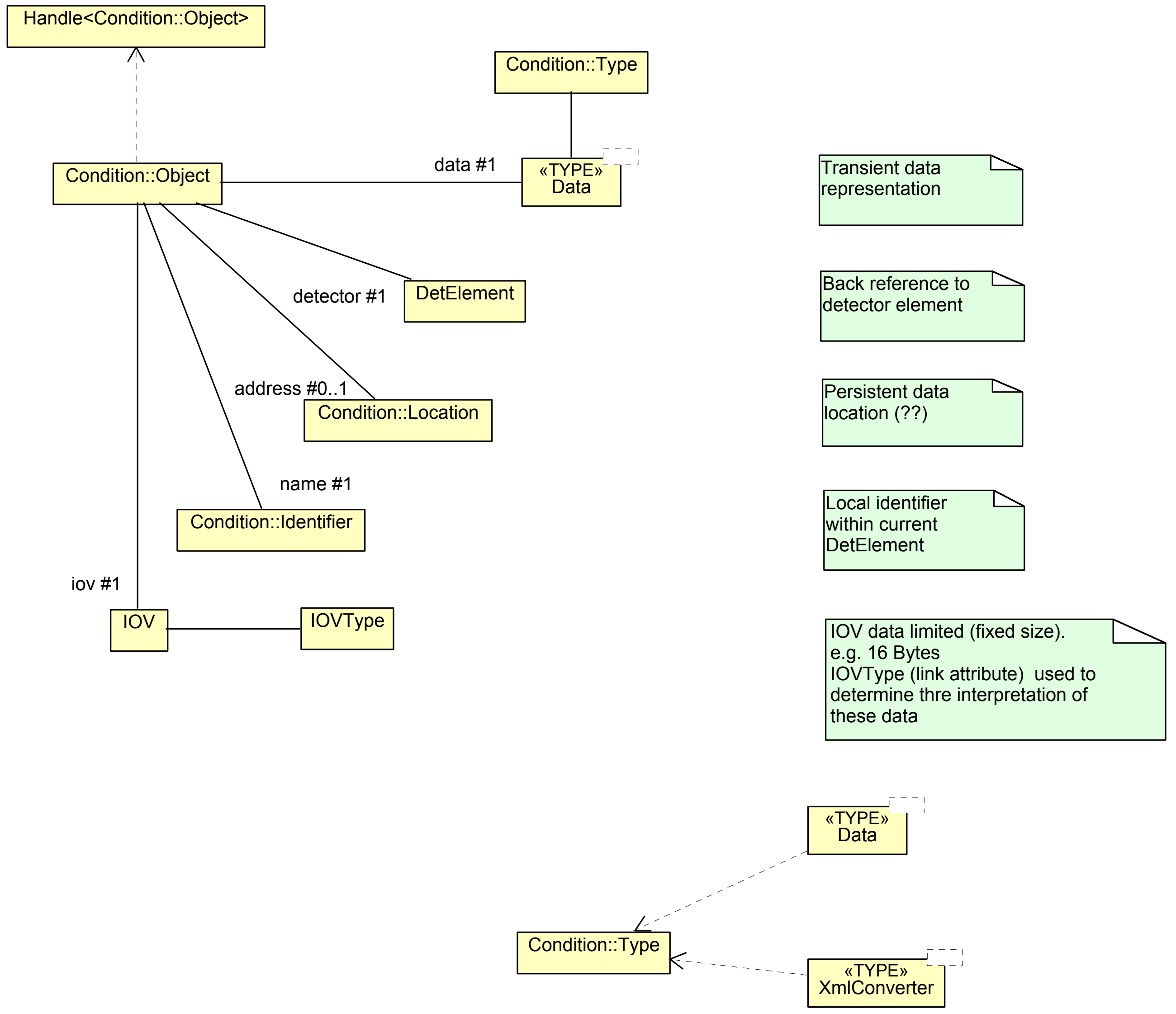

main
LCDD
Converter
fromCompact(fname)
convert

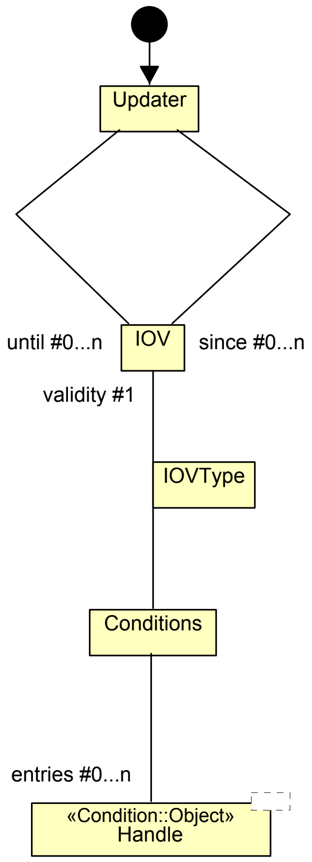

VolumeManager
+lookupContext(id_value:VolumeID): Context*
+lookupDetector(id_value:VolumeID): DetElement
+lookupDetElement(id_value:VolumeID): DetElement
+lookupPlacement(id_value:VolumeID): PlacedVolume
+worldTransformation(id_value:VolumeID): const TGeoMatrix&
-adoptPlacement(context:Context*): bool
-subdetector(id_value:VolumeID): VolumeManager
-addSubdetector(detector:DetElement, ro:Readout): VolumeManager
Ref_t
Access to the top level subdetectors identified in the compact xml
Access to all child DetElements of the subdetector, where a Placement is set
VolumeManager::Object
-detectors: map<DetElement,VolumeManager>
-managers: map<VolumeID,VolumeManager>
-volumes: map<VolIdentifier,Context*>
-detector: DetElement
-id: IDDescriptor
-top: TNamed*
-system: IDDescriptor::Field
-sysID: VolumeID
-flags: int
sysID #1
VolumeID
system #1
IDDEscriptor::Field
id #1..n
IDDescriptor
detector #0..1
DetElement
DetElement
detectors #0...n
VolumeManager
VolumeID
managers #0...n
VolumeManager
volumes #1...n
VolumeManager::Context
-identifier: VolumeID
-mask: VolumeID
-placement: PlacedVolume
-detector: DetElement
-toDetector: TGeoHMatrix
-toWorld: TGeoHMatrix
PlacedVolume
Access to all placed volumes using the full VolumeID as identifier
Handle multiple sections
Local manager data


Ref_t
VolumeManager
Access to the top level subdetectors identified in the compact xml
Access to all child DetElements of the subdetector, where a Placement is set
VolumeManager::Object
sysID #1
VolumeID
DetElement
system #1
IDDEscriptor::Field
detectors #0...n
VolumeManager
VolumeID
id #1..n
IDDescriptor
managers #0...n
VolumeManager
volumes #1...n
VolumeManager::Context
PlacedVolume
detector #0..1
DetElement
Access to all placed volumes using the full VolumeID as identifier
Handle multiple sections
Local manager data


Management block

-- hidden to user
Updater
descriptor = register()
ownerswhip
Condition::Descriptor
until #0...n
IOV
since #0...n
- Efficient ordering
- Efficient operator <
descriptor #1
Condition::Object
Ref_t


Handle<Condition::Object>
Condition::Type
«TYPE» Data
data #1
Condition::Object
DetElement
detector #1
address #0..1
Condition::Location
name #1
Condition::Identifier
iov #1
IOV
IOVType
Transient data representation
Back reference to detector element
Persistent data location (??)
Local identifier within current DetElement
IOV data limited (fixed size). e.g. 16 Bytes IOVType (link attribute) used to determine thre interpretation of these data
«TYPE» Data
Condition::Type
«TYPE» XmlConverter



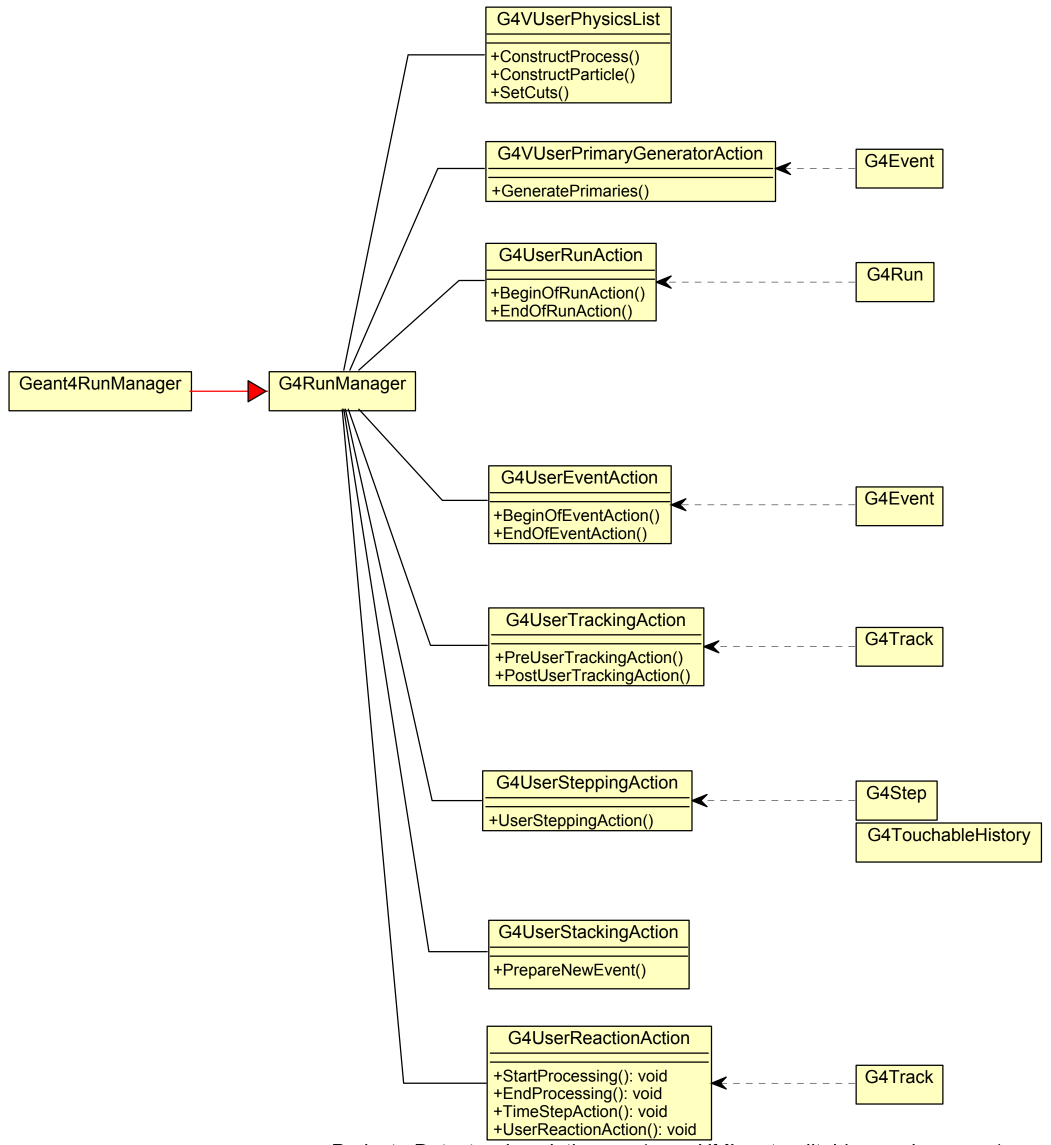
Updater determines from IOV which items to "update" according to criteria defined by the event processing:
- Event time
- Run number
...

IOV data limited (fixed size).
e.g. 16 Bytes
IOVType (link attribute) used to determine thre interpretation of these data

Container IOV describes the minimal range of validities of it's children.
Usage only to optimize lookups for conditions to be updated.

Geant4 provided user hooks

Any simulation program is supposed to overwrite the appropriate actions neccessary to perform the required functionality.
This means a flexible simulation program must hook into these callbacks and provide for each entry action queues to ease the implementation of small components.



Geant4Concept

General concept

Granularity:
-- One single G4 action entry leads to lengthy actions and spagetti-code.
Allow the seperation of each action into a sequence of individual action
each only serving a very special pupose and hence be very granular
To be seen: Is a mechansm necessary to interrupt an action sequence
and terminate the processing prematurely?

-- Sequencing:
Whereas for tracking-, stepping, run-, etc. actions the existing granularity looks sufficient,
for the event action sequences more "artificial" granularity may be desireable.

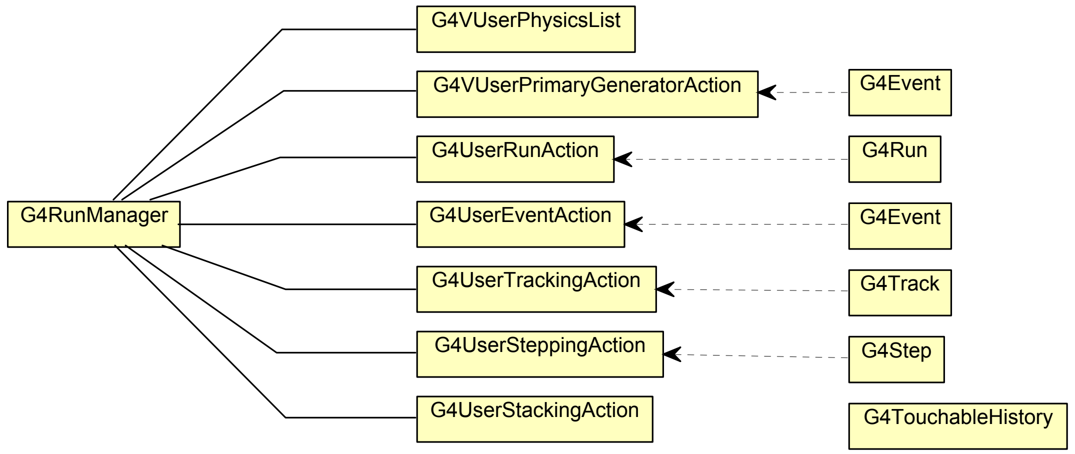
For example monitoring components of certain subdetectors
- wants to be initialized for each event,
- be called after processing the event
Such required functionality automatically leads to "Processing Phases",
where the simulation performed by Geant4 is only one of these.
Other phases may be the primary event generation or - as mentioned -
monitoring.

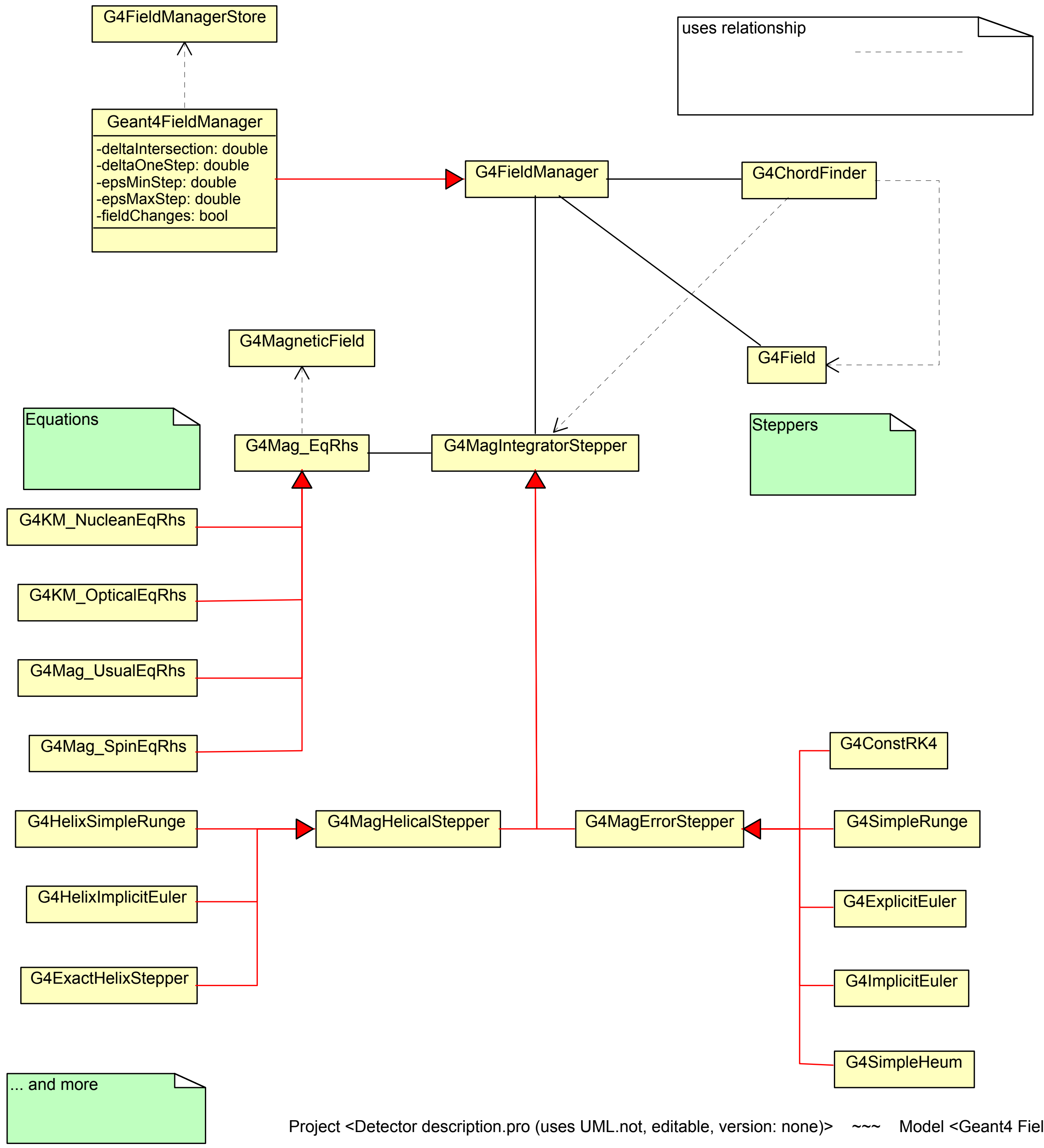
Important:
-- Any action may register for any phase supplying a member function as a callback
accepting the G4Event and the phase name as an argument

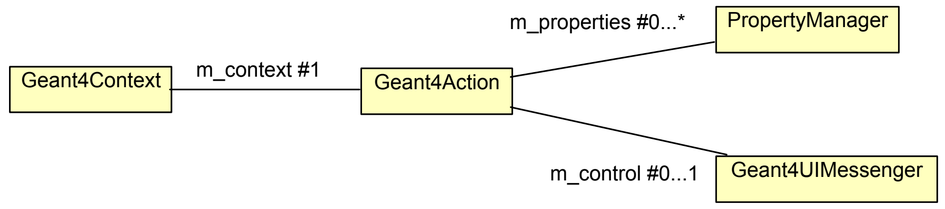
Since any action may register for any phase, this is sort of orthogonal to the action sequences,
which do not allow for calls of independent objects which do not belong to the same type of G4
callback.

If necessary the phase concept may be extended to runs.

-- Flexibility and Open-ness:
All actions must be simple.
Any setup functionality is outside the component.
Hence, various setup mechanisms may be attached: XML, python, Cint, ...


G4VUserPhysicsList
G4VUserPrimaryGeneratorAction
G4Event
G4UserRunAction
G4Run
G4RunManager
G4UserEventAction
G4Event
G4UserTrackingAction
G4Track
G4UserSteppingAction
G4Step
G4UserStackingAction
G4TouchableHistory

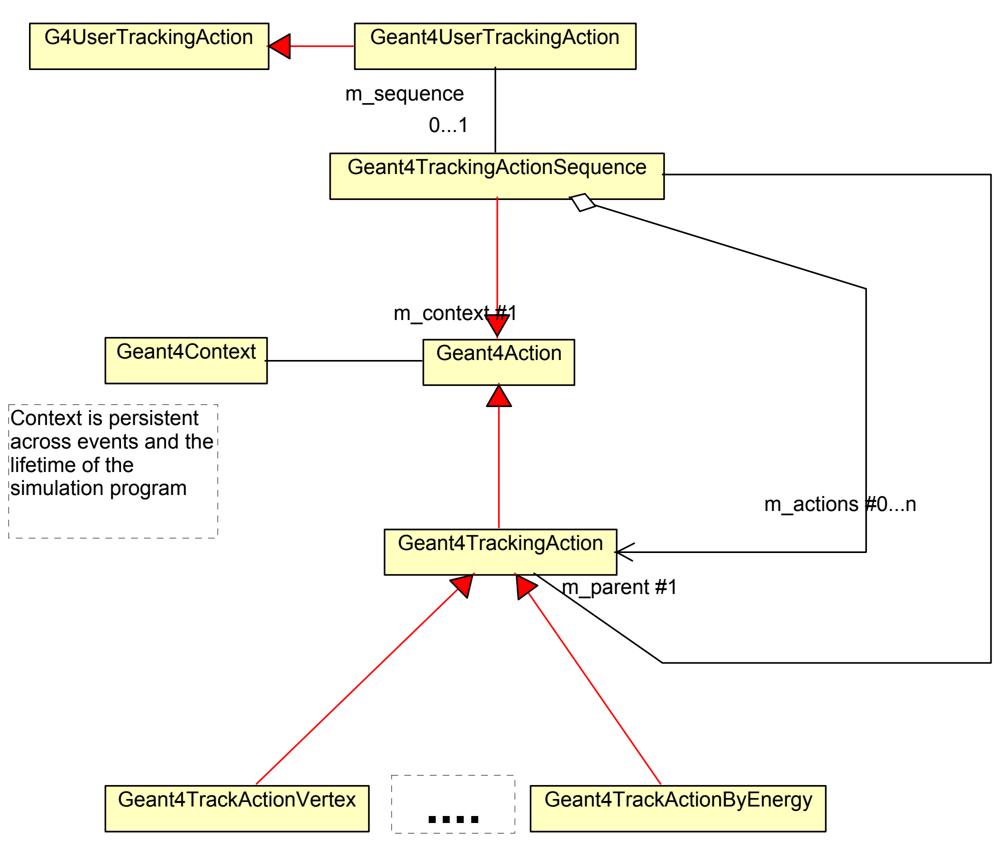

G4FieldManagerStore
Geant4FieldManager
-deltaIntersection: double
-deltaOneStep: double
-epsMinStep: double
-epsMaxStep: double
-fieldChanges: bool
uses relationship
G4FieldManager
G4ChordFinder
G4MagneticField
G4Field
Equations
G4Mag_EqRhs
G4MagIntegratorStepper
Steppers
G4KM_NucleanEqRhs
G4KM_OpticalEqRhs
G4Mag_UsualEqRhs
G4Mag_SpinEqRhs
G4ConstRK4
G4HelixSimpleRunge
G4MagHelicalStepper
G4MagErrorStepper
G4SimpleRunge
G4HelixImplicitEuler
G4ExplicitEuler
G4ExactHelixStepper
G4ImplicitEuler
G4SimpleHeum
... and more


m_properties #0...*
PropertyManager
Geant4Context
m_context #1
Geant4Action
Geant4UIMessenger
m_control #0...1

Geant4 Action example: G4UserTrackingAction

Similar construct for EventAction, RunAction, etc.

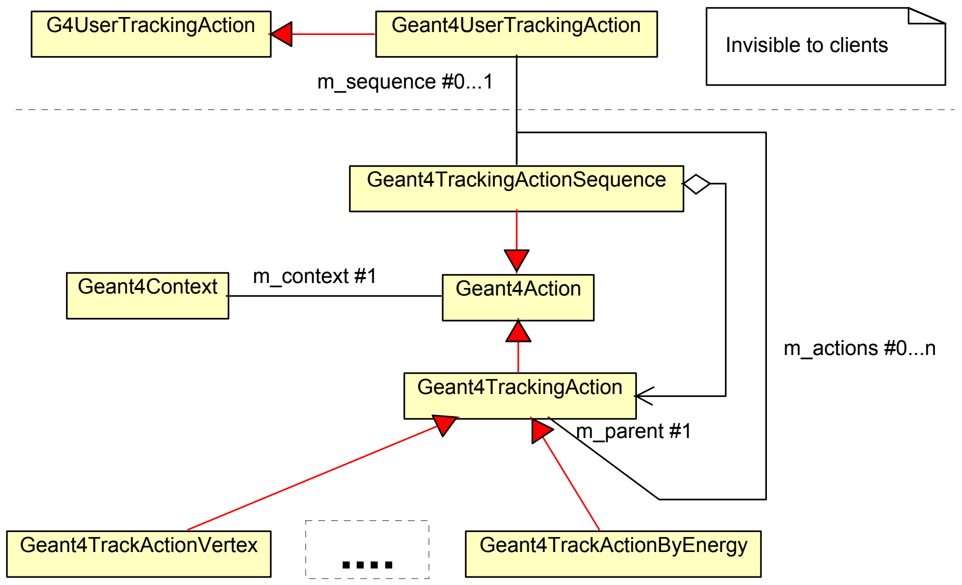
Concept lended from GiGa (I.Belyayev / ITEP)

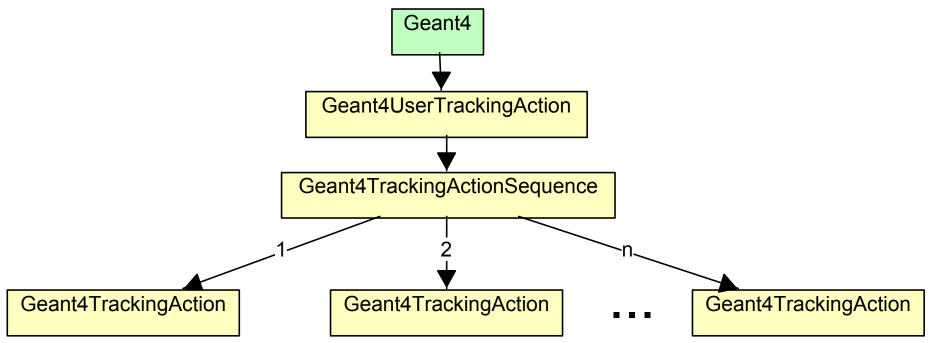


Every G4 user action type is divided into a sequence of 'smallish' components - each only addressing a small well defined job.

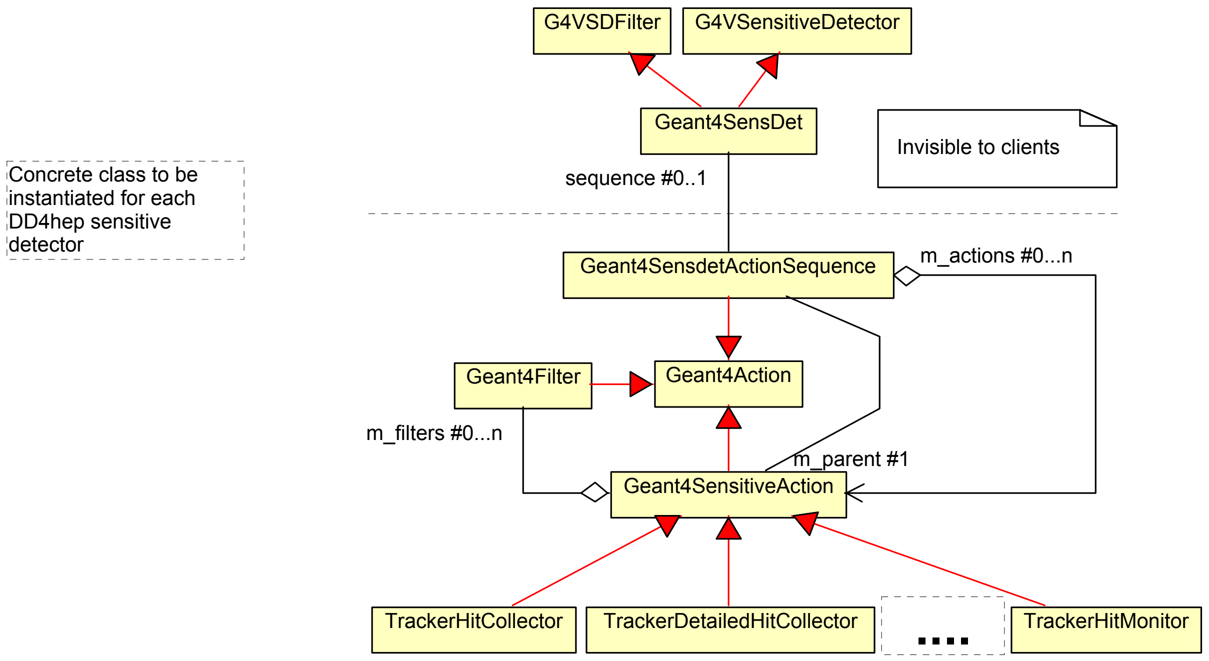
Common base class with common interface to all Geant4 user actions

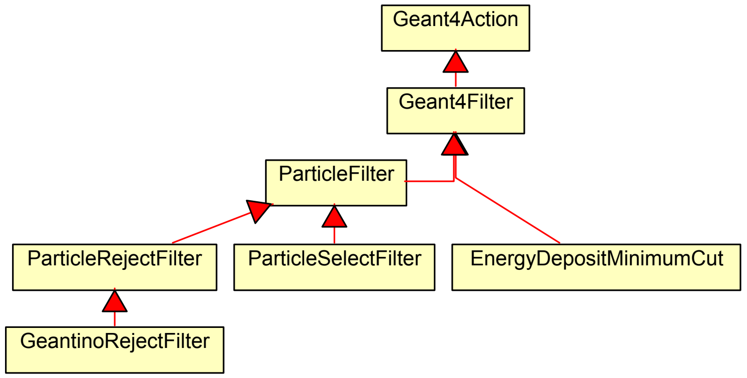
Specialized actions in the sequence for specialized functionality
-- track storage
-- track filtering
-- monitoring
-- etc.

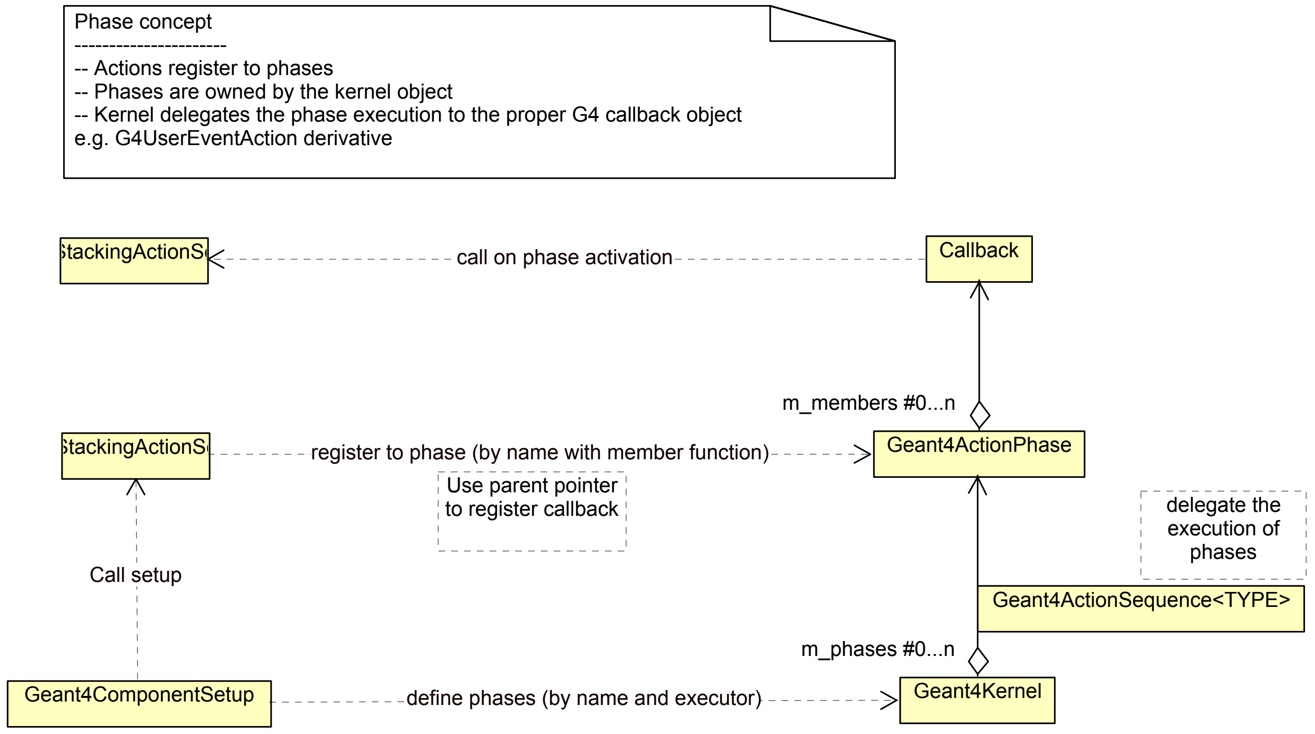

G4UserTrackingAction
Geant4UserTrackingAction
Invisible to clients
m_sequence #0...1
Geant4TrackingActionSequence
Geant4Context
m_context #1
Geant4Action
m_actions #0...n
Geant4TrackingAction
m_parent #1
Geant4TrackActionVertex
....
Geant4TrackActionByEnergy


Geant4
Geant4UserTrackingAction
Geant4TrackingActionSequence
1
2
n
Geant4TrackingAction
Geant4TrackingAction
...
Geant4TrackingAction

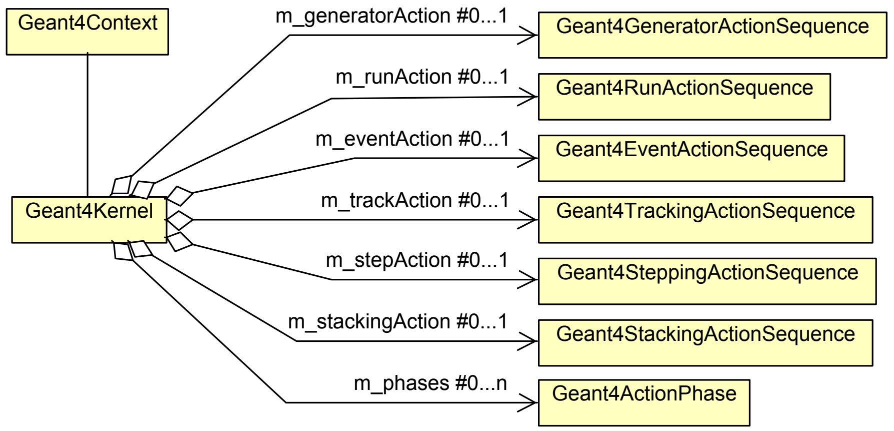
Another sequence example: Sequencing sensitive detector actions




Geant4Action
Geant4Filter
ParticleFilter
ParticleRejectFilter
ParticleSelectFilter
EnergyDepositMinimumCut
GeantinoRejectFilter


Phase concept

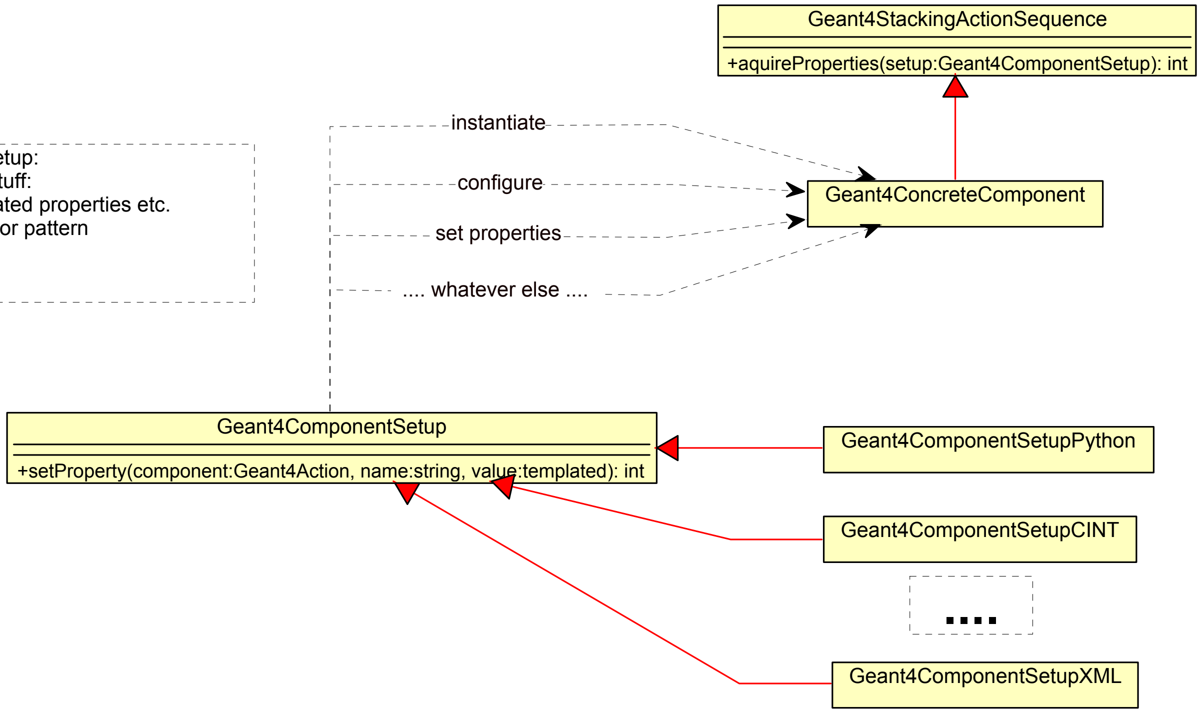
-- Actions register to phases
-- Phases are owned by the kernel object
-- Kernel delegates the phase execution to the proper G4 callback object
e.g. G4UserEventAction derivative
StackingActionS
call on phase activation
Callback
m_members #0...n
StackingActionS
register to phase (by name with member function)
Geant4ActionPhase
Use parent pointer to register callback
delegate the execution of phases
Call setup
Geant4ActionSequence<TYPE>
m_phases #0...n
Geant4ComponentSetup
define phases (by name and executor)
Geant4Kernel

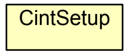

Geant4Context
Geant4Kernel
m_generatorAction #0...1
Geant4GeneratorActionSequence
m_runAction #0...1
Geant4RunActionSequence
m_eventAction #0...1
Geant4EventActionSequence
m_trackAction #0...1
Geant4TrackingActionSequence
m_stepAction #0...1
Geant4SteppingActionSequence
m_stackingAction #0...1
Geant4StackingActionSequence
m_phases #0...n
Geant4ActionPhase

Setup of actions and action sequences
-- Independent of the action implementation
-- Hence, several flavours possible
xml, python, cint, ...

Requires:
Well defined interface to interact with 'generic' Geant4Actions to set properties of all kinds. Otherwise the setup will be quite difficult and probably will require one setup component per action component, which is too much work!

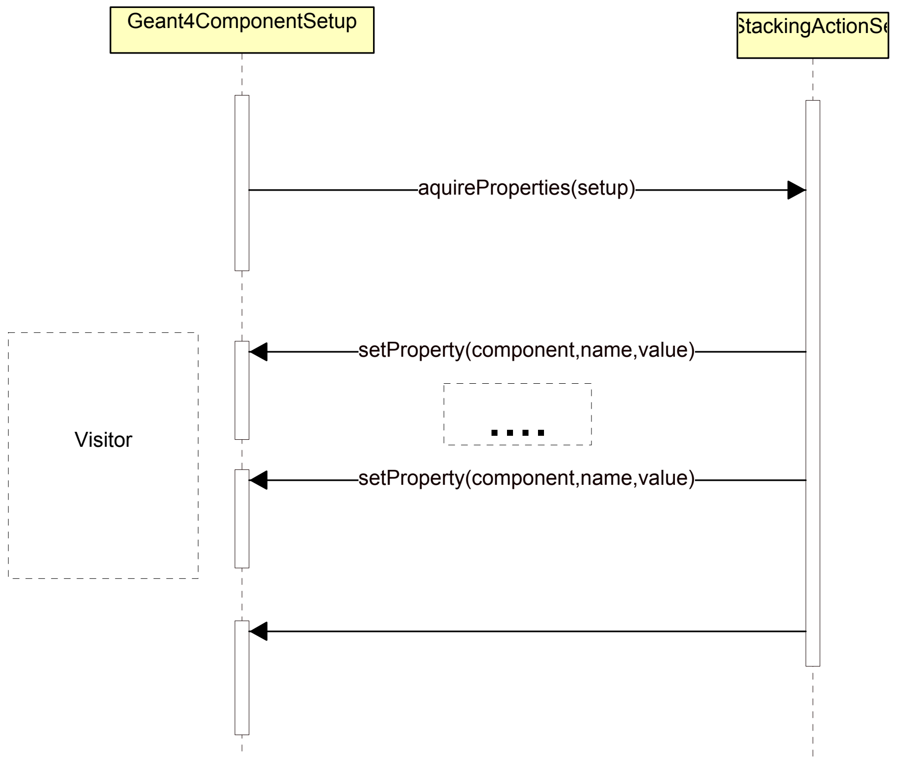
Property setup:
-- Simple stuff:
No deleated properties etc.
-- Use Visitor pattern

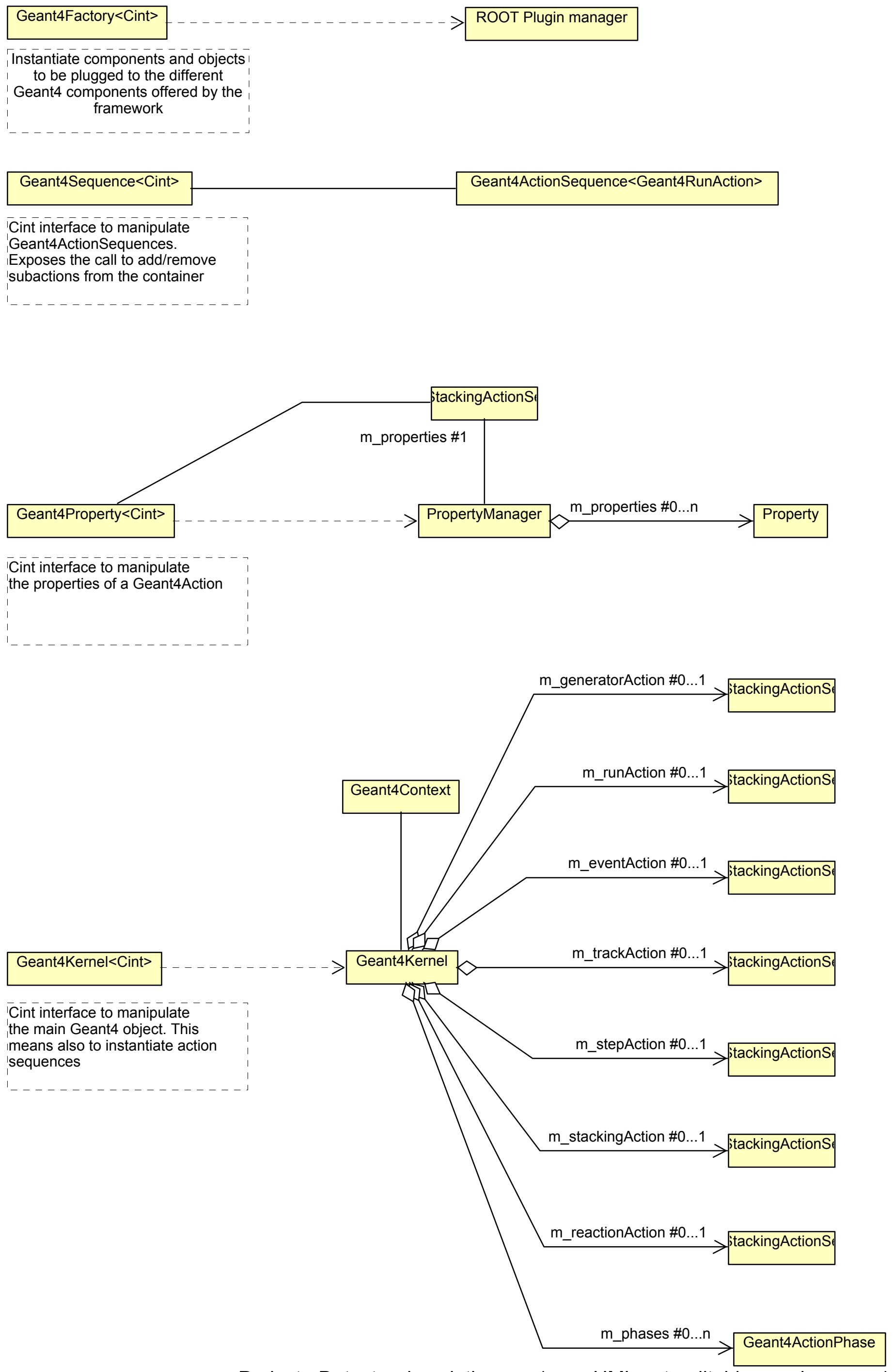

Geant4StackingActionSequence
+aquireProperties(setup:Geant4ComponentSetup): int
instantiate
configure
set properties
.... whatever else ....
Geant4ConcreteComponent
Geant4ComponentSetup
+setProperty(component:Geant4Action, name:string, value:templated): int
Geant4ComponentSetupPython
Geant4ComponentSetupCINT
....
Geant4ComponentSetupXML

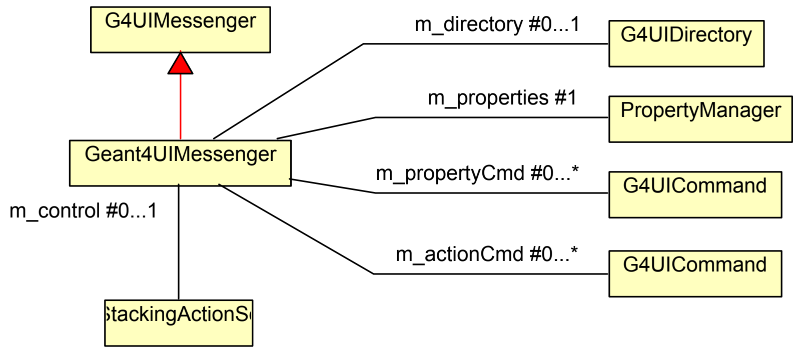


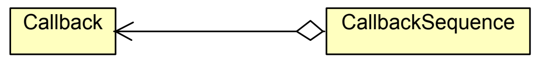
Property setup

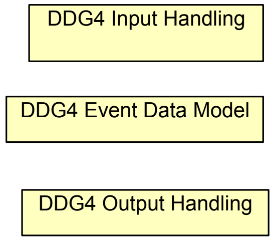
Note:
Property manipulation is NOT in the action.
Property manipulation is purely done by the setup component
This should lead us to a thin program, since after the setup all
objects concerned with the setup may be discarded

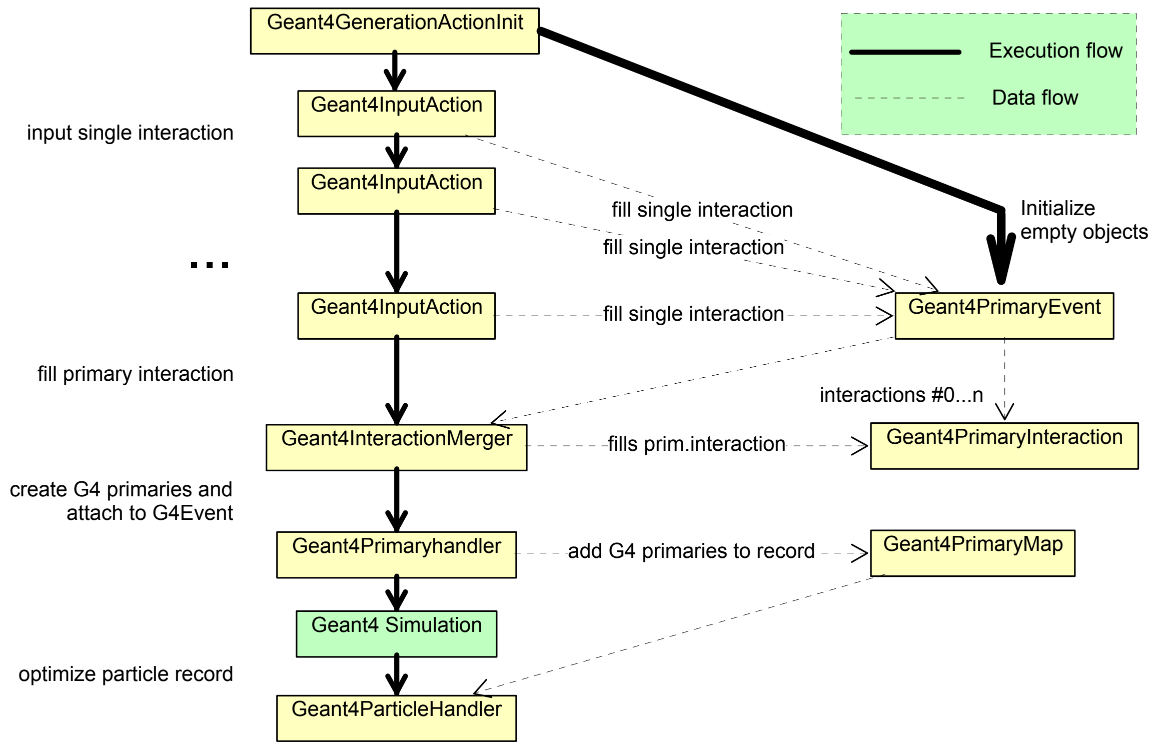


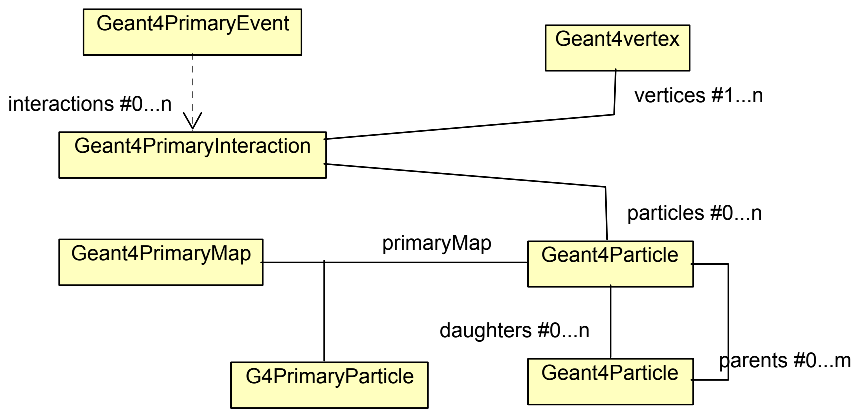

Geant4Factory<Cint>
ROOT Plugin manager
Instantiate components and objects to be plugged to the different Geant4 components offered by the framework
Geant4Sequence<Cint>
Geant4ActionSequence<Geant4RunAction>
Cint interface to manipulate Geant4ActionSequences. Exposes the call to add/remove subactions from the container
StackingActionSe
m_properties #1
Geant4Property<Cint>
PropertyManager
m_properties #0...n
Property
Cint interface to manipulate the properties of a Geant4Action
m_generatorAction #0...1
m_runAction #0...1
m_eventAction #0...1
m_trackAction #0...1
m_stepAction #0...1
m_stackingAction #0...1
m_reactionAction #0...1
m_phases #0...n
Geant4Context
Geant4Kernel<Cint>
Geant4Kernel
Geant4ActionPhase
Cint interface to manipulate the main Geant4 object. This means also to instantiate action sequences


G4UIMessenger
m_directory #0...1
G4UIDirectory
m_properties #1
PropertyManager
Geant4UIMessenger
m_propertyCmd #0...*
G4UICommand
m_control #0...1
m_actionCmd #0...*
G4UICommand
StackingActionSe


Callback
CallbackSequence


DDG4 Input Handling
DDG4 Event Data Model
DDG4 Output Handling

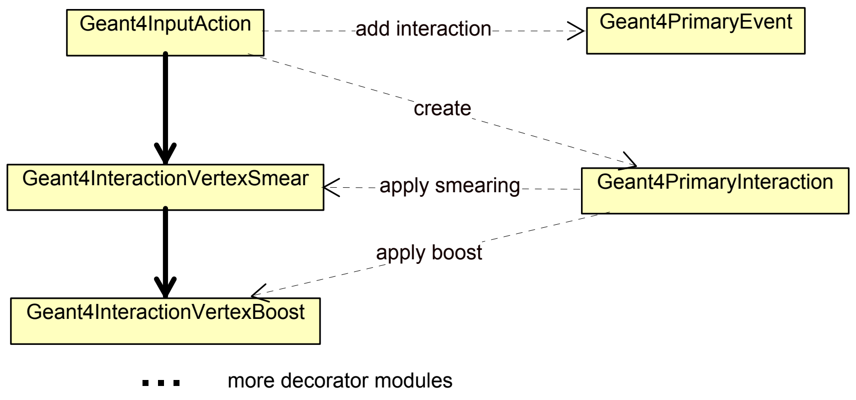

Geant4GenerationActionInit
Execution flow
Data flow
Geant4InputAction
input single interaction
Geant4InputAction
fill single interaction
fill single interaction
Initialize empty objects
...
Geant4InputAction
fill single interaction
Geant4PrimaryEvent
fill primary interaction
interactions #0...n
Geant4InteractionMerger
fills prim.interaction
Geant4PrimaryInteraction
create G4 primaries and attach to G4Event
Geant4Primaryhandler
add G4 primaries to record
Geant4PrimaryMap
Geant4 Simulation
optimize particle record
Geant4ParticleHandler



- A Geant4PrimaryEvent consists of one or several Geant4PrimaryInteraction objects

- A Geant4PrimaryInteraction consist of 2 lists: one for the associated Geant4Vertex objects describing the *primary* vertices. The particle list describes the associated particles (type Geant4Particle) in terms of their physical quantities.

- Particle relationships (class Geant4Particle) are described by the daughter parent relationships only. Any vertex information is explicitly aggregated into the particle.


Geant4InputAction
add interaction
Geant4PrimaryEvent
create
Geant4InteractionVertexSmear
apply smearing
Geant4PrimaryInteraction
apply boost
Geant4InteractionVertexBoost
more decorator modules