GLOBAL AND LOCAL ACTOR

User Guide

Mateusz Bała
bala.mateusz@gmail.com
Jagiellonian University
Kraków, Poland

http://koza.if.uj.edu.pl/pet/

CONTENTS

ACRONYMS

GA	Global Actor
LA	Local Actor

INTRODUCTION

Global and Local Actor are special designed classes of actors to global data acquisition. Actor is a GATE concept for data collecting during simulation from selected volume, and each one actor generate for attached to it volume single ROOT file with data. This standard concept of actor is useful when we want to focus on single volume and examine it in a special case. When comes to compare data from many volumes collected by standard actor (for example we have volume A and B and both are attached to the same actor, we receive two ROOT files) problems with analysis start. To skip this problem I created new design of actor - Global and Local Actor. In contrast to standard actors GA with LA can collect data from any number of volumes to single ROOT file with keeping the chronology of their recording. Moreover interface provided by GA and LA is simple and flexible. You can control in GATE macro what (particle name, energy, momentum, etc.) and when (for which process or/and particle, specific energy deposition, etc.). Just by writing few lines of macro you can manage your data acquisition in a completely different way than before, it was offered by other standard GATE actors. This guide shows you how to use GA and LA. After reading this guide you will know how and when to use the solutions offered here.

CHAPTER 1

GLOBAL AND LOCAL ACTOR HOW IT WORKS

1.1 Let's start with an example!

Let's suppose that you have two layers geometry detector (first layer has a name ”volA”, second ”voldB”) and you want to save data about Compton scattering of gammas in this detector. Moreover you want to know:

- name of volume where scattering happened
- energy deposition during scattering
- eventID and trackID
- gamma momentum direction before and after scattering

and this is all what you need for you hypothetical analysis.

All you have to do for setup this conditions is write simple macro and you will receive an output ROOT file with exactly this variables in tree structure.

Macro is short and easy to understand:

```
# Attaching to volume A with name "volA"
/gate/actor/addActor LocalActor LA_A
/gate/actor/LA_A/attachTo volA

# Attaching to volume B with name "volB"
/gate/actor/addActor LocalActor LA_B
/gate/actor/LA_B/attachTo volB

#Output file name
/gate/actor/LA_A/global/saveTo example.root

#Save only below listed things:

/gate/actor/LA_A/global/enableVolumeName
/gate/actor/LA_A/global/enableEnergyLossDuringProcess
/gate/actor/LA_A/global/enableEventID
/gate/actor/LA_A/global/enableTrackID
/gate/actor/LA_A/global/enableMomentumDirectionBeforeProcess
/gate/actor/LA_A/global/enableMomentumDirectionAfterProcess

#saves this data only when the following conditions are met:
/gate/actor/LA_A/global/filterParticleName gamma
/gate/actor/LA_A/global/filterProcessName Compton
```

1.2 How it works?

Global Actor can collect data from many volumes thank to Local Actors. Each one LA is a GA node which is attached to single volume. When particle go trough volume which is attached to LA the GA receive information about this and starts filtering and saving data for this event. If volume is not attached to LA then GA does not see this volume and events in it. You can control what to save by using commands starting with ”enable” and put conditions with commands begging with ”filter”.

1.3 When use GA and LA?

Use GA and LA always when you want to collect the same types of information from many volumes and you are aiming to reduce size of your output file.

1.4 Information from the author

Global Actor is still in development, so you may not be able to choose some of the options that may be useful from your perspective. If it is so then do not hesitate to write an email to me with a request to implement the new options.

CHAPTER 2

DATA ACQUISITION CONTROL *ENABLE*-COMMANDS

Global Actor can control which values can be saved do output ROOT file by commands starting with the phrase "**enable**". By default GA does not save any values and your task is always to choose what save - thank to this you can control size of output file and prevent saving unnecessary data.

2.1 Example

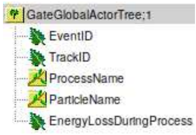
Let's assume that we have two volumes A and B - both attached to GA by Local Actors - as below:

```
# Attaching to volume A with name "volA"
/gate/actor/addActor LocalActor LA_A
/gate/actor/LA_A/attachTo volA

# Attaching to volume B with name "volB"
/gate/actor/addActor LocalActor LA_B
/gate/actor/LA_B/attachTo volB

#Output file name
/gate/actor/LA_A/global/saveTo example.root
```

In this example we want to save information about:

- eventID
- trackID
- particle name
- physics process name
- energy deposition during process

In this case the macro code looks as below (on the right you can see structure of output ROOT file for this macro):




```
/gate/actor/LA_A/global/enableEventID
/gate/actor/LA_A/global/enableTrackID
/gate/actor/LA_A/global/enableParticleName
/gate/actor/LA_A/global/enableProcessName
/gate/actor/LA_A/global/enableEnergyLossDuringProcess
```

2.2 List of commands

Command	Description
`enableVolumeName`	*Save volume (layer) name*
`enableScintilatorPosition`	*Save scintilator's center position*
`enableEventID`	*Save event ID*
`enableTrackID`	*Save track ID*
`enableEnergyBeforeProcess`	*Save particle energy (in keV) before process*
`enableEnergyAfterProcess`	*Save particle energy (in keV) after process*
`enableEnergyLossDuringProcess`	*Save particle energy different between before and after process*
`enableMomentumDirectionBeforeProcess`	*Save particle momentum direction before process*
`enableMomentumDirectionAfterProcess`	*Save particle momentum direction after process*
`enableProcessPosition`	*Save process name*
`enableEmissionPointFromSource`	*Save information about emission point from source*
`enableEmissionMomentumDirectionFromSource`	*Save particle momentum direction from source*
`enableEmissionEnergyFromSource`	*Save particle energy (in keV) from source*
`enableParticleName`	*Save particle name*
`enableParticlePGDCoding`	*Save particle PDG code*
`enableProcessAngle`	*Save angle between particle momentum direction before and after process*
`enablePolarizationBeforeProcess`	*Save particle polarization before process*
`enablePolarizationAfterProcess`	*Save particle polarization after process*
`enableProcessName`	*Save process name*
`enableParentID`	*Save parent ID*
`enableInteractionTime`	*Save interaction time (time of step)*
`enableLocalTime`	*Save time since the track was created*
`enableGlobalTime`	*Save time since the event was created*
`enableProperTime`	*Save time in its rest frame since the track was created*

Table 2.1 List of commands to control saving data.

CHAPTER 3

DATA FILTERING
FILTER-COMMANDS

The second property of GA is the ability to filter data during simulation. Using the GA commands starting with the phrase "**filter**" allows you to impose conditions on saving data.

To better understand the concepts of filtering, let's look at some examples with the macro code.

3.1 Example A

Let's assume that we have two volumes A and B - both attached to GA by Local Actors - as below:

```
# Attaching to volume A with name "volA"
/gate/actor/addActor LocalActor LA_A
/gate/actor/LA_A/attachTo volA

# Attaching to volume B with name "volB"
/gate/actor/addActor LocalActor LA_B
/gate/actor/LA_B/attachTo volB

#Output file name
/gate/actor/LA_A/global/saveTo example.root
```

In this example we want to save data only for specific kinds of particles: electrons,positrons and gammas. As you know from previous chapter, we need to call GA commands only once by local actor (it does not matter which one you take) and it works for all observed volumes by GA. In this case the macro code looks as below:

```
/gate/actor/LA_A/global/filterProcessName e-
/gate/actor/LA_A/global/filterProcessName e+
/gate/actor/LA_A/global/filterProcessName gamma
```

in this example it means:

Save only that data from volume A or B, when electron, positron or gamma goes through it.

3.2 Example B

Let's use the same volumes and local actors as in example A. In this example we want to save data only when:

- particle is gamma
- physics process is Compton scattering
- gamma is emitted from source placed at position (1,2,3) cm

In this case macro has a follow structure:

```
/gate/actor/LA_A/global/filterParticleName gamma
/gate/actor/LA_A/global/filterProcessName Compton
/gate/actor/LA_A/global/filterEmissionPoint 1 2 3 cm
```

3.3 Types of filtering commands

Filtering commands can be divided into two types:

- that can be called only once - the next time they are called, the last filter value will be overwritten with a new one.
- that can be called any number of times - each invocation of a command adds the next value to the list of conditions.

3.4 List of commands

Command	Description
`filterEmissionPoint`	*Save data only form particles emitted from this point*
`filterProcessAngle`	*Save data only for specific Compton angle*

Table 3.1 List of commands that can be called only once without overwriting its last state.

Command	Description
`filterProcessName`	*Save data only for process with this name*
`filterIgnoreProcessName`	*Save data only for process without this name*
`filterParticleName`	*Save data only for particle with this name*
`filterParticlePDGCode`	*Save data only for particle with this PDG code*

Table 3.2 List of commands that can be called any number of times without overwriting its last state.