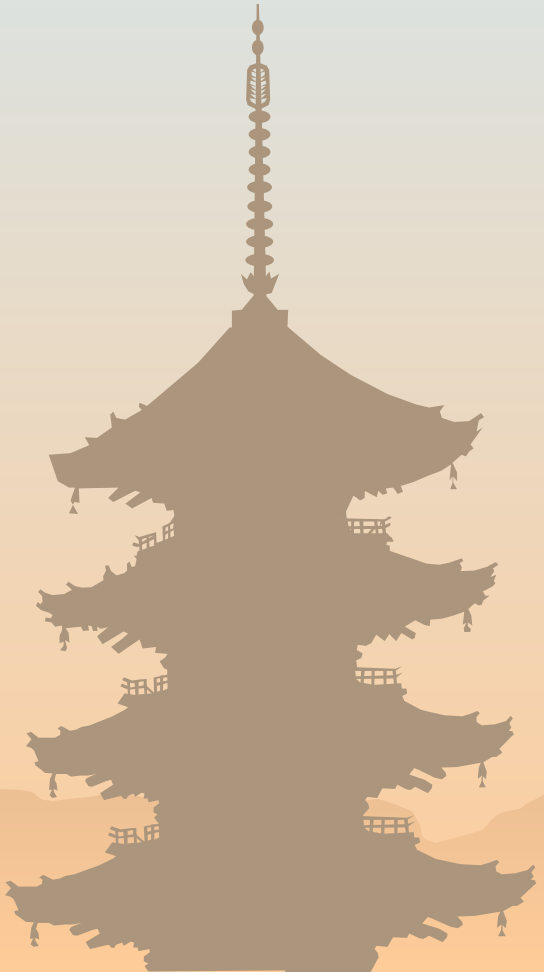
Stack management, Digitization and Pile-up

Makoto Asai (SLAC)

Geant4 Users Workshop @ CERN

Nov. 13th, 2002



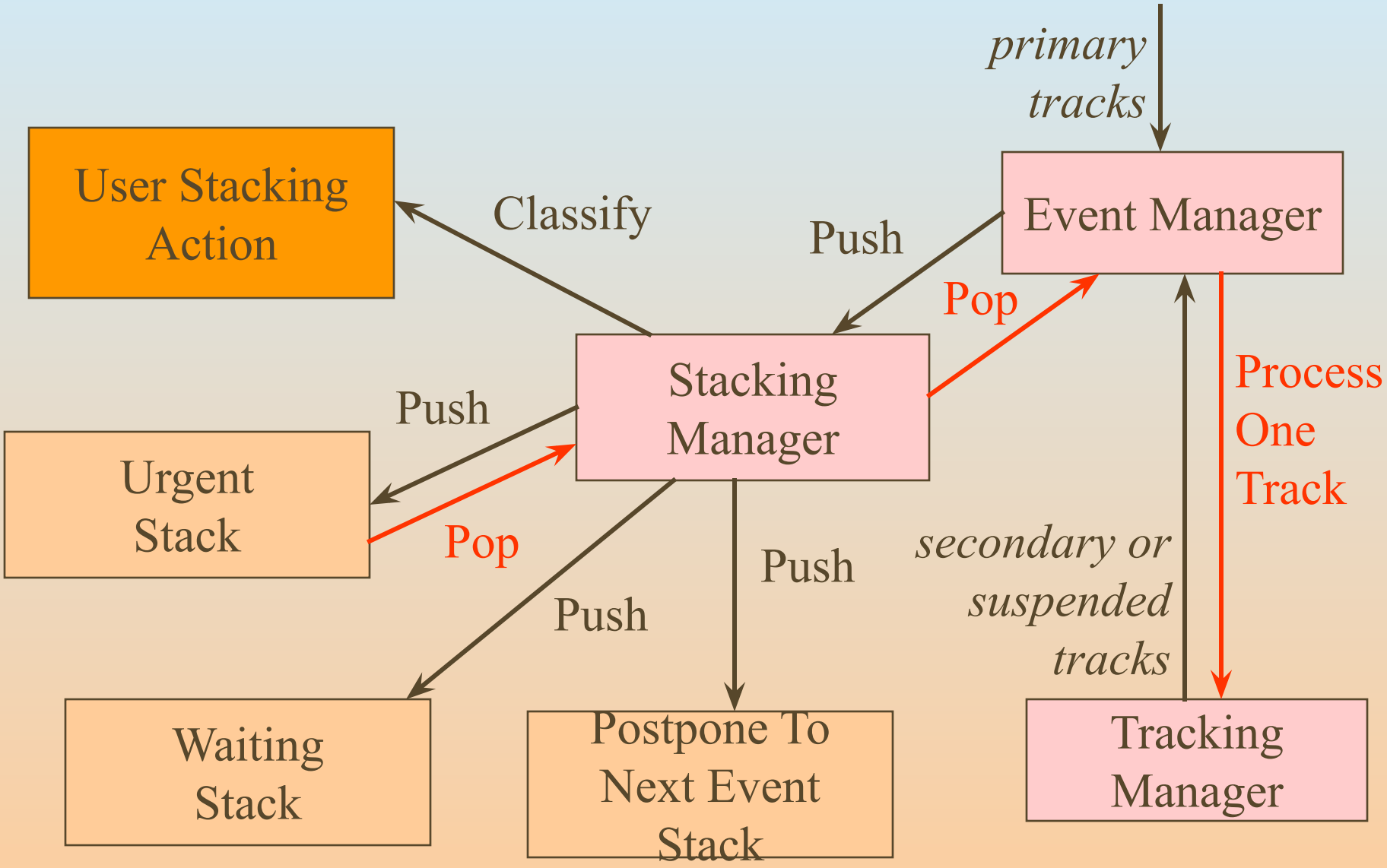
Introduction

- This tutorial covers the features of
  - Stack management
  - Digitization
  - Multiple events handling and pile-up
- These features are not necessary for naïve simulation. But by utilizing them, you can make your simulation program more flexible and efficient.

Stack management

- By default, Geant4 has three track stacks.
  - "Urgent", "Waiting", "PostponeToNextEvent"
  - Each stack is a simple Last-in-First-out stack.
  - Tracks are popped out only from Urgent stack.
  - Once Urgent stack becomes empty, tracks in Waiting stack are moved to Urgent stack.
  - UserStackingAction class classifies each track which stack to be pushed in (or killed).
  - By utilizing stacks, you can manage priorities of the tracks without achieving "highest priority scan".
  - You can add stacks for more intelligent stack management.
- Proper stack management and selection/abortion of event gives you better performance of your simulation.

Stack management



G4UserStackingAction

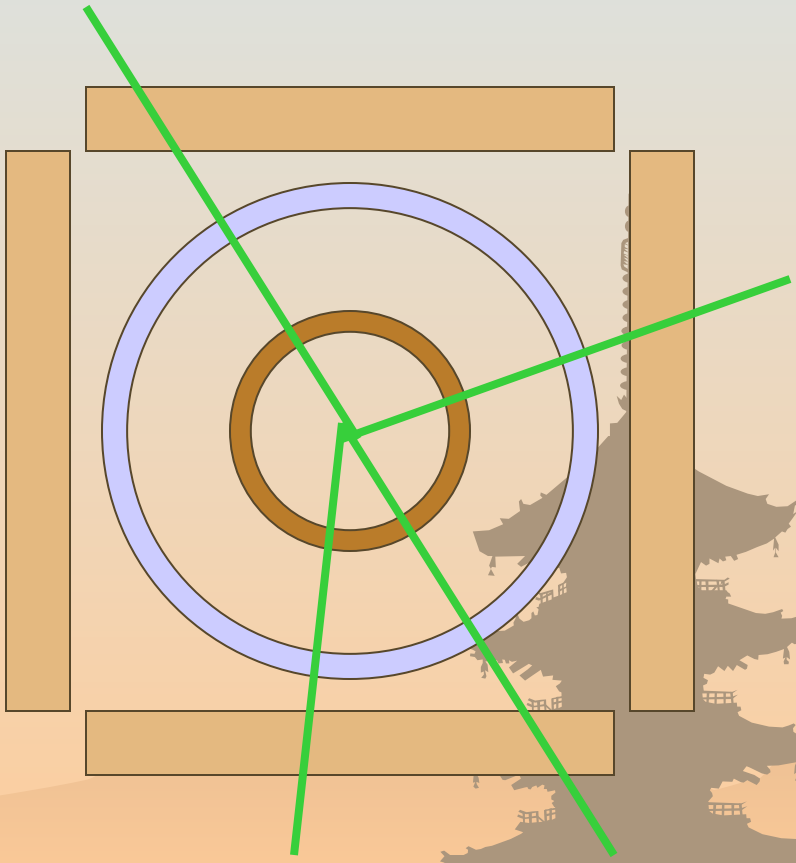
- G4UserStackingAction class has three methods.
- **`void NewStage()`**
  - Invoked when Urgent stack becomes empty
  - Invoked after tracks in Waiting stack had moved to Urgent stack
  - You can issue `stackManager->ReClassify()` thus all tracks in Urgent stack will be re-examined by `ClassifyNewTrack` method
- **`G4ClassificationOfNewTrack`**
  `ClassifyNewTrack(const G4Track*)`
  - Invoked every time a new track is pushed in
  - Classification
    - fUrgent // pushed into Urgent stack
    - fWaiting // pushed into Waiting stack
    - fPostpone // pushed into PostponeToNextEevnt
    - fKill // killed
- **`void PrepareNewEvent()`**
  - Invoked at the beginning of new event

G4UserStackingAction

ExampleN04 gives you a good example of stack management.

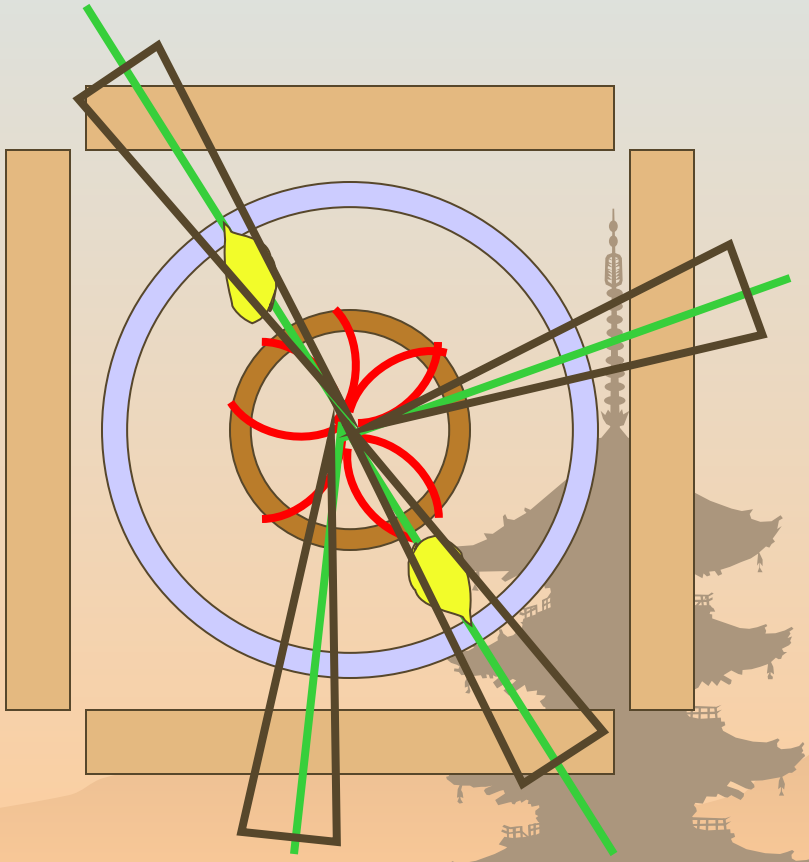
- This example has a simplified collider detector geometry and event sample of Higgs particle decays into muons via Zs.

- At the beginning of an event, only primary muons are pushed into Urgent stack and others to Waiting stack.
- Hits in muon counters are examined when Urgent stack becomes empty
  - Event is aborted if reasonable number of hits are NOT found in the muon counters, otherwise proceed to next stage.



G4UserStackingAction

- Only primary charged tracks are pushed into Urgent stack
  - Each track is traced in the tracking region and then suspended and pushed back to Waiting Stack once it reaches to calorimeter.
- Hits in tracking chambers are examined
  - Event is aborted if reasonable numbers of isolated muons are NOT found.
- Only tracks (both primary and secondary) in “Region of Interest” are pushed into Urgent stack and traced.
  - I.e. Shower is simulated only inside of *RoI*.



Digitization

- Digit represents a detector output (e.g. ADC/TDC count, trigger signal).
- Digit is created with one or more hits and/or other digits by a concrete implementation derived from G4VDigitizerModule.
- In contradiction to the Hit which is generated at tracking time automatically, the digitize() method of each G4VDigitizerModule must be explicitly invoked by the user's code (typically at EndOfEventAction).

Digitization

- The usages of methods in G4VDigitizerModule is quite similar to those of G4VSensitiveDetector.
  - Constructor
    - Registration of digits collection name(s)
  - Digitize() method
    - Get hits from proper hits collections and/or digits from proper digits collections
    - Generate digit(s) and store to dedicated digits collection(s)
    - Set created digits collection(s) to G4DCofThisEvent

Multi-event treatment and pile-up

- By design, G4EventManager **cannot** handle more than one events.
  - UserEventAction neither
- Only at the "Run" level, you can handle more than one events.
  - Create your own RunManager deriving from G4RunManager.
  - Implement your own AnalyzeEvent() method. Here you have access to more than one G4Event objects.
    - GetPreviousEvent(G4int) method
    - Don't forget to invoke SetNumberOfEventsToBeStored() method before starting an event loop.
  - You can invoke your Digitizer to generate digits / digits-collection.