FAYE: A Java Implement of the Frame/Stream/Stop Analysis Model.

Simon Patton
LBNL
Overview

• Review Frame/Stream/Stop model.
• FAYE Implementation.
  • Generic Portions.
    - freehep.
    - FAYE.
  • Experiment Specializations.
    - IceCube.
Frame/Stream/Stop Model

• Analysis uses an “electronic picture” of the experiment.

• Data between pictures change at different rates, e.g. geometry, HV, events.

• Related data, which all change at the same time, are grouped into *Records*.

• A *Frame* is a set of Records, of different types, related to the same time.

• A *Stream* is a set of Records, of the same type, from different times.
Building Frames from Streams

• Detector exist, HV off.
Building Frames from Streams

- HV on.

Geometry (t=0) → Frame (t=1)
Building Frames from Streams

• First Event read out.
Building Frames from Streams

- $n^{\text{th}}$ Event read out.
Specifying Which Frames are Supplied

• Specify Streams of interest as **Stops**.

• **Active Stops:**
  • Sequential (not nec. ordered) Stream.

• **Passive Stops:**
  • Response to (and precede) Active Stops.

• **Event Display Example:**
  - “Events” from a sequential source are Active Stops.
  - “Geometry” from a DB are Passive Stops, supplied whenever geometry changes.
**Active & Passive Stops**

• **Active Event Stops.**

- Geometry $(t=0)$
- HV $(t=0)$
- HV $(t=1)$
- Event $(t=2)$
- Event $(t=50)$

• **Preceding Passive Geometry Stop.**

- Geometry $(t=0)$
- HV $(t=0)$
- HV $(t=1)$
- Event $(t=2)$
Implementing Frame/Stream/Stop

• Three layers separate layers.
• Generic Record Loop.
  - Uses Java source/listener pattern.
  - RecordLoop is source, algorithm is RecordListener.
• FAYE (Frame AnalYsis Executable) layer.
  - Handles logic of supplying the Frame.
• Experiment Layer.
  - Defines experiment’s streams
  - Distributes Frames to analysis methods.
RecordListener Lifecycle
RecordListener Lifecycle

- Dormant
- Configured

configure
RecordListener Lifecycle

- Dormant
  - configure

- Configured

- Processing
  - recordSupplied
RecordListener Lifecycle

- Dormant
- Configured
- Suspended
- Processing

Events:
- configure
- recordSupplied
- suspend
RecordListener Lifecycle
RecordListener Lifecycle

- Dormant
  - configure
- Configured
  - recordSupplied
  - reconfigure
- Suspended
  - resume
  - suspend
- Processing
  - recordSupplied
RecordListener Lifecycle

- Dormant
  - configure
- Configured
  - reconfigure
  - resume
  - suspend
- Suspended
  - reconfigure
  - resume
  - suspend
- Processing
  - recordSupplied
  - recordSupplied
RecordListener Lifecycle

- **Dormant**
  - configure
  - finish

- **Configured**
  - suspend
  - reconfigure

- **Suspected**
  - resume

- **Processing**
  - recordSupplied

- **Active**
public interface RecordListener

    extends EventListener

{

    public void configure(ConfigurationEvent event);
    public void finish(RecordEvent event);
    public void recordSupplied(RecordSuppliedEvent event);
    public void reconfigure(ConfigurationEvent event);
    public void resume(RecordEvent event);
    public void suspend(RecordEvent event);

}
org.freehep.record Packages

• Provides classes to create composite RecordListener objects.
  • Sequences.
  • Branches.
  • Conditional execution.

• Defines the interfaces for sequential and interactive sources of record objects.
• **FayeSource**
  - Implementation of record source.
  - Contains FayeStopSource objects which are used to determine the next Stop to supply.
  - Uses FrameFactory to create new Frame for that Stop.
  - Return this Frame as Record to the loop.

• **FayeStopSource**
  - Supplies active Stop objects to FayeSource.
  - Supplies passive Stop objects based on active Stop.
  - Also a RecordListener (see below).
• **FayeListener**
  - Two phase composite `RecordListener`.
  - Phase one supplies new Frame to all `FayeStopSource` objects.
  - Phase two supplies filled Frame to a analysis `RecordListener` (can be composite).

• **FrameFactory**
  - Manages creation and lookup of `Frame` objects.
Experiment (IceCube) Layer

```java
class IceCubeSupport

    recordSupplied(RecordSuppliedEvent event);

class IceCubePlugin

    recordSupplied(RecordSuppliedEvent event);

    support.recordSupplied(event);

Frame frame =
    factory.lookupFrame(event.getRecord());
Stream stream = frame.getStopStream();
if (stream.equals(IceCubeStream.GEOMETRY)) {
    listener.geometry(frame);
} else if (stream.equals(IceCubeStream.HV)) {
    listener.hv(frame);
} else if (stream.equals(IceCubeStream.EVENT)) {
    listener.event(frame);
}
```
Summary

• The Frame/Stream/Stop model provides a flexible framework in which to develop analyses.

• The Java implementation of this (FAYE) is based on a freehep foundation so it can be easily used elsewhere, e.g. JAS3.

• Experiment specialization can be done by providing around half a dozen simple classes.