Framework Structure

- Framework is an event loop. Events are delivered by input sources, processed by workers, and consumed by outputs.
- There is a master executable used to run framework jobs, cmsrun, artrun, etc.
- The master executable takes as a required parameter the job configuration file name which describes the job to be run.
- CMS and Art have different job configuration languages, but the result of processing the job configuration is a ParameterSet. The parameter set is queried by the framework subsystems for control and configuration information.
- A framework job is assembled from the configuration by dynamically loading a set of software components that provide the behavior required by the job.
- Components are implemented as a shareable libraries, possibly linked against experiment-specific and external product libraries.

Software Components

- Kinds
  - Services. Setup at start of job, exist for life of job, accessible by all components, provide behavior likely to be needed by all parts of the job, including each other.
  - Workers. Perform sub-tasks.
- Workers
  - Two categories of worker exist, and different kinds of worker per category.
  - Trigger path workers.
    - Event source. Retrieve events from an arbitrary source, frequently a root data file.
    - Event producer. Create events themselves instead of retrieving them from somewhere, frequently an event simulator of some kind.
    - Event filter. Pass or Reject events for further processing. Result is recorded in the trigger results object for later use.
  - End path workers.
    - Event analyzer. Processes event data in some way and possibly adds further data products to the event.
    - Event output. Uses the final form of the event after filtering and analysis to produce output of some kind. Frequently a new root data file, but may be an event display, or summary statistics.

Services

- Common
  - Floating point control
  - Timing
  - Tracer

- CMS specific
  - CPU
  - InitRootHandlers
  - JobReport
  - LoadAllDictionaries
  - Memory
  - PrintEventSetupDataRetrieval
  - PrintLoadingPlugins
  - ProcInfoFetcher
  - ProfParse
  - SiteLocalConfig
  - Sym
  - UnixSignal
  - VertexTracker

- Art specific
  - FileCatalogMetadata
  - TriggerNames
  - CurrentModule
  - TfileService
  - TrivialFileDelivery
  - TrivialFileTransfer
  - RandomNumberGenerator
  - SimpleMemoryCheck
  - SimpleInteraction

Master Executable

- Reads configuration file.
- Starts configured services.
- Creates and initializes configured workers. Determines order workers will be run in.
- Runs the event loop.
- Performs job termination duties.
- Error handling.

Extension Mechanism and Configurable Behavior

- Callbacks on state transitions
- SelectEvents
- Exception Handling

Data Model

- Dumb Data
- Intra-Event pointers by unique product id
- MetaData
  - Process history
  - Provenance (parentage)
- Primary data file format is root-based, metadata is stored with data.

Major Subsystems

- State machine
- Plugin manager
- Service manager
- Path and Schedule
- Input Sources
- Output Modules
- Message Logging
- Callout Manager

Problems to be addressed

- On-disk data format compatibility.
- ParameterSet compatibility.
- Metadata compatibility.
- Product id compatibility.
- Plugin systems have major differences. Demands on build system.
- How to make framework an external library?
- Does a common framework have a build system? A software distribution system? How would you install it? How do you use it with your build system/distribution system?