CMS experience in turning ROOT error messages into exceptions

Christopher Jones, Matti Kortelainen, Dan Riley
HEP-CCE SOP
16 October 2024

Turning ROOT error messages into exceptions

- CMSSW uses exceptions for error handling
  - Usual reaction is to terminate the application in a graceful way
    - At higher level (WM, comp ops) actions are based on the error types
- CMSSW has a service, InitRootHandlers, that registers a custom message handler for ROOT by calling SetErrorHandler()
- CMSSW implementation of the message handler (link)
  - Maps the ROOT messages to CMSSW messages according to the ROOT message severity
  - Severity of a message can be upgraded (in practice to Fatal) based on the location and content
  - Severity of a message can be downgraded (in practice to Info) based on the content of the message, or on the location where the message was issued
  - All messages above Info (i.e. Fatal, SysError, Error, Warning) are thrown as exceptions
    - Unless the message corresponds to a pending signal, in which case the message is printed

Challenges with this approach

- We effectively assume ROOT code is not exception safe
  - Not that much of CMSSW code would really be exception safe either…
- We have to base some decisions on severity on the message content
  - Based on our experience and limited understanding on ROOT internals
    - “Message A is classified as Info by ROOT, but we want to terminate CMSSW in that case”
    - “Message B is classified as Error by ROOT, but we do not want to terminate CMSSW in that case”
  - Brittle: we have no guarantees on the stability of e.g. formatting of these error messages
- Coarse grained
  - Framework gets to know there was a (fatal) error in ROOT, but not much of the cause
    - Without parsing the message content

(Non-exhaustive) Wishlist

- We would like to clearly distinguish e.g.
  - File open errors
  - File read errors
    - Possibly finer grained, e.g. ROOT metadata is corrupt, error in user data decompression, …
- Distinction could be made with multiple exception types, or with one (or few) type(s) and error codes stored in the exception object
  - Up to date documentation of the situations corresponding to each exception type / code would be helpful
-