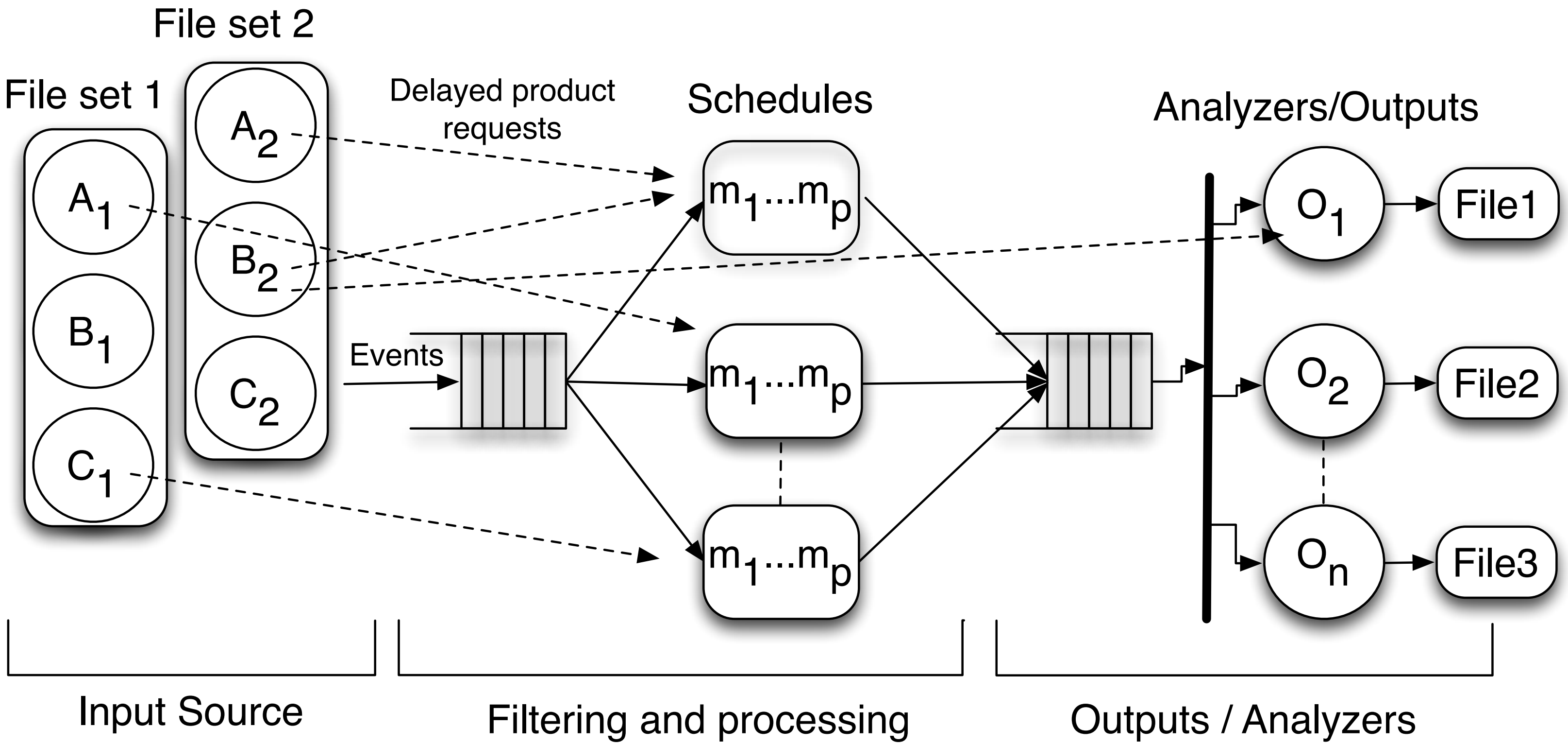
I/O desires from the art team

Jim Kowalkowski

(Presented by Philippe Canal)

Architecture for art

- Input
  - Multiple sets of files open at the same time
  - The data products for a single event are spread across the files in the set
  - The sets can all come from the same “run”
- Output
  - Each output is independent of the others
  - Closes and opens do not necessarily align with open and close of input files



Simultaneous and efficient access

- From multiple threads, reading
  - Same/different leaf, same/different branch, same/different tree, same/different file
  - Permitted with fast cloning active of branches or trees within same file
- Use of on-demand loading of data products during filtering, processing, and analysis stages
- Resources for carrying out I/O operations under our control
  - No fixed pool of dedicated cores or threads for ROOT to own

We would like to see …

- Published data format (not just an API for data access)
- Option to use native byte ordering for binary data and immunity to compiler alignment and padding
- Support for C++11 and modern C++ (14 and beyond)
  - Take advantage of move constructs
  - Underlying ROOT standard containers more STL-like
  - Dictionaries for new containers and template constructs

Other useful things …

- Support for all Reflex operations within TClass system, including
  - Additions from C++11
  - Operations for querying and using template parameters
- Portability of serialized data format across upcoming platforms such as ARM and PowerPC
- Writes to single output file from multiple processes

Histograms

- Desire lightweight histogram object
  - To be used for data collection purposes only
  - Guaranteed thread safety: no use of global state
  - Mapped into bigger THxx objects
  - Support reduction operations for building THxx objects