The Kanga Event Store for BaBar

Matthias Steinke
Ruhr Universität Bochum

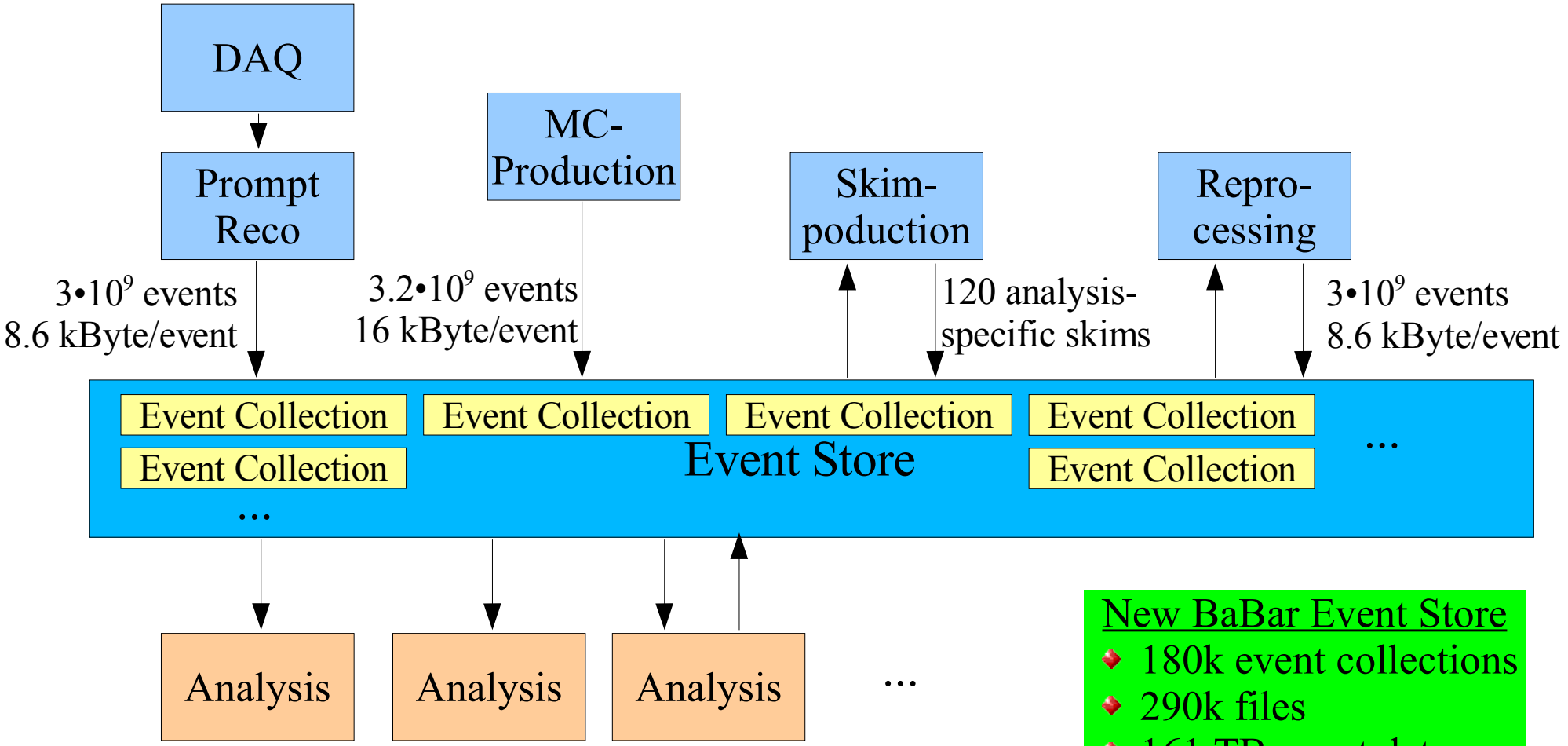
for the BaBar Computing Group

CHEP 2004, Interlaken

Outline

- Requirements for an Event Store for BaBar
- BaBar Event Store History
- Computing Model 2 Event Store
  - Event Structure
  - Trees, Files and Directories
  - File Access Methods
  - USR Data and Interactive Data Access
- Current Status of the new BaBar Event Store

Requirements



New BaBar Event Store
- 180k event collections
- 290k files
- 161 TB event data

Analysis jobs at 5 Tier A sites and at university sites
@SLAC: typically 2000 analysis jobs running in parallel

BaBar Event Store History

Original BaBar event store was based on Objectivity OODBMS

- Objectivity is a database technology and was not designed as an event store
  - transactions not needed for readonly access in physics analysis
  - poor control of event data ↔ file association
    - "level of detail" navigation effectively impossible for staged data
  - latency for new data becoming available for analysis too large
  - data size overhead too large
- Data export to other sites was difficult and resource intensive
- Objectivity is commercial software
  - release policy blocks migration to new compiler and OS versions
- High costs of administration and development
- Many users were not happy with the overall performance

BaBar Event Store History

- Temporary solution: ROOT I/O based event store for “Micro” data
  - was developed in a short time
  - fast access to Micro data
  - no navigation to more detailed event components
  - easy to export
  - very popular for Micro based analysis
  - not usable for reconstruction and MC production
- Running two event stores in parallel can not be a long term solution

Kanga Event Store

BaBar Computing Model 2

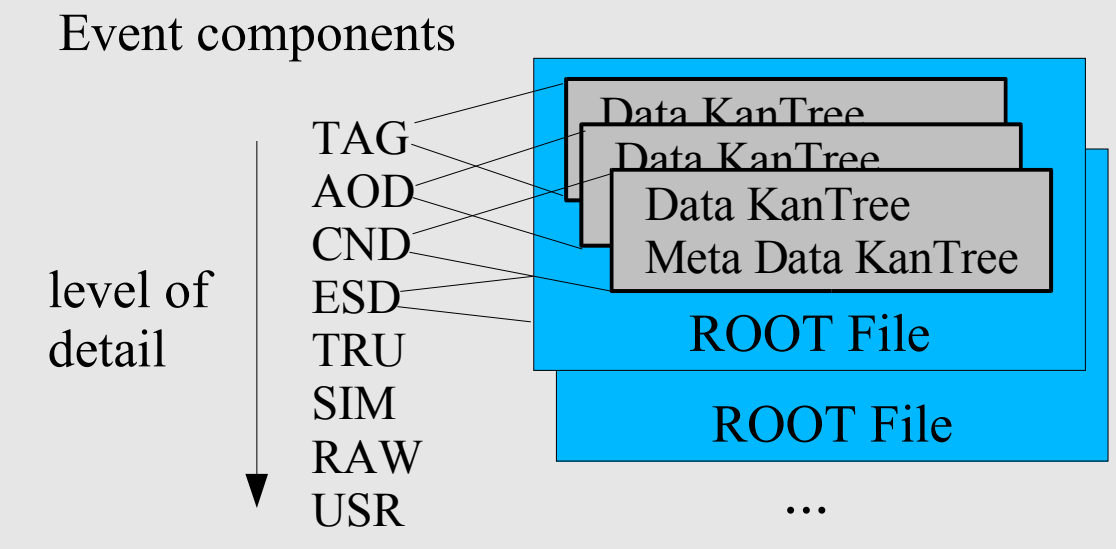
- In 2003 BaBar implemented a new computing and analysis model: *Computing Model 2*
- One key component: a full featured, fast, ROOT I/O based event store: *Kanga*
- New features: Customizable USR data and interactive data access
- Kanga is **not** a GRID project

Kanga Event Store

- What is the event structure?
- How does the file access work?
- Interactive Access

Event Layout

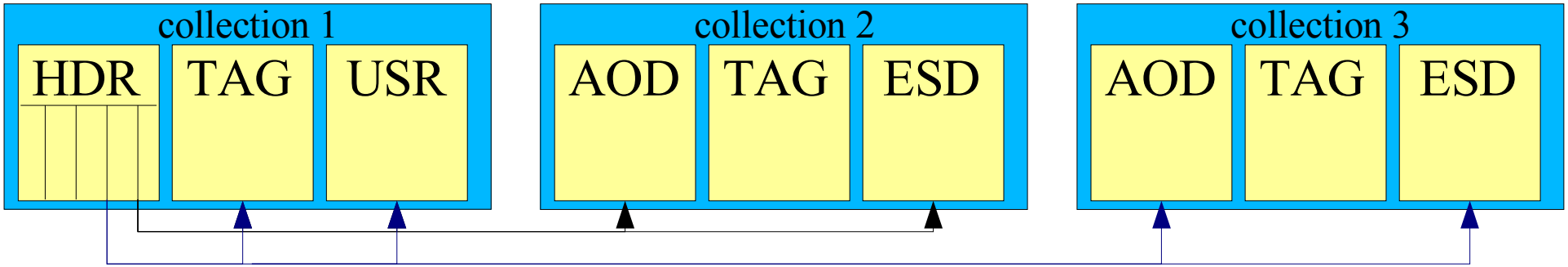
- Events consist of *components* which correspond to different levels of detail of the data
  - a component is implemented as 1 TTree for data and 1 TTree for metadata
- The Event Header is the key part of the event
  - each event header holds a pointer to the event data of each component with the corresponding **L**ogical **F**ile **N**ame and a tree offset
- Object relations are persisted using a custom reference



- Components can be clustered to one or more files
- File size is limited to 2 GByte
  - continuation files are created automatically

Event Header

- Events are organized in *event collections*



- Collections can
  - *own* a component: the collection holds a deep copy of the component data
  - *borrow* a component from another collection: just a pointer to the component data is stored
    - Reprocessing/reskimming: rewrite just TAG, AOD, CND and USR data
- Collections are relocatable: data export is reduced to a simple file copy

File Access

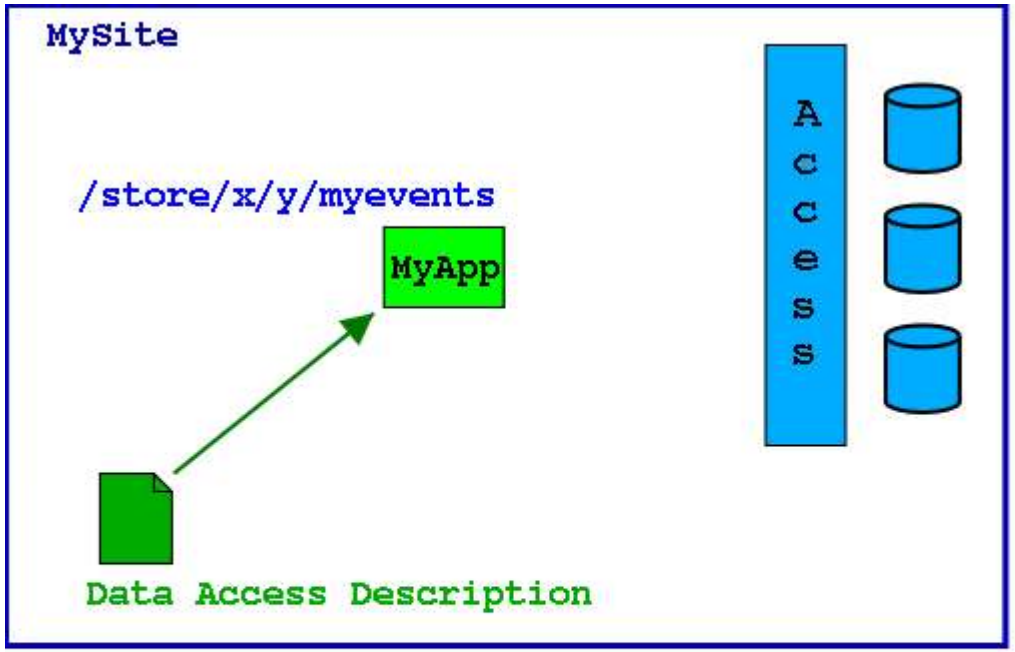
- A Kanga event store consists of ROOT files stored in a directory tree
- The **L**ogical **F**ile **N**ame of the base file is derived from the collection name by ***naming convention***
  - Example: */store/PRskims/R12/14.3.2/B0ToccKFinal/00/B0ToccKFinal_0031*
  - LFN: */store/PRskims/R12/14.3.2/B0ToccKFinal/00/B0ToccKFinal_0031.01.root*
- LFNs are stored in the event header
- **P**hysical **F**ile **N**ames are hidden from the user and get constructed using a simple configuration file (no file catalogue, no database access):

```
read /store/SP/* file /nfs/workdisk/skims/
read /store/PRskims/* xrootd kanolb-a:1094/
```

root://kanolb-a:////*store/PRskims/R12/14.3.2/B0ToccKFinal/00/B0ToccKFinal_0031.01.root*

File Access

- Read and write access can be configured to be
  - direct file access
  - xrootd
- Job reads site specific access description file when it is actually run



- xrootd allows a fault tolerant, load shared event store access
- xrootd and TXNetFile available in latest ROOT version

```
LFN = /store/x/y/myevents.01.root
PFN = root://mysiteaccess//store/x/y/events.01.root
PFN = rfio://castor/zzz/store/x/y/events.01.root
PFN = /mnt/bigdiskarea/store/x/y/events.01.root
etc. etc.
```

Talk of Andrew Hanushevsky, Monday16:30, Theatersaal
Poster [128] by Fabrizio Furano, poster session 2

USR Data and Interactive Access

USR Data

- special component to hold user specified data
- USR data can be associated to an event or to a particle candidate
- talk of David Brown, Thursday 15:40, Kongress-Saal

Interactive Access to Event Store Data

- BaBar model separates transient and persistent data
- BaBar analysis runs on transient data. ROOT analysis runs on persistent data
- Basic design: Cache transient representations of data
- Add functionality to persistent classes for unpacking and access to transient objects
- Replace n-tuple based analysis and avoid data duplication

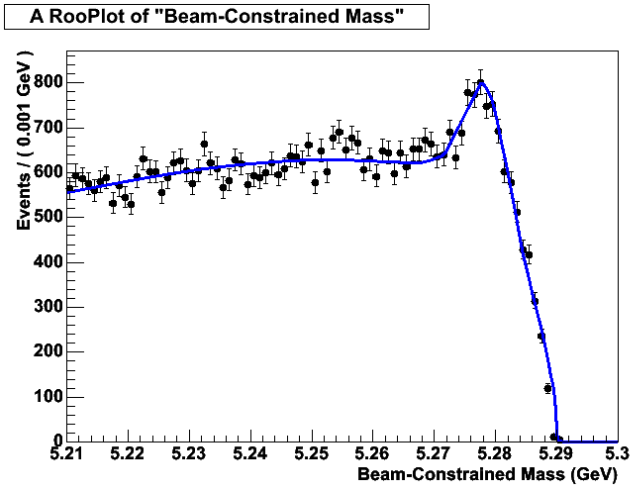
Interactive Example Session

root[0] gSystem->Load("libKanga.so");
This load all the symbols to use the Kanga data.

root[1] KanEventSource* input = KanEventSource::mini();
This builds an object to read the events (TChain).

root[2] input->Add("/a/collection/of/events");
root[3] input->Add("/an/other/collection/of/events");
This opens some collections for input. Collections are accessed the same way as in framework applications.

root[4] input->Draw("Pid_DchPids.dedx()");
This is the simplest possible use, plotting a trivial member function of a stored object. Namely the ionization in the Drift Chamber.



Using Kanga with RooFitCore:

root[5] TH1FmesHist("mes","mes",100,5.2,5.3);

root[6] input->Project("mes","BSemiExcl_mES");

root[7] gSystem->Load("libRooFitBabar_root.so");

root[8] FitBMass(mesHist);

Tools

- Objectivity to Kanga conversion tool
- Event collection merge tool
  - merge output of jobs running in parallel to limit number of files
- Tools to inspect collections
- Collection administration tool
  - remove collections
  - create user areas
- Import tools
  - import data from MC production and prompt reconstruction sites
  - import streams to tier A and and tier C sites for Analysis
- Bookkeeping tools [talk of Douglas Smith, Thursday 17:10, Theatersaal]

Status of the BaBar Kanga Event Store

- All Run1 to Run3 and the last MC production Micro and Mini data were converted to Kanga
- Run4 reconstruction wrote directly to the Kanga event store
- Simulation production uses Kanga event store since Feb. 2004 [talk of Douglas Smith, Wednesday 14:00, Ballsaal]
- Analysis jobs are moving to CM2 event access since January 2004
- BaBar Kanga event store contains
  - 290k files
  - 184k event collections
  - 161 TB data
- Typically 2000 analysis jobs are accessing the Kanga event store at SLAC in parallel

Status of the BaBar Kanga Event Store

First experiences:

- All necessary features available
- Arguably the data latency went down from 1 week to 2 days
- Better data compression, less overhead
- Faster event access than with CM1 event store
  - End of last data taking: After 6 days all data was calibrated, reconstructed, QAed and skimmed.
  - 11 days later the first paper based on the data was submitted to PRL
- More stable
  - no lock server problems
  - no AMS problems
  - no NFS client problems

CM2 Kanga Event Store seems to be a success