LHCb Collaboration

France: Clermont-Ferrand, CPPM Marseille, LAL Orsay
Italy: Bologna, Cagliari, Ferrara, Firenze, Frascati, Genova, Milano, Univ. Roma I (La Sapienza), Univ. Roma II (Tor Vergata)
Netherlands: NIKHEF
Spain: Univ. Barcelona, Univ. Santiago de Compostela
Switzerland: Univ. Lausanne, Univ. Zürich

CERN
Brazil: UFRJ
China: IHEP (Beijing), Tsinghua Univ. (Beijing)
Romania: IFIN-HH Bucharest
Russia: BINR (Novosibirsk), INR, ITEP, Lebedev Inst., IHEP, PNPI (Gatchina)
Requirements on Data Rates and Computing Capacities
LHCb Technical Design Reports

LHCb MAGNET

Submitted: January 2000
Recommended by LHCC: March 2000
Approved by RB: April 2000

LHCb Calorimeters

Submitted: September 2000
Recommended: November 2000

LHCb RICH

Submitted: September 2000
Recommended: November 2000
Defining the architecture

Issues to take into account

- Object persistency
- User interaction
- Data visualization
- Computation
- Scheduling
- Run-time type information
- Plug-and-play facilities
- Networking
- Security
General categorization of systems [2]

- **user-centric**: focus on the direct visualization and manipulation of the objects that define a certain domain.
- **data-centric**: focus upon preserving the integrity of the persistent objects in a system.
- **computation-centric**: focus is on the transformation of objects that are interesting to the system.

Our applications have elements of all three. Which one dominates?
Getting Started

First crucial step was to appoint an architect - ideally skills as:
- OO mentor, domain specialist, leadership, visionary

Started with small design team ~ 6 people, including:
- developers, librarian, use case analyst

Control activities through visibility and self discipline
- meet regularly - in the beginning every day, now once per week

Collect URs and scenarios, use to validate the design

Establish the basic design criteria for the overall architecture
- architectural style, flow of control, specification of interfaces
Development Process

- Incremental approach to development
  - New release every few (~ 4) months
  - Software workshop timed to coincide with new release

- Development cycle is user-driven
  - Users define priority of what goes in the next release
  - Ideally they use what is produced and give rapid feedback
  - Frameworks must do a lot and be easy to use

- Strategic decisions taken following thorough review (~1 /year)

- Releases accompanied by complete documentation
  - Presentations, tutorials
  - URD, reference documents, user guides, examples
Possible migration strategies

1. **SICb**
   - Fast translation of Fortran into C++

2. **SICb**
   - Wrapping Fortran

3. **SICb**
   - Framework development phase
   - Transition phase
   - Hybrid phase
   - Consolidation phase
How to proceed?

Physics Goal:

To be able to run new tracking pattern recognition algorithms written in C++ in production with standard FORTRAN algorithms in time to produce useful results for the RICH TDR.

Software Goal

To allow software developers to become familiar with GAUDI and to encourage the development of new software algorithms in C++.

Approach

choose strategy 3
start with migration of reconstruction and analysis code
simulation will follow later
New Reconstruction Program - BRUNEL

- Benefits of the approach
  - A unified development and production environment
    - As soon as C++ algorithms are proven to do the right thing, they can be brought into production in the official reconstruction program
  - Early exposure of all developers to Gaudi framework
  - Increasing functionality of OO 'DST'
    - As more and more of the event data become available in Gaudi, it will become more and more attractive to perform analysis with Gaudi
  - A smooth transition to a C++ only reconstruction
Integrated System - databases

Slow Control Database

Readout System Database

The power supply on that VME crate

Detector description
Frontend Electronics

- Data Buffering for Level-0 latency
- Data Buffering for Level-1 latency
- Digitization and Zero Suppression
- Front-end Multiplexing onto Front-end links
- Push of data to next higher stage of the readout (DAQ)
Timing and Fast Control

- Provide common and synchronous clock to all components needing it
- Provide Level-0 and Level-1 trigger decisions
- Provide commands synchronous in all components (Resets)
- Provide Trigger hold-off capabilities in case buffers are getting full
- Provide support for partitioning (Switches, ORs)
IBM NP4GS3

Features

- 4 x 1Gb full duplex Ethernet MACs
- 16 special purpose RISC processors @ 133 MHz with 2 hw threads each
- 4 processor (8 threads) share 3 co-processors for special functions
  - Tree search
  - Memory move
  - Etc.
- Integrated 133 MHz Power PC processor
- Up-to 64 MB external RAM
Event Building Network Simulation

- Simulated technology: Myrinet
- Nominal 1.28 Gb/s
- Xon/Xoff flow control
- Switches:
  - ideal cross-bar
  - 8x8 maximum size (currently)
  - wormhole routing
  - source routing
  - No buffering inside switches
- Software used: Ptolemy discrete event framework
- Realistic traffic patterns
  - variable event sizes
  - event building traffic
**Event Building Activities**

- **Studied Myrinet**
  - Tested NIC event-building
  - Simulated switching fabric of the size suitable for LHCb
  - Results show that switching network could be implemented (provided buffers are added between levels of switches)

- **Currently focusing on xGb Ethernet**
  - Studying smart NICs (→ Niko’s talk)
  - Possible switch configuration for LHCb with ~today’s technology (to be simulated...)

Multiple Paths between sources and destinations!
Network Simulation Results

Results don’t depend strongly on specific technology (Myrinet), but rather on characteristics (flow control, buffering, internal speed, etc)

FIFO buffers between switching levels allow to recover scalability 50 % efficiency “Law of nature” for these characteristics
Alteon Tigon 2

Features
- Dual R4000-class processor running at 88 MHz
- Up to 2 MB memory
- GigE MAC+link-level interface
- PCI interface

Development environment
- GNU C cross compiler with few special features to support the hardware
- Source-level remote debugger
Common integrated controls system

- Detector controls
  - High voltage
  - Low voltage
  - Crates
  - Alarm generation and handling
  - etc.

- DAQ controls
  - RUN control
  - Setup and configuration of all components (FE, Trigger, DAQ, CPU Farm, Trigger algorithms, ...)
  - Consequent and rigorous separation of controls and DAQ path

Same system for both functions!

Scale: ~100-200 Control PCs
many 100s of Credit-Card PCs

By itself sizeable Network!
Most likely Ethernet