The LHCb Way of Computing

The approach to its organisation and development

John Harvey
CERN/ LHCb

DESY Seminar

Jan 15th, 2001

Talk Outline

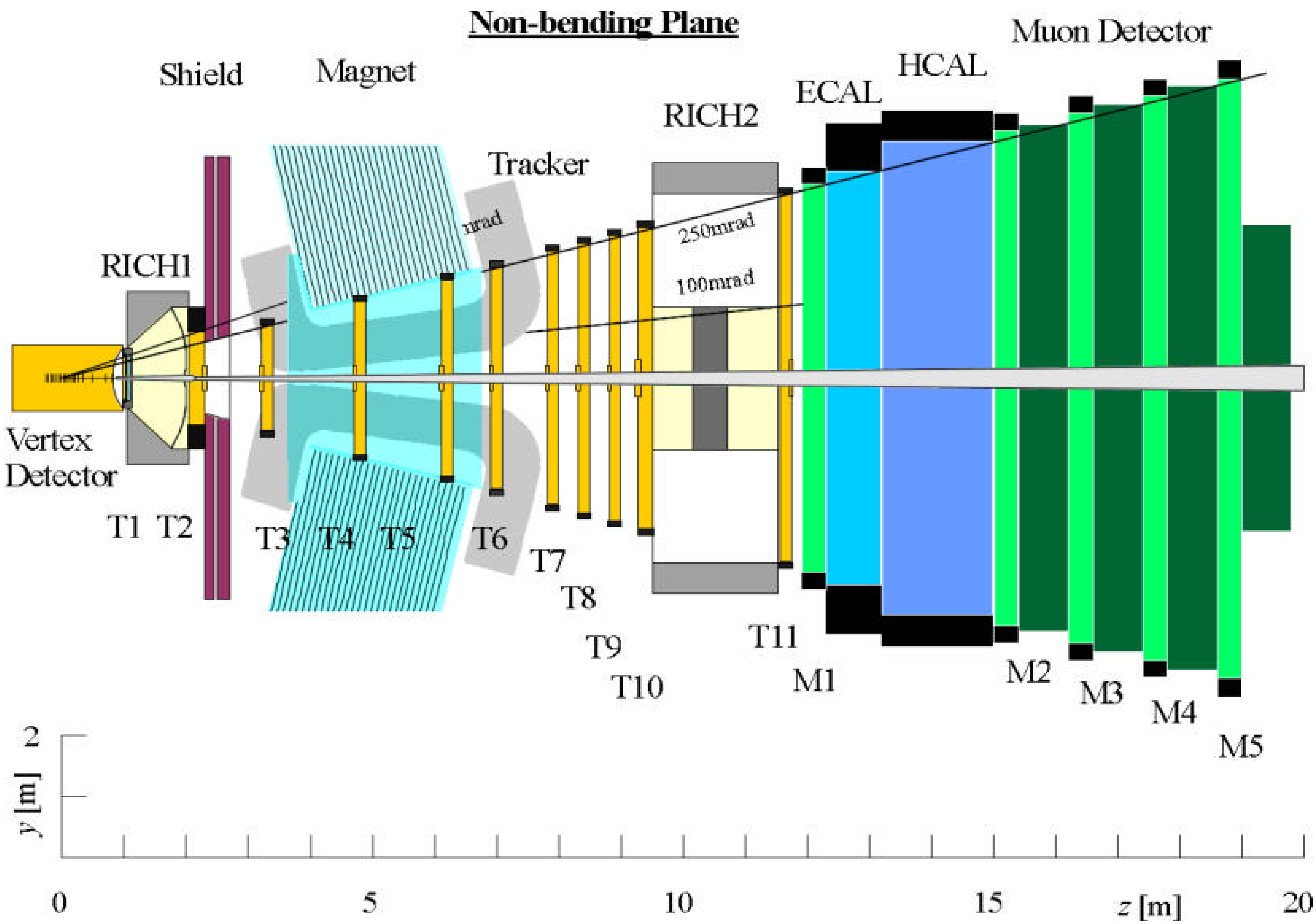
- Brief introduction to the LHCb experiment
  - Requirements on data rates and cpu capacities
- Scope and organisation of the LHCb Computing Project
  - Importance of reuse and a unified approach
- Data processing software
  - Importance of architecture-driven development and software frameworks
- DAQ system
  - Simplicity and maintainability of the architecture
  - Importance of industrial solutions
- Experimental Control System
  - Unified approach to controls
  - Use of commercial software
- Summary

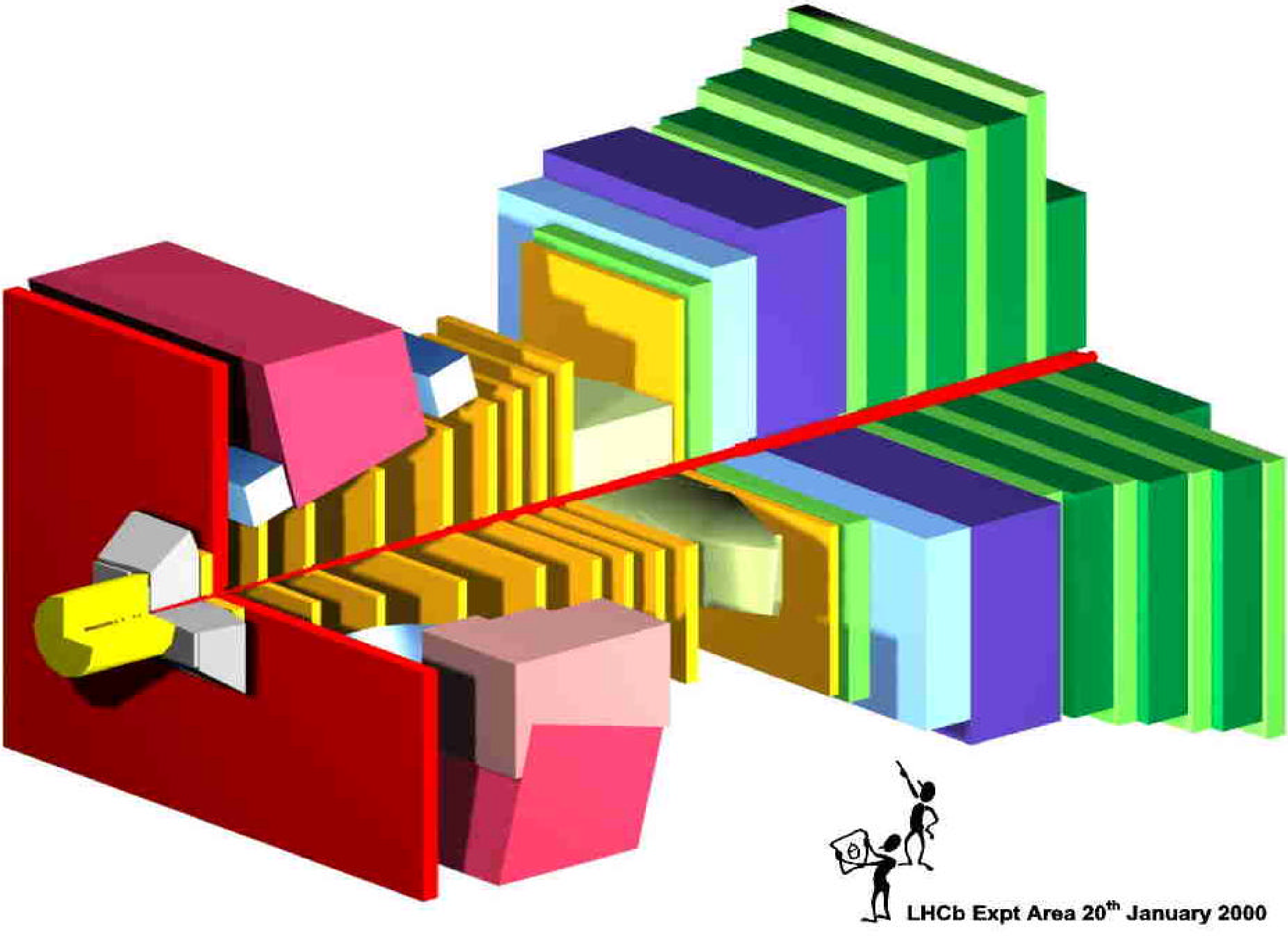
Overview of LHCb Experiment

The LHCb Experiment

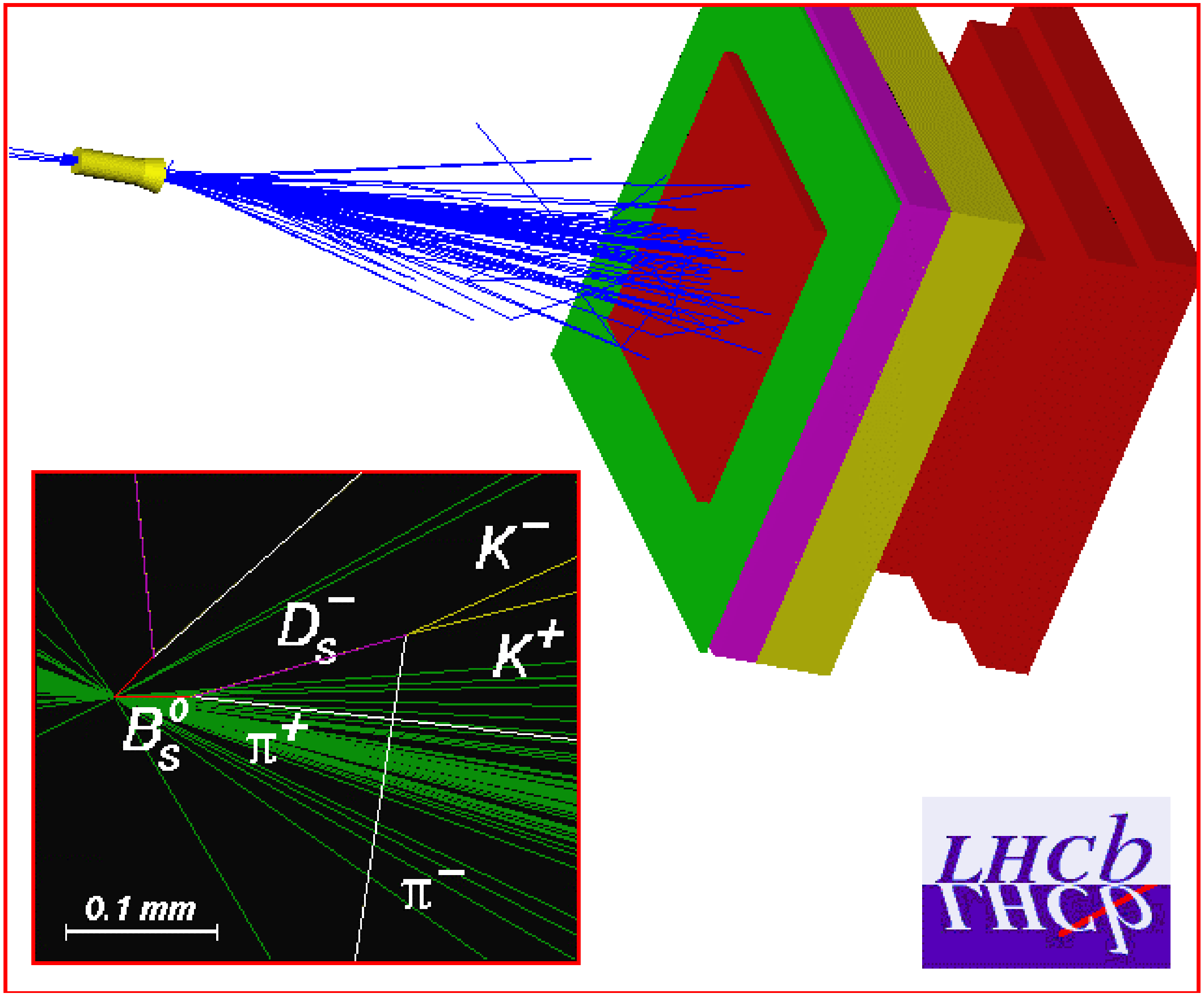
- Special purpose experiment to measure precisely CP asymmetries and rare decays in B-meson systems
- Operating at the most intensive source of B_u, B_d, B_s and B_c, i.e. the LHC at CERN
- LHCb plans to run with an average luminosity of $2 \times 10^{32} cm^{-2} s^{-1}$
  - Events dominated by single pp interactions - easy to analyse
  - Detector occupancy is low
  - Radiation damage is reduced
- High performance trigger based on
  - High p_T leptons and hadrons (Level 0)
  - Detached decay vertices (Level 1)
- Excellent particle identification for charged particles
  - K/?: ~1GeV/c < p < 100GeV/c

LHCb Detector Layout



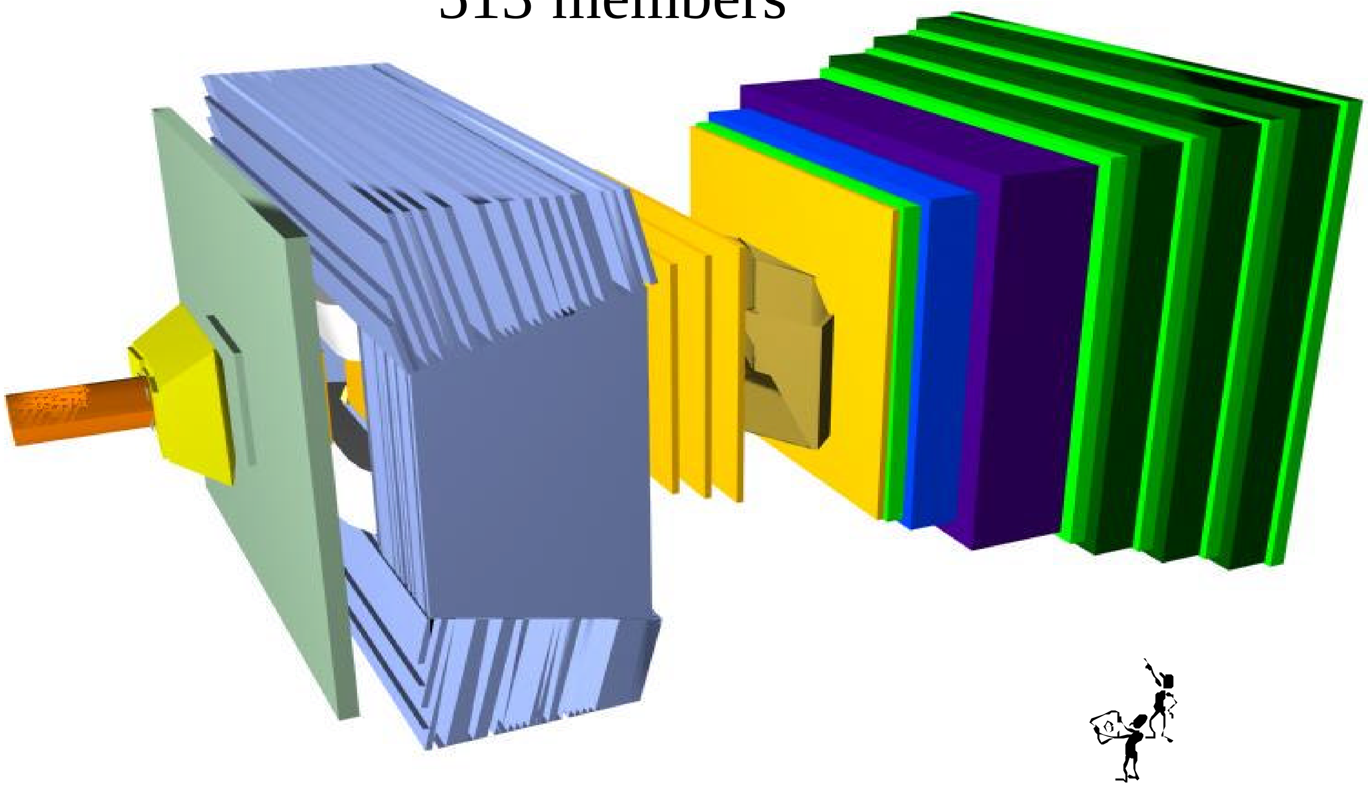

LHCb Expt Area 20th January 2000

Typical Interesting Event



The *LHCb* Collaboration

49 institutes
513 members



Brazil

Finland

France

Germany

Spain

Ukraine

UK

Switzerland

Italy

Netherlands

PRC

Poland

Romania

Russia

LHCb in numbers

- Expected rate from inelastic p-p collisions is ~15 MHz
- Total b-hadron production rate is ~75 kHz
- Branching ratios of interesting channels range between 10^{-5}-10^{-4} giving interesting physics rate of ~5 Hz

Bunch crossing rate	40 MHz
Level 0 accept rate	1 MHz
Level 1 accept rate	40 kHz
Level 2 accept rate	5 kHz
Level 3 accept rate	200 Hz
Number of Channels	1.1 M
Event Size	150 kB
Readout Rate	40 kHz
Event Building Bandwidth	6 GB/s
Data rate to Storage	50 MB/s
Total raw data per year	125 TB
Total ESD per year	100 TB
Simulation data per year	350 TB
Level 2/3 CPU	35 kSI95
Reconstruction CPU	50 kSI95
Analysis CPU	10 kSI95
Simulation CPU	500 kSI95

Timescales

- LHCb experiment approved in September 1998
- Construction of each component scheduled to start after approval of corresponding Technical Design Report (TDR) :
  - Magnet, Calorimeter and RICH TDRs submitted in 2000
  - Trigger and DAQ TDRs expected January 2002
  - Computing TDR expected December 2002
- Expect nominal luminosity ($2x10^{32}$ $cm^{-2}sec^{-1}$) soon after LHC turn-on
  - Exploit physics potential from day 1
  - Smooth operation of the whole data acquisition and data processing chain will be needed very quickly after turn–on
- Locally tuneable luminosity ? long physics programme
  - Cope with long life-cycle of ~ 15 years

LHCb Computing
Scope and Organisation

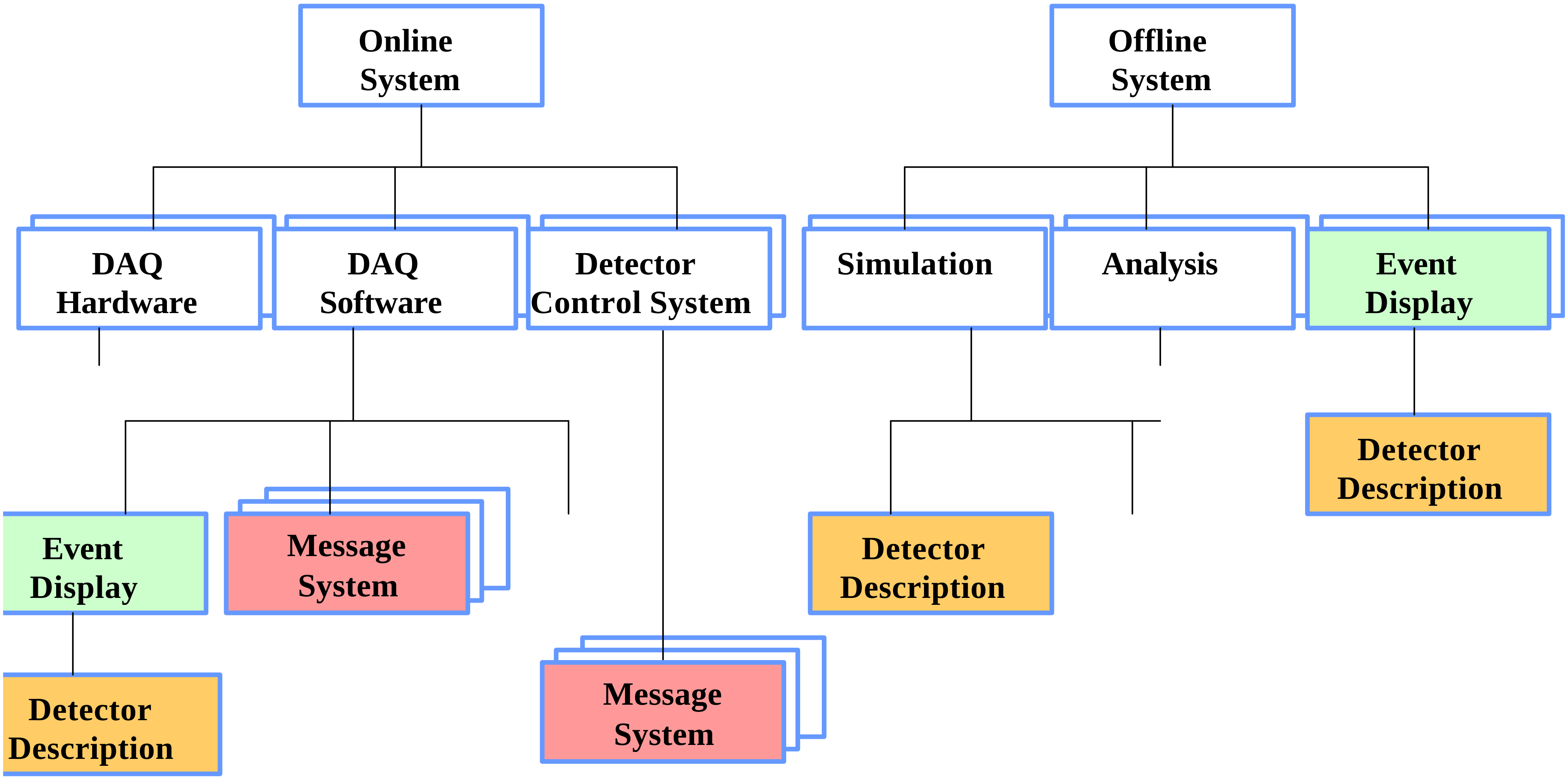
Requirements and Resources

- More stringent requirements …
  - Enormous number of items to control - scalability
  - Inaccessibility of detector and electronics during datataking - reliability
  - intense use of software in triggering (Levels 1, 2, 3) - quality
  - many orders of magnitude more data and CPU - performance
- Experienced manpower very scarce
  - Staffing levels falling
  - Technology evolving very quickly (hardware and software)
  - Rely very heavily on very few experts (1 or 2) - bootstrap approach
- The problem - a more rigorous approach is needed but this is more manpower intensive and must be undertaken under conditions of dwindling resources

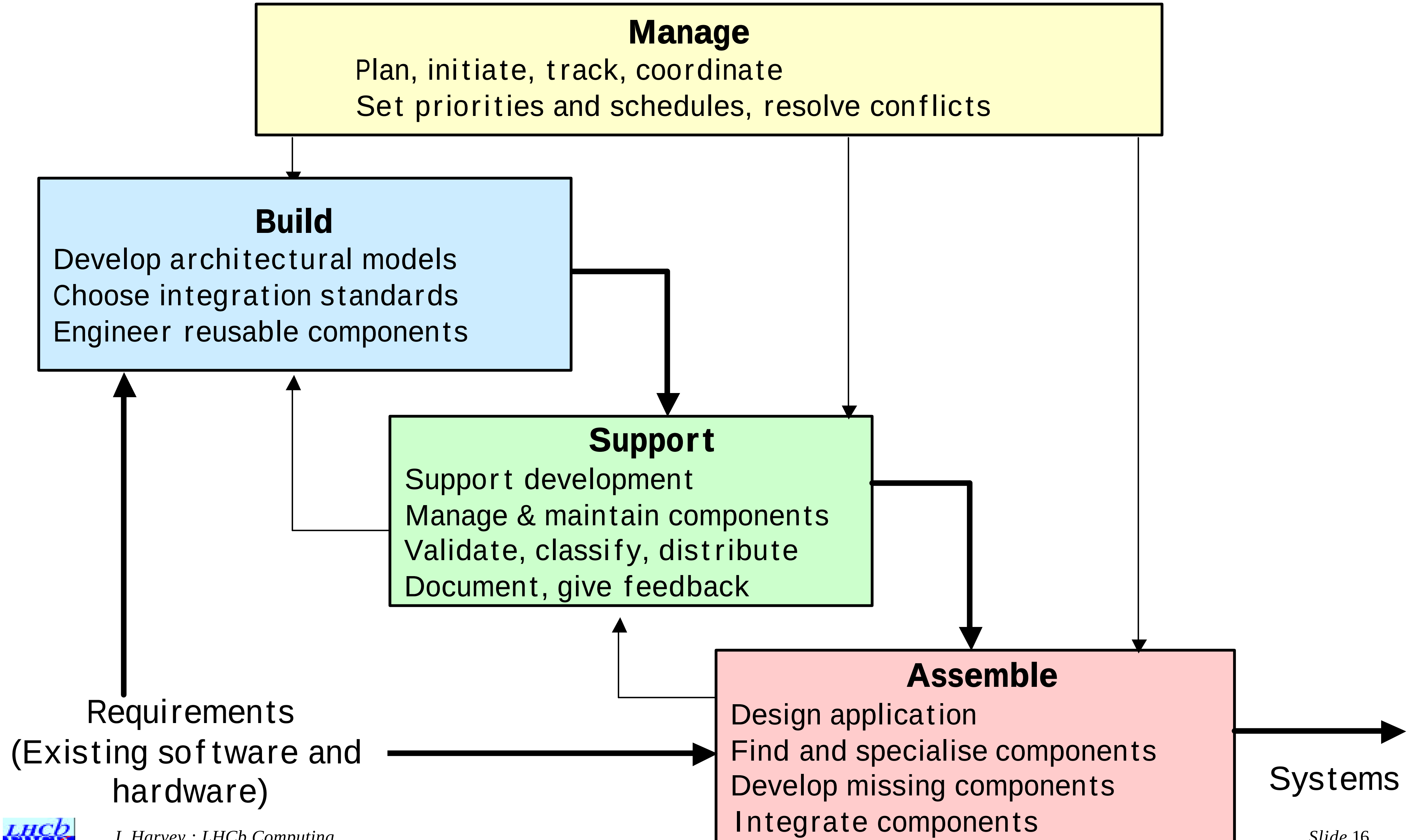
Importance of Reuse

- Put extra effort into building high quality components
- Become more efficient by extracting more use out of these components (reuse)
- Many obstacles to overcome
  - too broad functionality / lack of flexibility in components
  - proper roles and responsibilities not defined (e.g. architect)
  - organisational - reuse requires a broad overview to ensure unified approach
    - we tend to split into separate domains each independently managed
  - cultural
    - don’t trust others to deliver what we need
    - fear of dependency on others
    - fail to share information with others
    - developers fear loss of creativity
- Reuse is a management activity - need to provide the right organisation to make it happen

Traditional Project Organisation



A Process for reuse



LHCb Computing Project Organisation

National Computing Board
RC RC RC C

Computing Steering Group
A M C A

Technical Review
E E M A

M
Software Development Support
Code Mgt,
Release Mgt,
Tools,
Training
Documentation
Web

M **Operations**

M **Simulation**

M **Trigger**

M **Reconstruction**

M **Analysis**

A **GAUDI Framework**
Architecture Spec.
Det Desc, Visualisation
GEANT4, XML,…

M **Experiment Control System**
Detector controls
Safety System
Run Control system

Controls Framework
Architecture Spec,
SCADA, OPC, …

M **DAQ System**
Timing Fast Control,
Readout Unit
Event Builder
Event Filter Farm

DAQ Framework
Architecture Spec,
Simulation Model,
TTC, NP, NIC,..

M
Distributed Computing Facilities
CPU Farms
Data storage
Computing Model
Production Tools
GRID

Manage | Assemble | Build | Support

C Computing Coordinator
M Project Manager
E Project Engineer
RC Regional Centre Rep
A Software Architect

Data Processing Software

Software architecture

- Definition of [software] architecture [1]
  - Set or **significant decisions** about the **organization** of the software system
  - Selection of the **structural elements** and their **interfaces** which compose the system
  - Their **behavior** -- collaboration among the structural elements
  - **Composition** of these structural and behavioral elements into progressively larger **subsystems**
  - The **architectural style** that guides this organization

- The **architecture** is the blue-print (architecture description document)

[1] I. Jacobson, et al. “The Unified Software development Process”, Addison Wesley 1999

Software Framework

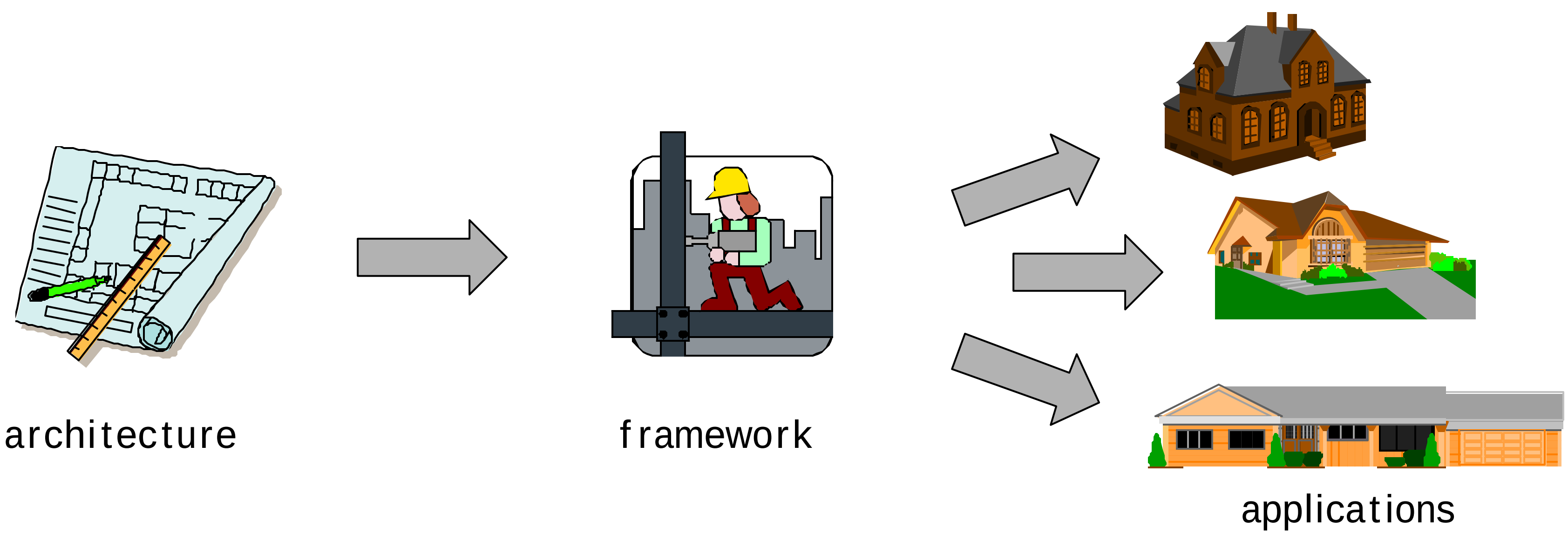
- Definition of [software] **framework** [2,3]
  - A kind of micro-architecture that codifies a particular domain
  - Provides the suitable knobs, slots and tabs that permit clients to customise it for specific applications within a given range of behaviour
  - A framework realizes an architecture
  - A large O-O system is constructed from several cooperating frameworks
- The **framework** is real code
- The **framework** should be easy to use and should provide a lot of functionality

[2] G. Booch, “Object Solutions”, Addison-Wesley 1996

[3] E. Gamma, et al., “Design Patterns”, Addison-Wesley 1995

Benefits

- Having an architecture and a framework:
  - Common vocabulary, better specifications of what needs to be done, better understanding of the system.
  - Low coupling between concurrent developments. Smooth integration. Organization of the development.
  - Robustness, resilient to change (change-tolerant).
  - Fostering code re-use



What's the scope?

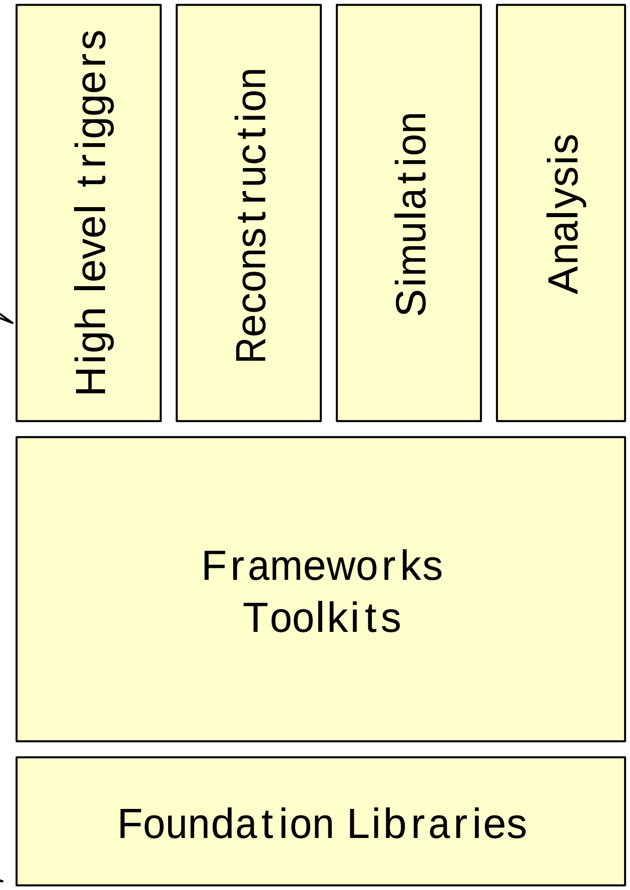
- Each LHC experiment needs a framework to be used in their **event data processing applications**
  - physics/detector simulation
  - high level triggers
  - reconstruction
  - analysis
  - event display
  - data quality monitoring,...
- The experiment framework will incorporate other frameworks: persistency, detector description, event simulation, visualization, GUI, etc.

Software Structure

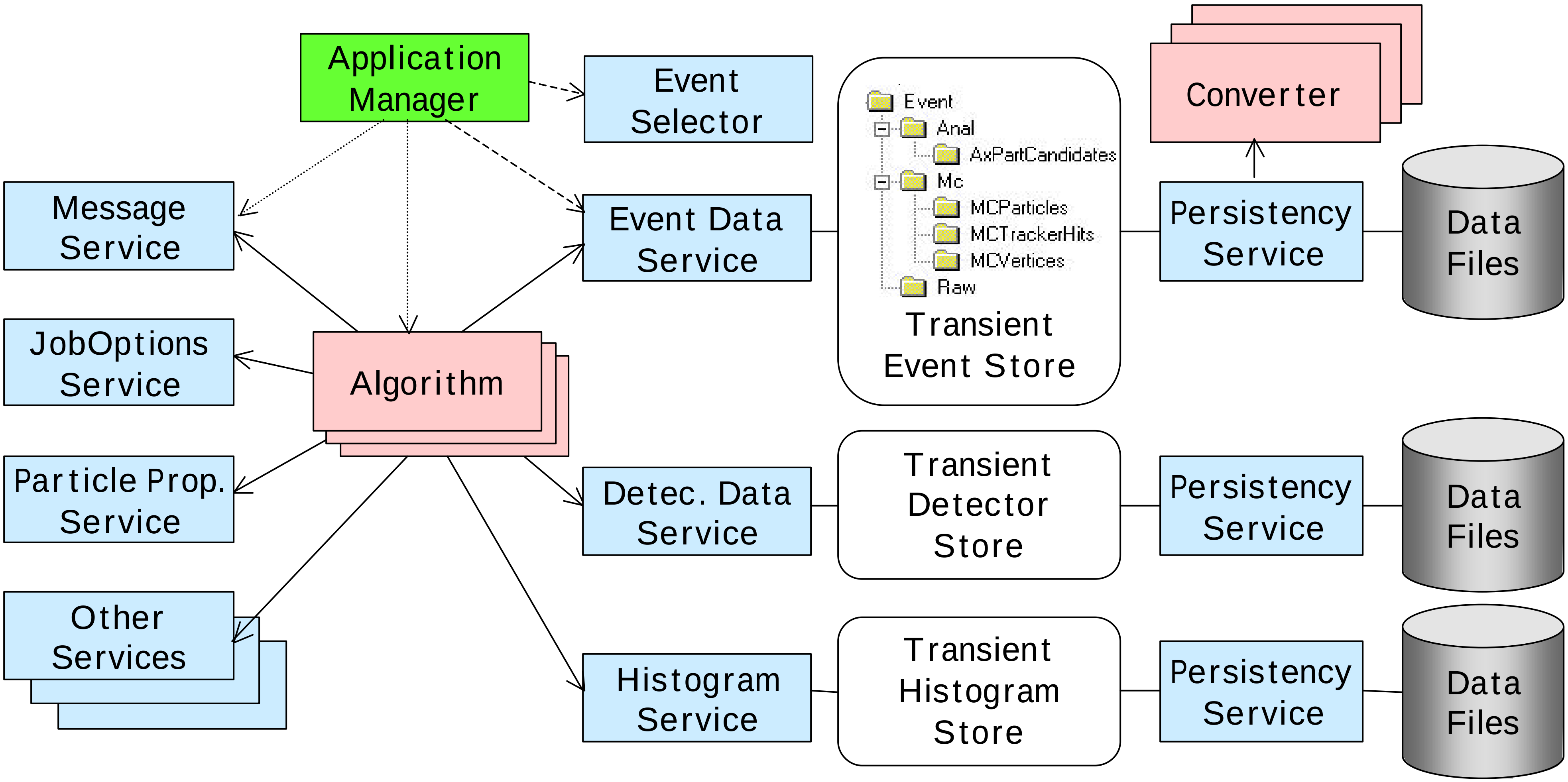
Applications built on top of frameworks and implementing the required physics algorithms.

One main framework
Various specialized frameworks: visualization, persistency, interactivity, simulation, etc.

A series of basic libraries widely used: STL, CLHEP, etc.



GAUDI Object Diagram



GAUDI Architecture: Design Criteria

- Clear separation between **data** and **algorithms**
- Three basic types of data: **event**, **detector**, **statistics**
- Clear separation between **persistent** and **transient** data
- **Computation-centric** architectural style
- **User code** encapsulated in few specific places: algorithms and converters
- All components with well defined **interfaces** and as **generic** as possible

Status

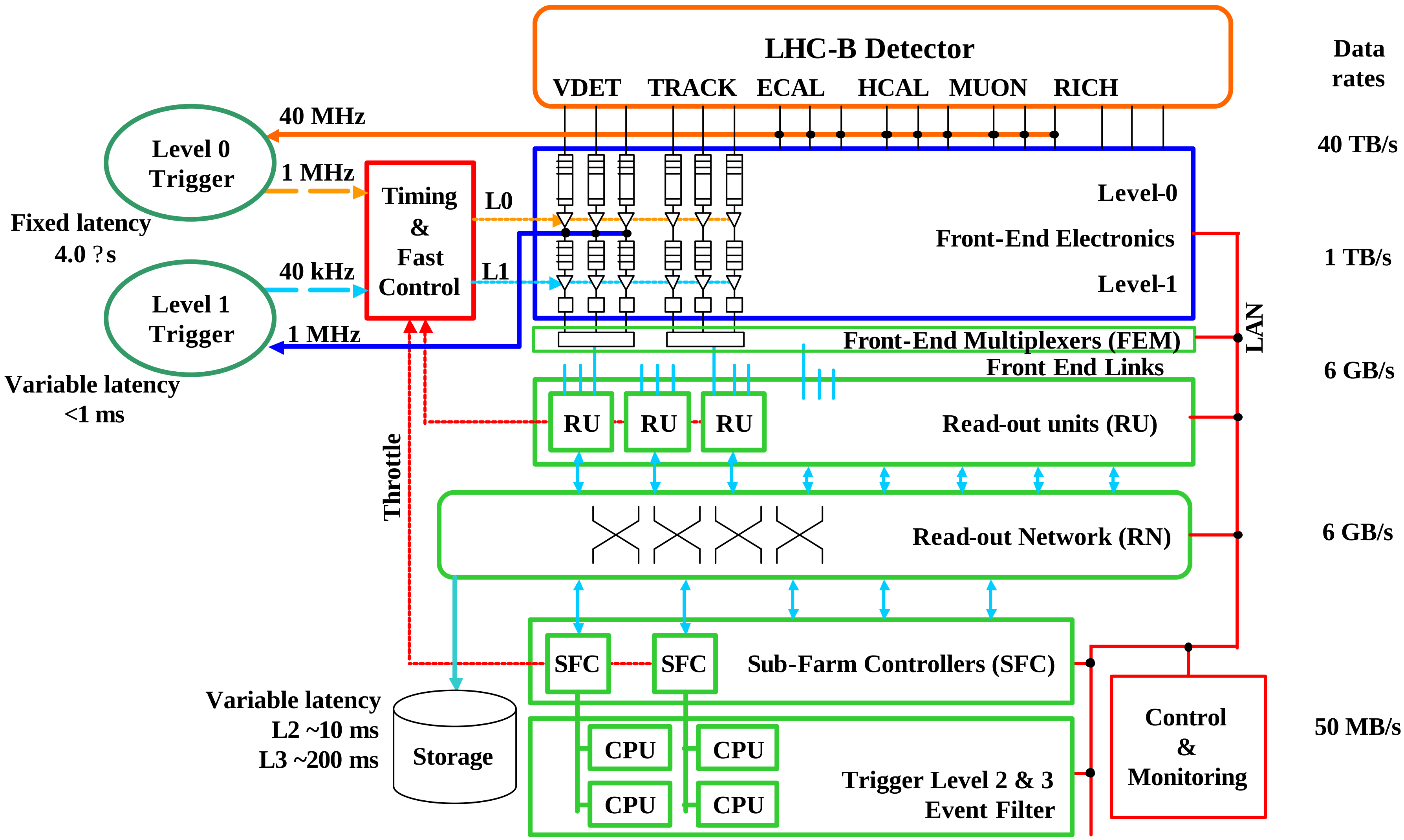
- Sept 98 – project started GAUDI team assembled
- Nov 25 '98 - 1- day architecture review
  - goals, architecture design document, URD, scenarios
  - chair, recorder, architect, external reviewers
- Feb 8 '99 - GAUDI first release (v1)
  - first software week with presentations and tutorial sessions
  - plan for second release
  - expand GAUDI team to cover new domains (e.g. analysis toolkits, visualisation)
- Nov '00 – GAUDI v6
- Nov 00 – BRUNEL v1
  - New reconstruction program based on GAUDI
  - Supports C++ algorithms (tracking) and wrapped FORTRAN
  - FORTRAN gradually being replaced

Collaboration with ATLAS

- Now ATLAS also contributing to the development of GAUDI
  - Open-Source style, expt independent web and release area,
- Other experiments are also using GAUDI
  - HARP, GLAST, OPERA
- Since we can not provide all the functionality ourselves, we rely on contributions from others
  - Examples: Scripting interface, data dictionaries, interactive analysis, etc.
- Encouragement to put more quality into the product
- Better testing in different environments (platforms, domains,..)
- Shared long-term maintenance
- Gaudi developers mailing list
  - tilde-majordom.home.cern.ch/~majordom/news/gaudi-developers/index.html

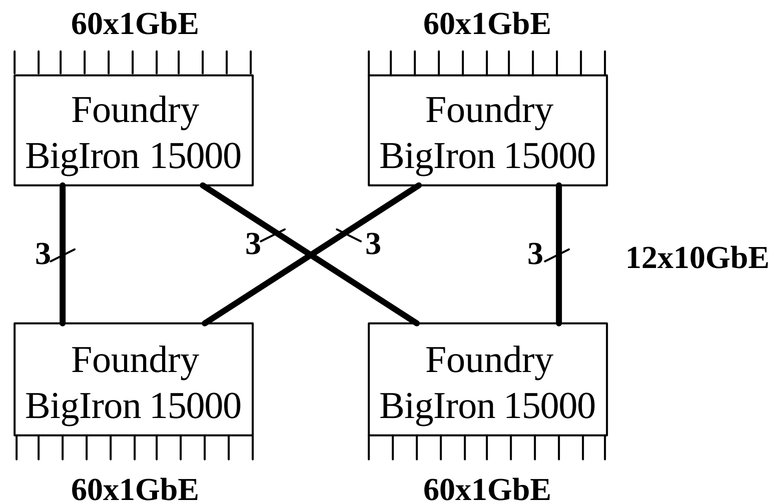
Data Acquisition System

Trigger/DAQ Architecture



Event Building Network

- Requirements
  - 6 GB/s sustained bandwidth
  - Scalable
  - ~120 inputs (RUs)
  - ~120 outputs (SFCs)
  - commercial and affordable (COTS, Commodity?)



- Readout Protocol
  - Pure push-through protocol of complete events to one CPU of the farm
  - Destination assignment following identical algorithm in all RUs (belonging to one partition) based on event number
  - Simple hardware and software
  - No central control ? perfect scalability
  - Full flexibility for high-level trigger algorithms
  - Larger bandwidth needed (+~50%) compared with phased event-building
  - Avoiding buffer overflows via 'throttle' to trigger
  - ? Only static load balancing between RUs and SFCs