2

Introduction to Gaudi

LHCb Software Strategy

Develop an Architecture (‘blueprint’) and a Framework (real code) to be used at all stages of LHCb data processing

- HLT, simulation, reconstruction, analysis

Avoid fragmentation and duplication of computing efforts

Transparent use of third-party components wherever possible

Applications are developed by customizing the Framework

What is a Framework?

Framework Definition [1,2]

- An architectural pattern that codifies a particular domain. It provides the suitable knobs, slots and tabs that permit clients to use and adapt to specific applications within a given range of behavior.

In practice

- A skeleton of an application into which developers plug in their code and provides most of the common functionality.

[1] G. Booch, "Object Solutions", Addison-Wesley 1996

[2] E. Gamma, et al., "Design Patterns", Addison-Wesley 1995

Framework Benefits

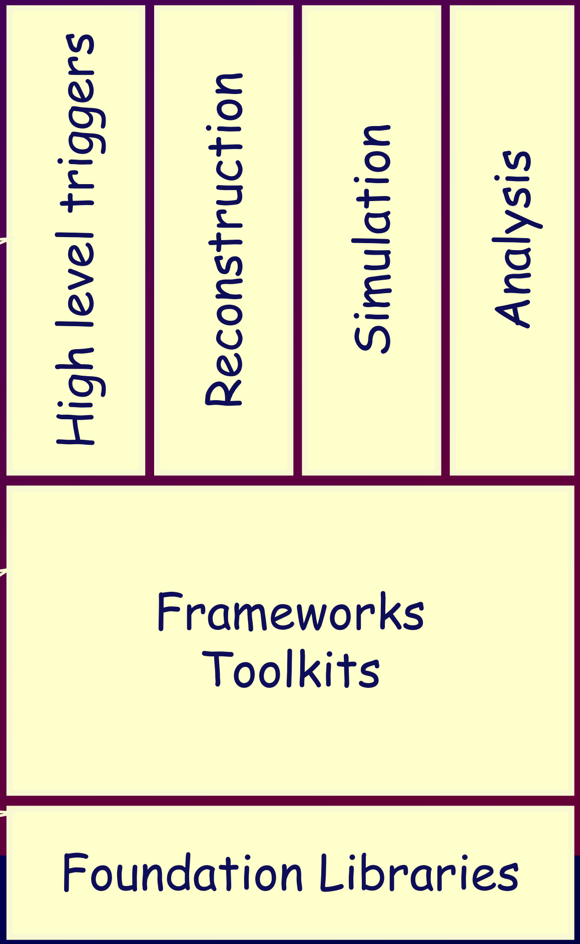
- Common vocabulary, better specifications of what needs to be done, better understanding of the system.
- Low coupling between concurrent developments. Smooth integration. Organization of the development.
- Robustness, resilient to change (change-tolerant).
- Fostering code re-use

Software Organization

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

One framework for basic services + various specialized frameworks:
- detector description,
- visualization, persistency,
- interactivity, simulation, etc.

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



High level triggers

Reconstruction

Simulation

Analysis

Frameworks
Toolkits

Foundation Libraries

Gaudi Architecture

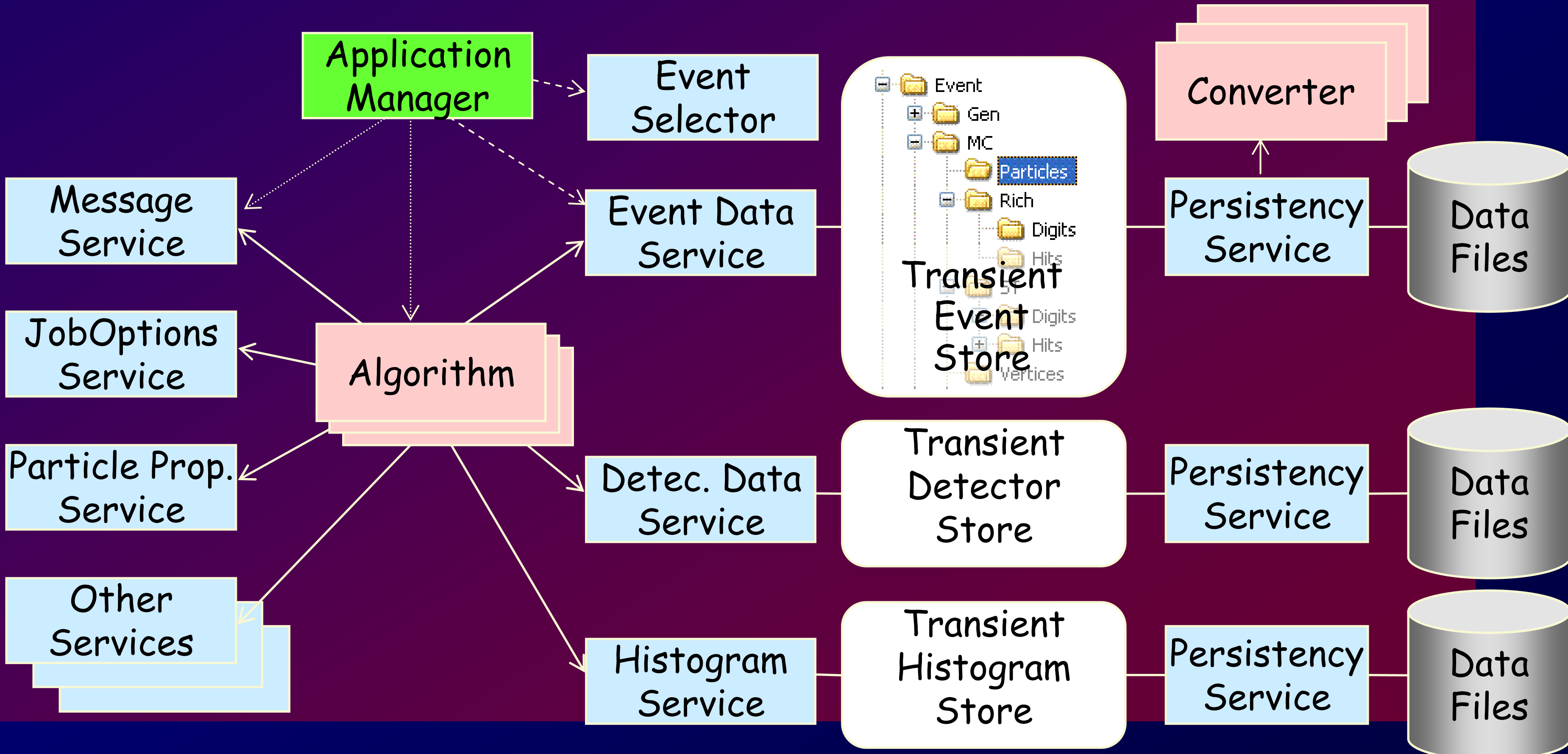
GAUDI is an architecture and framework for event-processing applications (simulation, reconstruction, etc.)

- Initially developed for LHCb, it has been adopted and extended by ATLAS and adopted by several other experiments including GLAST and HARP

Main Design Choices

- Separation between “data” and “algorithms”
- Three basic categories of “data”: event data, detector data, statistical data
- Separation between “transient” and “persistent” representations of data
- Data store-centered (“blackboard”) architectural style
- “User code” encapsulated in few specific places
- Well defined component “interfaces”

Gaudi Object Diagram



Definition of Terms

- Algorithm
  - Atomic data processing unit (visible & controlled by the framework)
- Data Object
  - Atomic data unit (visible and managed by transient data store)
- Transient Data Store
  - Central service and repository for data objects (data location, life cycle, load on demand, …)

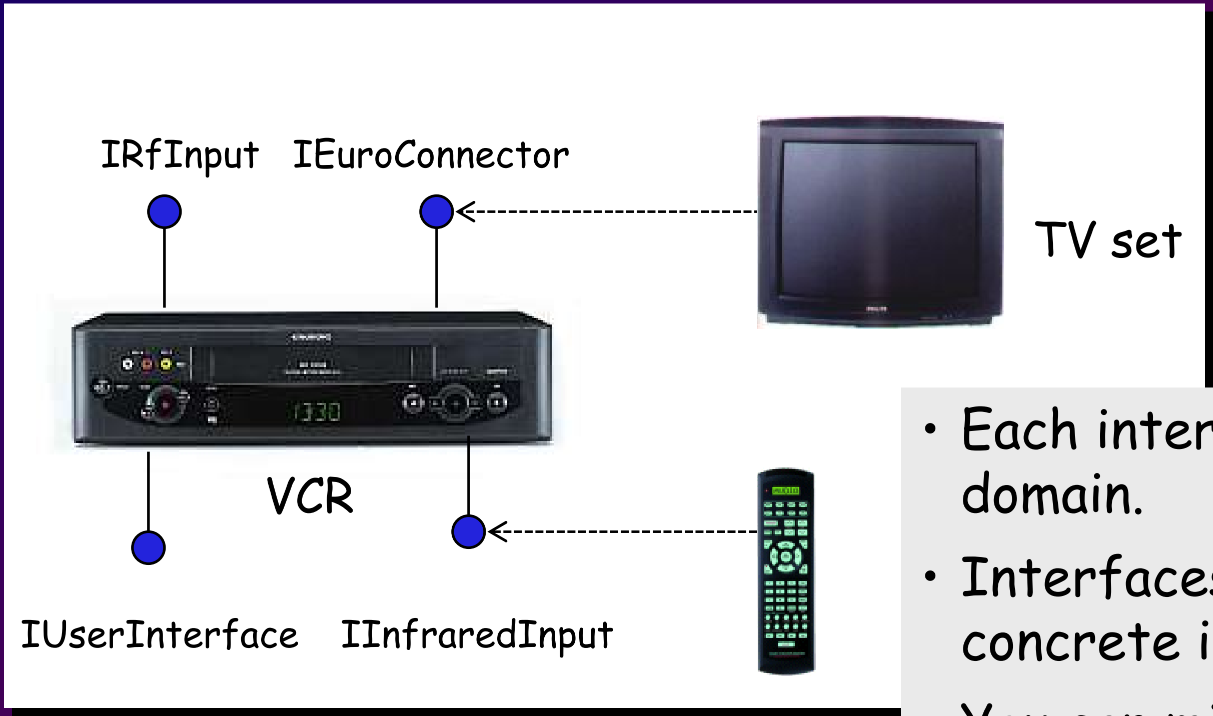
Definition of Terms (2)

- Services
  - Globally available software components providing framework functionality
- Tools
  - Globally or locally available components to allow sharing of code between algorithms
- Data Converter
  - Provides explicit/implicit conversion from/to persistent data format to/from transient data
- Properties
  - Control and data parameters for Algorithms and Services

Algorithm

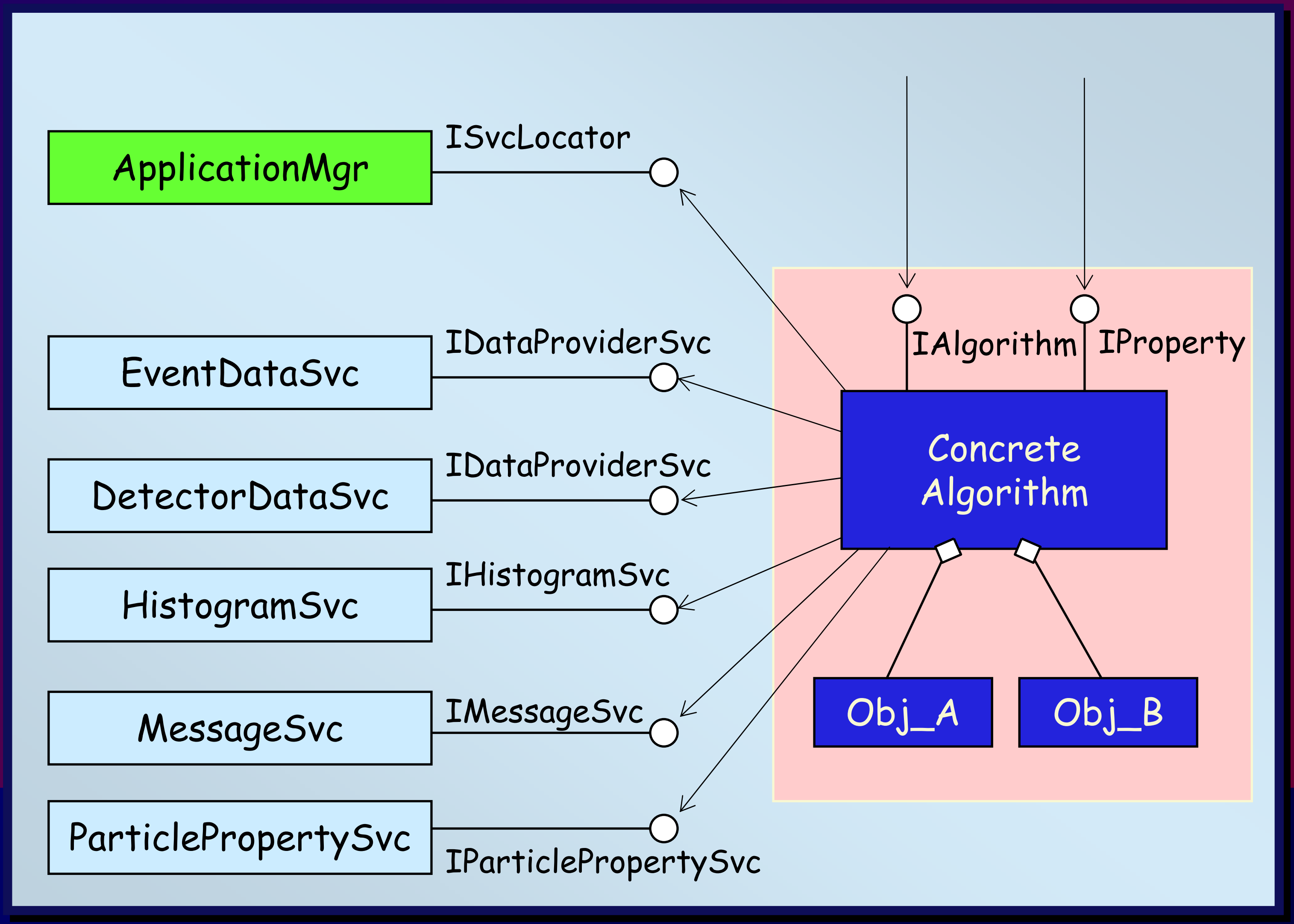
- Users write Concrete Algorithms
  - Most of the tutorial will be devoted to that
- It is called once per physics event
- Implements three methods in addition to the constructor and destructor
  - initialize(), execute(), finalize()

VCR Interface Model



- Each interface is specialized in a domain.
- Interfaces are independent of concrete implementations.
- You can mix devices from several constructors.
- Application built by composing.
- Standardizing on the interfaces gives us big leverage.

Gaudi Interface model



Interfaces in Practice

IMyInterface.h

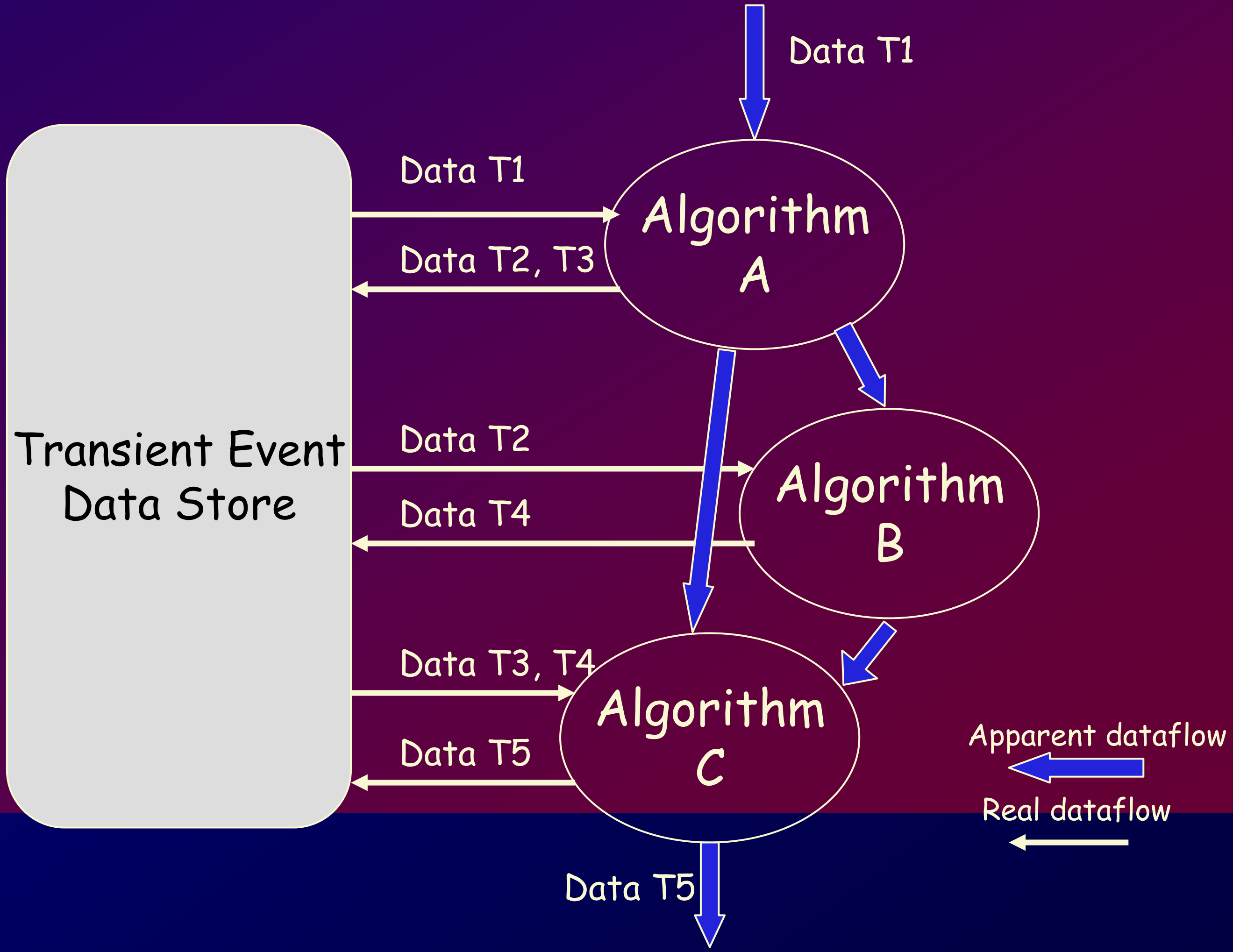
```
class IMyInterface {
  virtual void doSomething( int a, double b ) = 0;
}
```

ClientAlgorihtm.cpp

```
#include “IMyInterface.h”

ClientAlgorithm::myMethod() {
  // Declare the interface
  IMyInterface* myInterface;
  // Get the interface from somewhere
  myInterface = svc<IMyInterface>(“MyServiceProvider”);
  // Use the interface
  myInterface->doSomething( 10, 100.5);
}
```

Algorithm & Transient Store



Transient Event Data Store

Data T1

Algorithm A

Data T1

Data T2, T3

Data T2

Algorithm B

Data T4

Data T3, T4

Algorithm C

Data T5

Data T5

Apparent dataflow

Real dataflow

Gaudi Services

- JobOptions Service
- Message Service
- Particle Properties Service
- Event Data Service
- Histogram Service
- N-tuple Service
- Detector Data Service
- Magnetic Field Service
- Random Number Generator
- Chrono Service
- (Persistency Services)
- (User Interface & Visualization Services)
- (Geant4 Services)

Gaudi Product Sheet

- Current release
  - v18r10 (November 06)
- Supported Platforms
  - Scientific Linux (CERN) 3 & gcc 3.2.3
  - Scientific Linux (CERN) 4 & gcc 3.4
  - Windows 2000,XP & VisualC++ 7.1
- Web address.
  - http://cern.ch/proj-gaudi/

Documentation

- Gaudi User Guide
  - A 220 pages document targeted to end-users
- C++ Documentation
  - Generated from code (Doxygen)
    - Uses special comments in code, e.g. Tutorial solutions
    - http://cern.ch/proj-gaudi/releases/GAUDI/doc/html/index.html
- Self help
    - lhcb-soft-talk@cern.ch