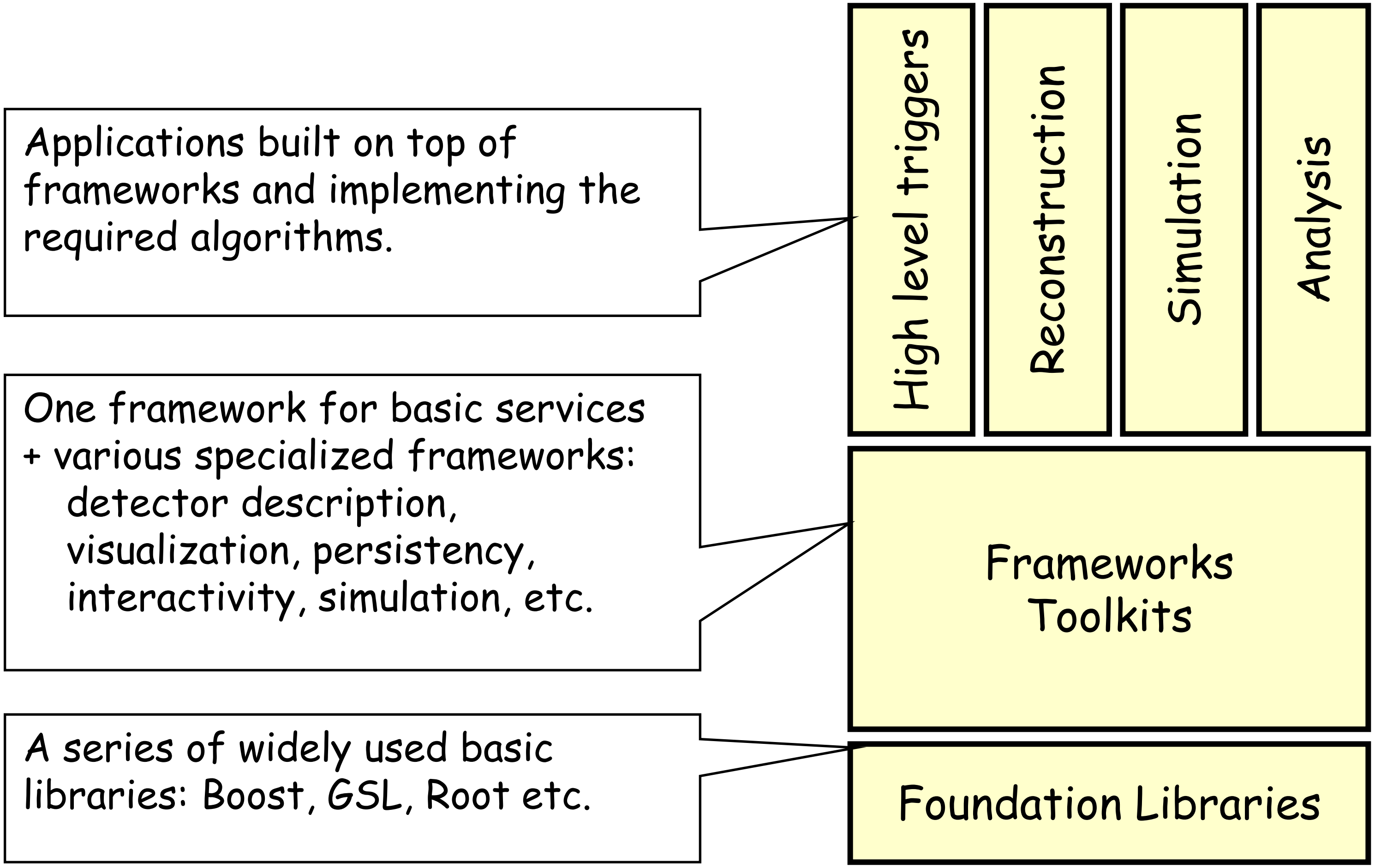
Introduction to Gaudi

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

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 widely used basic libraries: Boost, GSL, Root etc.

High level triggers

Reconstruction

Simulation

Analysis

Frameworks
Toolkits

Foundation Libraries

What is a Framework?

- Framework Definition [1,2]

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

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

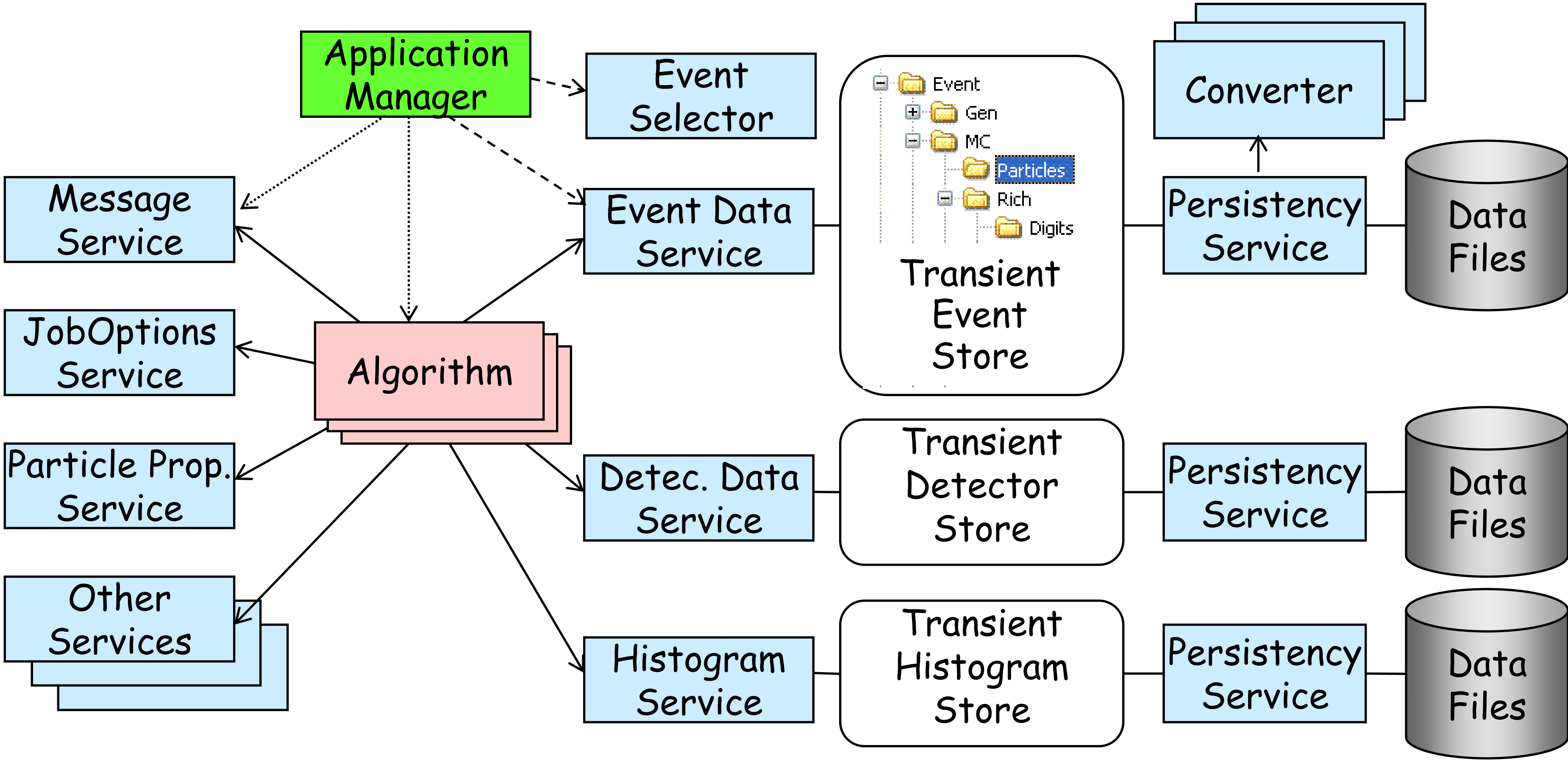
  - An architectural pattern that codifies a particular domain. It provides the suitable knobs, slots and tabs that permit clients to use and adapt to specific applications within a given range of behavior.
- In practice
  - A skeleton of an application into which developers plug in their code, and which provides most of the common functionality.
- Benefits
  - Common vocabulary, better specifications of what needs to be done, better understanding of the system.
  - Low coupling between concurrent developments.
    - Smooth integration.
  - Robustness, resilient to change (change-tolerant).
  - Fostering code re-use

Gaudi Architecture

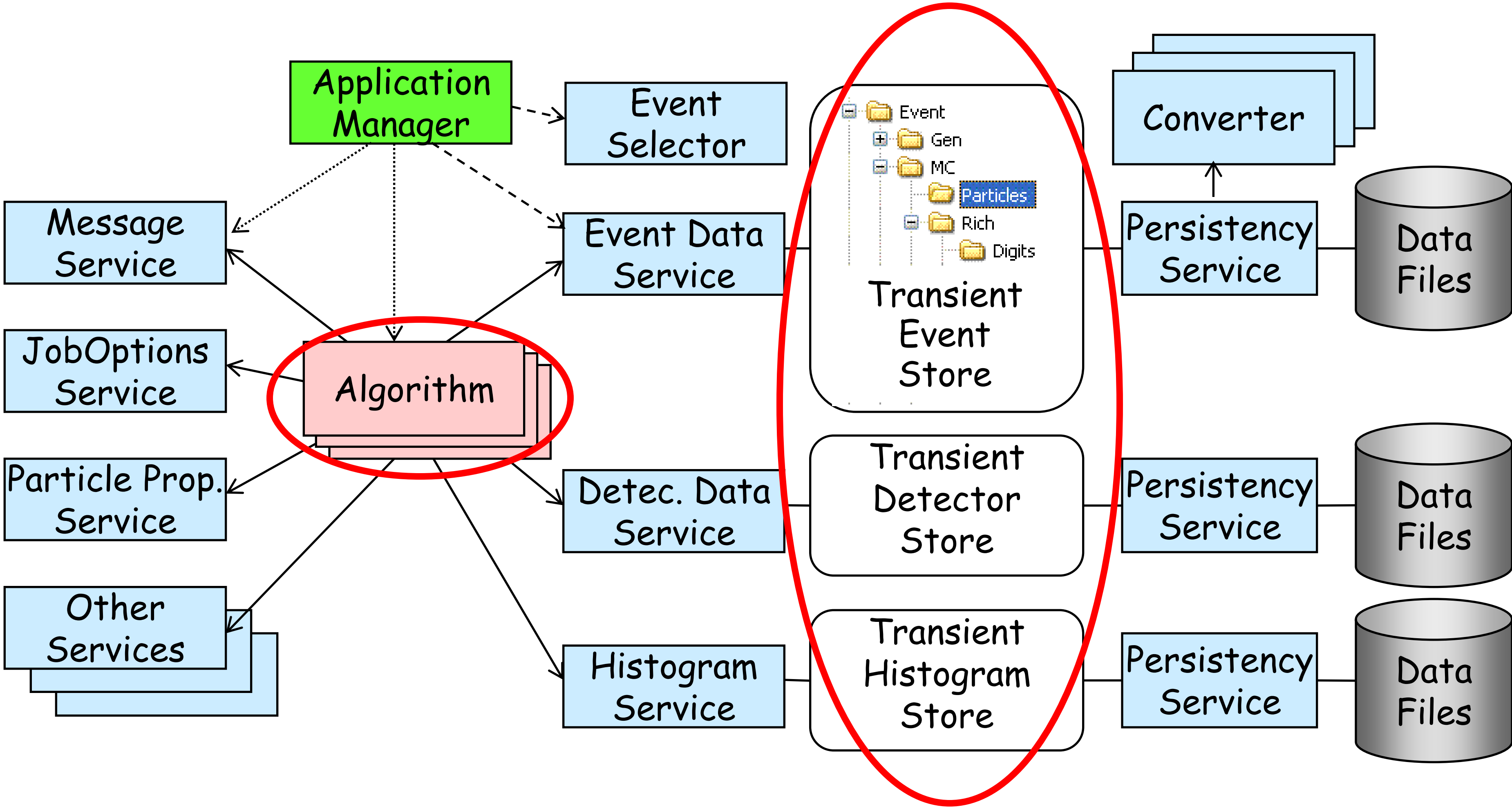
- GAUDI is an architecture and framework for event-processing applications (simulation, reconstruction, etc.)
  - Initially developed for LHCb, it has been adopted and extended by ATLAS and adopted by several other experiments including Fermi Space Telescope, HARP, Minerva, Daya Bay

- Main Design Choices best illustrated by looking at the Gaudi Object diagram
  - A snapshot of the components in memory, once the job is initialized and running

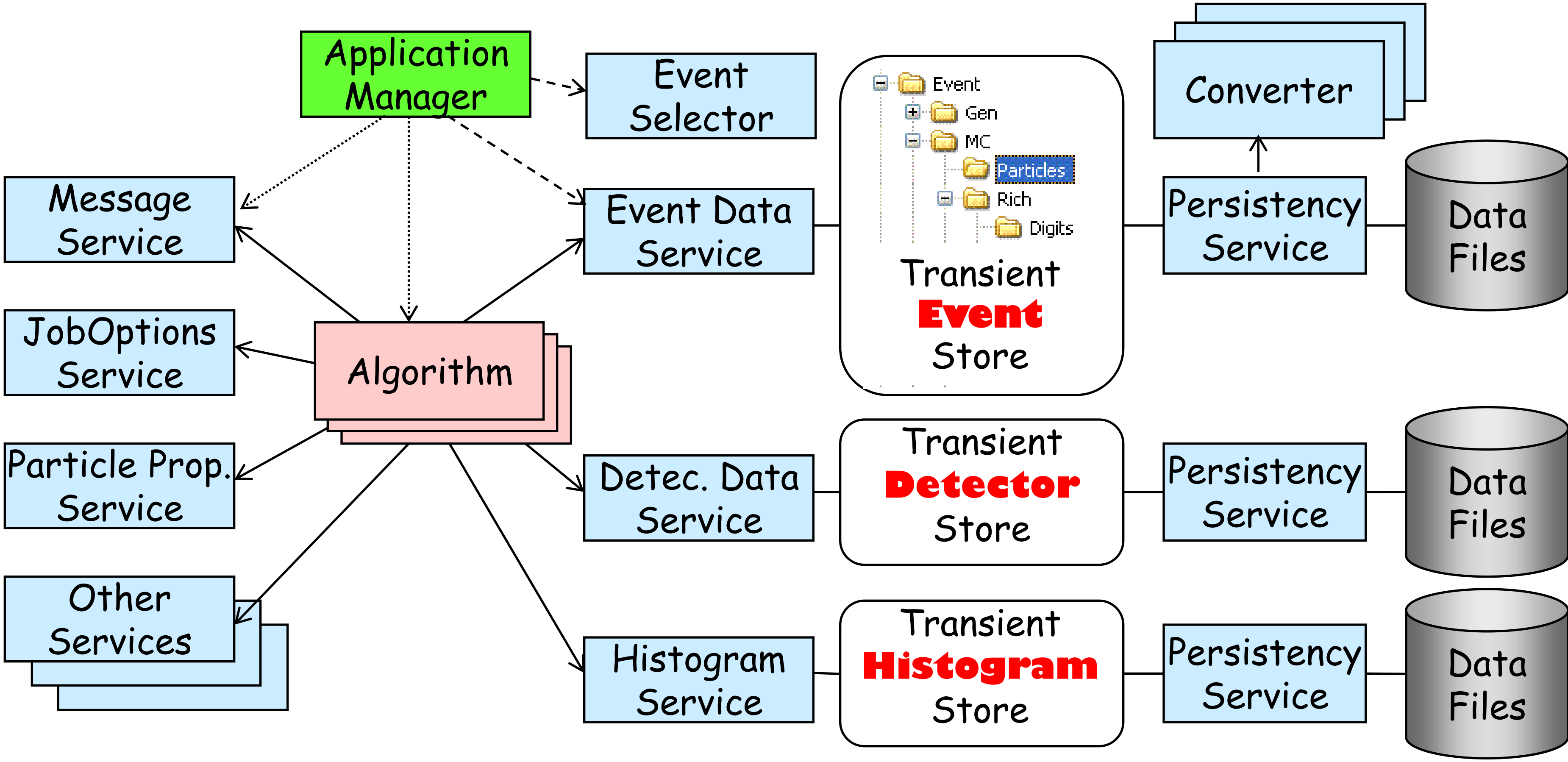
Gaudi Object Diagram



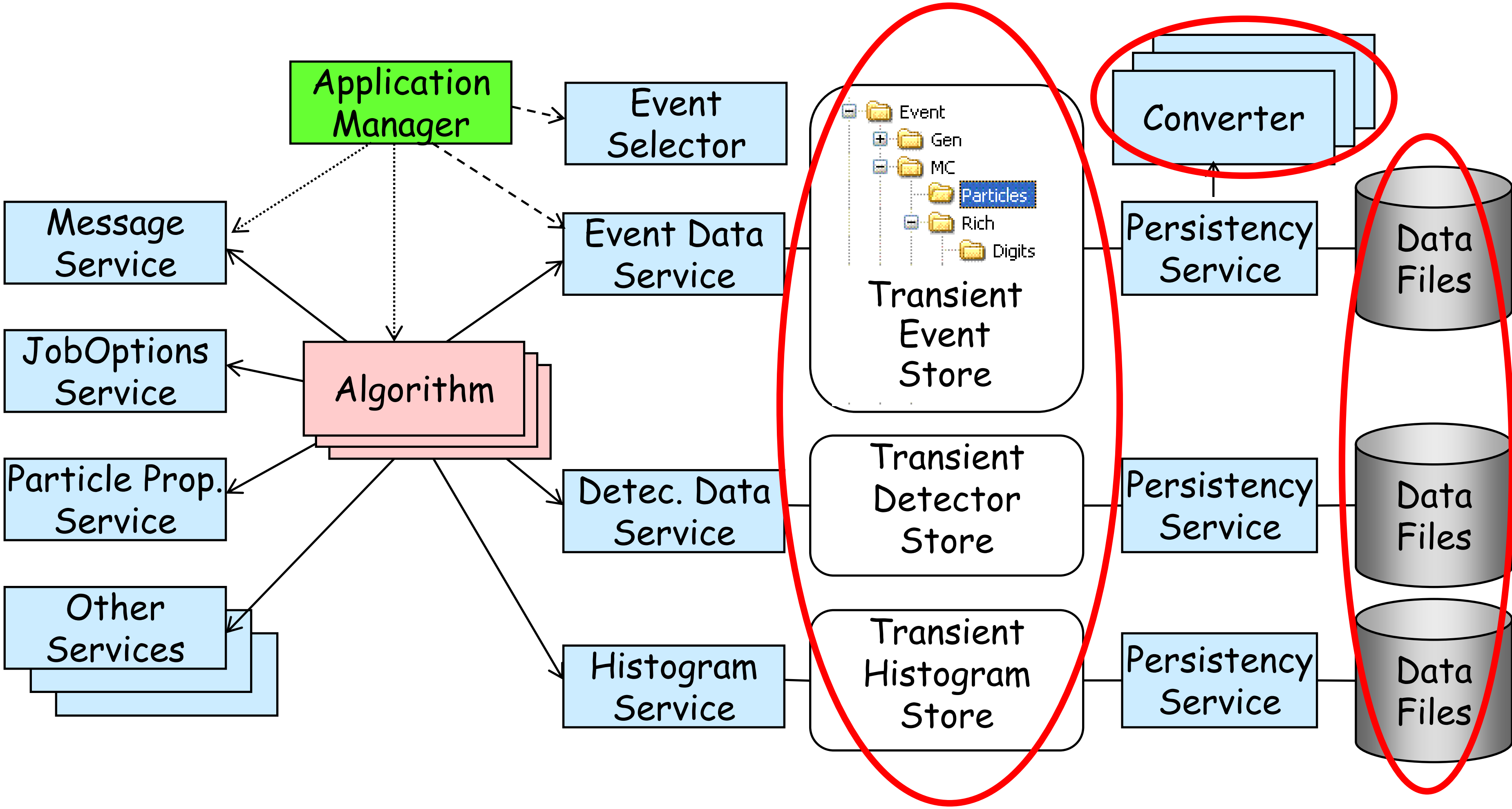
Separation between “data” and “algorithms”



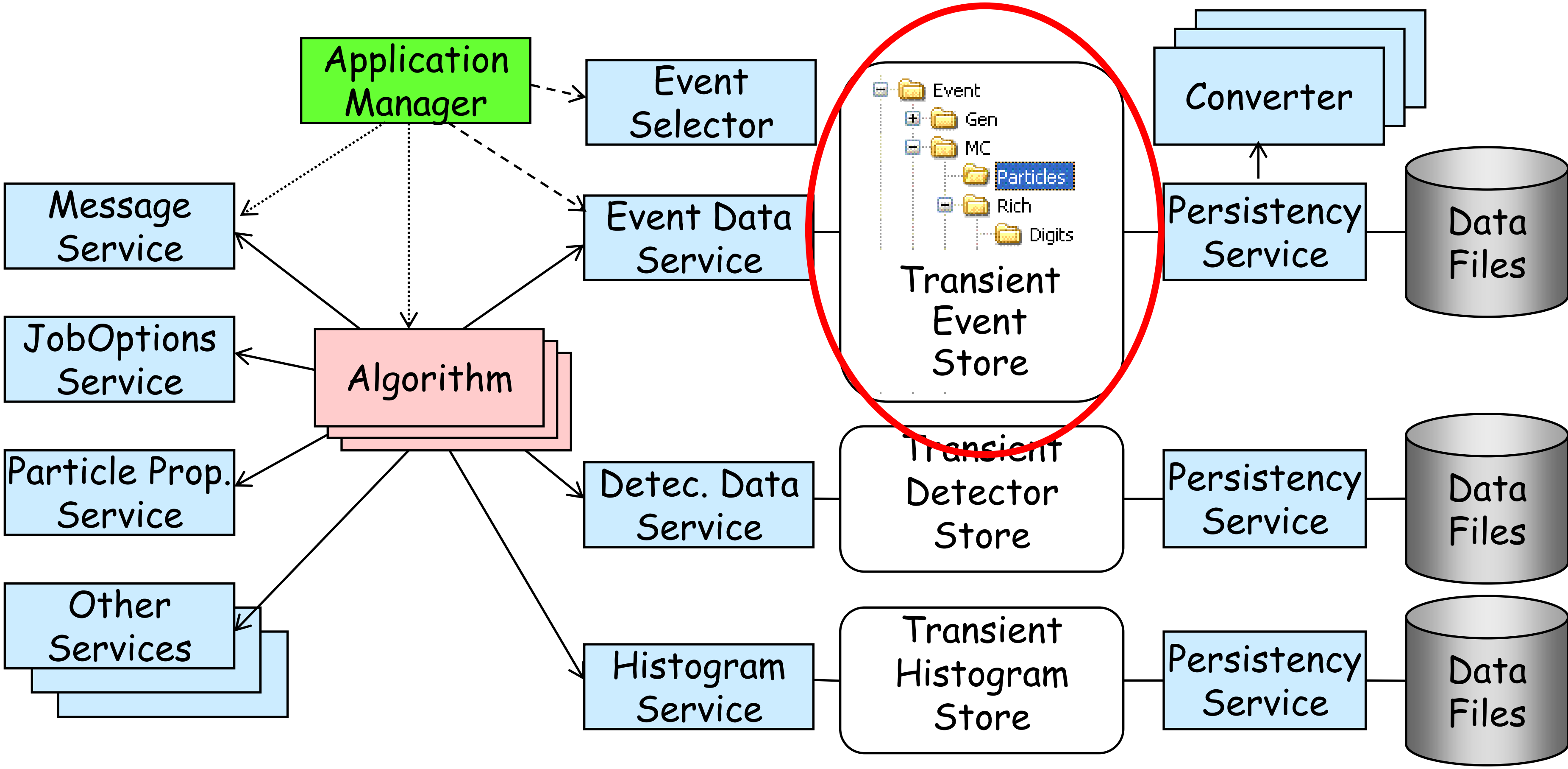
Three basic categories of data characterized by their “lifetime” in the job



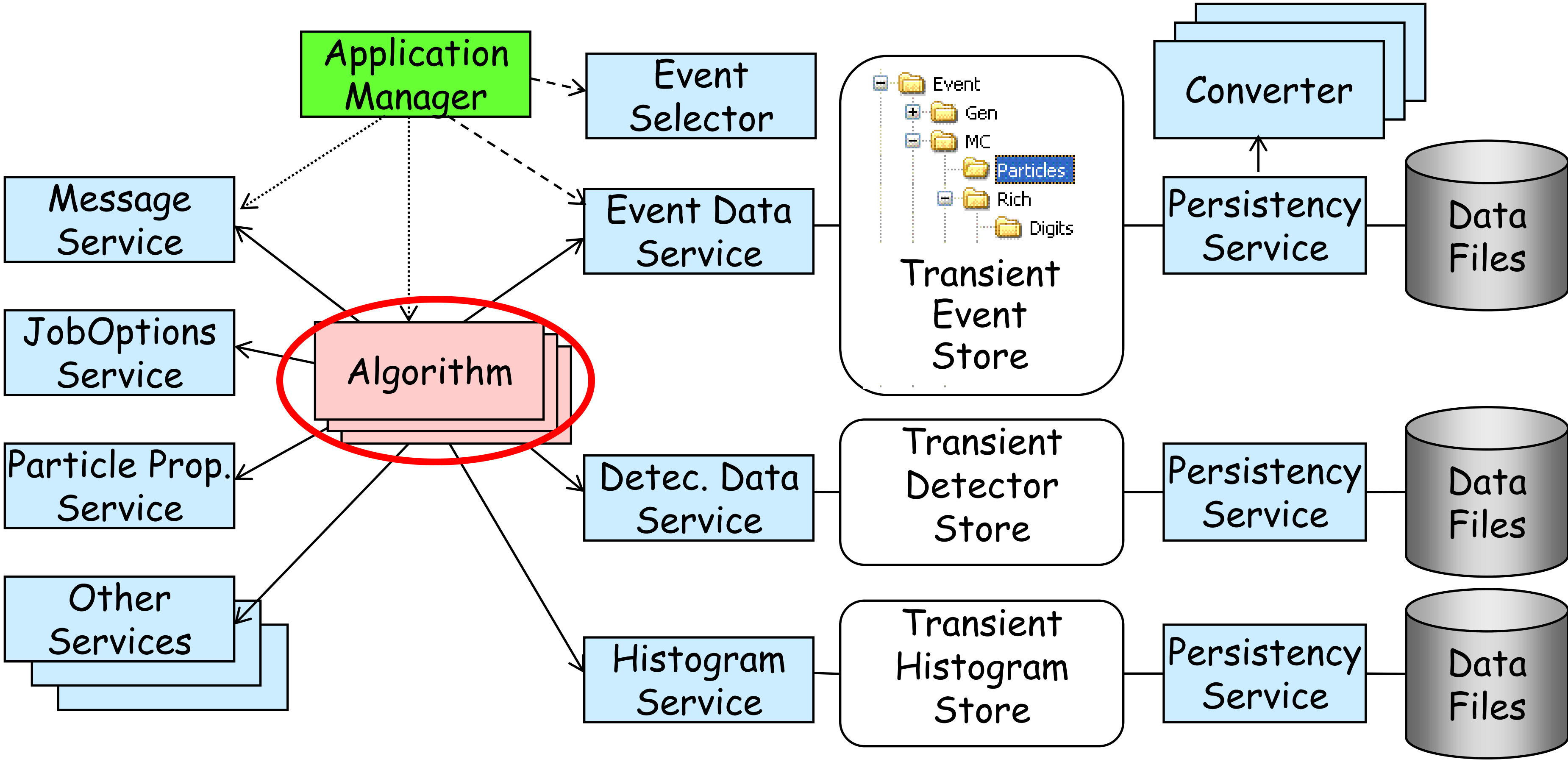
Separation between “transient” and “persistent” representations of the data



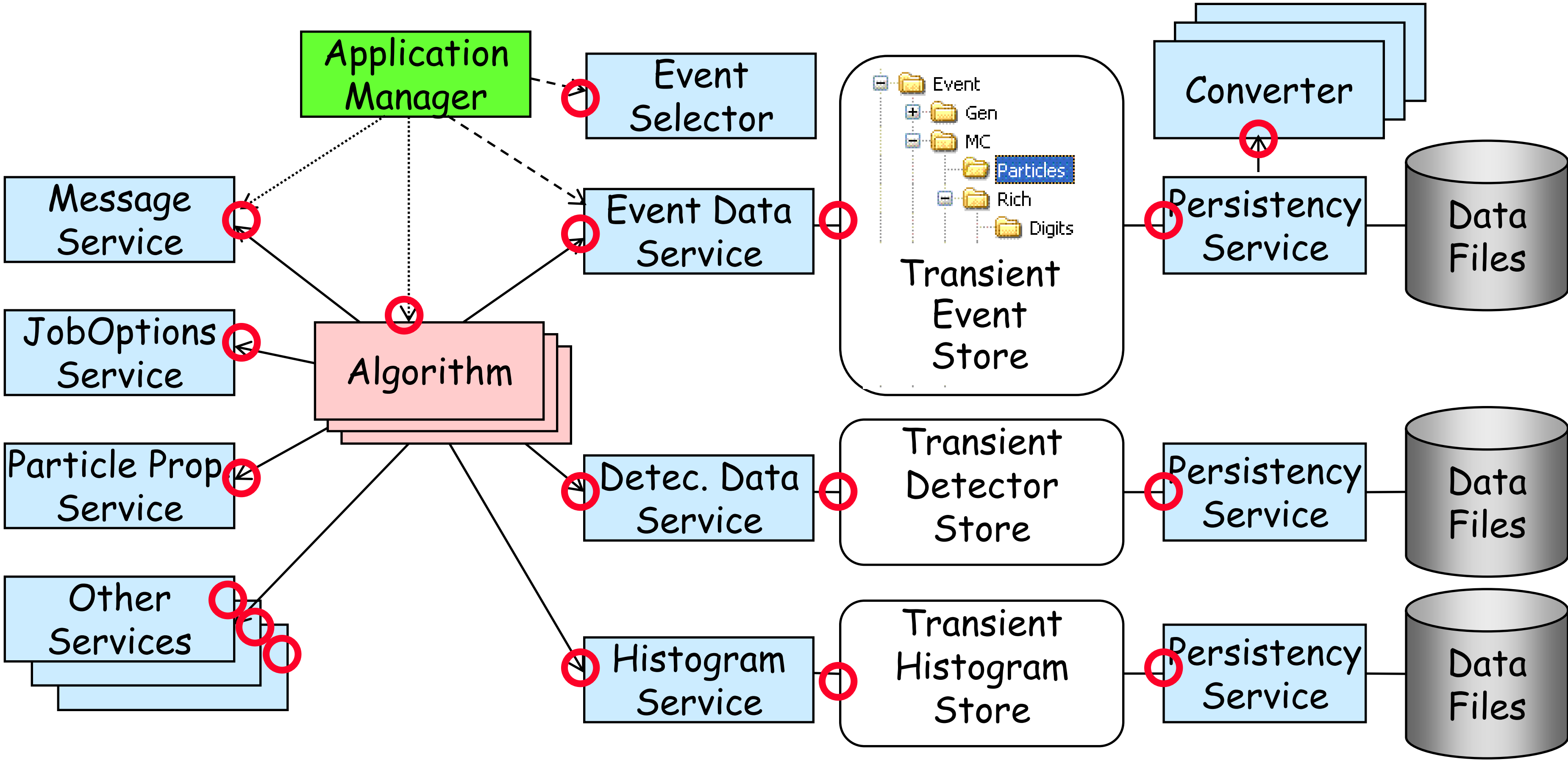
Data store-centered (“blackboard”) architectural style



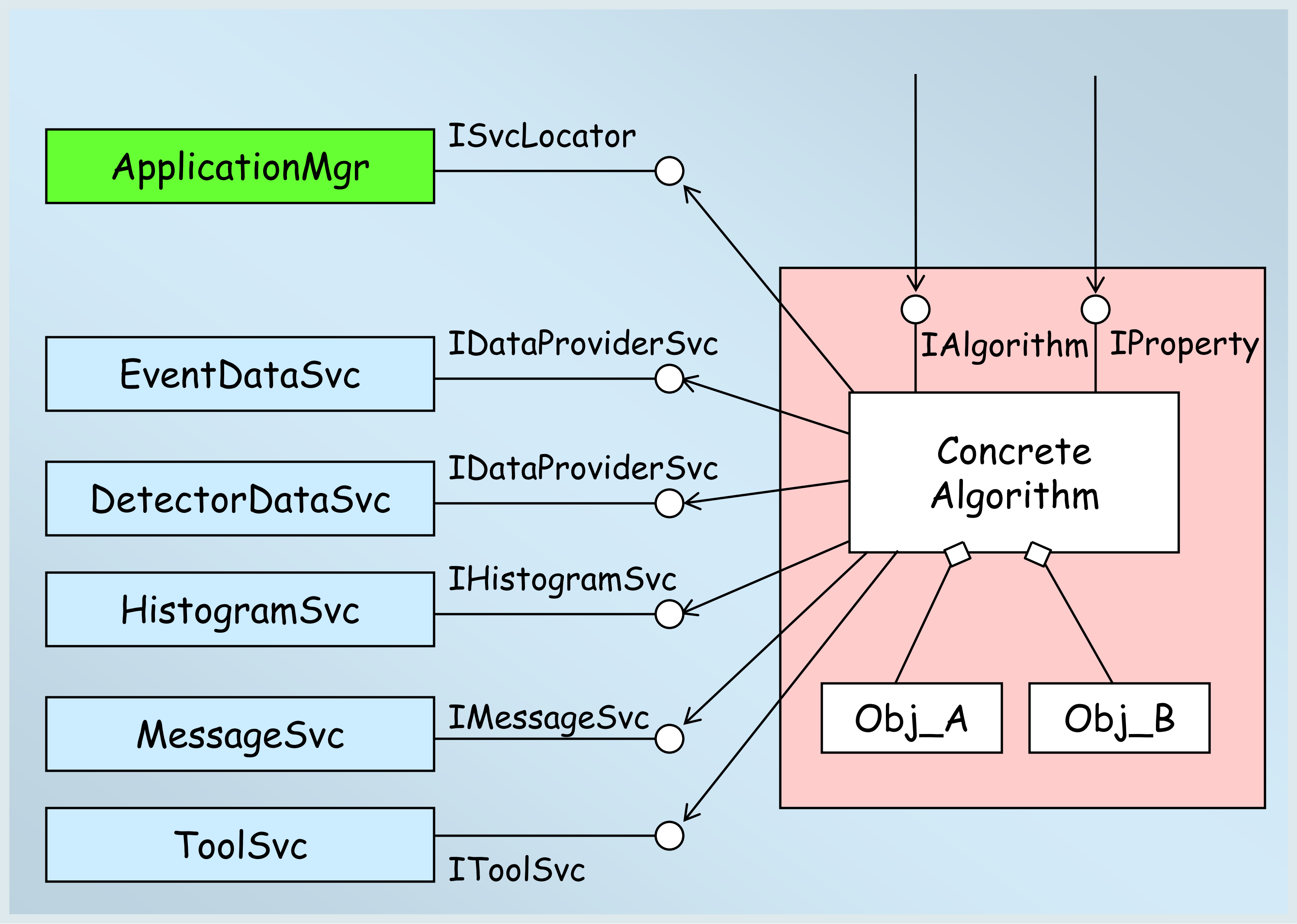
"User code" encapsulated in few specific places



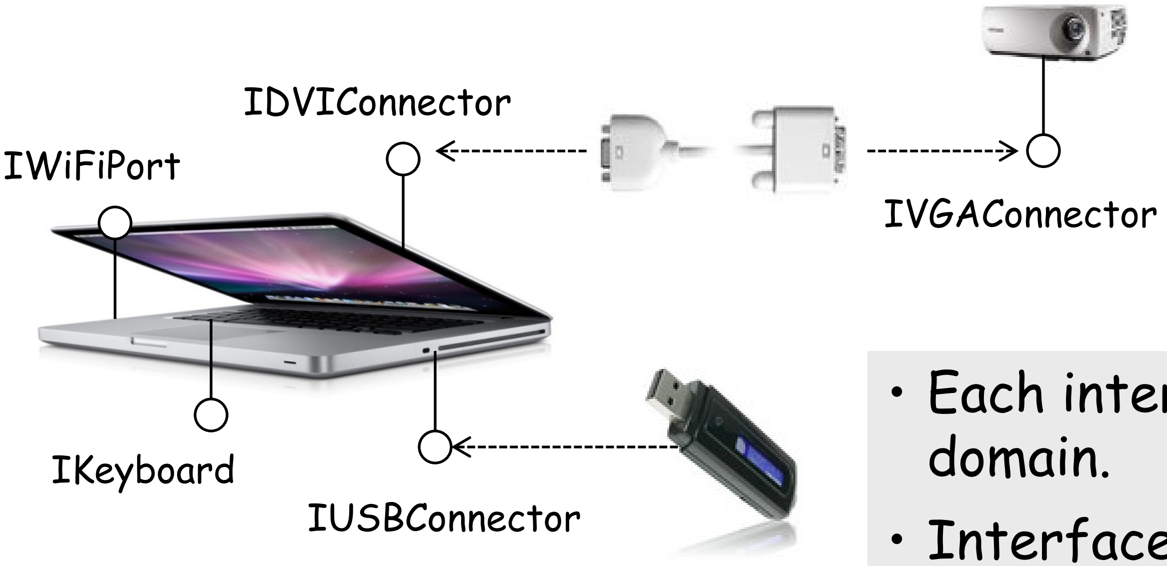
Well defined component "interfaces"



Gaudi Interface model



Laptop 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.

Electrical plug interface model

Don’t define too many interfaces for the same job!

Interfaces in Practice

IMyInterface.h

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

MyDoSomethingTool.h

```
#include “IMyInterface.h”

class MyDoSomethingTool : public GaudiTool,
                          virtual public IMyInterface {
  inline double doSomething( int a, double b )
  { return b * (double)a; }
}
```

ClientAlgorithm.cpp

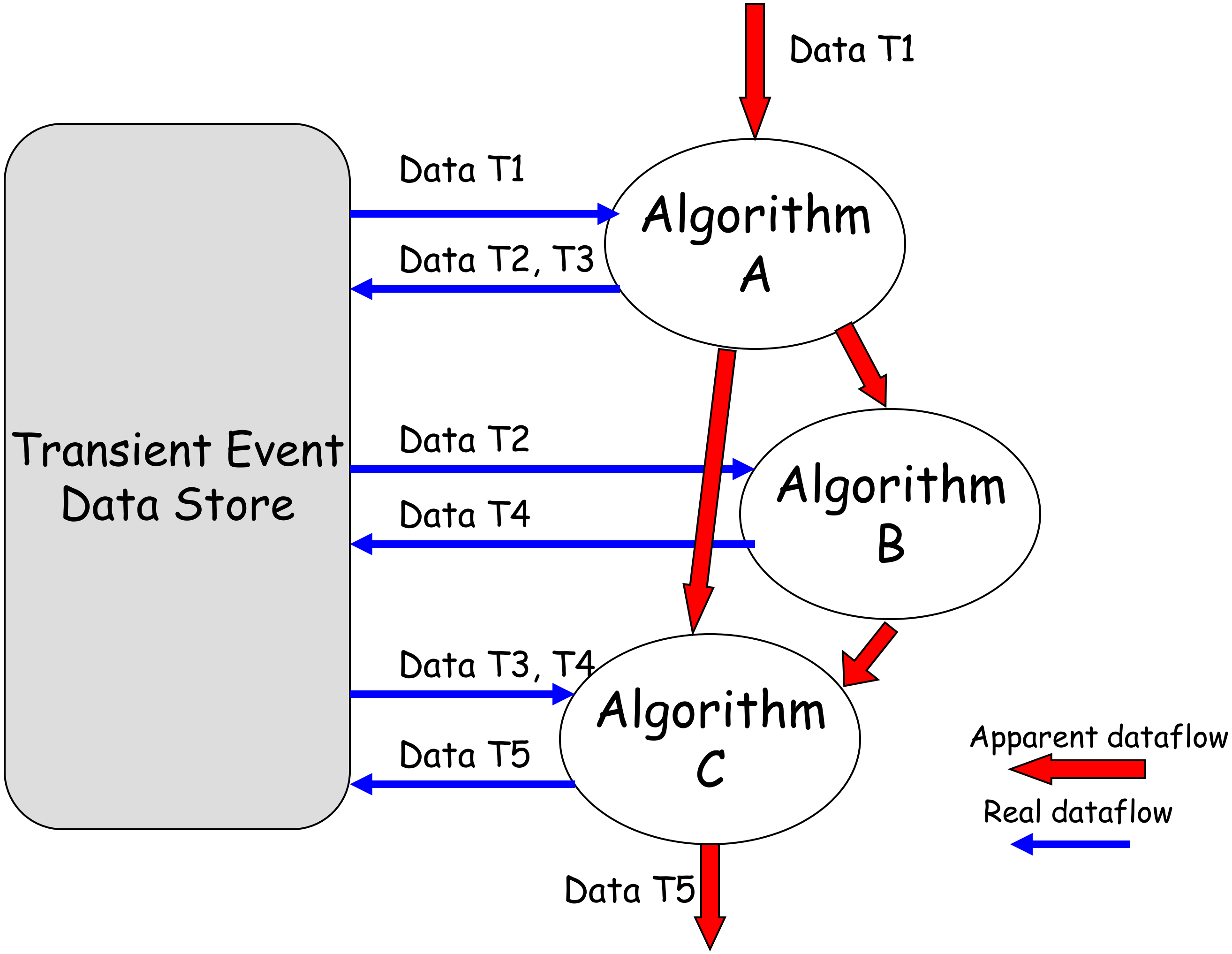
```
#include “IMyInterface.h”

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

Gaudi Component Model

- Algorithms, Converters, Services and Tools are known as Components
  - Self contained objects with behaviour
  - Loaded at run time from component libraries (“plugins”)
  - Interact with the framework and with each other via Interfaces
  - Behaviour controlled by Properties (“Job options”)

- Algorithms are where users interact with the framework
  - Implement three methods invoked by the framework
    - initialize() called once at the start of the job
    - execute() called in the event loop, once per event
    - finalize() called once at the end of the job
  - Communicate with each other via the Transient Event Store

Algorithm & Transient Store



Tools and Services

- Algorithms have some limitations
  - Only called once per event, only share data via TES
    - □ What about private data?
- Tools are components providing algorithmic functionality
  - Callable many times per event, through specific interface and with arguments
    - □ IBdlTool, IDistanceCalculator, IHltSummaryTool, IMassVertexFit, IProtoParticleFilter, ISTReadoutTool, ITrackFitter….
  - Can be private: owned and accessible only by the component creating it
  - Or public: owned by the framework and accessible globally
- Services:
  - Similar to public tools, created by framework to provide global functionality
    - □ Job Options Service, Message Service, Particle Properties Service, Event Data Service, Histogram Service, Ntuple Service, Detector Data Service, Magnetic Field Service, Random Number Service, Persistency Services…

Data

- Data Object
  - □ Atomic data unit (visible and managed by transient data store)
  - □ No algorithmic code
- Transient Data Store
  - □ Central service and repository for data objects
  - □ Manages data location, life cycle, load on demand, …
- Data Converter
  - □ Provides explicit/implicit conversion from/to persistent data format to/from transient data

Gaudi Product Sheet

- Current release
  - v22r1 (Feb 2011)
- Supported Platforms
  - Scientific Linux (CERN) 5 & gcc 4.3 on x86_64 and i686
  - Scientific Linux (CERN) 5 & icc 11 on x86_64
  - Windows XP & VisualC++ 9 on i686
  - (gcc 4.3 with OSX 10.6)
- Web address.
  - http://cern.ch/proj-gaudi/

Documentation

- Gaudi User Guide
  - A 220 pages document targeted to end-users
- FAQ: Mixture of Gaudi and LHCb specific topics
  - □ https://twiki.cern.ch/twiki/bin/view/LHCb/FAQ/LHCbFAQ
- C++ Documentation (generated from code)
  - Doxygen:
    - □ Uses special comments in code, e.g. Tutorial solutions
    - □ http://cern.ch/proj-gaudi/releases/latest/doxygen/
  - Lbglimpse: indexed search within released code
    - □ Lbglimpse <search string> <Project> <version>
      - □ Lbglimpse IJobOptionsSvc Gaudi v22r1
- Self help mailing lists:
  - □ lhcb-soft-talk@cern.ch, gaudi-talk@lists.bnl.gov