
The GAUDI Framework



Pere Mato, CERN
15th February 2000

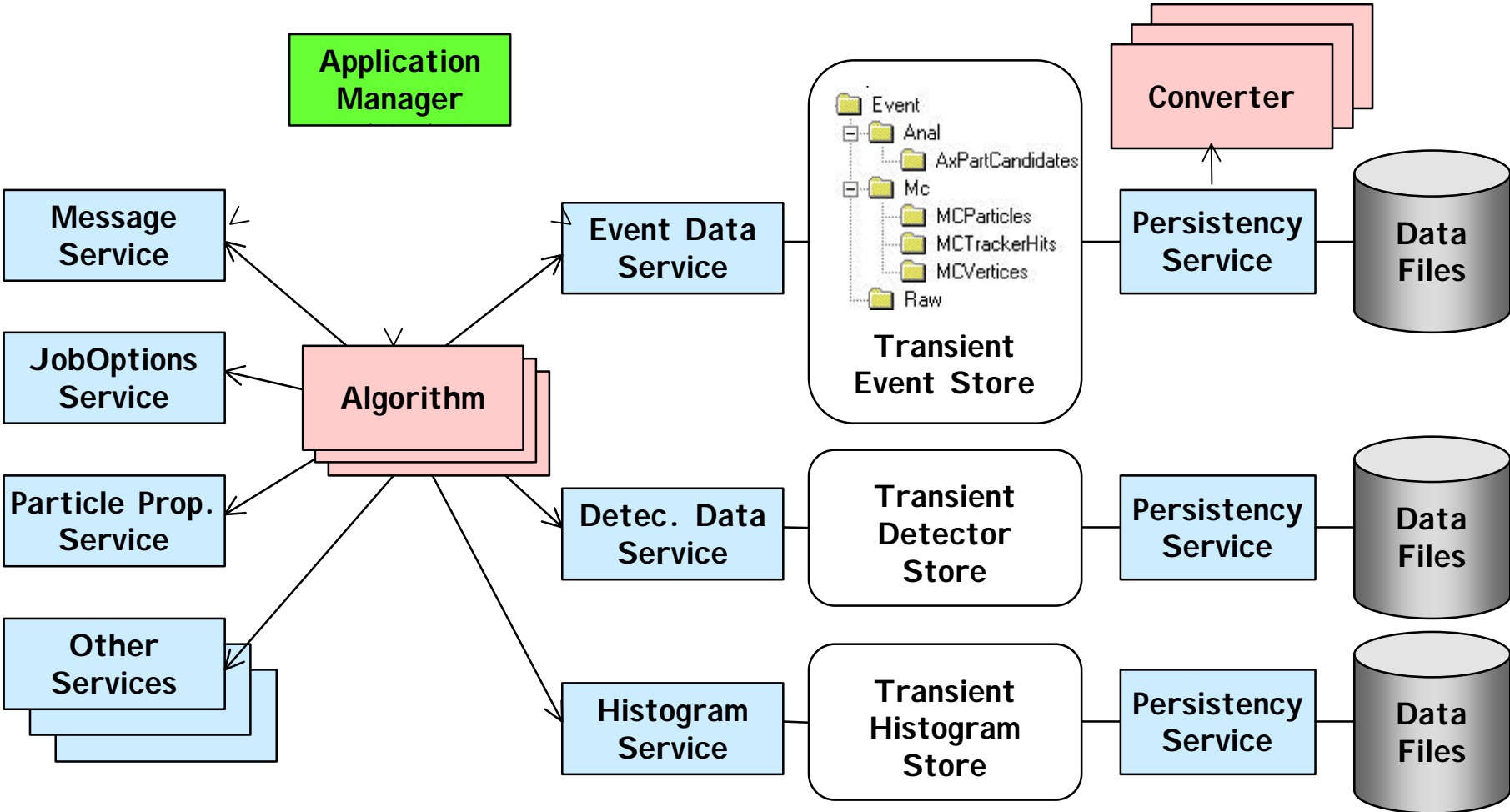
Outline

- ◆ Followed Strategy
- ◆ Architecture (selected issues)
- ◆ Project History
- ◆ Possible Collaboration

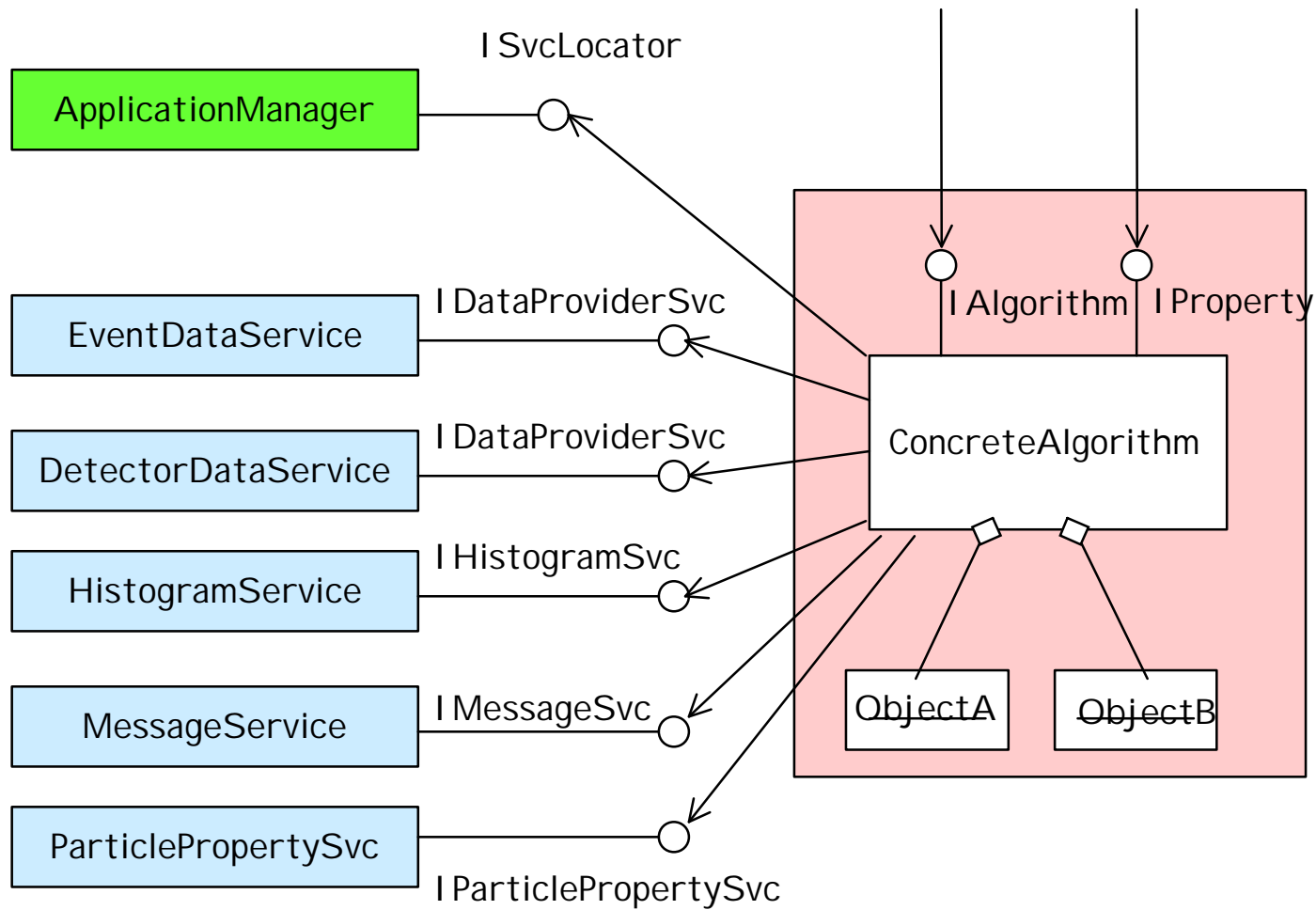
Followed strategy

- ◆ Start with small design team of 6-8 people
 - architect, librarian, domain specialists with design/programming experience
- ◆ Collect User Requirements and use-cases
- ◆ Establish basic criteria for the overall design
- ◆ Make technology choices for implementation of initial prototypes
- ◆ Incremental approach to development.
 - Release every ~4 months.
 - Releases accompanied by complete documentation
 - Development cycle driven by the users: priorities, feedback, etc.
- ◆ Strategic decisions after thorough design review (~1/year)

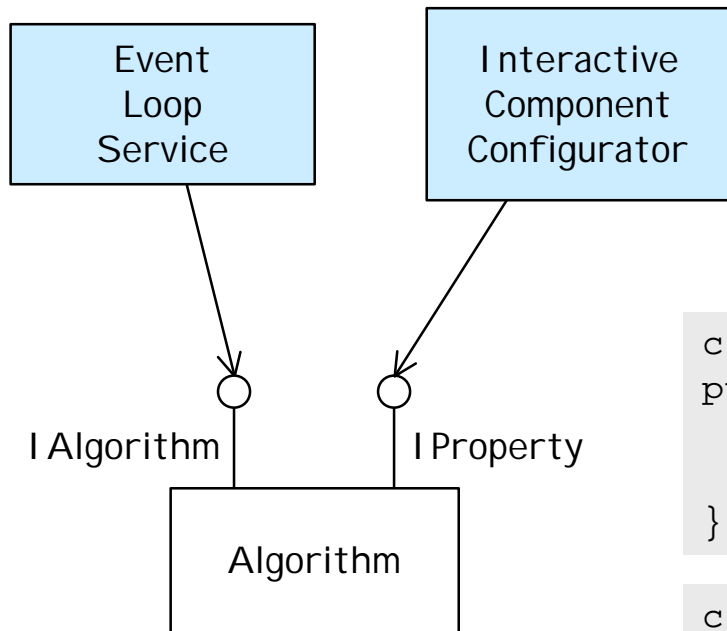
GAUDI Architecture



Interface Model



Interface Model (2)



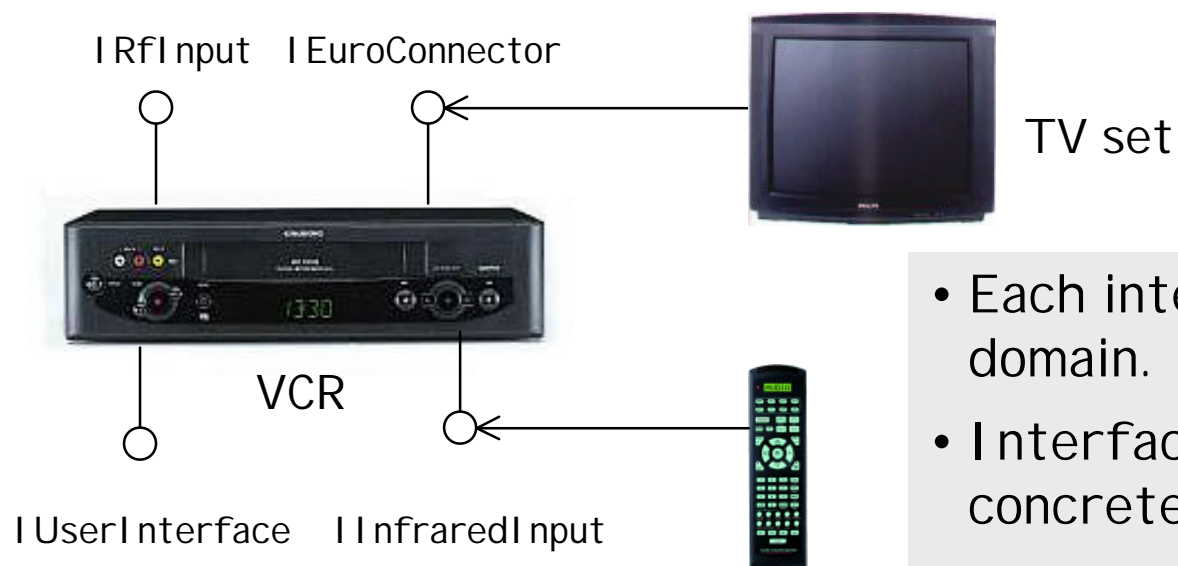
```
class IAlgorithm : virtual public IInterface {
public:
    virtual StatusCode initialize() = 0;
    virtual StatusCode execute() = 0;
    virtual StatusCode finalize() = 0;

    virtual const std::string& name() const = 0;
    virtual StatusCode sysInitialize() = 0;
    virtual StatusCode sysFinalize() = 0;
};
```

```
class IProperty : virtual public IInterface {
public:
    virtual StatusCode setProperty(const Property& p) = 0;
    virtual StatusCode getProperty(Property *p) const = 0;
};
```

```
class Algorithm : virtual public IAlgorithm,
                  virtual public IProperty {
public:
    ...
}
```

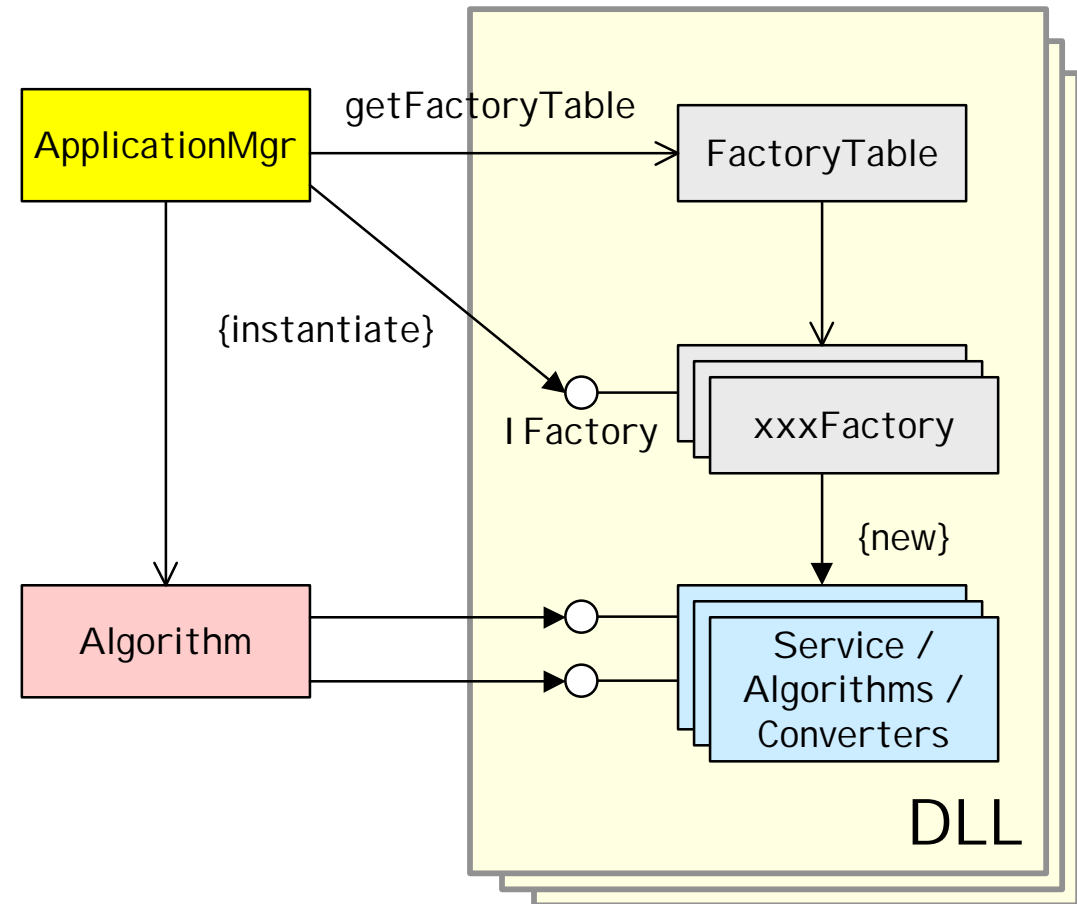
VCR Interface model



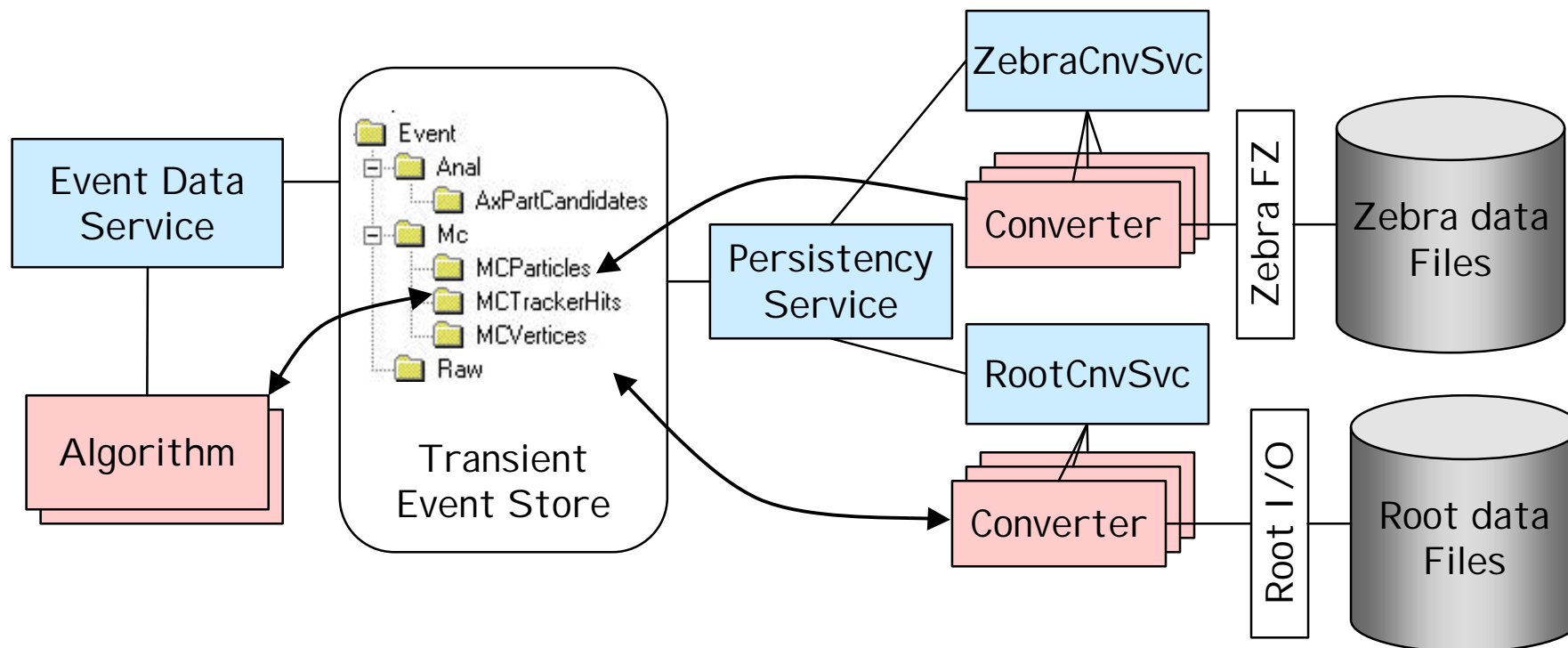
- Each interface is specialized in a domain.
- Interfaces are independent of concrete implementations.
- You can mix devices from several constructors.
- Application built by composing.
- Standardizing on the interfaces gives us a big leverage.

Factories & Dynamic Loading

- **Plug-and-Play**
- **Factory pattern** to avoid using concrete implementation.
- Run-time discovery of components.
- Only pure abstract classes (**interfaces**) are accessible.

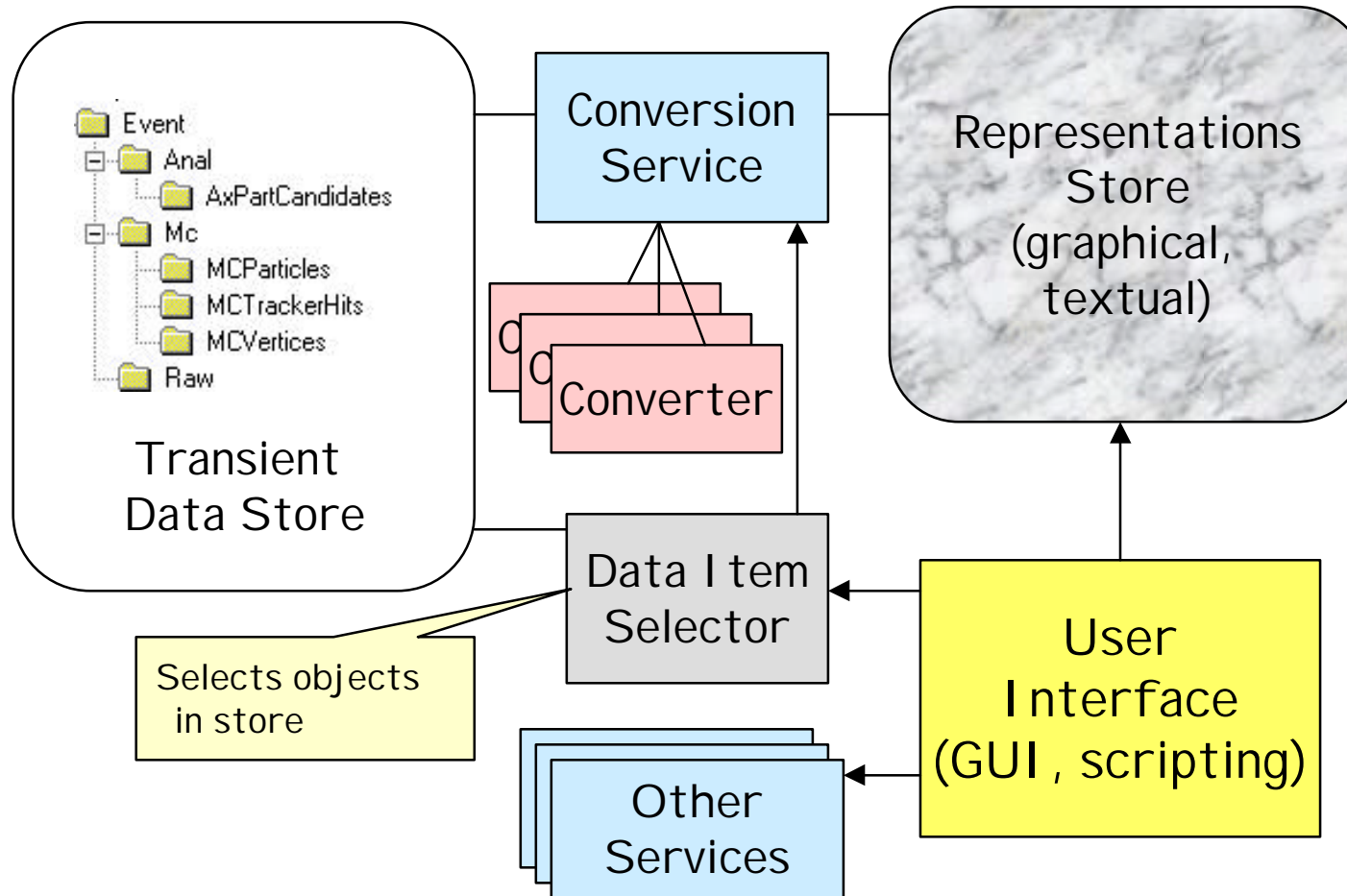


Persistency






- Various technologies available in the same program: Objy, Root, Zebra,...
- **Converters** transform objects from one representation to another.

User Interaction / Visualization



Project History

- ◆ Sep '98 - architect appointed, design team (6 people) constituted
- ◆ Nov 25 '98 - external architecture review
 - objectives, architecture design document, URD, scenarios
-  ◆ Feb 8 '99 - first GAUDI release
 - first software week, presentations, tutorials
 - plan second release (together with users)
 - expand GAUDI team
-  ◆ May 30 '99 - second GAUDI release
 - second software week, plan third release with users, expand team.
-  ◆ Nov 23 '99 - third GAUDI release and software week
 - plan deployment for production applications
- ◆ Spring '00 - second external review

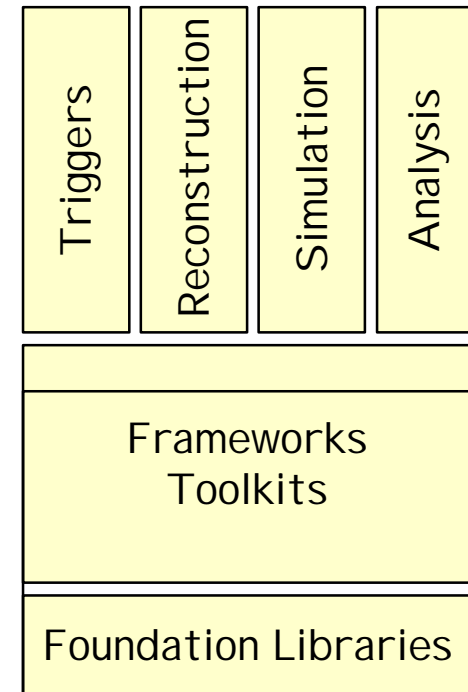
Possible Collaboration

◆ Scope

- Common foundation libraries
- Common interface model
- Common frameworks (interfaces + basic services)
- Different *Event Model* and *Algorithms*
- Different Applications

◆ Benefits

- Better design
- Sharing development of basic infrastructure services (higher quality)
- CERN/IT efforts better focussed (single request may fulfill more than one experiment) (AIDA project)
- Better communication (same vocabulary)



Possible Collaboration (2)

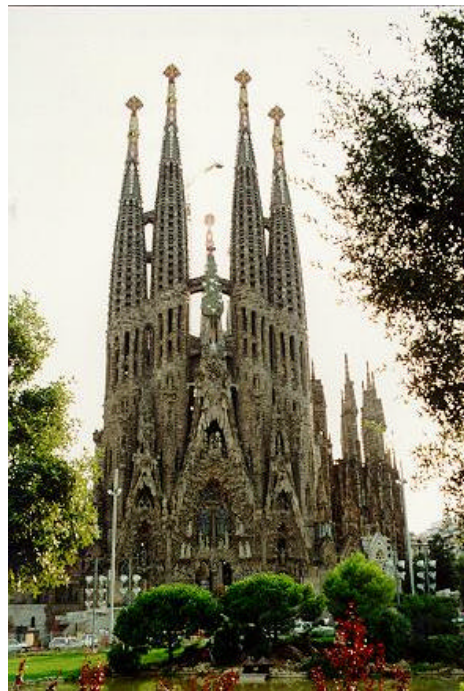
- ◆ Disadvantages

- Less freedom
- Needs more formality (change procedures, upgrades, etc.)
- It may fail

- ◆ Practical aspects

- Regular meetings, workshops, ...
- Mailing lists and other collaborating tools
- Common code repository ?

Discussion



Antoni Gaudí
Barcelona (1852-1926)

