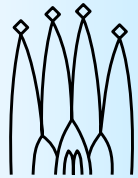


*CHEP2009 - Prague, March 24, 2009*

# *Recent Developments in the Gaudi Software Framework*

Marco Clemencic

marco.clemencic@cern.ch

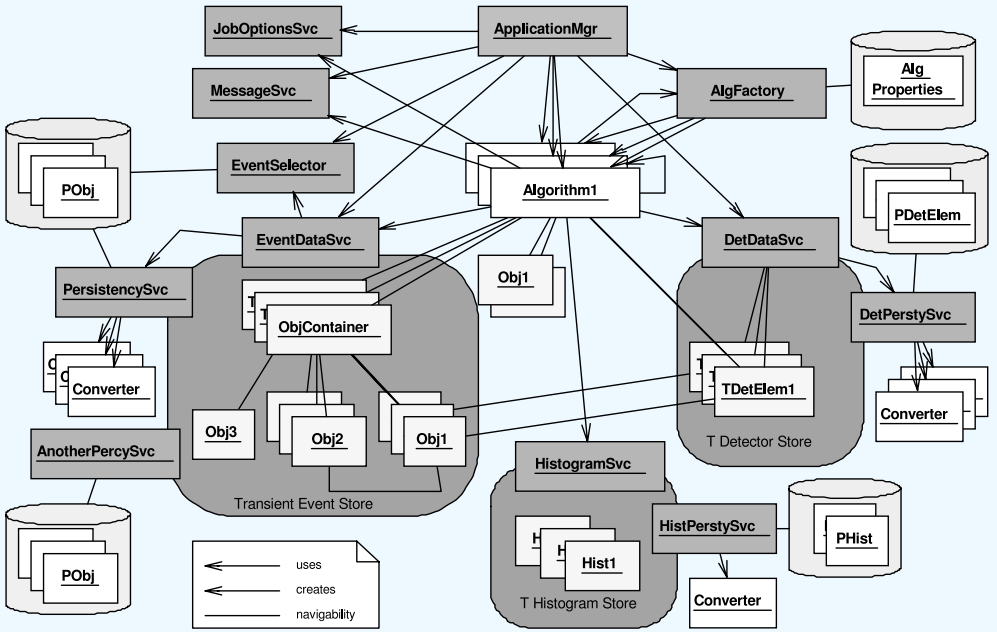
# Overview

- A bit of history
- Issues
- Developments
- Status and Conclusions

# *A bit of history*

# Once upon a time…

- First documents Q4 1998
  - Scenario and Requirements
  - Architecture Design Document



# Presented at CHEP

- **CHEP 2000, Padova, Italy**
  GAUDI - A Software Architecture and Framework for building HEP Data Processing Applications
  Barrand G. *et al.*
- **CHEP 2001, Beijing, China**
  Status of the GAUDI event-processing framework
  Cattaneo M. *et al.*

# A shared framework

- Designed an implemented by LHCb
- Adopted and extended by Atlas in 2000
- Used by other experiments:
  - GLAST
  - HARP
  - DayaBay
  - MINERvA
- Main developments: LHCb and Atlas

# Continuous evolution

- Few releases per year
- Mainly bug-fixes and small improvements
- Few big steps
  - Plug-in loading based on Reflex/ROOT

    *Pere Mato*
  - Job configuration via Python options

    *Wim Lavrijsen, Martin Woudstra, Sebastien Binet, Pere Mato*
  - Extended state machine

    *Marco Clemencic*
- November 2008: mini-workshop
  *plan for improvements and consolidation*

# *Issues*

# Issues

- Interfaces limitations
  - Inheritance not allowed by the design
  - `queryInterface` implementation error prone
- Finalization order of services
  - ToolSvc and AuditorSvc may be needed by other services (not easy to decide what to finalize first)
- New state machine

# *Proposals/Developments*

# Interfaces

- Old implementation

```
// Declaration of the interface ID
static const InterfaceID IID_IMessageSvc("IMessageSvc",
                                         1 , 1);
// Interface (abstract class)
class IMessageSvc: virtual public IInterface {
public:
  /// Retrieve interface ID
  static const InterfaceID& interfaceID() {
    return IID_IMessageSvc;
  }
  // ...
};
// Implementaion
class MessageSvc: public Service,
                  virtual public IMessageSvc {
  // Implementation of IInterface::queryInterface()
  virtual StatusCode  queryInterface(
                        const InterfaceID& riid,
                        void** ppvUnknown);
  // ...
};
```

# Interfaces cont.

- Simplified with some template meta-programming techniques and cpp macros

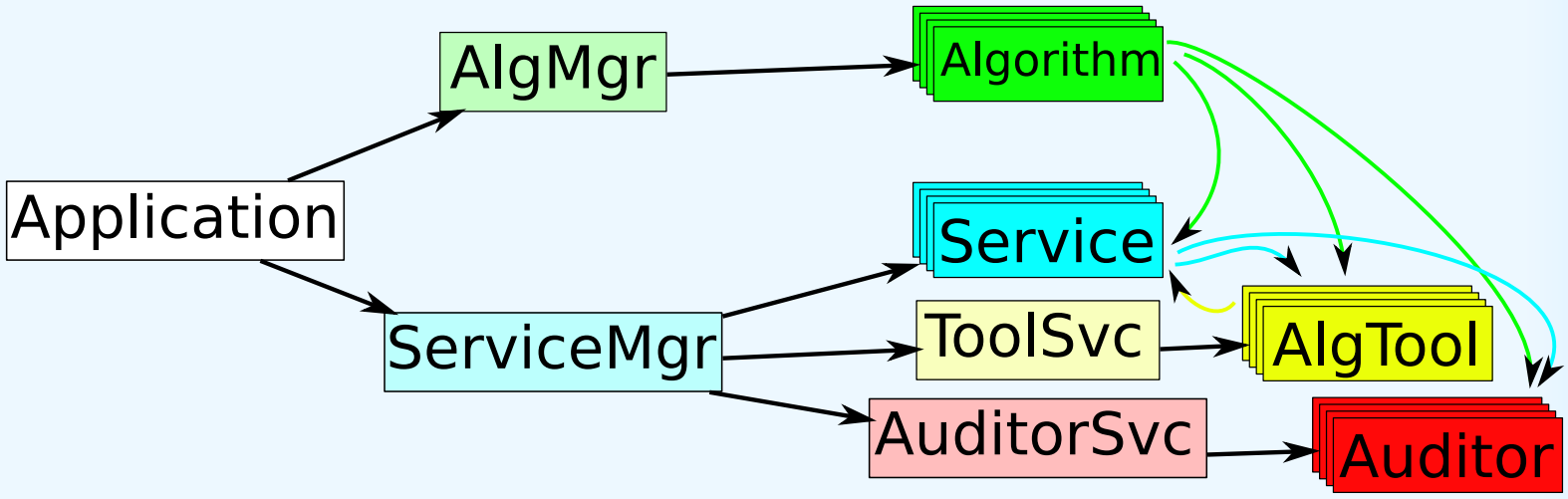
```
// Interface (abstract class)
class IMessageSvc:
  virtual public extend_interfaces1<IInterface> {
public:
  /// InterfaceID
  DeclareInterfaceID(IMessageSvc,2,0);
  // ...
};
// Implementaion
class MessageSvc: public extends1<Service,
                                  IMessageSvc> {
  // queryInterface() is automatic
  // ...
};
```

# Interfaces cont.

- The templated base classes allow to implement automatic book-keeping of the inheritance tree of the interfaces.
- Automatic implementation of `queryInterface` and other methods that give access to the list of interfaces.
- Implementation of common methods in a single place (e.g. reference counting).

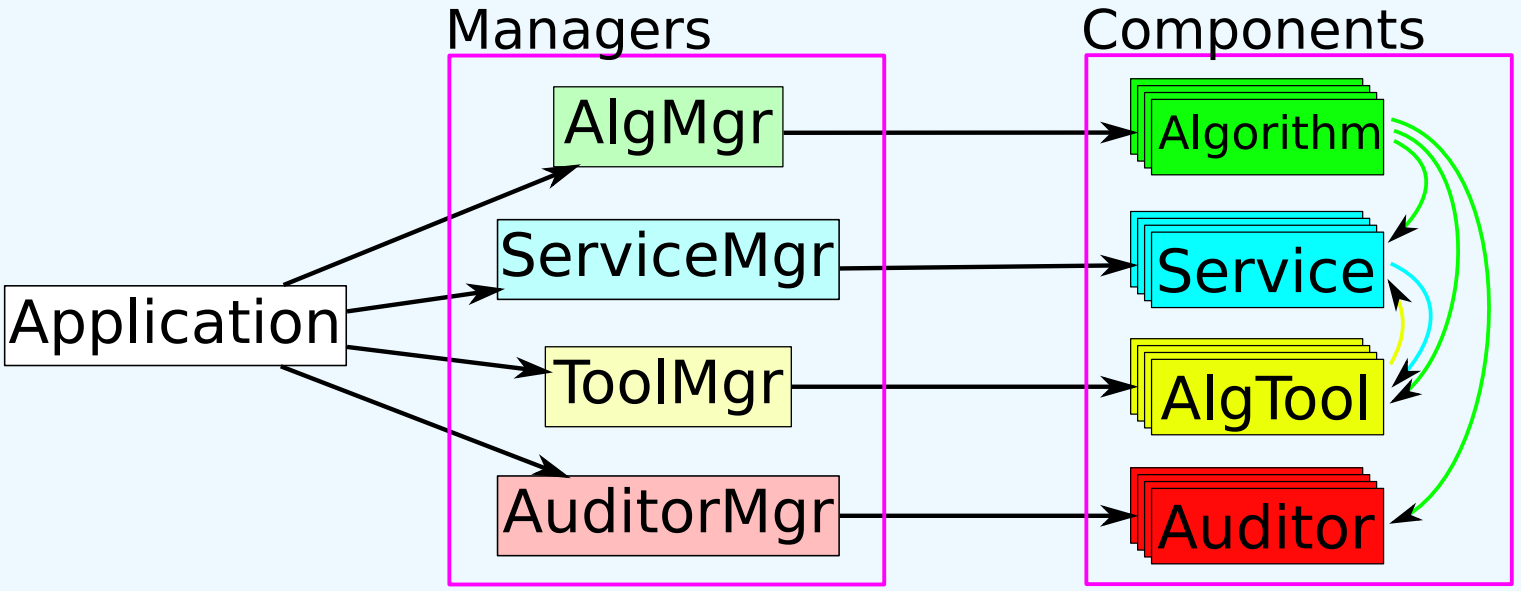
# Finalization order of services

- We have special services:
  *managers of components*
  - `ToolSvc` → `AlgTools`
  - `AuditoSvc` → `Auditors`
- Their life-time cannot follow the rules of services



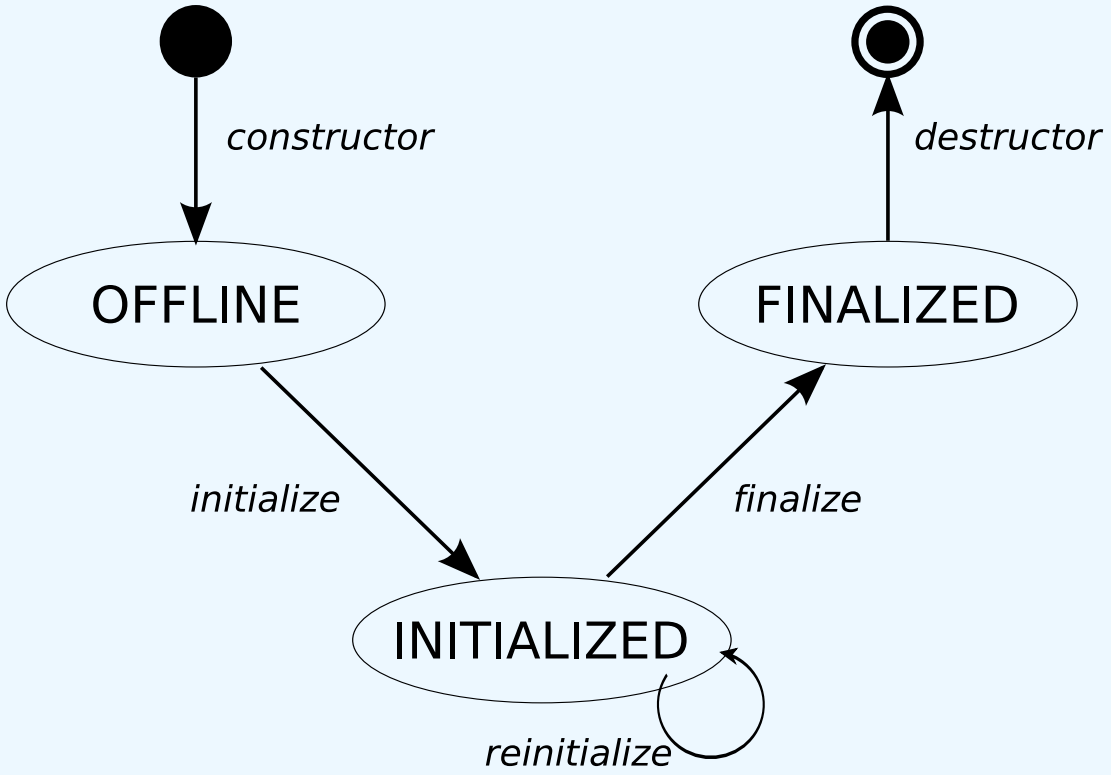
# Finalization order of services cont.

- Adding the concept of *managers*



# New state machine implementation

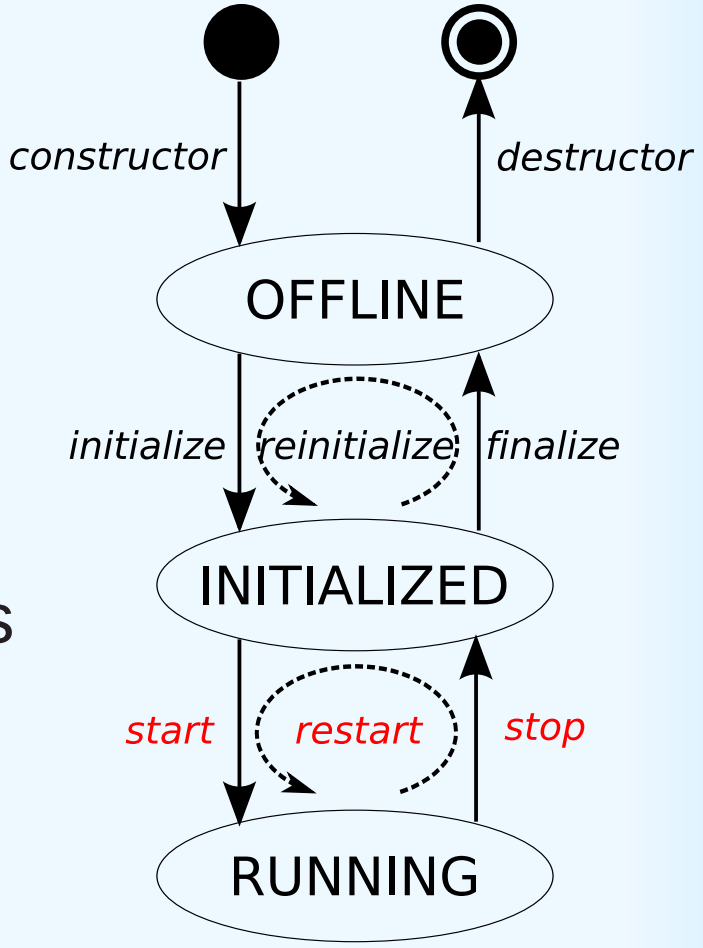
- Original implementation (for services)




- Does not fit the needs of Online system

# New state machine implementation cont.

- New transition added to accommodate the separation between server and client start up
- To be reviewed
  - another transition
  - redefine the re* methods



# *Conclusions*

# Conclusions

- We need both stability and changes
  - No functional changes: the principles are good
  - Introduce backward-incompatible changes with hacks to allow old code to work
- The consolidation work not yet completed
  - Will need a couple of releases
  - Some areas need further discussions with the users

Thanks to all the collaborators for the help and the useful feedback.