The MICO

CORBA Components

Project

Frank Pilhofer
fp@fpx.de

Sponsored by

Executed by

Motivation for CORBA Components

- CORBA 2: A server . . .
  - is a monolithic program that
  - operates by Request and Response

$\Rightarrow$ But these are relicts from the batch-programming of mainframes!

- CORBA 3:
  - Lightweight, modular, reusable Components
  - Programming by assembly of existing Components

Definition of “Component”

- Existing definitions are not very helpful:

> *A Component is a self-contained unit of software code consisting of its own data and logic, with well-defined connections or interfaces exposed for communication.*

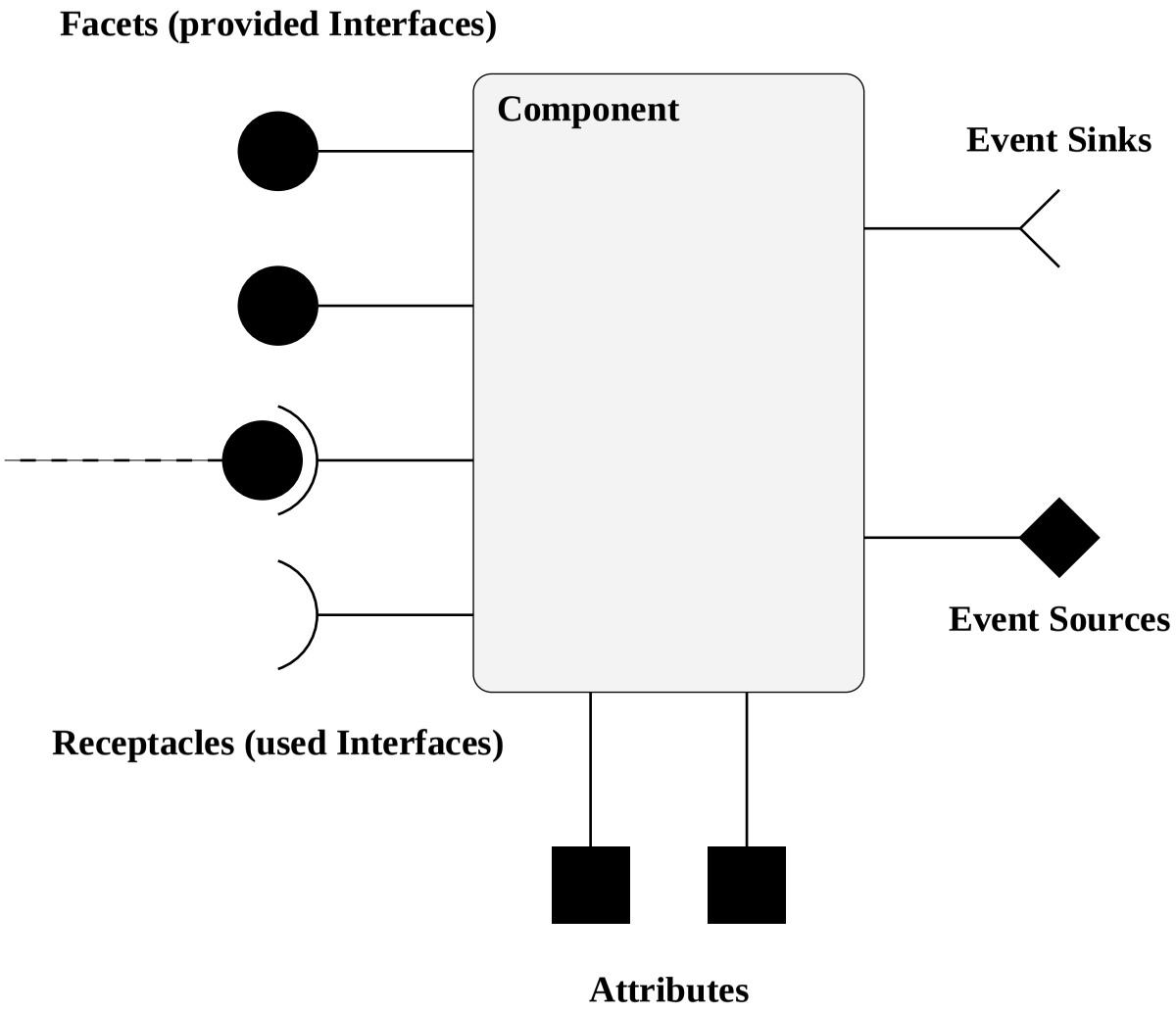
OMG Tutorial, Edward Cobb

> *A Component is an object.*

Arno Puder

- Why Components, if we already have objects?

External view of a CORBA Component



CORBA Components

- Not a new paradigm, but a specialization of objects
- Focus on the interfaces to other components
- Graphical handling of Components is possible:
  - Connecting facets to receptacles
  - Connecting event sources and event sinks
  - Configuring initial attributes

$\Rightarrow$ Goal: Application *assembly* rather than *development*

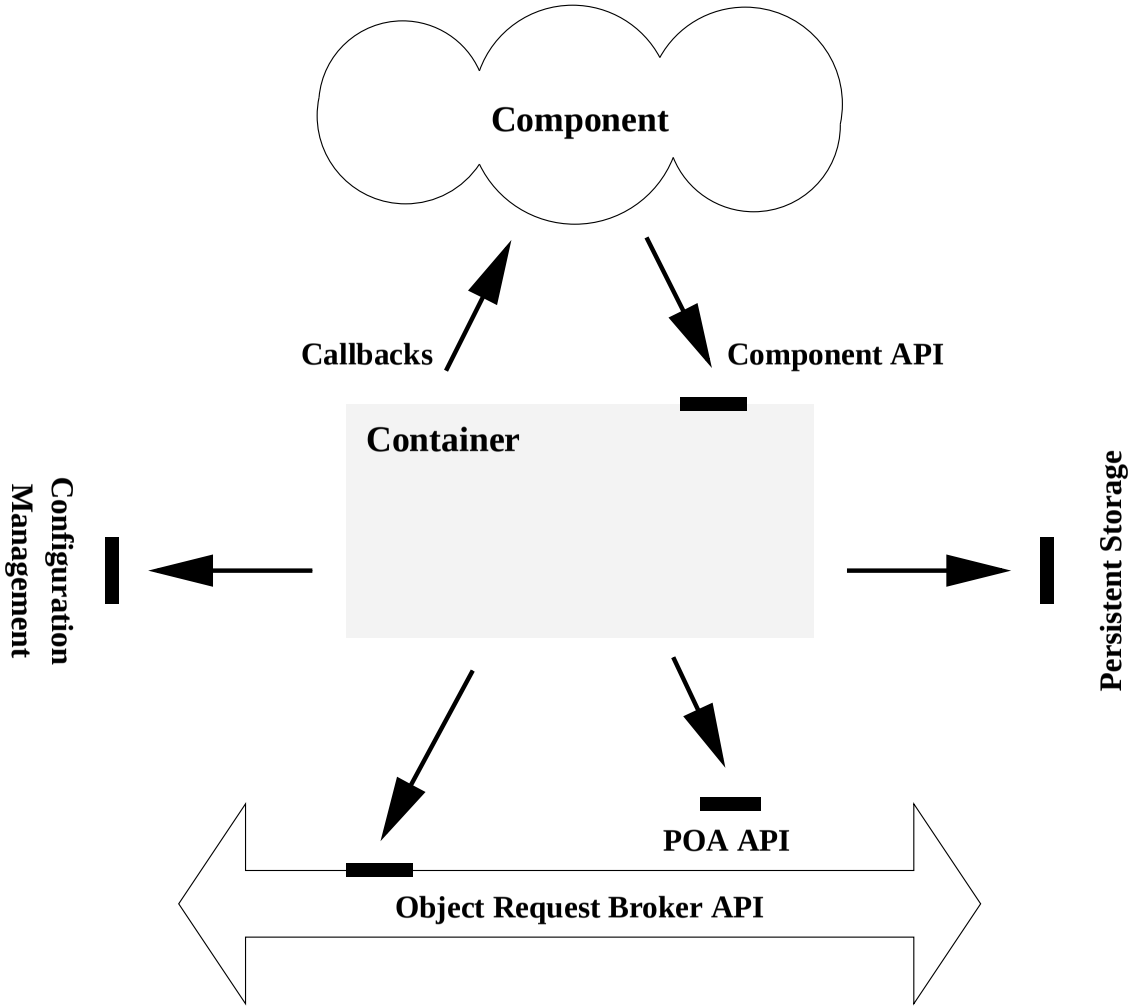
Container

- Each server is re-inventing lots of wheels:
  - Object reference management (via ORB and POA)
  - Persistence (via DBMS or PSS)
  - Configuration management (files)

$\Rightarrow$ Use a “Container” to automate these tasks.

- Pre-configured containers for common applications
- Component development becomes easier, components more lightweight

Container


Component
Callbacks
Component API
Container
Configuration Management
Persistent Storage
POA API
Object Request Broker API

Components Summary

The goal of CORBA Components is twofold:

- Easy assembly of reusable and configurable Components by expressing their interactions with other components

  $\Rightarrow$ similar to Java Beans and ActiveX

- Easy implementation of persistent and transactional Components by introducing Containers

  $\Rightarrow$ analogous to Enterprise Java Beans

- “Buzzword compliance”

Open Questions and Criticism

- “Vaporware,” originally planned for 1999
- Complex, as yet unverified specification
- Some parts still unfinished, such as CIDL and language mappings
- Third, non-orthogonal implementation strategy
- Is the ideal of application assembly ever possible?
  - Will component providers distribute their source code?
  - Non-source code Components are always bound to an ORB, but multiple ORBs per application are not desirable

The MICO CCM Project

- Sponsored by Alcatel
- Results will be included into the MICO distribution under normal GPL/LGPL license
- Support of Basic Components (EJB equivalent) by July 2001
- At the same time evaluation of Extended Components
- Feedback to the OMG if necessary