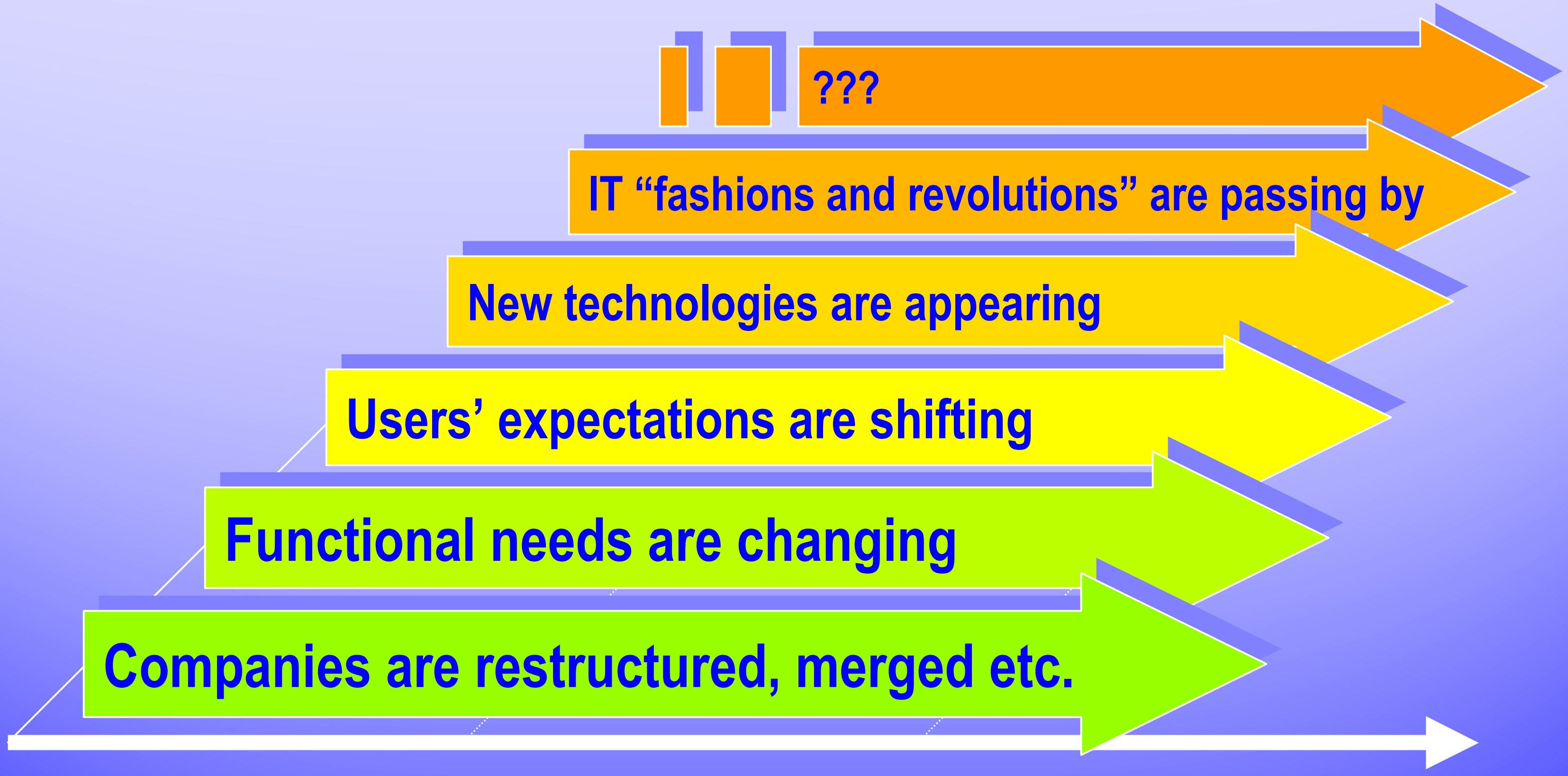
delta
SOFTWARE TECHNOLOGY

Enterprise Application Integration
with
Generated Adaptive Frameworks

by Rüdiger Schilling
smDelta Software
rschilling@delta-software.de

Everything Moves ...



???

IT “fashions and revolutions” are passing by

New technologies are appearing

Users’ expectations are shifting

Functional needs are changing

Companies are restructured, merged etc.

While your EA-Systems are growing ...

Continuous Growth and Integration

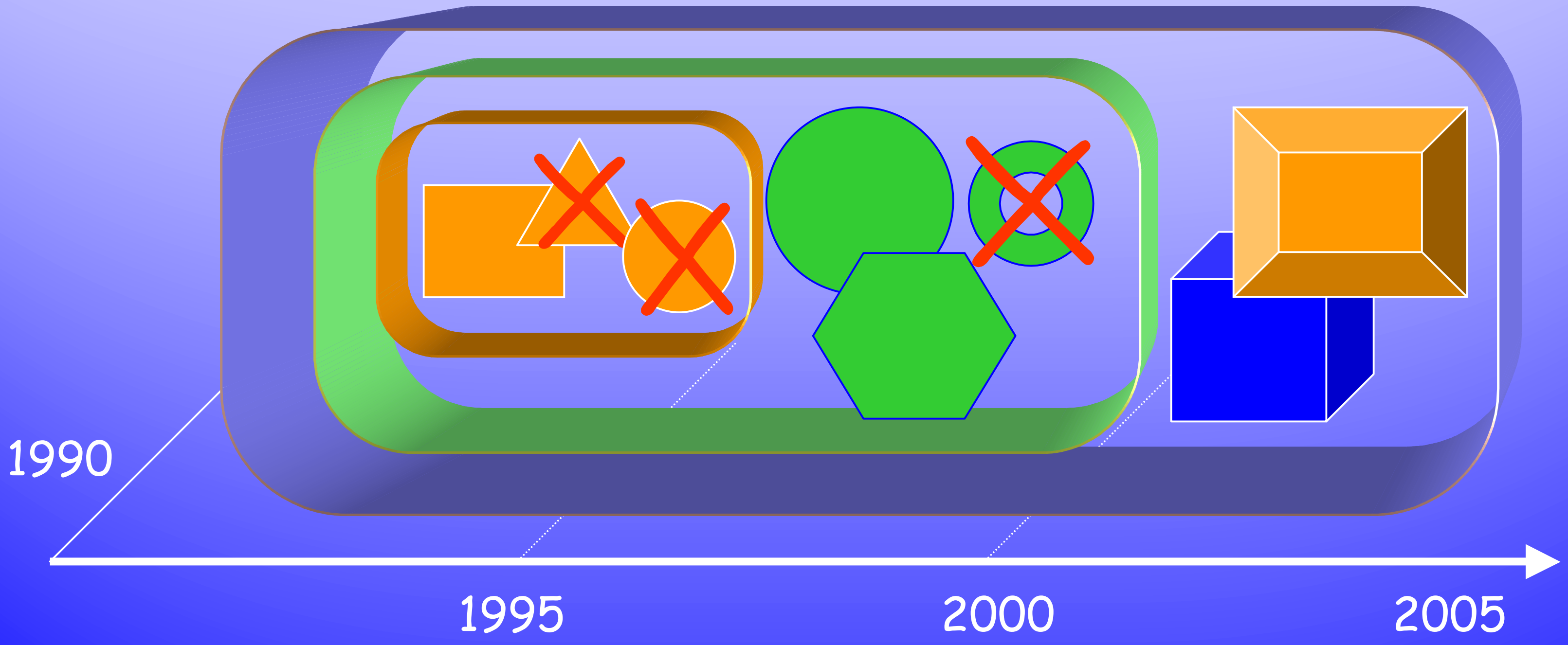
First Apps are bought, build, inherited

These have to be integrated

More Apps are added, some dropped

Next integration takes place

And so on ...



EAI - Not a Single Process

- **Application integration is** not a single process but **a continuous task**
- The application we are developing today will have to be integrated tomorrow
- Integration is not like fixing a bug,

  it is the development principle of the future
- All our experiences tell us:

This is not the last integration

EAI Architecture

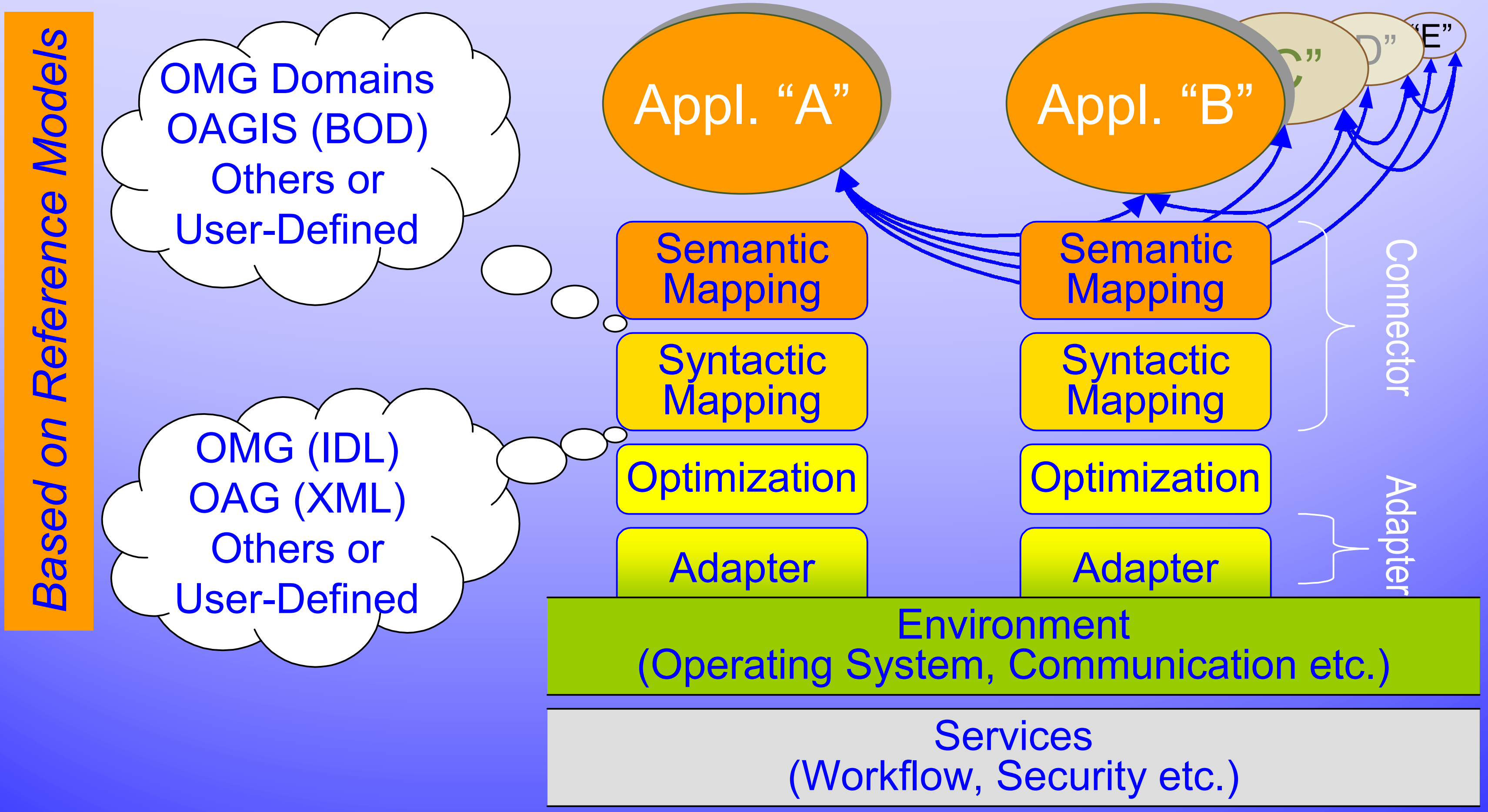
- **Most important goal**:
  Prepare applications for “repeating integration” - since see above: this is not ...!
- Clear, robust modularization - “componentizing”
  - The principles for this are known for a long time and lead to reliable, (re)usable components
- Separation of application functions from technical infrastructure
  - The biggest obstacle to problem-free integration of different subsystems is the mixing of these levels

The Gauge

An Integration Architecture is only good if it is also good for the development of new systems.

Otherwise it is merely a one-way wrapping which will still increase the integration problems in the future

(EA) Integration Architecture



EAI and Components

- Components are for **composition** [C.Szyperski]
- Components are for **reuse**
- Composition and reuse are major issues of EAI
  - Converting applications into sets of well-defined components will allow easy integration now and in future

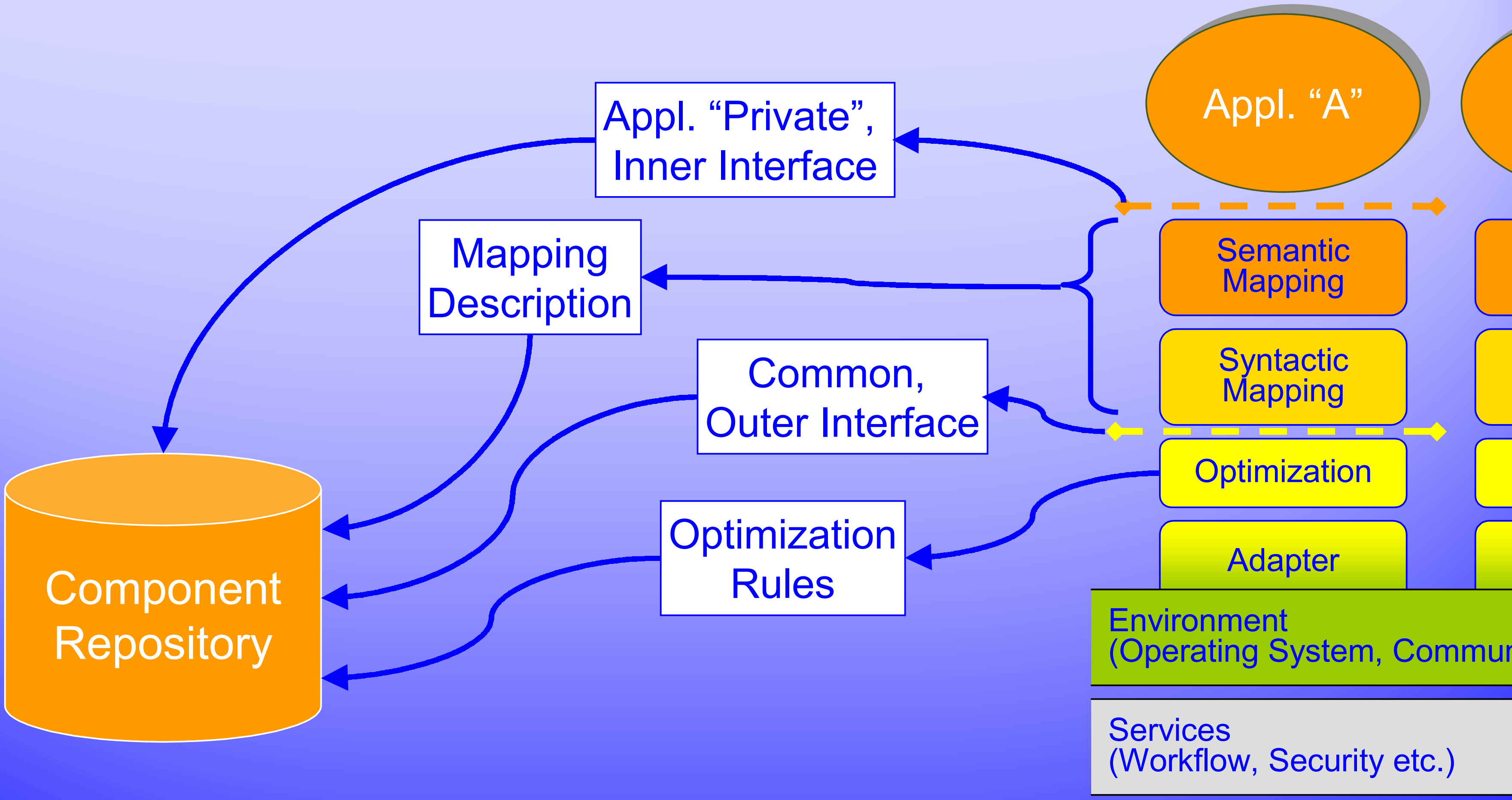
A software component is

- a unit of composition
- with **contractually specified interfaces**
- and **explicit context dependencies** only.

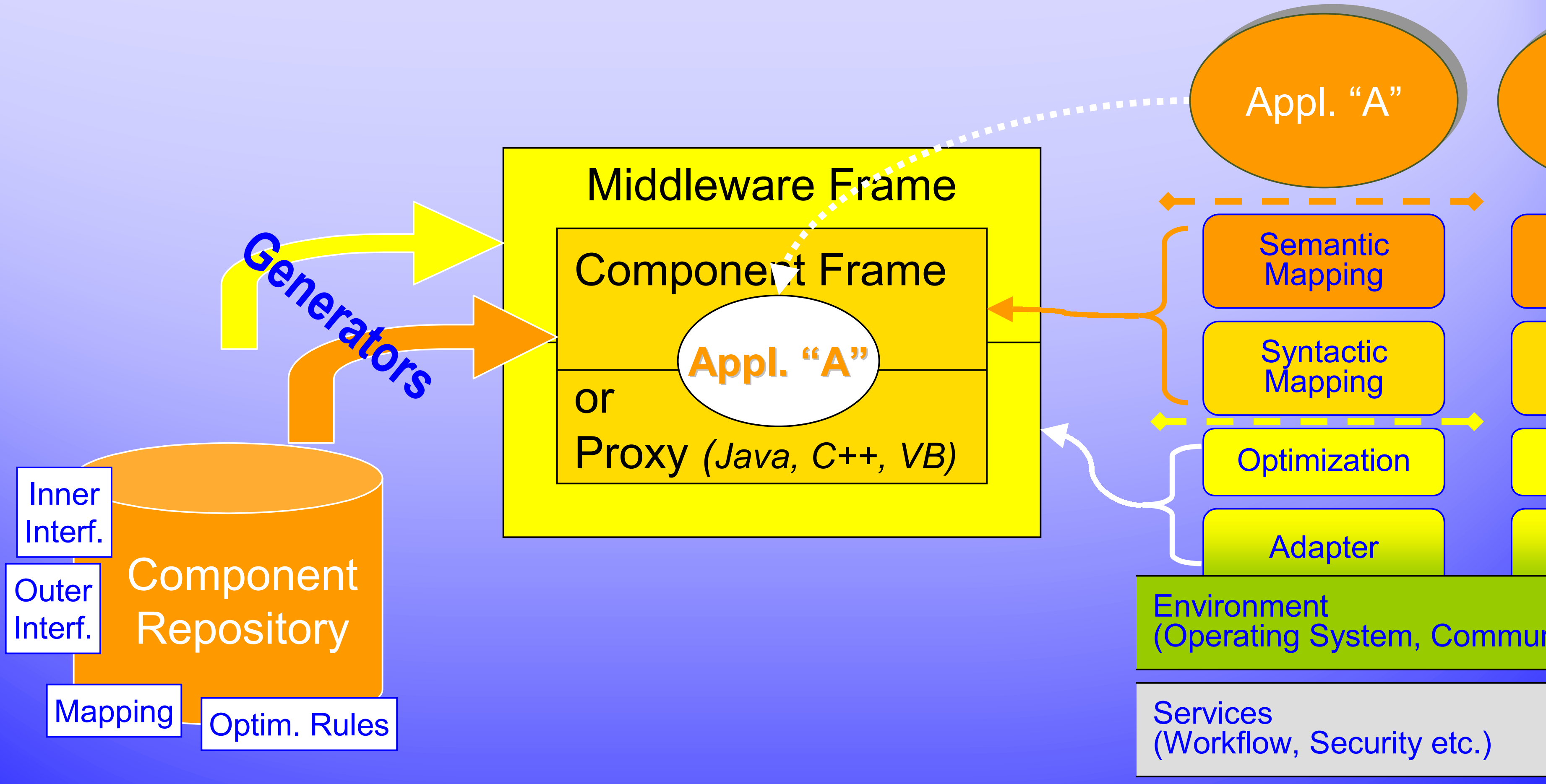
[ECOOP 1996]

The less, the better

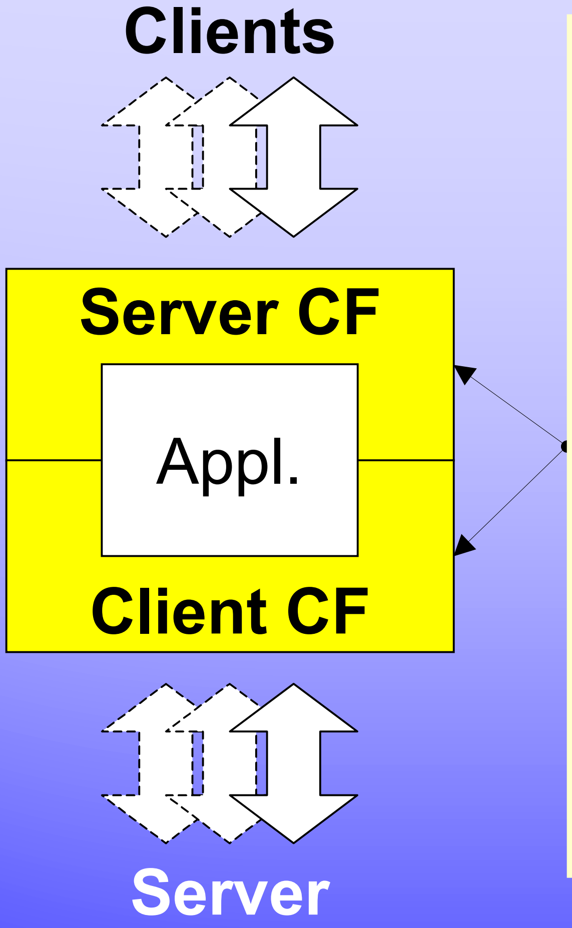
Feeding the Repository



Generating the Framework



Component Frame

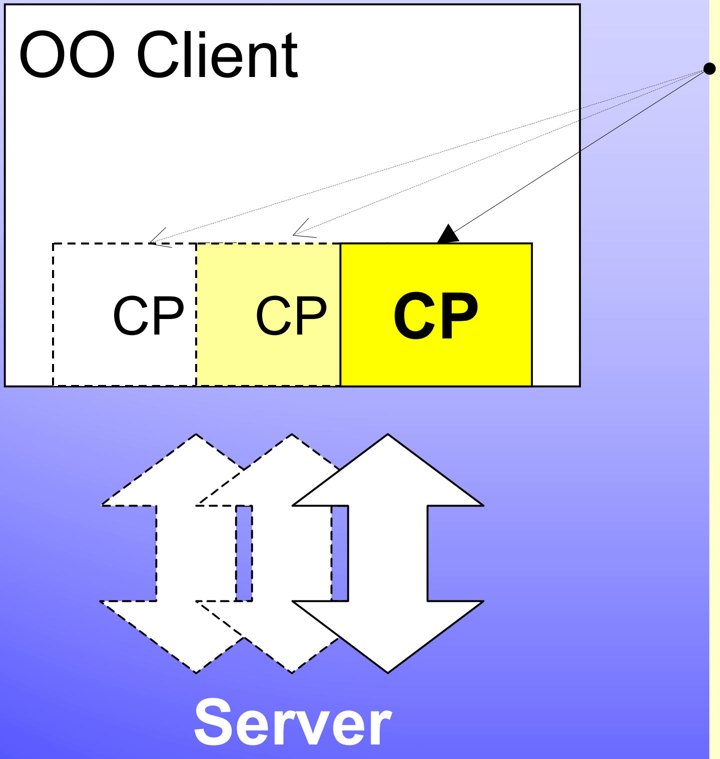


Primary Goal:

Neutral, stable and reliable component interface

- ▲ (Schema) Mapping, internal vs. external view
- ▲ Request/Response packaging
- ▲ Interface versioning
- ▲ State/Workspace management
  (via middleware frame)

Component Proxies



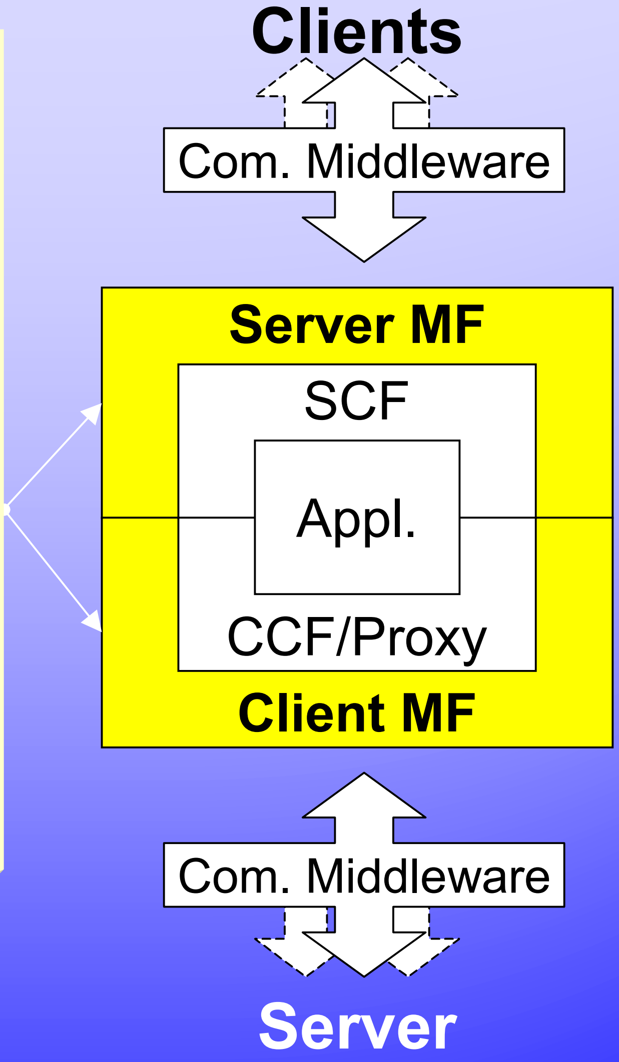
- For OO clients
  - Java
  - C++
  - VisualBasic
- Completely generated class definition and implementation
- Delegation instead of class/interface inheritance
- Client CF-compatible
- Seamlessly integrated in client IDE

Middleware Frame

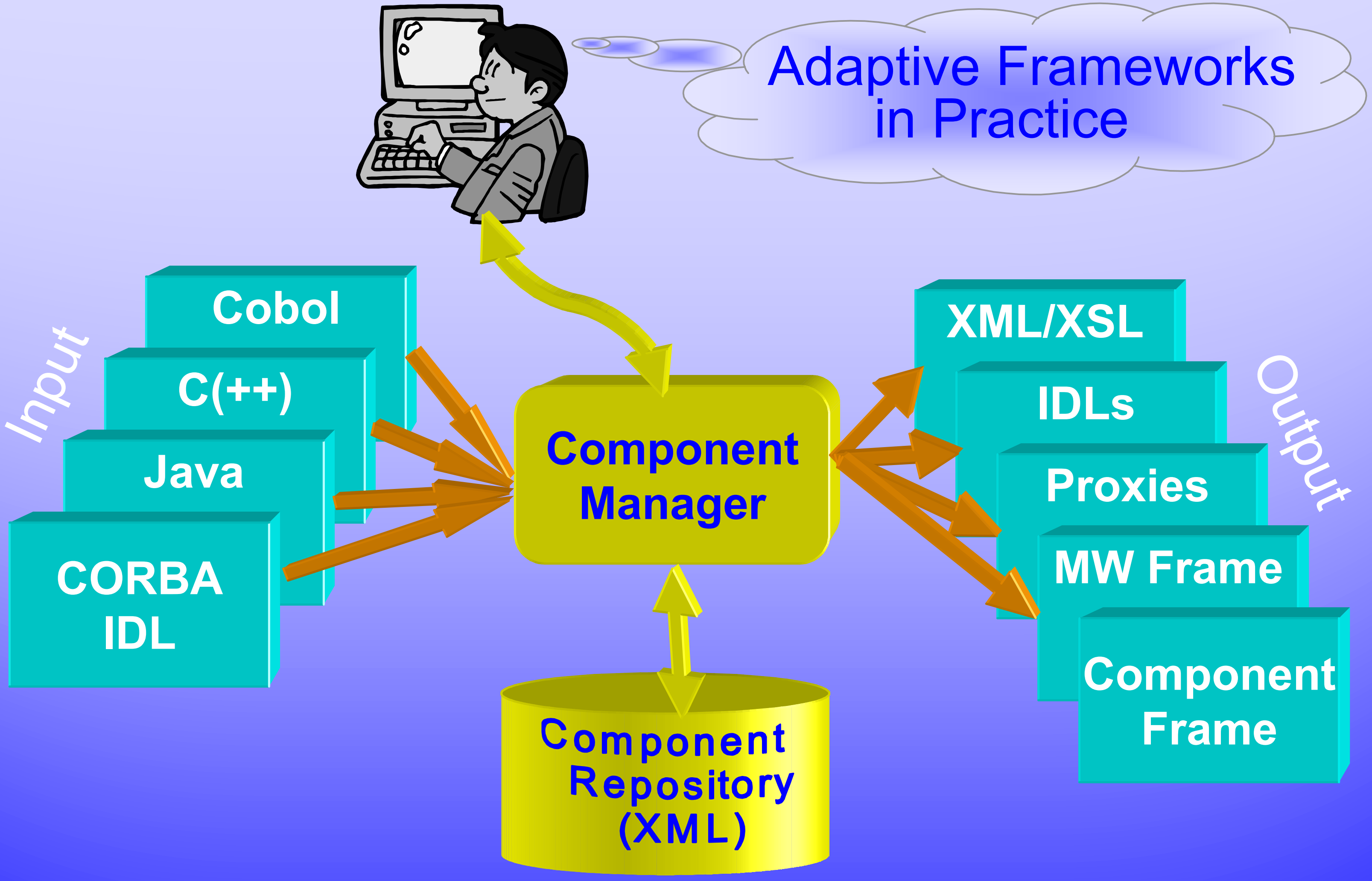
Primary Goal:

Less context dependency

- Linkage to communication middleware
  - OLTP (CPI-C, ATMI,ECI)
  - CORA
  - RMI/IIOP
  - MQS
  - COM
- Linkage to transaction services
- IDL generated



Score/Integration Suite



Summary

- Integration is a continuous task
  - The applications we are building today will have to be integrated tomorrow
- Base concepts of an EAI architecture are:
  - Proper modularization and separation of application functions from technical infrastructure
- Adaptive frameworks - a layered concept of frames and proxies
  - repositories and generators help to produce reliable component structures
- **SCORE/Integration Suite** of smDelta Software

to put theory into practice

The Last Page

- The company “smDelta Software Entwicklung GmbH”, founded in 1994, is the head of “Delta Software Technology” group
- It has its seat in Germany with subsidiaries and distributors in Central Europe
- Several hundred companies are using smDelta’s development tools to build enterprise-wide applications
- Delta™ and SCORE™ are European Trademarks of smDelta Software

delta
SOFTWARE TECHNOLOGY

More information about
SCORE/Integration Suite:
at www.adaptive-frameworks.com