# TEKZONEFORUM036

MO. 25. AUG. 2003, 17:00 UHR

MIDDLEWARE FOR DISTRIBUTED COMPUTING

# DIE QUAL DER WAHL: OBJEKTE ODER PROZEDUREN?

MIT ROUNDTABLE-DISKUSSION

WIRD PRÄSENTIERT VON

netcetera

MEDIENPARTNER

netzwoche

TEKZONEFORUM036

OMEX

# Middleware for Distributed Computing

## Die Qual der Wahl: Objekte oder Prozeduren

**Werner Froidevaux**
Dipl. Informatik Ing. ETH
werner.froidevaux@omex.ch

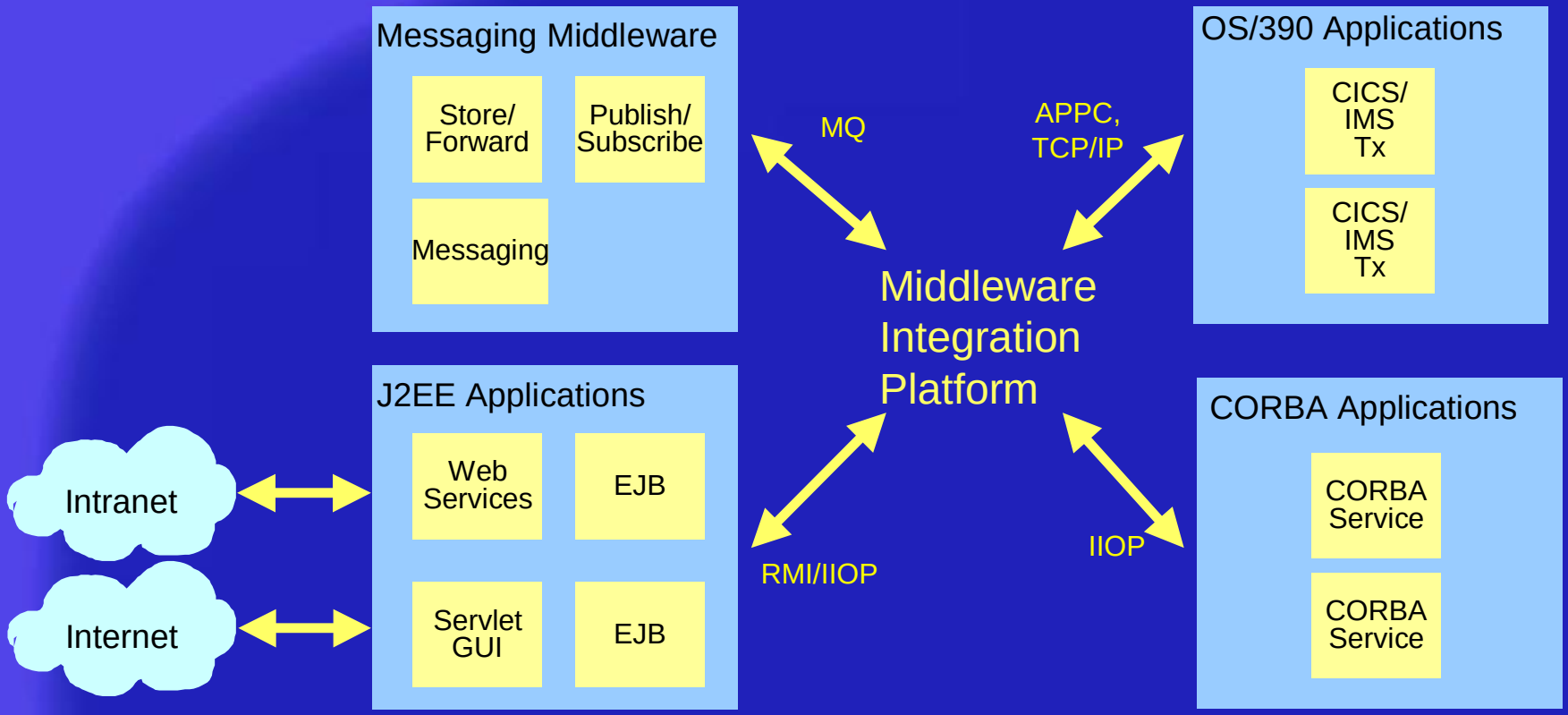
**OMEX AG**
Technoparkstr. 1, CH-8005 Zürich, Switzerland

# Why Distributed Computing?


- Key **benefits** of distributed computing:
  - Enabler for **migration** & **evolution** of IT systems
  - Allows **integration** of heterogeneous platforms
  - Increases **flexibility** & **reusability** of applications

# Distributed Computing

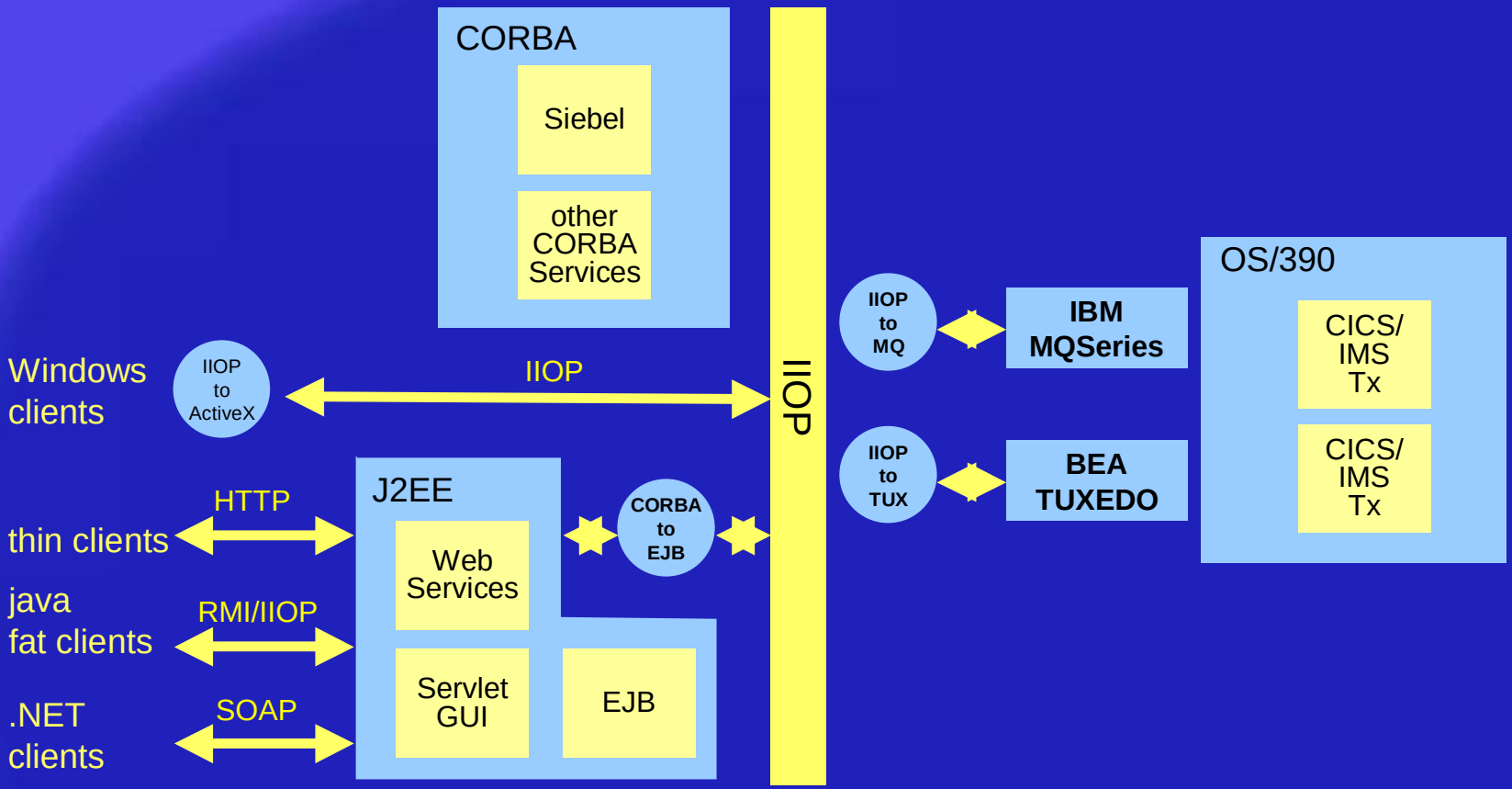
## Middleware integrates complex IT environments




Integration possible either through native middleware protocols (IIOP, RMI/IIOP, SOAP) or through specific middleware integration adapters.

# Distributed Computing

## Real-world sample



# Middleware Infrastructure

Standards and Technologies

OMEX

- Middleware Standards and Technologies:
  - **RPC.** Remote Procedure Call from SUN (and others)
  - **WebServices:** WSDL (Web service description language) and SOAP (simple object access protocol) defined by W3C.
  - **CORBA**: Common Object Request Broker Architecture from OMG (Object Management Group)
  - **J2EE/EJB**: Enterprise Java Beans from SUN.
  - … and many others

# Middleware Infrastructure

## Problems solved by Middleware

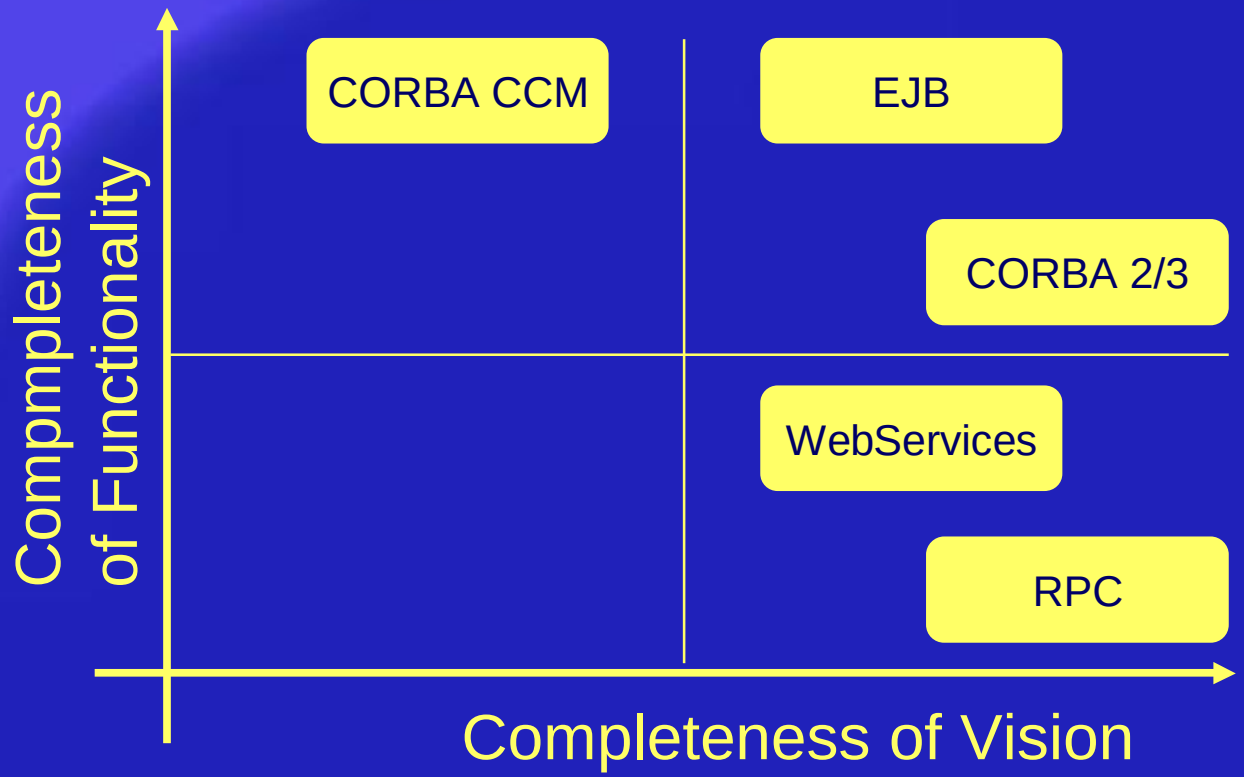
- Which problems does middleware solve?
  - How to **find** remote services?
  - How to **communicate** with remote services?
  - How to handle (de-)**marshaling** issues?
  - How to **activate** remote services?
  - How to keep **state** persistent and consistent?
  - How to solve **security** issues?
  - How to solve **failover**?
  - How to solve **scalability**?
  - …

# Middleware Infrastructure

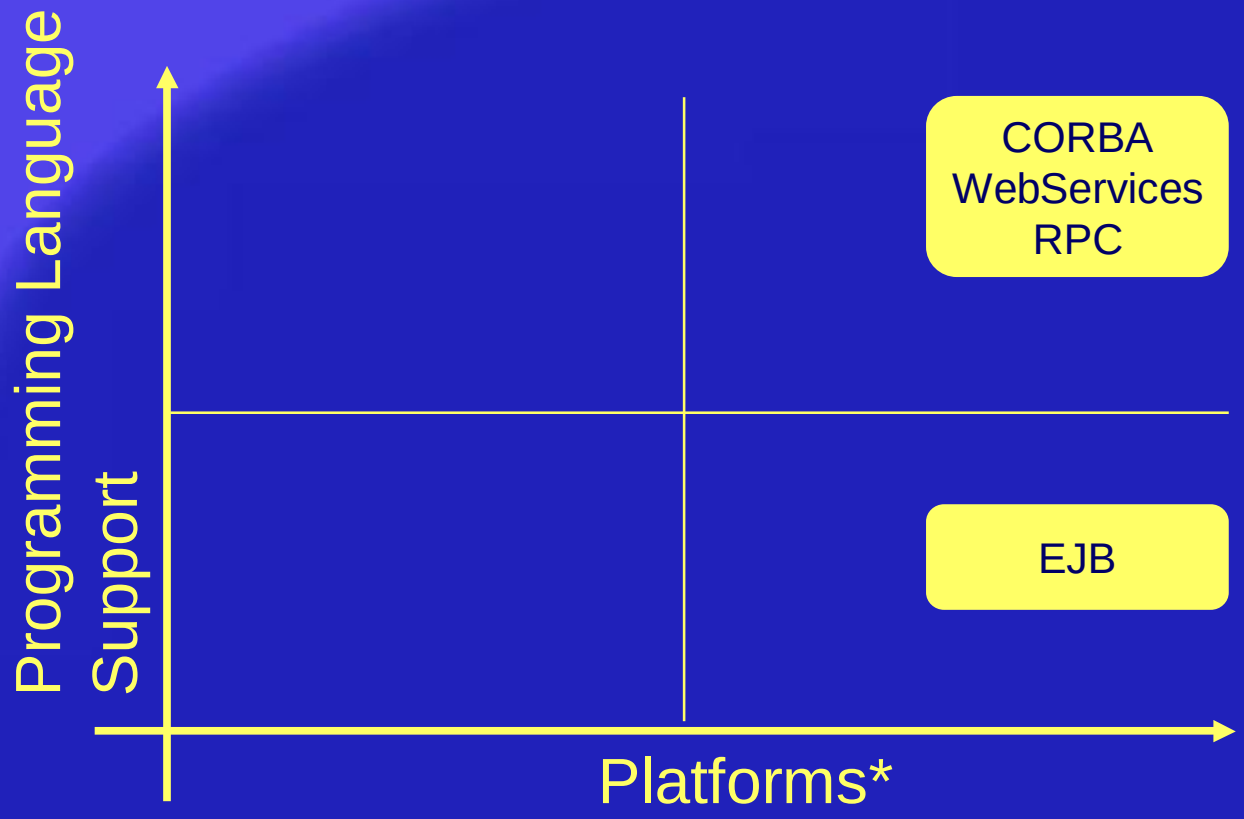
## Types of Services

- Middleware such as CORBA, EJB, WebServices, RPC all support one or more of the following service types:
  - Service **interaction**:
    - request/response
    - callbacks
    - asynchronous invocation
    - store and forward
    - publish and subscribe
  - Service **state**:
    - stateless
    - during a session
    - persistent
  - Service **granularity**:
    - process (e.g. AddressManager)
    - application object (Address with id '0815')

# Middleware Comparison *[1]*

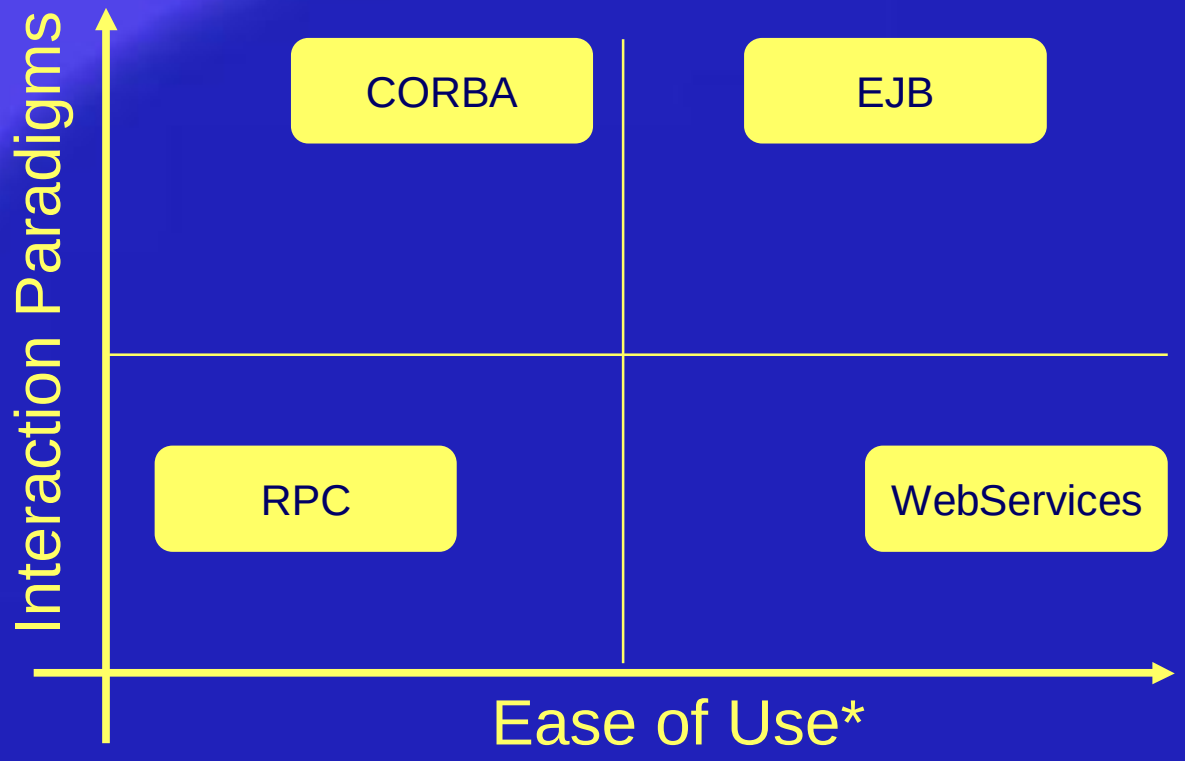


# Middleware Comparison *[2]*



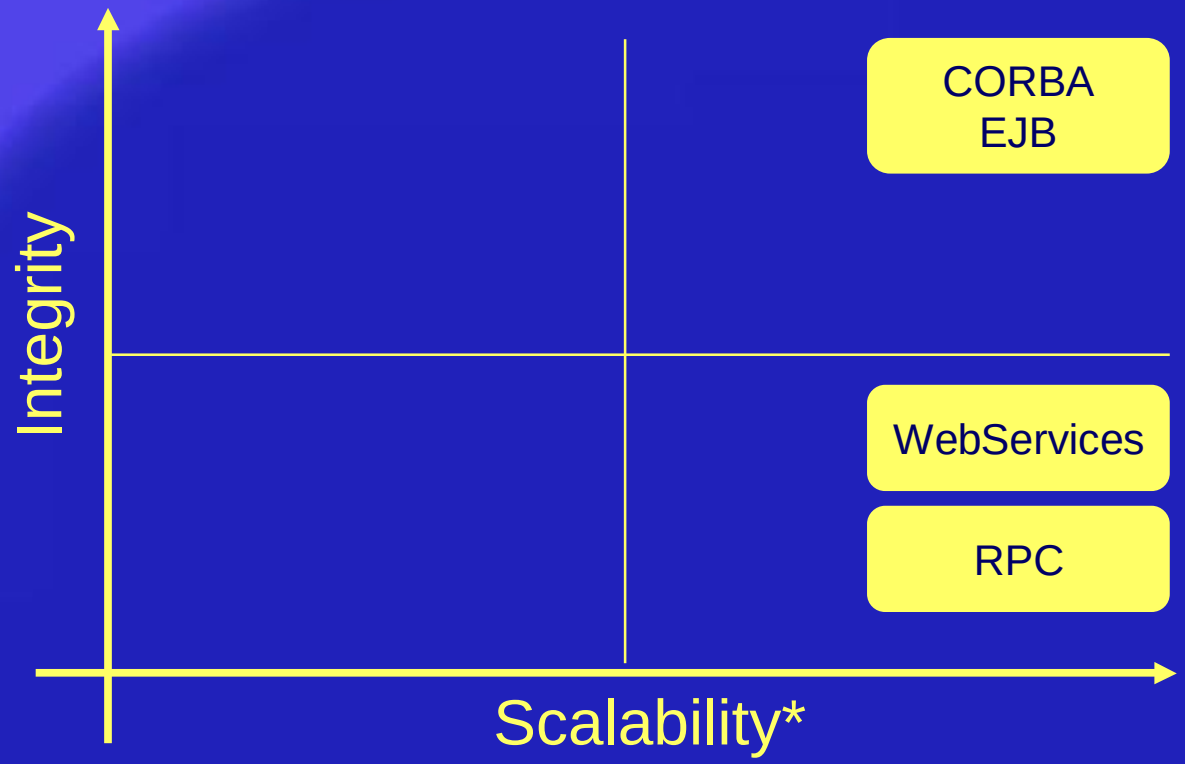

*NOTE: a standard designed for multi-platform support does not imply that products implementing the standard are available on all possible platforms.

# Middleware Comparison *[3]*




*NOTE: development and operations.

# Middleware Comparison *[4]*




*NOTE: a standard designed for scalability does not imply that products implementing the standard do always scale well.

# Middleware Comparison

## Difficulties


- Application programming is difficult (learn CORBA, EJB, WebServices, …)
- Application integration is even more difficult
- Application code is non-portable (CORBA application can not easily be ported to EJB)

- **==> SW development projects put a lot of effort in middleware technologies instead of application logic**

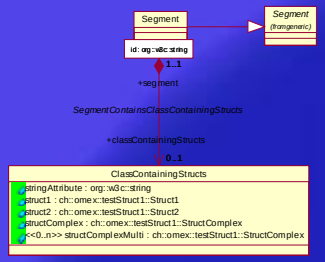
# Next Generation Middleware

## The Model Driven Approach - MDA

- OMG's model driven architecture (MDA) initiative abstracts from middleware technologies.
- Application are modeled and implemented platform/middleware independent
- **Platform-independent model** is **transformed** at development or runtime to a specific platform e.g. EJB, CORBA, WebServices.

# Next Generation Middleware

## MDA with openMDX




**PIM**
**(platform-independent models)**

uses

**PIP**
(platform-independent **plugins**)

**PSP**
(platform-specific **plugins**)

platform-independent component contract

JMI | MOF | XMI | JDO+

openMDX - PIM Platform

**Local** | **J2EE** | **CORBA** | **.NET** | **WebServices**

Platform

**Language and Standard Library Platform**
(e.g. JDK, .NET CLR)

*NOTE: minimal use of generators, application-logic implemented in Java (no new 4GL, …)

# Next Generation Middleware


## MDA with openMDX

```
// openMDX JMI sample
List addresses = person.getAddress();
for(
  Iterator i = addresses.iterator();
  i.hasNext();
) {
  Address address = (Address)i.next();
  System.out.println("address=" + address.getLine(0);
  System.out.println("address=" + address.getCity();
  System.out.println("address=" + address.refGetValue("country"));
}
```

**NOTE**:
- standard JDK APIs (no CORBA, EJB, WebServices interfaces)
- Iterator implementations of openMDX are batching (handling of large input and result sets)
- Behaviour configured at deployment time