



**Poseidon House
Castle Park
Cambridge CB3 0RD
United Kingdom**

TELEPHONE:
INTERNATIONAL:
FAX:
E-MAIL:

**Cambridge (01223) 515010
+44 1223 515010
+44 1223 359779
apm@ansa.co.uk**

Training

ANSAwise - CORBA in the Real World

Chris Mayers

Abstract

Organizations considering purchasing CORBA technology wish to know the state of the market and the practical problems that they may encounter.

CORBA technology is still new; products are incomplete, have serious bugs, and statements by vendors cannot be taken at face value.

This session of the ANSAwise training programme explains the evaluation work carried out by the ANSA Phase 3 project, and the findings so far.

This session is available only to sponsors.

[Note: this module includes material from APM.1194 (Comparison of CORBA-compliant platforms), and should be maintained in line with that document.]

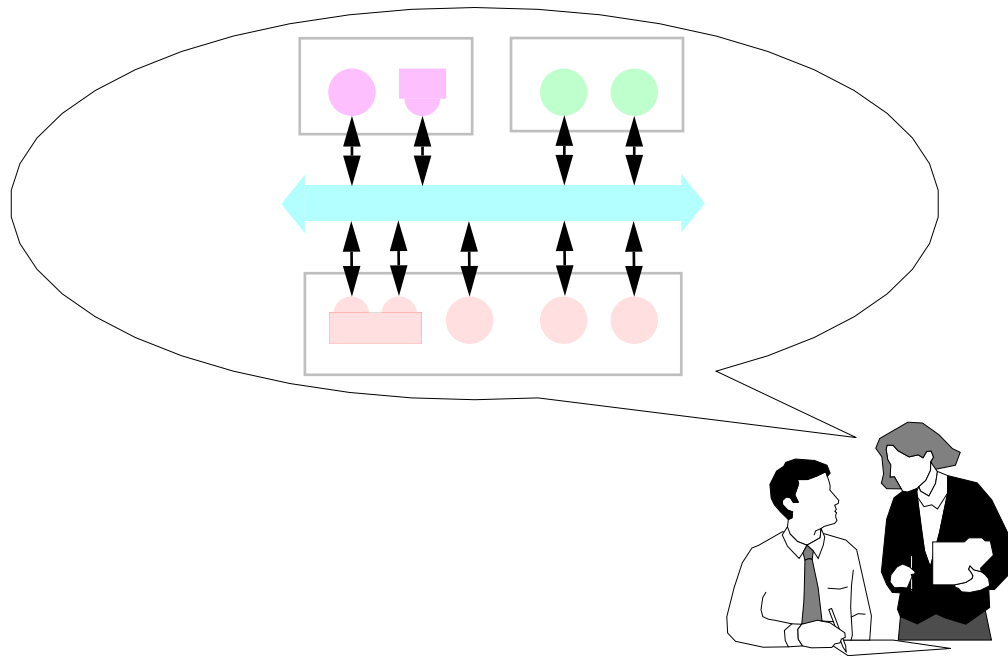
APM.1379.03

Approved
Briefing Note

2nd April 1996

Distribution:
Supersedes:
Superseded by:

CORBA in the Real World



This presentation is confidential to the ANSA consortium



In this session

- **Review some CORBA products**
 - as evaluated as part of the ANSA Phase 3 project
- **Explain their current limitations**
 - and likely future developments
- **Note other points that arose during the evaluation**



CORBA products evaluated to date

- **Iona's Orbix, version 1.2 (and a beta version 1.3)**
- **Xerox's ILU (Inter-Language Unification), version 1.7**
- **Post-Modern Computing Technologies' ORBeline**
- **ICL's DAIS (pre-release 2.1)**
- **DEC's Application Control Architecture Services (ACAS), version 2.1**
 - **now known as ObjectBroker**
- **Object-Oriented Technologies' DOME**

Evaluations of new and updated CORBA products continue



Approach to evaluation

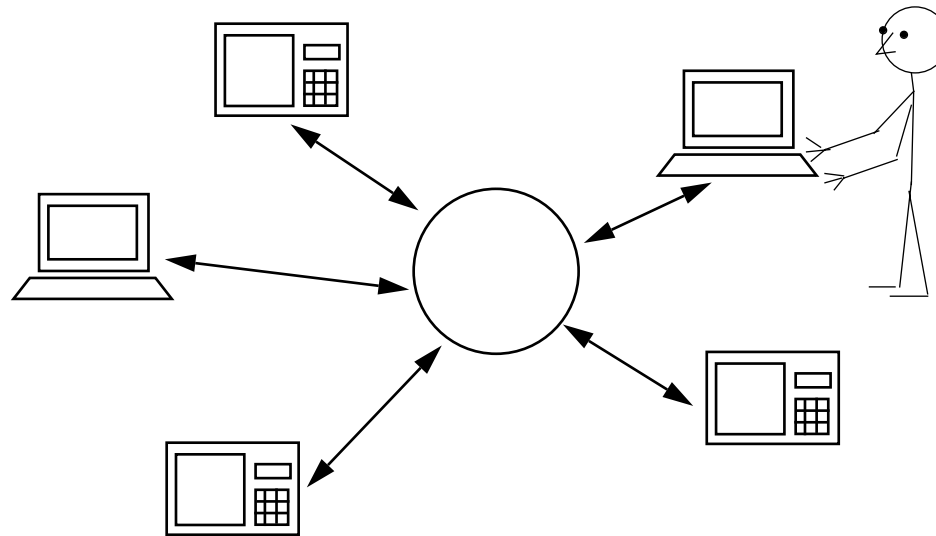
- **Check conformance with the CORBA 1.2 specification**
 - CORBA 2 products were not then available
- **Port standard example programs**
 - the Echo and Simple Bank applications
- **Look beyond the ORB specification**
 - multi-threading
 - trading (or its equivalent)
- **Evaluate the product's ease of use, including documentation**



What conformance means

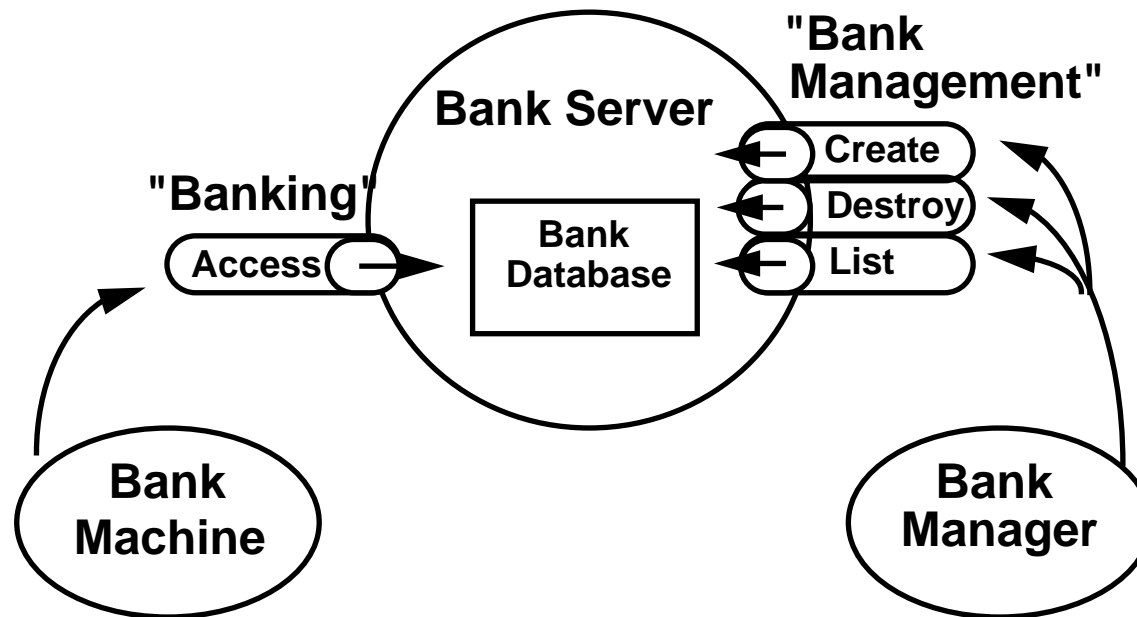
- **Conformance means that an implementation matches a specification**
 - here we are interested in an ORB implementation conforming with the CORBA specification
- **In the future, conformance will be formally validated against an international CORBA standard**
 - probably by X/Open...
 - ...meanwhile, we must check the vendors' assertions ourselves

Simple Bank Example



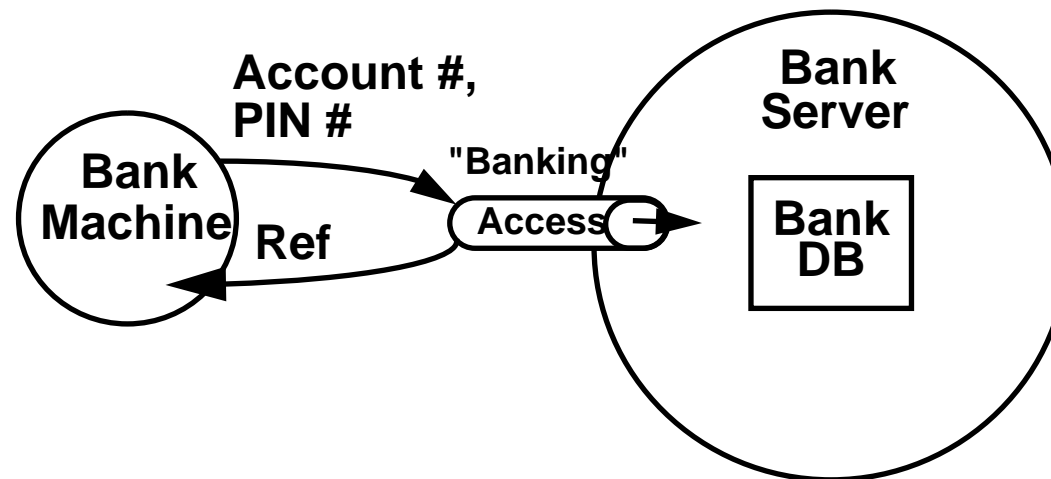
- **A Bank Machine (Automatic Teller Machine) system**
 - **Users accessing their accounts**
 - **Bank managers administering these accounts**

Bank Server - Interfaces

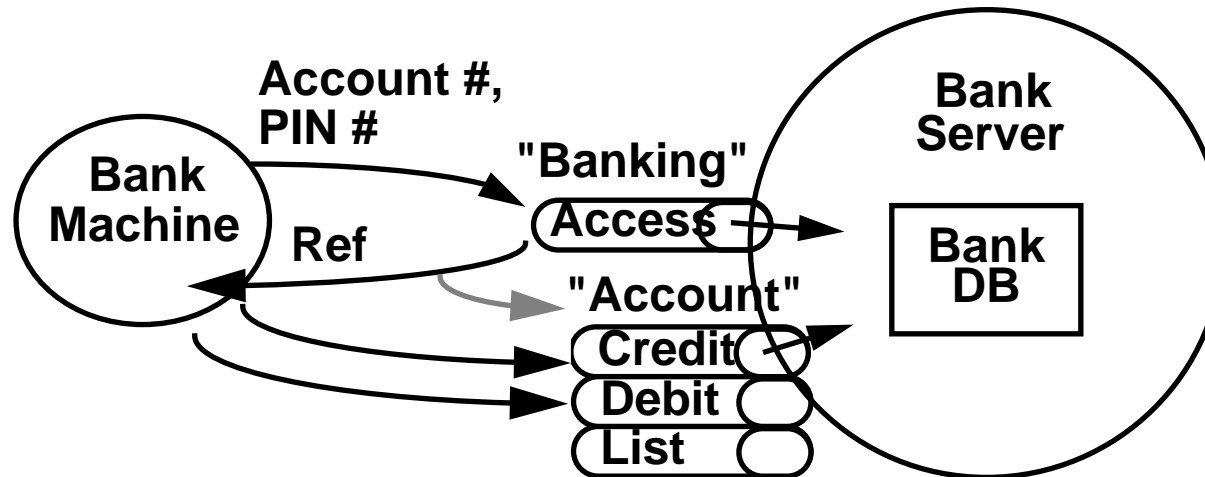


Banking Interface

- User should only be able to interact with account if PIN is valid
- Do not want Accounts to be public
 - customers should only access their own accounts
- If "Access" succeeds, "Account" is created and its object reference returned



Banking and Account interfaces



- If "Access" succeeds, "Account" is created, and its object reference returned
- "Account" interface provides operations on one particular account
- Only that client knows the "Account" object reference



Simple Bank Interfaces

- **Banking (SBank)**
 - for access by customer via Bank Machine, with PIN
- **Account (Account)**
 - for credit/debit/list by customer via Bank Machine
- **Bank Management (SBankMgmt)**
 - for create/destroy/list by manager



Simple Bank - general definitions

```
typedef unsigned long AccountNumber;
```

```
typedef unsigned long PersonalIdentificationNumber;
```

```
exception NoSuchAccount {};
```

```
exception InvalidPin {};
```

```
exception InsufficientFunds {};
```

```
exception ResourcesExhausted {};
```

```
exception StaleAccountReference {};
```



Account Interface

- **Account - credit/debit/list account**
- **An Account object is created (via the SBank interface's Access operation) for each account being accessed**
- **So there is no need to pass account numbers as arguments to the operations in the Account interface**



Simple Bank - interface Account

```
interface Account
{
    struct AccountRecord {
        string owner;
        float balance;
        string lastaccess;
    };
    void Credit(in float Amount) raises(StaleAccountReference);
    void Debit(in float Amount)
        raises (InsufficientFunds, StaleAccountReference);
    void List(out AccountRecord List_R1)
        raises(StaleAccountReference);
};
```



Banking (SBank) Interface

- **SBank - PIN based access to Account interfaces**
- **Returns an Account interface for the account identified by an account number and corresponding PIN**
- **Equivalent to the user interface offered by a real bank machine to a human user**



Simple Bank - interface SBank

```
interface SBank
```

```
{
```

```
    Account Access(in AccountNumber acct,  
                   in PersonalIdentificationNumber pin)  
        raises(NoSuchAccount, InvalidPin, ResourcesExhausted);
```

```
};
```



Bank Management (SBankMgmt) Interface

- **SBankMgmt - management operations for:**
 - creating an account
 - destroying an account
 - listing an account
 - listing all accounts
- **Note the use of an IDL sequence to return a variable amount of data when listing all accounts**



Simple Bank - interface SBankMgmt: types

```
interface SBankMgmt
{
    struct CreateRecord {
        AccountNumber acct;
        PersonalIdentificationNumber pin;
    };
    struct FullRecord {
        AccountNumber acct;
        PersonalIdentificationNumber pin;
        string owner;
        float balance;
        string lastaccess;
    };
    typedef sequence<FullRecord> ListAllResult;
};
```



Simple Bank - interface SBankMgmt: operations

```
void Create(in string owner,  
           in float balance, out CreateRecord Create_R1)  
    raises(ResourcesExhausted);
```

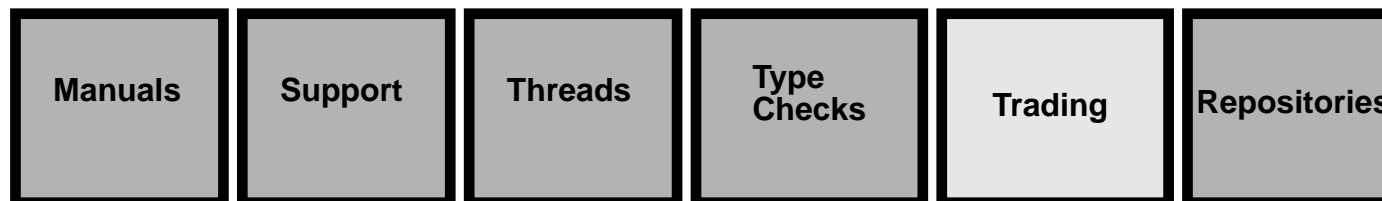
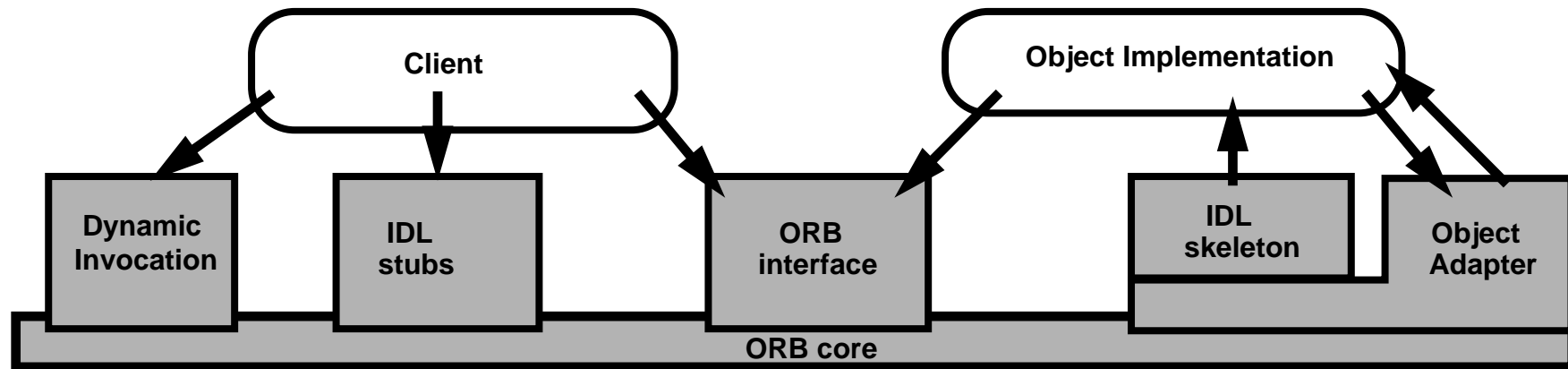
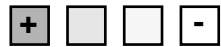
```
void Destroy(in AccountNumber acct) raises(NoSuchAccount);
```

```
void ListOne(in AccountNumber acct, out FullRecord ListOne_R1)  
    raises(NoSuchAccount);
```

```
void ListAll(out ListAllResult ListAll_R1);
```

```
};
```

Orbix





ORBIX - Iona Technologies

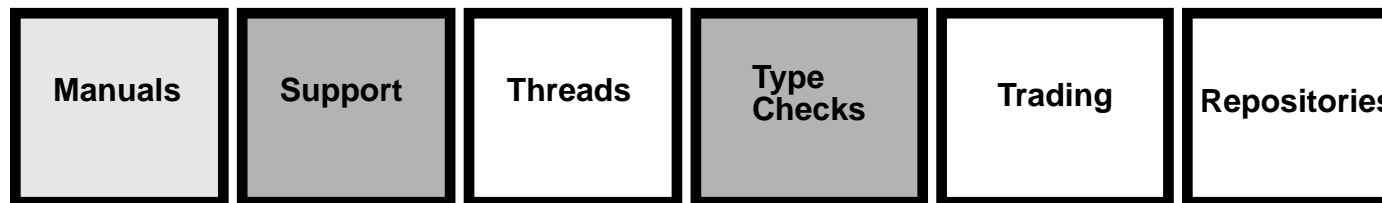
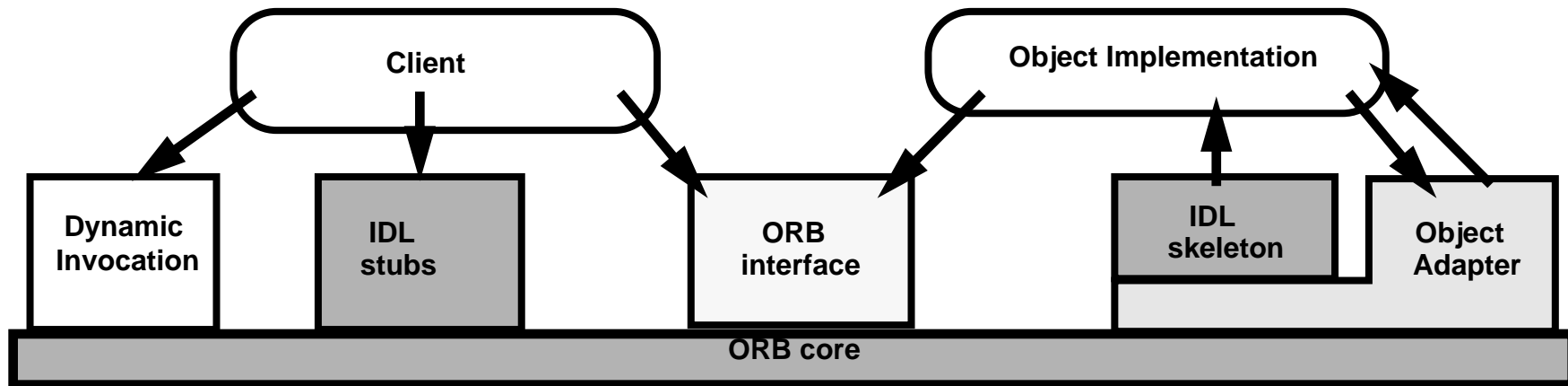
- **Implements all of the ORB**
 - all IDL, the Dynamic Invocation Interface, and exceptions
- **Clients and servers are written in C++**
- **No direct support for multi-threading**
 - but Orbix examples show how to use an operating system threads package



Other ORBIX findings

- **The Interface Repository is simply a directory containing the IDL source files**
 - **eminently straightforward...**
 - **... but the Interface Repository server must reparse all the IDL source files when it starts up**
- **Executable binaries are small; they use shared libraries**

ILU





ILU - Xerox

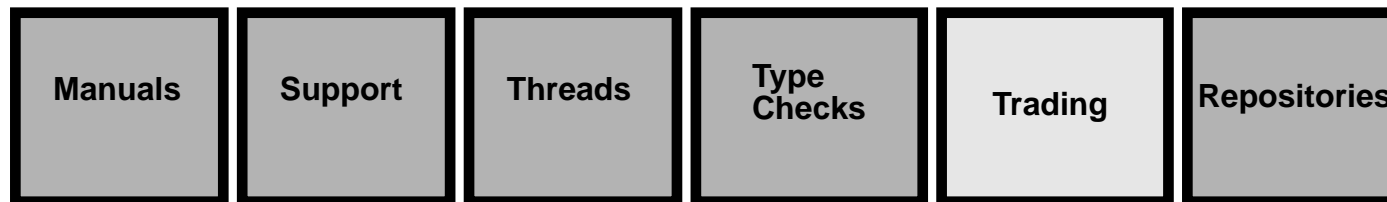
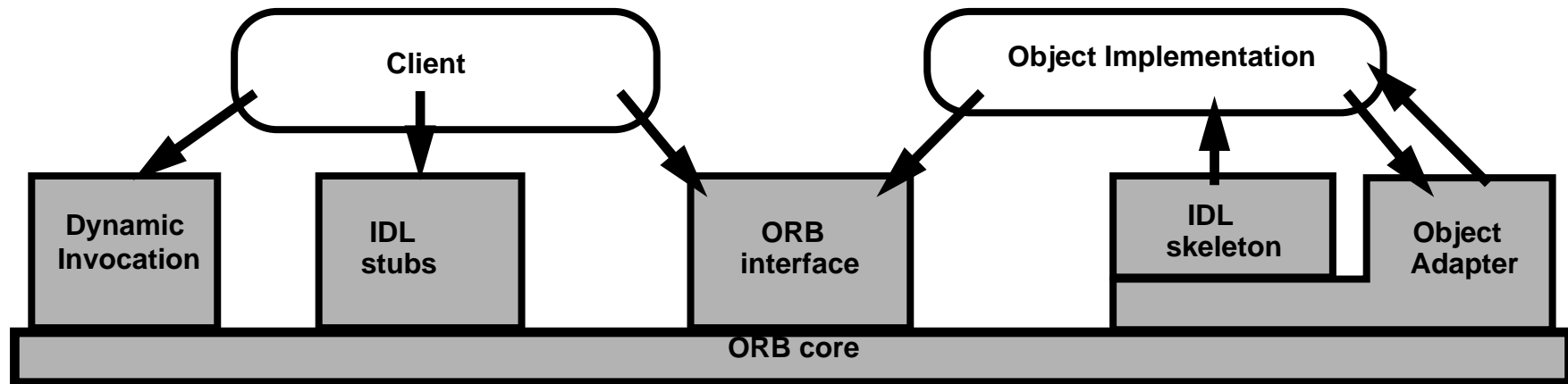
- **Incomplete ORB implementation**
 - a few IDL features missing (*Object* and *any* types)
 - no Dll support
- **Clients and servers are written in C, C++, Common Lisp, and also Modula-3**
- **Supplied as C source**
 - implementation is free!
- **No direct support for multi-threading**



ILU - other findings

- **No Interface or Implementation Repository**
- **Executable binaries are large; no shared library support**
 - **not surprising for a product supplied in source form**

ORBeline





ORBeline - Post-Modern Computing Technologies

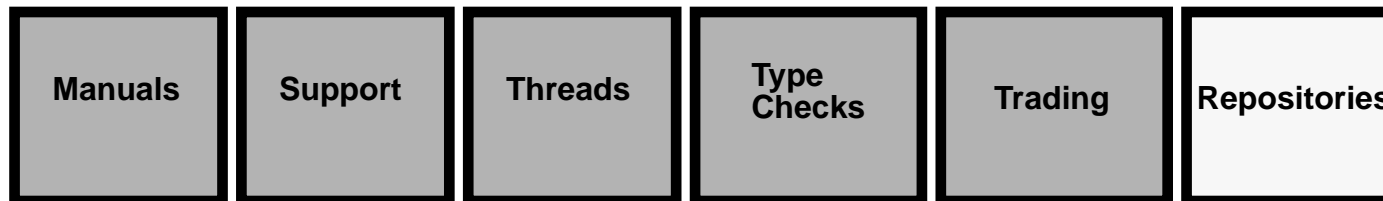
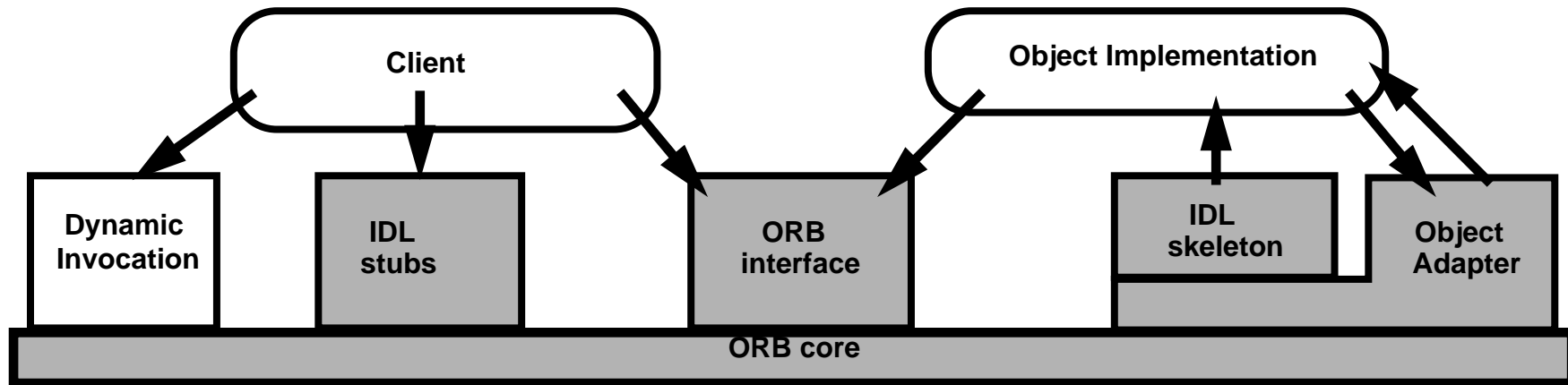
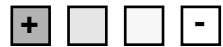
- **Implements all of the ORB**
- **Clients and servers are written in C++**
 - **but do not quite conform with the anticipated C++ Language Mapping**
- **Multi-threading is supported**
 - **in a separately-packaged version**



ORBeline - other findings

- **Both Interface Repository and Implementation Repository supported**
- **Supports event handling, object migration, and object replication**
- **Executable binaries are small; they use shared libraries**

DAIS





DAIS - ICL

- **Implements almost all of the ORB**
 - **no Dll support**
- **Clients and servers are written in C**
- **Multi-threading is supported**



DAIS - other findings

- **No Interface Repository or Implementation Repository**
- **Support for object relocation and migration**
- **Executable binaries are large; no shared library support**
- **Extended with additional services**
 - **alerts, configuration, database integration, transactions, and security**



ACAS - DEC

- **Incomplete ORB implementation**
 - no IDL support (no stubs), Dynamic Invocation Interface only
 - no constructed values (e.g. sequences)
- **Clients and servers are written in C**
- **No direct support for multi-threading**
 - and no obvious way of achieving it



ACAS - other findings

- **Servers can be implemented using shell scripts**
 - flexible and useful for prototyping
- **There is a DDE (Dynamic Data Exchange) interface for Microsoft Windows clients**
- **Executable binaries are small; they use shared libraries**



DOME - Object-Oriented Technologies

- **Incomplete ORB implementation**
 - limited IDL support
 - no typedefs
 - no constructed values (e.g. sequences)
 - no CORBA interfaces (proprietary only)
- **Clients and servers written in C++**
- **No direct support for multi-threading**
 - it may be possible to achieve this with C++ implementation classes



DOME - other findings

- **No Interface or Implementation Repository**
- **Installation was difficult**
 - **and there were several serious bugs**
- **Executable binaries are large; no support for shared libraries**



General summary - conformance

- **Not all vendors fully support IDL**
 - although they aim to in future
- **For language mappings**
 - **C++ is generally supported, but not always C**



General summary - engineering

- **Lack of shared library support for C++ leads to large executable binaries**
 - typically 1 Mbyte without shared libraries, 150 Kbyte with them
 - a general problem for C++, not confined to CORBA...
 - ... shared libraries require support from the C++ compiler/linker *and* the operating system *and* the ORB vendor



Aspects not evaluated

- Evaluation is not a formal test of conformance
- Evaluation is not carried out with a particular application in mind
- Evaluation does not test many important non-functional characteristics (e.g. compilation speed, scalability, performance)
- Only the ORB was evaluated
 - not Object Services
- Interoperability was not evaluated



Other technical considerations

- **Leapfrogging ORB product versions and compatibility**
 - **when you are using multiple operating systems or compilers**
- **Bundled administration tools**
- **Integration with other tools, for example**
 - **make/configuration control**
 - **CASE tools**
 - **GUI toolkits**



Other commercial considerations

- **Licensing conditions**
 - **development and run-time licenses**
- **Vendor qualification**
- ...



Summary

- **There are complete CORBA implementations available now**
 - upgrades for ORB 2.0 in progress
- **Most implementations support C++**
 - but check for conformance with the CORBA C++ Language Mapping
- **New and updated implementations are continuing to arrive**
- **Few implementations yet support real-time applications**
 - but ANSA Phase 3 is working on this
- **For more on this topic**
 - see *Comparison of CORBA-compliant platforms (APM.1194)*



Supported Hardware/OS

HW/OS	Orbix	ILU	ORBeline	DAIS
SunOS4	Yes (4.1)	Yes (4.1.3)	Yes	Yes
Solaris 2	Yes	Yes (2.3)	Yes (2.3)	Yes (2.3)
HP/UX	Yes (9.x)	Yes (mdrb!)	Yes	Yes (8.x, 9.x)
OSF/1	Yes (3.0)	Yes (untested)	Yes (2.0)	available internally†
other UNIX	IRIX, AIX, Ultrix, UnixWare, Sinix	IRIX, Linux (Xerox) AIX, DG/UX, SCO others should work	AIX, UnixWare VxWorks, LynxOS, pSOS	SVR4 Sparc, SCO AIX soon UnixWare, Pyramid†
MS Win 3	Yes (3.1)		Yes (3.1)	Yes (3.11)
OS/2				Yes (2.1)
NT	Yes (3.5)		Yes (3.5)	soon
VMS				Yes (6.1)
Mainframes			Cray	VME SV 294