Poseidon House
Castle Park
Cambridge CB3 0RD
United Kingdom

TELEPHONE: **Cambridge (01223) 515010**
INTERNATIONAL: **+44 1223 515010**
FAX: **+44 1223 359779**
E-MAIL: **apm@ansa.co.uk**

Training

EPFL Course September 95: Architecture for ODP, Part 2

Mark Madsen

Abstract

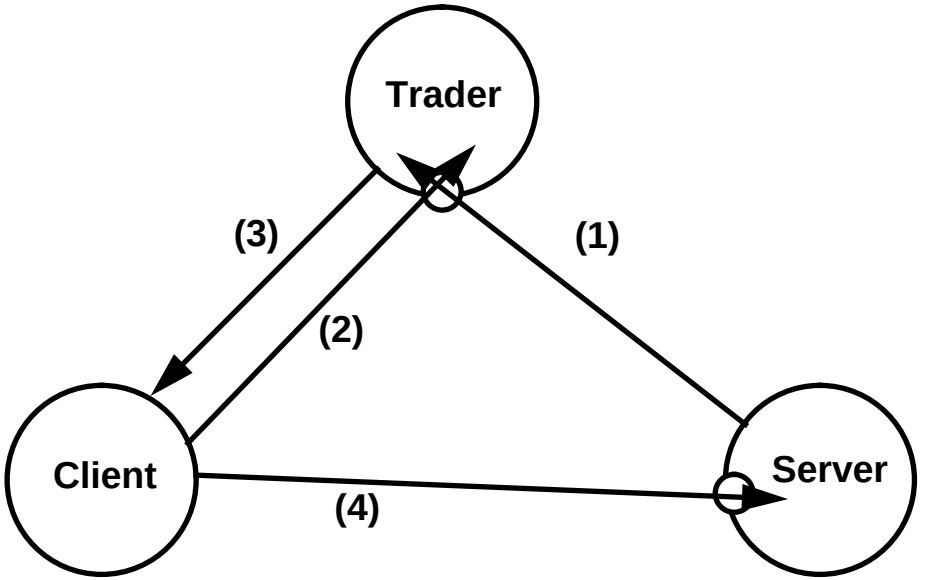
This presentation is based on APM.1330.02 “ANSAwise - Trading and Federation”.

It was adapted for presentation as part of Module M3 “Distributed Systems” of the course on Communication Networks given at Ecole Polytechnique Federale de Lausanne in September 1995.

APM.1565.01 **Approved** 8th September 1995
Briefing Note

Distribution:
Supersedes:
Superseded by:

ODP Architecture II: Trading and Federation

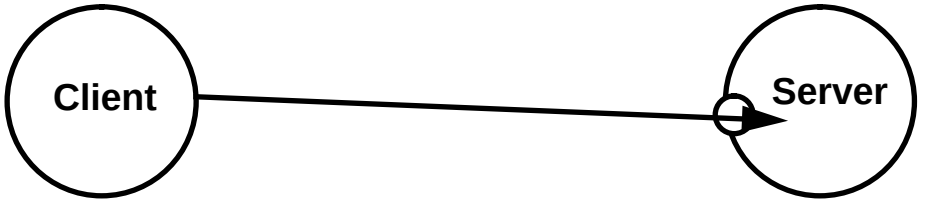


In this session

- *Explain the purpose of trading in a distributed system*
  - the roles of client, server, and trader
- *Show a simple form of trading in action*
- *Explain the significance of federation*
- *Explain federated trading*

Client and Server are Roles, not Types of Machine

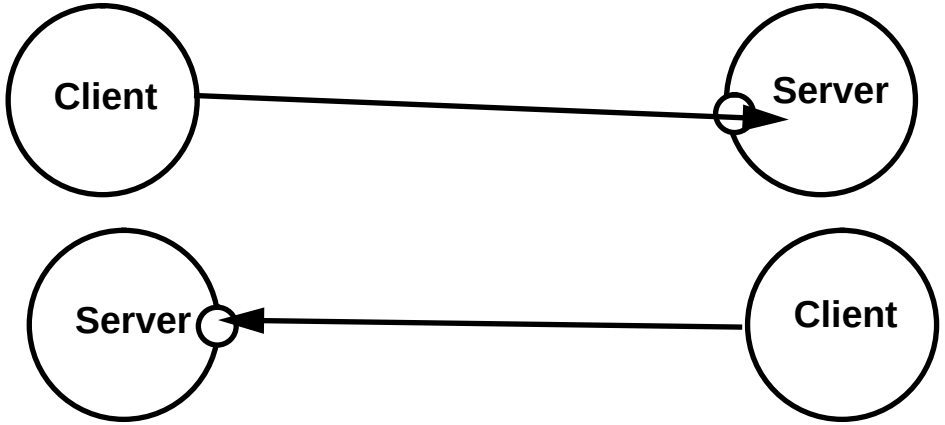
- *We use the terms client and server in a particular sense...*
- *... Client and Server are roles in a specific interaction between objects*

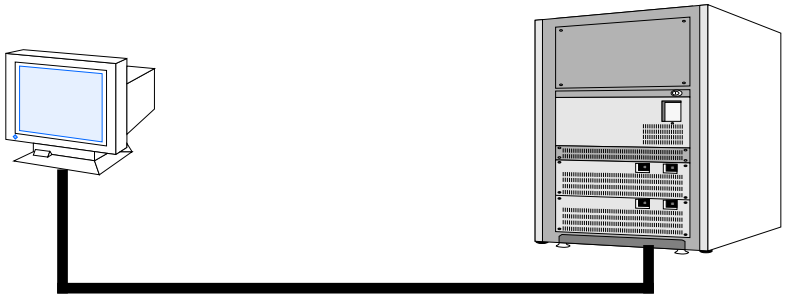


- *Nothing is implied about the technology that supports them*
  - the client could be a mainframe...
  - ...the server could be a PC
  - ... they could both be on the same Unix machine
- *Technology is not part of the Computational Model*

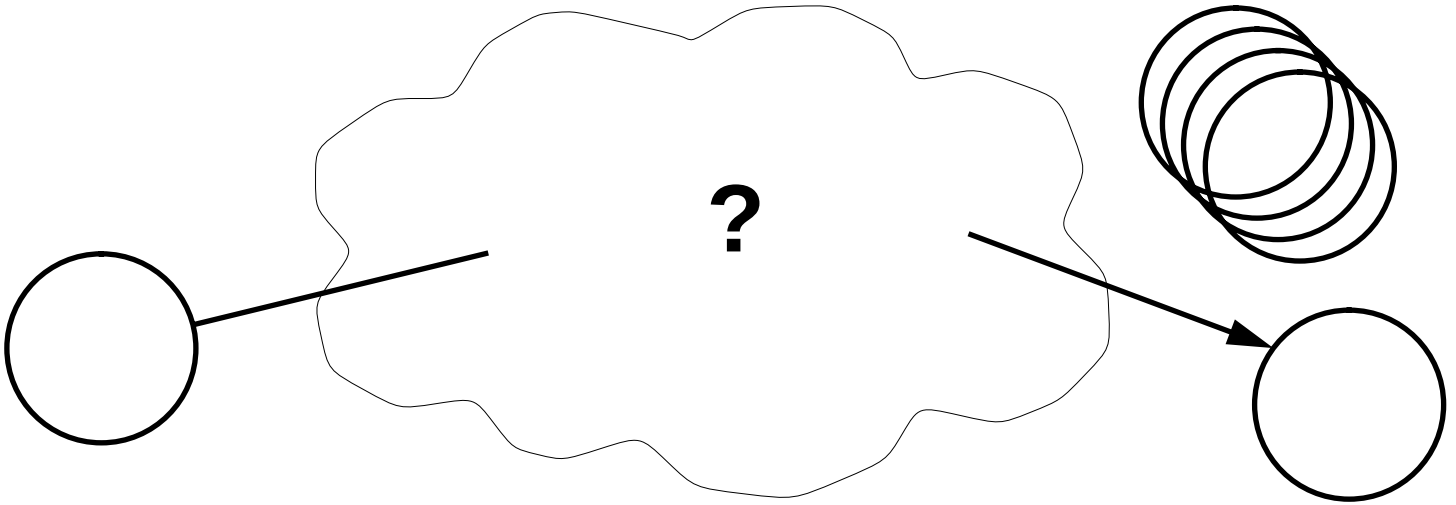
Client and Server Roles

- ***The roles are determined by the objects themselves***





The need for Trading



- ***How can clients find servers that provide the services that they need?***
  - **in the future, there will be millions of interconnected servers around the world**
  - **clients will come and go dynamically**
  - **servers will come and go dynamically**

Stating service requirements

- *There are two potential ways to identify something*
  - by naming it (as in a “white pages” telephone directory)
  - by describing it (as in a “yellow pages” telephone directory)
- *These functions are sometimes combined in a directory service*
  - but they are logically separate, just as real telephone directories are
- *CORBA has separate Naming and Trading services*

Naming is not enough

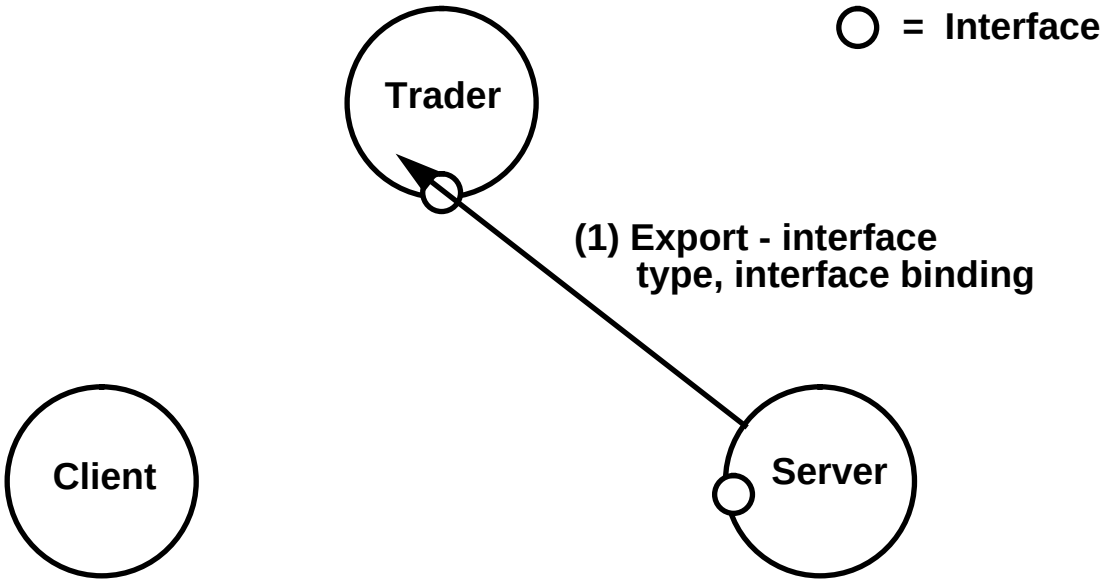
- ***Trading is necessary***
  - **we cannot rely on clients being able to name servers...**
  - **....the server may not even exist when the client was created**
- ***Trading works by matching descriptions provided by clients and servers***

Trading - Basic needs

- ***Server must state what it provides***
  - **it must export a service offer**
- ***Client must state what it requires***
  - **it must import a service offer**
- ***Trading must find a service offer that matches the request***
  - **there may be many such offers...**
  - **...there may be none**

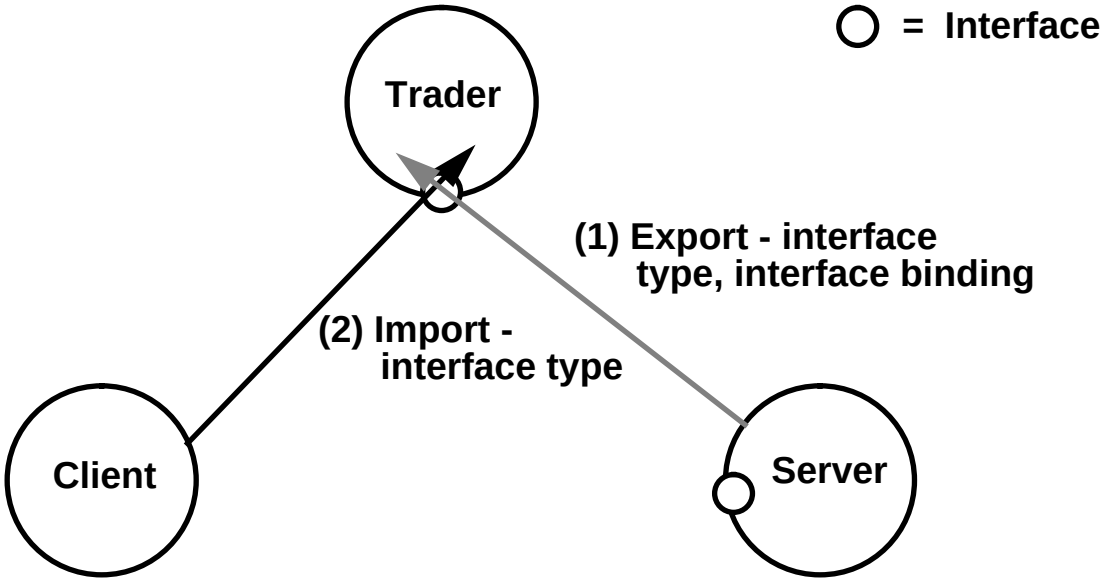
Steps in Trading - (1)

- *(1) Server exports a service offer to the Trader*



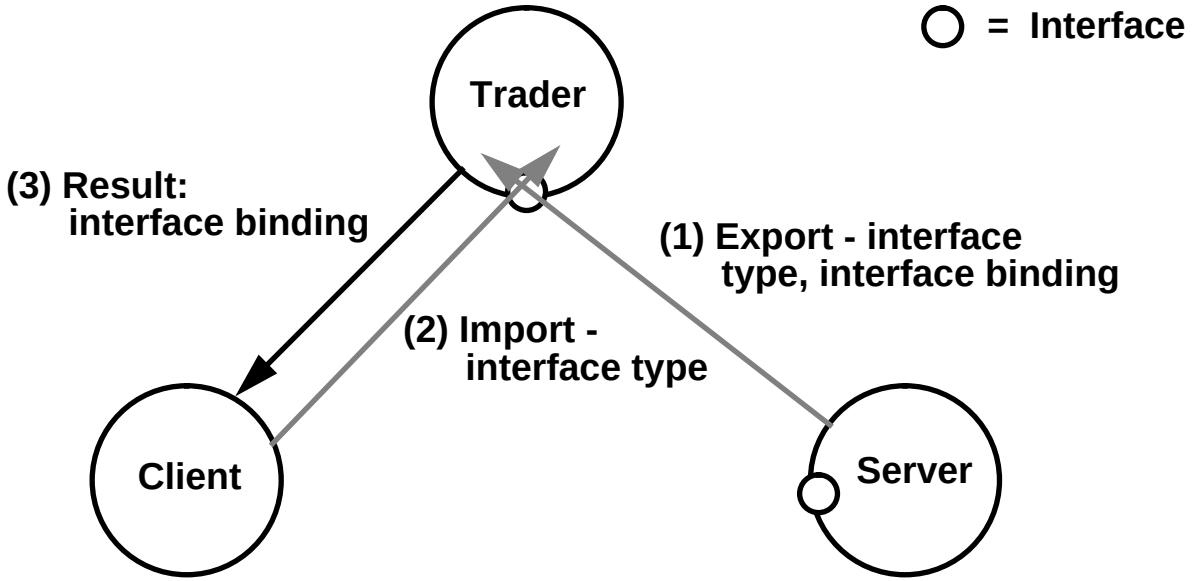
Steps in Trading - (2)

- *(2) Client requests a service offer from the Trader*



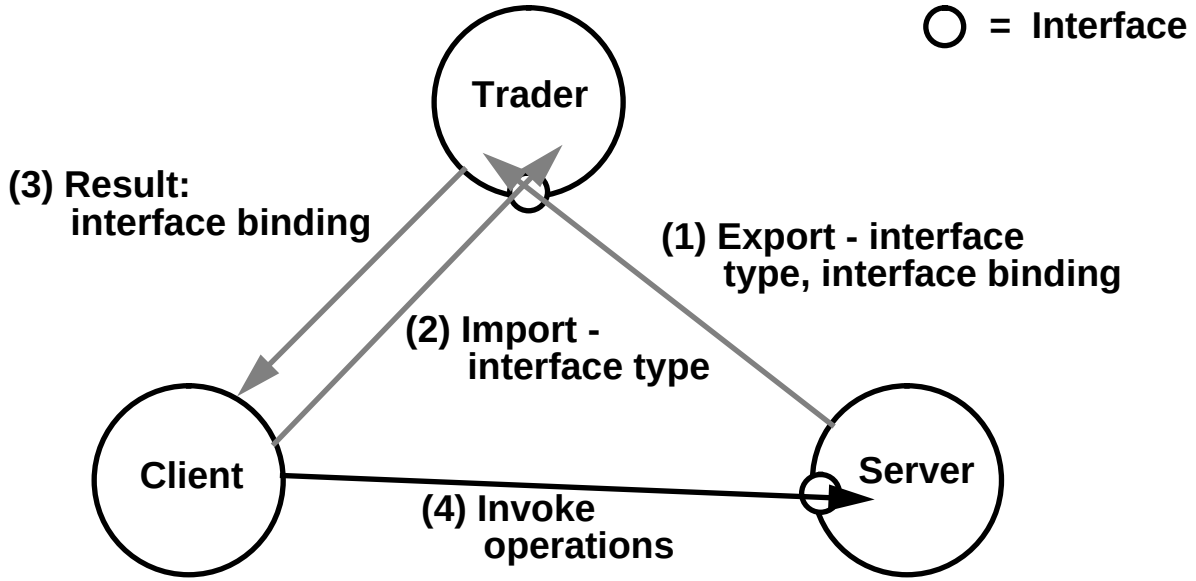
Steps in Trading - (3)

- *(3) Trader returns a matching service offer to the client*
  - it returns the interface binding given by the server



Steps in Trading - (4)

- *(4) Client uses interface binding to invoke the server's operations*
  - Trader takes no further part in the interaction
  - an interface binding from the Trader is invoked just like any other



Matching requests with offers - type conformance

- ***How does Trading decide whether a client request matches a server offer?***
  - **it uses the interface *type conformance* concept from the Computational Model**
- ***If the client request and server offer interface types do not conform, they are incompatible, and cannot match***

Other matching criteria

- *Type conformance is not sufficient*
  - who owns the service?
  - what will it cost to use?
  - where is the service, and can it be reached?
- *These criteria are known as properties*
  - (name, value) pairs
- *Preference criteria sort matching offers into order*
- *Scope criteria control where to look for offers*

Using properties

- ***Property name/value pairs are specified when offer is exported***
- ***Clients importing services may specify:***
  - **Acceptable property constraints, for example**

```
(PaperSize=='A4')
```

  - **Selection of offer with maximum (or minimum) value of a property**
- ***Arbitrary combinations of constraints may be specified in the form of a logical expression, for example***

```
(Node == 'basilisk') and (Price <= 200)
```

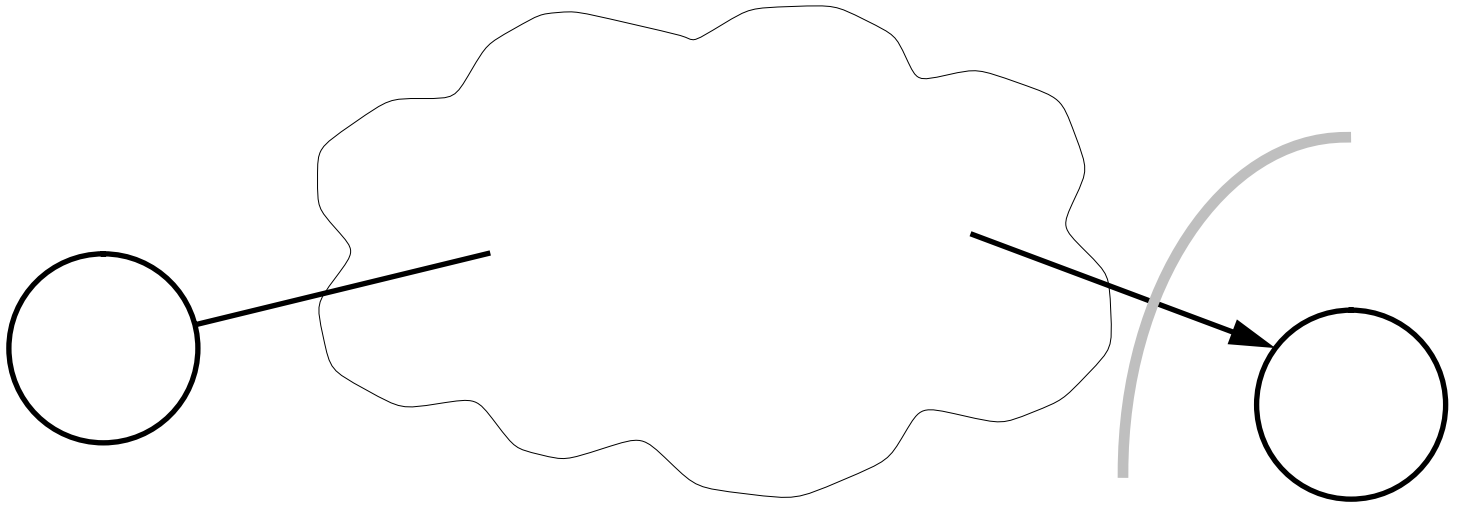
Trading in large distributed systems

- ***Because there will be millions of servers in the world:***
  - **there will be many Traders providing the Trading service**
  - **... Traders must be interconnected**
  - **... the Trading service must itself be distributed for scalability**
  - **... and cannot be centralized**
  - **this is called *federated trading***
- ***And also because organizations will wish to control their own Traders:***
  - **to determine who sees which services**

The need for federation

- ***Organizations want to offer electronic services to other organizations***
  - **services provided by applications in the usual way...**
  - **...services provided by distributed objects**
- ***Organizations want to make money***
  - **to *sell* services provided by distributed objects**
- ***Organizations need to keep control of their objects***
  - **control over their security, for instance**

Federation is concerned with boundaries



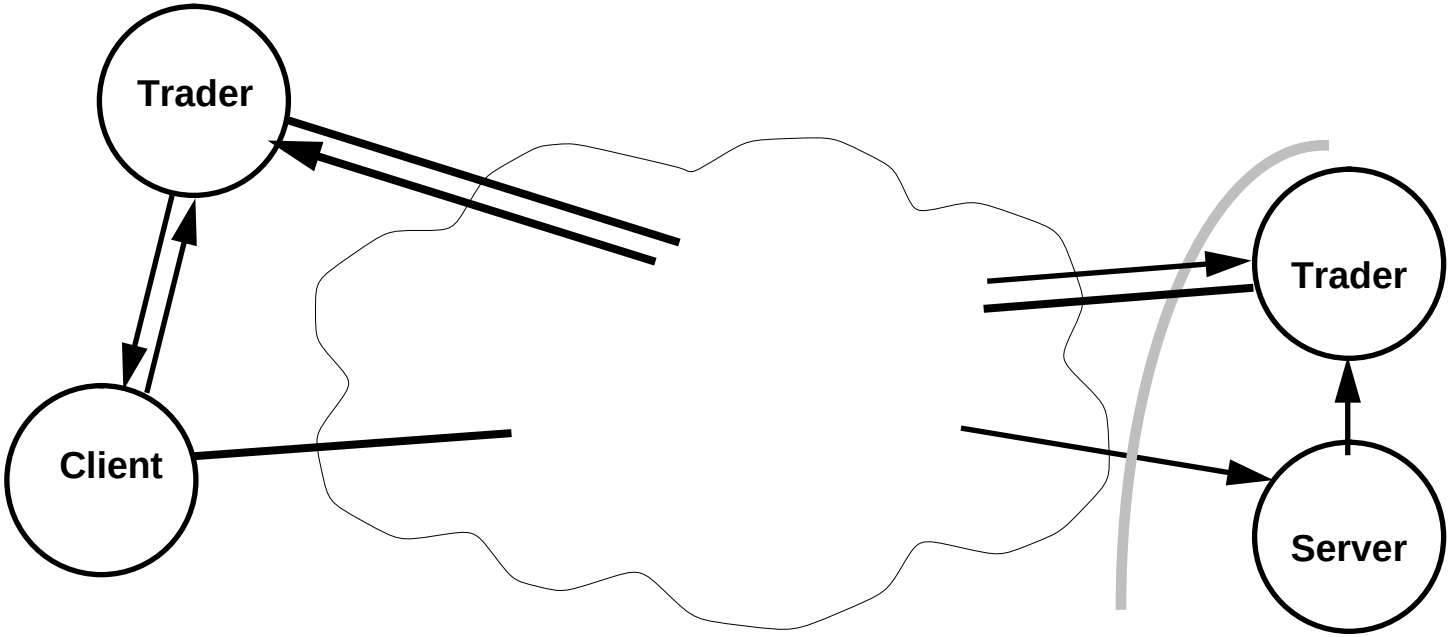
- *Controls can be enforced naturally at boundaries*
- *Distributed objects are ideal*
  - their encapsulation provides a boundary

Kinds of boundaries

- *There may be many kinds of boundaries*
  - Administrative
  - Judicial
  - Political
  - Technological
- *Openness entails bridging boundaries (in a controlled manner)*
  - not abolishing them
- *But we want these boundaries to be transparent to applications*

Federated Trading

- ***The traders are federated, but the client and server need not be aware of the boundary***



Different policies for different organizations

- ***Federation raises policy issues including***
  - **Authority**
  - **Billing**
  - **Security**
- ***Organizations that sell electronic services must be able to enforce their policies***
  - **they cannot rely on trust**

Federated Trading

- ***Distributed objects in the Computational Model already satisfy most of the needs of federation...***
- ***... but there are new issues***
  - **federation of naming contexts**
  - **technology boundaries**
  - **security**
- ***These issues affect all services, including the trading service itself***

Implementing Traders

- ***Traders can be implemented in several ways***
  - **as small, self-contained objects: optimized for locality and speed**
  - **as databases: optimized for large numbers of offers, and sophisticated query facilities**
  - **integrated with the naming service: optimized to share the federation facilities of the naming service**
- ***Whichever implementation is chosen, traders can be federated***
- ***The research community has used all these implementations and proved the scalability of traders***

Standards in Trading

- ***Standards work has been carried out in ISO/IEC using the ODP Reference Model***
  - **issued as DIS 13235 (ODP Trading Function)**
  - **Annex A includes a CORBA IDL Specification**
- ***OMG CORBAservices COSS RFP 5 is for a Trader of exactly this kind***

Summary

- ***Trading is necessary to support large distributed systems***
  - **it is the activity of choosing a service offer that matches the service requirement**
- ***The Trading service (trader) is provided just as any other service***
- ***A trader manages a database of service offers and matches requirements to offers***
  - **this matching is done on the basis of type conformance**
- ***Once a trader has introduced a server to a client, it plays no further part in the interaction***

Trading and Federation - more information

- *For a general discussion of trading and federation, see The ANSA Model for Trading and Federation (paper: AR.005.00; ftp: APM.1005)*
- *For naming issues, see The ANSA Naming Model (paper: AR.003.01; ftp: APM.1003)*
- *For security issues, see A Framework for Federating Secure Systems (paper: AR.008.00; ftp: APM.1008)*
- *For a discussion of Quality of Service issues in multimedia applications, see Integrating Multimedia into the ANSA Architecture (paper: TR.028.00; ftp: not yet released)*

Trading - Extra questions

- ***If Trading is a service, how does a client find the Trading service to start with?***
  - **Yes, there has to be a way to bootstrap the client. This can be done by building a Trader interface-reference into the infrastructure, by broadcasting, or some other special means. This is really an Engineering issue**
- ***How does Trading understand what the service offers mean?***
  - **It doesn't need to. It just needs to decide whether they match service requests**

Advanced trading - negotiation

- ***The 4-step approach described earlier will satisfy many needs, but not all...***
  - **...if several offers match, which one should be chosen?**
- ***Different offers may have a different quality-of-service, or cost***
  - **This will require more sophisticated patterns of *negotiation*...**
  - **... for trade-offs between quality and cost**
  - **... involving another service, a *negotiation service***
  - **the Trading service is neither accountable nor responsible for the quality of service described in service offers**

Advanced trading - different lifecycle epochs

- *The description so far implies that trading happens at run-time*
  - this is typical
- *Does it make sense for trading to happen at earlier epochs during the service development lifecycle?*
  - at application link time?
  - at application compile time?
  - at design time?
  - at specification time?

Advanced trading - compile/link time epoch

- *At compile/link time, special tools could be used...*
  - to test that a set of services could trade successfully
  - to determine an optimum set of bindings within this set
  - to replace the trading operations by this set of bindings
- *... to optimize a group of closely-related services*

Advanced trading - design/specification epoch

- ***Applications and service designers need to know what services already exist***
  - **to avoid re-inventing the wheel**
- ***This information is already available in the trading service***
  - **intelligent browsing tools can aid the designer**

Advanced trading - new service wanted...

- *Suppose the designer does not find the service they require...*
  - ...but they do not want to implement the service themselves
- *They can simply place a service request for this non-existent service*
  - ...like a Service Wanted advertisement
- *A special service can intercept this request*
  - and create a dummy service

Use of MatchMaking Service