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**

ANSA Phase III

A Designers' Introduction To Trading

Yigal Hoffner

Abstract

This paper discusses trading in the context of the process of setting up the co-operation between objects in distributed systems. The discussion concerns the large scale of future systems and applications, the need for co-operation across boundaries, and the lack of trust which results from the multiplicity of authorities.

The result is a model of co-operative trading which encompasses the process, the information and the possible configurations of trading necessary to deal with and exploit the complexity of large scale distributed systems. The model helps outline the problem space of trading and is used to indicate to designers of such systems what are the central issues which must be considered. The model also serves as a starting point for the discussion of the issues involved with the implementation of traders.

APM.1140.00.10 **Draft** 12th December 1994

Request for Comments (confidential to ANSA consortium for 2 years)

Distribution:
Supersedes:
Superseded by:

A Designers’ Introduction To Trading

A Designers' Introduction To Trading

Yigal Hoffner

APM.1140.00.10

12th December 1994

The material in this Report has been developed as part of the ANSA Architecture for Open Distributed Systems. ANSA is a collaborative initiative, managed by Architecture Projects Management Limited on behalf of the companies sponsoring the ANSA Workprogramme.

The ANSA initiative is open to all companies and organisations. Further information on the ANSA Workprogramme, the material in this report, and on other reports can be obtained from the address below.

The authors acknowledge the help and assistance of their colleagues, in sponsoring companies and the ANSA team in Cambridge in the preparation of this report.

Architecture Projects Management Limited

Poseidon House
Castle Park
CAMBRIDGE
CB3 0RD
United Kingdom

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

Contents

1 Introduction

1.1 Objective

Trading is an essential part of the process of setting up co-operation between objects in distributed systems. This report addresses trading in large scale systems in which the need for co-operation occurs across many types of boundaries, and has to respect the trust relationship between the multiplicity of authorities involved.

This report sets out a model of ***co-operative trading*** which encompasses information and services necessary to perform trading in large scale distributed systems. The model helps outline the problem space of trading and is also used to generate a check-list of issues which should be considered by designers wishing to implement trading in such systems. It thus serves as a starting point for discussing implementation issues. Further discussion of implementation issues can be found in [APM.1110 94], [APM.1091 93] and [APM.1177 94].

Other documents relating to trading are [RC.101 90], [ARM 93] and [APM.1005 93].

1.2 The necessity of trading in distributed systems

The trading process provides information about available services. Because distributed systems change and evolve, information about services must be timely, up-to-date and readily available. Trading is the process of obtaining such information and using it to discover compatible clients and servers. With suitable infrastructure support, the provision of service information allows:

- **deferred decision making**: this provides flexibility in choosing which service to use as late as possible (late binding). Clients do not have to know the exact services available at any point in time provided they can locate and use a service compatible with their requirements when it is necessary
- **deferred creation of services**: advertising potential services as well as existing ones facilitates the deferment of their creation to when and where they are needed. This avoids wasteful consumption of resources when they are not needed.

1.3 Benefits for future systems

The model of co-operative trading presented in this paper is part of a study by the federation group designed to:

- enable low cost, rapid deployment of new services and modification of existing services. This is provided by facilitating easy access to services as well as their description

- assist in the design of distributed applications where users, designers, implementors and vendors find out about, and locate services, as well as advertize their own. This can be achieved by providing trading functionality both at and before run-time, and by facilitating browsing with incomplete requirement specification and incomplete matching
- support commercially based co-operation between clients and servers of distributed applications. This can be achieved by incorporating a wider description of objects which includes management and remuneration aspects of applications for example, as well as dialogue and negotiation structures to achieve agreement among the co-operating parties
- support applications spanning multiple boundaries of different kinds whilst maintaining the guaranteed quality of service. This can be provided through the extended information about clients and servers and the dialogue and negotiation facilities. This is dependent on the support provided by the infrastructure.

1.4 Overview of the document

The following is a summary of the chapters in this document.

Chapter 1 contains this introduction.

Chapter 2 introduces the motivation for trading and an explanation of how the large scale of future distributed systems requires a more general approach to trading than that described in [APM.1005 93].

Chapter 3 provides a series of micro-scenarios showing the circumstances in which trading will be necessary.

Chapter 4 describes the stages necessary in setting-up co-operation between objects and thus sets the context for the process of trading.

Chapter 5 provides a process view of trading by describing what are the phases of trading.

Chapter 6 discussed the information the client and server have to provide in order to support trading.

Chapter 7 provides a view of how the phases of the trading process can be configured.

Chapter 8 is an implementation view of trading. It shows how enterprise issues such as performance and security affect the decisions which face the designer.

2 Co-operation in large scale distributed systems

2.1 Motivation for the model of co-operative trading

Trading is basically about exchanging information between service providers and consumers in order to find out whether they can, and wish, to co-operate successfully with each other. Trading is a common process carried out by individuals and organizations in every day life and covers many facets of the service provider and consumer and their environments. As more of the functions performed by people and organizations are subsumed by computers, there is a need to support such a process and information inside distributed systems. In such systems trading should enable:

- the trading of services which are inside the system. In this sense, computing resources are treated as commodities which can be traded like any other services
- the trading of services outside the system. In this sense the computer holds information about commodities outside the system and allows people and organizations to trade them.

In order to achieve the above in computer systems, users, designers, implementors and vendors require assistance in order to:

- find out about, and locate available services as well as advertize their own
- find out about the quality of services offered
- find out about how to use services that are compatible with their requirements.

The exchange of information and the match making activity necessary to ensure successful co-operation between service consumers and providers can be carried out at different times:

- design time
- implementation time
- instantiation
- run-time
- maintenance time.

Co-operative trading described in this paper is aimed at meeting the above needs in the context of large scale distributed systems and is seen as the linchpin of distributed systems of the future.

This model is an essential ingredient of the work of the federation group. Together with “Boundaries and Domains“[APM.1139 94] they form the basis of the information model for trading developed in “Information Model for Federation“[APM.1229 94].

2.2 Scope of co-operation

Co-operation may span organizational boundaries and therefore poses technical as well as non-technical problems in such areas as management, remuneration and security:

- objects wishing to co-operate will be developed separately and be subject to the control of independent authorities
- authorities may not trust any trading agents outside their domain of control.

Further problems are raised when examining the above mentioned issues:

1. Co-operation spanning boundaries points to:
   - a development process that is potentially distributed: the design and development of clients and servers can take place in different organizations, at different times, using different tools and environments. Divergence of designs and implementations is therefore highly likely
   - the management of objects and resources has to deliver quality of service that is potentially distributed. Policies concerning how things are done can differ between domains thus complicating attempts to deliver what is promised.
2. Independent authorities raise the need for:
   - dialogue, negotiation and agreement: systems and their independent authorities require negotiation to overcome the differences in institutional, remuneration, administrative and technical approaches. There is a need to reach agreements concerning the guarantees, responsibilities and liabilities which are given by each side of the co-operation. The problem of the representation of agreements in computer systems must also be addressed.
3. Issue of trust raises the necessity for:
   - clients, servers, or their agents to acquire sufficient and trustworthy information about each other to enable them to:
     — decide whether they wish to interact with each other. This involves testing for compatibility at different levels
     — operate with their counter-part to achieve the necessary effect in areas such as remuneration and management
   - carrying out certain processes (such as compatibility tests) in more than one place to ensure that the results are correct and consistent with local objectives and constraints. This will be further discussed in §7.4
   - independent trusted agents such as trading agents (as well as arbiters, authentication servers and electronic banks) where this is possible.

2.3 Relation with current work on federation

Current work in the federation group is concentrated on three areas [APM.1193 94]:

1. investigation of the architectural issues involved with the federation of large scale systems:
   - providing the architectural basis for the prototyping work
   - showing the general direction in which systems are going thus providing an evolutionary path for both the research and prototyping work.

   This paper fits within this category.
2. Construction of scenarios [APM.1064 94] and micro-scenarios [APM.1095 94] to motivate the design and implementation of prototypes, as well as provide a background to the development of the architecture.
3. Design and implementation of prototypes to aid technology transfer [APM.1091 93], [APM.1193 94] and [APM.1177 94]. Prototypes are derived from the architectural work and the scenarios, and based on technologies such as:
   - distributed system platforms: ORBIX [IONA 93a] [IONA 93b]
   - database technology: SQL [ANSI X3.135 92]
   - object oriented approach [RUMBAUGH 93].

3 Co-operative trading scenarios

The importance of clients and servers having the appropriate information about each other, the means to obtain it and the way it is used for testing their compatibility at several levels, are demonstrated in the following examples.

Note that the issues raised and their order corresponds to those discussed in the model of co-operative trading as described in chapters §6 and §7.

3.1 Information issue scenarios

The following sections outline scenarios concerned with different trading information issues.

3.1.1 Symmetric information exchange between the client and server

Clients will want to obtain information about servers and similarly, servers will want to have information about potential clients (§6.2).

A client wishes to use a service of type X and requests a trading process to locate a service which matches the specification. The trading process locates such a service if it exists. The server providing the service is not willing to co-operate with any client unless the client provides some information about itself. In particular the server wishes to know with a known level of confidence:

- the identity of the client and its credentials
- the ability of the client to pay for the services consumed.

3.1.2 Comprehensive information exchange

The information exchanged between the client and server includes more than just interface signature definitions (§6.3).

The information required by the client and server to allow them to make a decision concerning co-operation includes a description of functionality, as well as issues such as the guaranteed quality of service, management and remuneration, or at least information about where to get such information.

3.1.3 Client and server requirements/conditions and offers/guarantees

Clients have expectations of servers which they state in their requirements, while servers make statements about what they offer. Similarly, servers have expectations of clients which they must state, and client must make clear what they offer to ensure successful co-operation with the server (§6.4).

Examples of what a server may require from a client are:

- specify the behaviour of the client (concerning such issues as performance, dependability and security, for example). This may be crucial where the quality of the service depends on the way in which the client behaves
- the guarantees[1] the client offers with regard to its relevant behaviour
- client's ability and willingness to pay
- penalty clauses in case the guarantees are not met.

3.1.4 Servers register offers - clients register requests

Clients can register their interest in services in a similar fashion to servers registering their service offers (§6.5).

Clients who are looking for a service either as an on-going concern over long stretches of time may want to register their interest in a similar way to that done by service providers. Such a repository of client request information will allow servers to search for potential clients. This use of trading resembles the notion of tendering and will allow servers to compete for service requests. In order to facilitate the binding of the clients to the servers the client has to provide an interface to which the information from a successful match will be sent to.

3.1.5 Browsing and matching on partial information

Obtaining trading information may be an on-going process distributed in time and space, based on vague requirements of a service, where the objective is to obtain information other than just the binding information to a specific server (§6.6).

Browsing of repositories which contain information about clients and servers is required for different purposes:

- the designer of a client may have a vague idea of the type of functions which services of type X provide. It may also be that there are slight but important variations. The client designer wishes to investigate before committing the design in one direction or another or would like to incorporate a number of options.
- the designer may change the specification of the client it is designing after browsing through the specification of servers conforming to service type X.
- the designer may negotiate a change in the way the service is provided by interacting with the management of the service. This may involve a change in the design of the server or just a re-configuration of resources to provide a higher quality of service in exchange for more pay.
- the interaction with the trading process may take the form of a dialogue in which successive interactions allow the designer/user to refine its view of the services provided.

A trading process can provide different views of the information available in different OMG repositories, for example.

1. Note the distinction between the specification and the guarantees. Specification describes the expected behaviour of the system primarily from a technical point of view. Guarantees are promises within a legal context concerning the validity of the specification.

3.1.6 Incomplete match between client and server

An approximate match between the client and server may be sufficient with regard to issues where interception technology can help overcome the differences (§6.6).

For example, if the format of the information is different (for example in a video on demand application - PAL vs NTSC) from the one expected but transformers are available as services.

3.1.7 Trading at different epochs

Trading activities do not have to be confined to run-time and may take place at different stages of an object's life cycle (§6.8).

For example, the designer of a client wishes to investigate what range of services exist in a certain environment before committing to a specific design. Of particular interest to the designer is the range of quality of service guarantees offered by the different services as well as the assurances given by the service providers about the guarantees.

3.2 Configuration issue scenarios

The following sections outline scenarios concerned with different trading configurations.

3.2.1 Surrogate trading

In a large system there will be multiple trading processes each having access to information about different parts of the system. Such trading processes may have to interact and trade on behalf of each other (§7.2).

This may arise for a number of reasons. For example:

- specialized trading processes may exist for special areas of expertise, for example, for control applications or real-time applications
- for commercial reasons, some repositories of trading information may only allow access to them through specific trading processes and not others.

In order to extend the trading scope of a system (in order to advertize or find out about services on a larger scale), traders will have to interact with other traders on behalf of their clients. Also, traders can trade traders thus allowing networks of traders to evolve.

3.2.2 Cascaded trading

A client-server configuration may in fact be a complex set-up of multiple clients and servers. Setting-up such a configuration may require relating successive trading processes between multiple trading processes (§7.3).

A client wishes to consume a service whose use requires some intermediate services. The end to end trading process has to be broken down into a series of local trading processes between the cascaded clients and servers. For example, in order to gain access to a 3D simulation environment the client program has to access a remote database as well as a super-computer. The links between them are provided by a number of different networks. The overall trading

process has to be broken down into a number of separate trading stages between the different entities involved.

3.2.3 Peer trading

Peer trading is a configuration in which independent parallel trading processes interact with each other, acting on behalf of the client and server respectively (§7.4).

Carrying out the trading process concurrently in two peer trading processes is similar to the situation in which two law firms work out the agreement between two organizations. It is unacceptable in legal situations to have a single law firm doing so. In cases where the relationship between the client and server is sensitive and any dissatisfaction with guarantee provision may prove costly or disastrous, there is a need for each side to be represented by an independent trading process, at least for the match/negotiation process.

This may arise in one of the following situations:

- a service provider is sensitive about its well-being and reputation. Given that a failure may originate in the client, it wishes to minimize the possibility of this happening. It will therefore want to conduct its own checks of the compatibility of the client with its server. It may also know some things about the service that it may not wish to make public to external trading processes
- for commercial reasons and availability of advertizing information, the service provider may wish to advertize the service in other traders as well, even though it does not trust their trading capability
- some trading processes may be able to carry out the trading process but have no authority to endorse the results. This is similar to a travel agency which can take bookings but cannot issue the tickets
- the trading process of the service provider may be expensive leading to a preference to use a local but inexpensive trader.

The peer trading processes will have to negotiate the matching of the client and sever requirements.

3.2.4 Centralized trading

Where trust in a third party trading agent exists, the result can be a centralized configuration of the trading process (§7.5).

Over time, services specializing in the various aspects of trading will emerge. Their use is likely to become common provided they will be backed by reliable organizations which can provide sufficient guarantees concerning the quality of the offered trading service.

3.2.5 Intra-trading chain

The trading process can be broken into stages which may be carried out separately and at different times (§7.6).

This configuration can support different distributions in time and space of the stages of the trading process as suggested in §3.1.7.

For example, the search for compatible client-service configuration may be carried out by one process, while the test for compatibility may be carried out by another. The negotiations with the candidates may be carried out by yet

another process. The actual signing of the agreement may be done by the representatives of the client and server.

3.2.6 Multi-level chain

Different aspects of the client and server descriptions (as described in §3.1.2) may be compared and tested for compatibility separately and at different times (§7.7).

Each trading process which concerns a specific compatibility issue can rely on the compatibility and agreements reached in previous trading stages.

Organizations may, for example, decide that the negotiations concerning remuneration for certain classes of services will be conducted outside the computer system prior to any attempt to match any specific service and clients. Another area in which agreement prior to any run-time trading may be useful concerns management.

Another example is that of matching of client and server in terms of semantics and interface signatures can be carried out at compile-time, leaving the matching of the comms for later. In fact, this is what the trading and binding processes in ANSAware [ARM 93] do.

3.3 Layered trading

Each layer of trading builds on the established compatibility and binding of the previous layer (§7.8).

In a home shopping application, the communication infrastructure compatibility and binding, platform and shopping application compatibility and binding will have to be established before the users can trade for the goods they are seeking. Such a sequence can be regarded as a layered process of trading.

4 Setting-up co-operation

This chapter describes the stages necessary in setting-up any degree of coupling between objects as the process of splicing. Splicing stages are often aggregated into phases such as: agreement, trading, resource set-up and interaction.

Co-operation is described as encompassing four degrees of coupling: inter-connection, inter-working, inter-operability and integration. Splicing can be applied to achieve any of the four degrees of coupling between objects.

4.1 Setting-up the coupling between objects - a process view

The process of setting-up any degree of coupling between objects is called ***splicing***[1].

In order to establish purposeful, meaningful and successful coupling of any degree it is necessary to go through the following stages of the splicing process:

- reach **agreement** between the authorities of the service provider and consumer concerning the wish to co-operate[2] (*dialogue and negotiation*)
- **advertize** and **search** in appropriate repositories (e.g. name services, X.500 directories [ALMGREN 91], OMG repositories [OMG 92], DCE CDS [OSF 92] ANSA [APM.1003 93] and ANSAware trader [ARM 93]) to store and locate information about existing or creatable objects (*store and retrieve/export and import*)
- **compare** specifications of offers and requirements made by service providers and prospective clients with a view to their compatibility (*compatibility/match-making*).
- **negotiate** between the involved parties where options are available or modifications are possible to the offered or required services (*dialogue and negotiation*)
- **select** objects for co-operation based on the comparison of their specifications either by finding a complete match or an incomplete one which can be resolved
- **resolve** any resolvable barriers to co-operation. This may involve the use, or the dynamic creation of interceptors/gateways/adaptors (*interception*). It may also involve the dynamic creation of services or the re-configuration of existing ones (*configuration management*)

1. Splicing is defined as joining two ends of rope by untwisting and interweaving the strands of each [HAWKINS 79].

2. This is not to be confused with the agreement between the parties concerning the details of the coupling set-up itself, for example, which communications protocols to use.

- **configure** the engineering and communications infrastructure necessary for the co-operation between the objects (*binding*)
- carry out the **interaction** between the objects (*invocations*).

4.2 The major phases of splicing

By separating the concerns of the different stages of splicing described in §4.1, and grouping related ones together, it is possible to discern four major phases:

- ***agreement***: the basic agreement to co-operate. This usually encompasses the agreement to trade
- ***trading***: is primarily concerned with clients and servers finding out sufficient information about each other in order to decide whether they wish and can co-operate. Trading involves the *advertizing*, *searching*, *compari*ng, *negotiating* and *selecti*ng stages of the splicing process
- ***binding***: the set-up of resources to enable the interaction to take place with the agreed quality of service. This involves the *resolution* and *configuration* stages of the co-operation set-up process
- ***interaction***: involves the actual consumption of services as well as the remuneration and management processes, for example.

4.3 Co-operation set-up between objects - a layered view

The process of **co-operation set-up** is about the steps necessary to ensure that effective (i.e. purposeful, meaningful and successful) interaction between objects take place. Co-operation implies a degree of coupling between the objects which encompasses the following increasing degree of complexity (Figure 4.1):

- **Inter-connection**: basic data transfer
- **Inter-working**: providing a means of interaction in terms of dialogue structures, for example, remote procedure call (RPC) including location and access transparencies
- **Inter-operability**: additional transparencies such as replication and migration, and quality of service guarantees such as security, dependability and performance
- **Integration**[1]: application semantics compatibility. This degree of co-operation is the hardest to achieve and the most difficult in terms of the issues it raises.

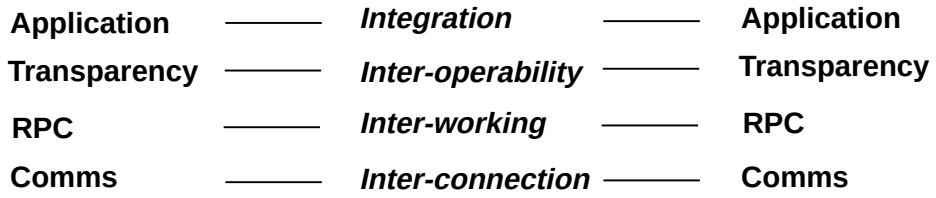
4.4 Observations about the process of co-operation set-up

Some important observations can be derived from considering the stages of splicing:

1. There are multiple *levels of dialogue*, negotiation and agreement, possibly taking place at different stages. There has to be an overall agreement

1. The word integration is used to express the coming together of two separate entities to co-operate, not necessarily by bringing them together physically. In other words no notion of centralization is implied here.

Figure 4.1: A layered view of the degree of co-operation between objects



between the authorities of different domains as to whether any co-operation is desired at all. This has to be accompanied by further agreements concerning the specific application which requires the co-operation of objects in different domains.

2. The co-operation set-up process can be applied *recursively* to some of the stages within it. For example, the configuration of resources necessary for co-operation (i.e. binding) may itself require comparison, selection, resolution and configuration stages to determine which transparency mechanisms and communication protocols to use for co-operation.
3. The stages need not be *co-located in time or space*, nor be carried out by the same agents in the same domain. The stages of the co-operation set-up process may take place at different points in the application life cycle, by different agents or tools. For example: at design time by a designer or a development tool, at implementation time by a pre-processor or compiler, at initiation time by a linker or a configuration management tool, or at run-time by traders or management agents.
4. Some stages may be carried out locally or remotely. For example, some trusted compilers or trading agents may reside locally or remotely.
5. In some cases the *sequence* in which the steps are carried out can vary. For example, the dialogue and negotiation concerning agreement to co-operate may take place before any of the other stages, or just prior to interaction. It may in some cases be implicit in the willingness to interact.
6. Different *information models and representations* of the necessary information will be used by the different agents involved in the process. There is reason to assume, given the heterogeneous nature of distributed systems, that different systems will use different models, representations and storage mechanisms for the processes involved[1]. Overcoming these differences suggests the application of the same process recursively as described in (2).
7. The co-operation set-up process ties together *trading*, *interception* [APM.1042 93], *configuration management* and *binding* [APM.1314 94] as different parts of the same process, thereby providing a more general context for the discussion of these topics.

4.5 Co-operative trading and splicing

The following chapters concentrate on the process of **trading** which is primarily concerned with the advertizing, searching, comparison, negotiation

1. The multiple IDLs currently in existence [APM.1042 93] are an example of such a trend.

and selection stages of splicing. The result is a ***model of co-operative trading*** which discusses:

- *process* issues: the phases of the trading process
- *information* issues:
  - — the information necessary for a comprehensive trading system
  - — the means and ways of accessing this information
  - — ways of using the information
- *configuration* issues - the different ways in which the trading process and information can be organized. This results in a number of configurations:
  - (i) surrogate trading
  - (ii) cascaded trading
  - (iii) peer trading
  - (iv) centralized trading
  - (v) intra-trading chain
  - (vi) multi-level trading
  - (vii) layered trading.
- *implementation* issues: Information and configuration issues outline the space of possible implementations. This section describes the link between enterprise type considerations and the choice of implementation from those outlined in earlier sections.

4.6 Trading processes and trading agents

The **trading process** is the sequence of actions which have to be taken in order to find a compatible client/server. A **trading agent** is an entity engaged in the process of trading or part of it on behalf of a client or a server. The distinction between trading and the agent(s) carrying it out is important as the stages of the trading process can be carried out in different places by different agents, even by the client or server objects themselves. In this paper, the term trading agent does not necessarily imply an agent which is independent and outside the client or server. The agent or part of it may reside inside the client or server.

5 Process view of co-operative trading

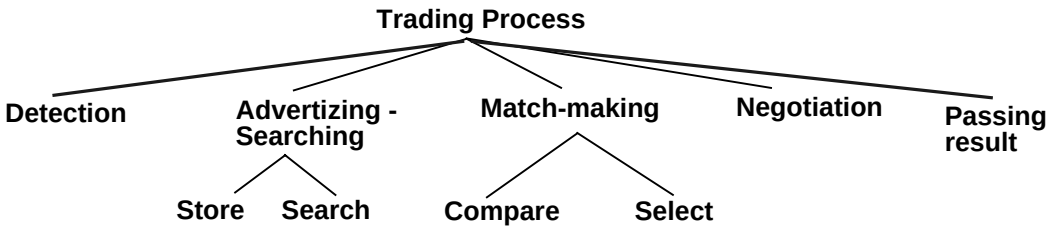
5.1 Introduction

This chapter forms part of the co-operative trading model description. It presents a description of the stages involved in the trading process.

5.2 The stages of the trading process

Trading consists of several major stages (Figure 5.1):

Figure 5.1: Stages of the trading process



1. **detection** of the offer or request for co-operation from the client or service
2. **advertizing/searching**: the storing or the retrieval of the request or offer of service
3. **match-making**: comparison between the offer and the request specifications and selection of options
4. **negotiation** process [GRIFFETH 92] aimed at:
   - matching non-functional elements in the specification, e.g. constraints in ANSAware [ARM 93]
   - either:
     - setting an agreement to co-operate (within the appropriate institutional framework)
     - or agreeing not to co-operate.
5. **passing** of the results to the next stage.

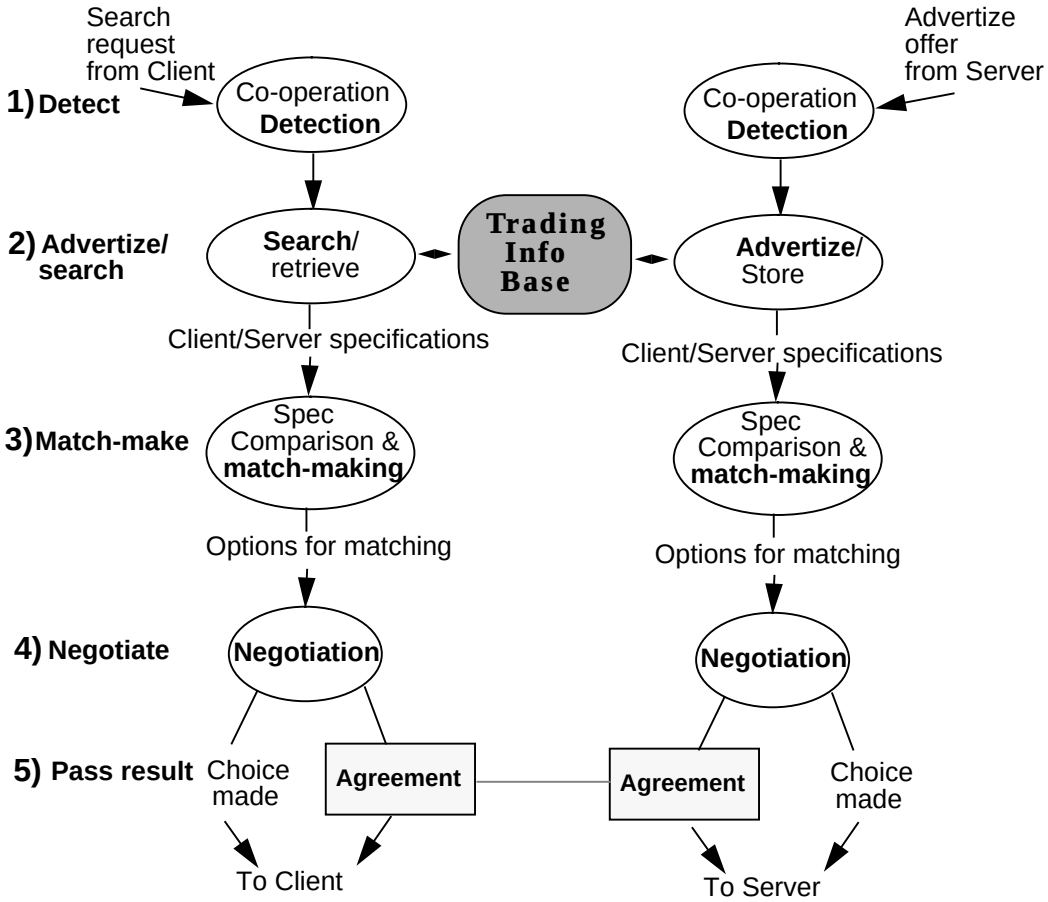
These stages are shown in Figure 5.2.

All of the above stages have to be carried out according to the policy of the trading process, the client and the server.

The result of the negotiation, i.e. the agreement and the information the objects have about each other can be passed to the client or server who in turn can pass it to:

- a *binder* in the case where no barriers for co-operation are present

Figure 5.2: The stages of the trading process



- an *interceptor factory*[1] if interceptors are necessary to enable the co-operation [APM.1142 94]
- a *configuration manager/factory* if the service/client have to be re-configured or instantiated.

Note the similarity of the trading model stages to those of the interception model [APM.1142 94]. This is because the two processes are basically concerned with the same problem but with different aspects of it. Trading is primarily concerned with identifying compatibility and finding compatible objects, while interception is concerned with identifying incompatibility and how to overcome it.

1. The task of interceptors and binders is similar. Binding can be viewed as interception aimed at overcoming the distance gap by using some form of communication.

6 Information view of co-operative trading

This chapter forms part of the co-operative trading model and is about the information required to provide a comprehensive trading service as well as the issues involved in how this information can be used.

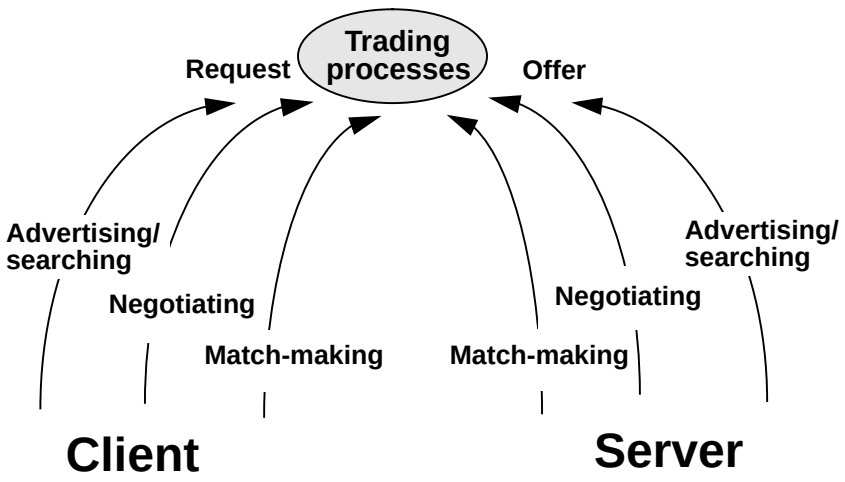
Check-list items are included to highlight the pertinent issues which arise from the model described in the following chapters.

6.1 Information for the different stages of the trading process

Each one of the stages of the trading process (§5.2) requires information in order to carry it out (Figure 6.1):

1. **client/service advertising/searching information**: requirements given to the trading service concerning the manner in which the advertising and searching is to be carried out. For example, the trading service may be requested by the service provider to restrict the visibility of the service, to only advertise it in special circumstances and with instructions to provide the advertiser with certain information about the potential customers. Similarly, potential service consumers may wish to restrict the search for a service to a geographical area, for example.
2. **client/service description information**: a specification of what the service and client offer and require from a potential co-operator necessary for the match-making process.
3. **client/service negotiation information**: options, procedures and agents for negotiation. This requires information about what negotiations are possible, what are the interfaces through which they should be conducted and what information is necessary to do so.

Figure 6.1: Types of information required for trading

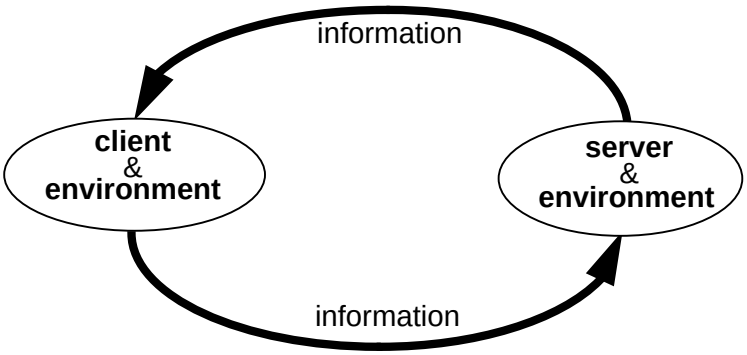


The rest of this paper concentrates on the information required for describing clients and servers with a view to match-making them. Although advertizing and negotiation issues are important they are outside the scope of this paper.

6.2 The symmetric information exchange between the client and server

The basic information model for trading is symmetric: the client and its environment must have information about the server and its environment, and vice versa (Figure 6.2).

Figure 6.2: Structural symmetric of the information flow in trading



The major reason for the two way exchange of information concerns:

- the commercial reality of large scale systems using services and paying for their use. The small scale of provision and consumption of services within a department or an organization not requiring accounting, billing and paying no longer holds in large scale systems
- issues of trust: it is common when buying goods to provide the supplier with some identification which links the buyer to some paying capability. This may be the availability of cash, a credit card or familiarity and trust
- the need to provide overall management across boundaries requires the client and server management agents to interact. This requires the server and client to provide each other with information about their respective management capabilities and restrictions.

The two way exchange helps the server find out about the client and vice versa, to help provide some assurances concerning security, dependability and remuneration capability.

Check list

- ***Does your service and its environment require any information about the client?***
- ***Consider the way in which the import and export operation arguments and results are designed and their impact on the trading process.***

The implementation of the model may take many forms and does not have to preserve the symmetry in all cases. This is discussed in the following sub-sections.

6.2.1 One way information transfer

In special environments or circumstances the model may be simplified to include only the server passing information to the client. For example, the server may not be concerned with who is invoking it when remuneration and security issues are not important. Some public or free services may belong to this category.

6.2.2 Different routes and strategies of information exchange

The route used to pass information from the server to the client is not necessarily the same as the one through which the client uses to pass information to the server. For example, a client may get information from a trading agent about the server's billing strategy while the client may be required to send billing information about itself directly to the server. From an information point of view "what" information is needed matters; how it is sent or who sent it is immaterial.

6.2.2.1 Request-reply coupling

In general there are two ways of passing information from a trading agent to an object:

- reply parameter of an invocation asking for information
- invoking an interface supplied as a parameter by the invocation asking for information. The interface can belong to the object, a management entity, a factory or any agent acting on behalf of the requesting object.

The choice between these two options will influence the design of the trading agent as well as the object interacting with it. For example, the way in which the Import() and Export() operations are designed will be influenced.

Check list

- ***Does the import and export operation structure fit your requirements?***

6.2.2.2 Registration of interest

An object can register with a trading agent that it is interested in certain events, for example, any new offers or requests for a service of a specified type and attributes. Alternatively it may be interested in their withdrawal. The object will have to supply an interface reference which the trader can invoke when such an event has occurred. Such an interface reference can be used to notify the object of certain events of interest.

Check list

- ***Does your application require to be notified of events taking place in the trading agent?***

6.2.2.3 Passing interface references

An interface reference is a structure which carries information necessary for the client to establish a binding to the server and to subsequently invoke it.

As will be discussed in §6.8.1, not all the trading process stages have to be carried out at the same epoch or by the same agent. The passing of interface

references is a special example where the information may be passed to the client either directly from a server or through a third party. Any passing of interface references or interaction between objects should be the consequence of trading conducted at earlier stages either implicitly or explicitly.

6.3 Comprehensive information exchange

The successful co-operation of service providers and consumers involves finding suitable objects to interact with, and then resolving any barriers which arise due to incompatibility[1]. The compatibility checks necessary to ensure successful co-operation are concerned with a wide range of functional and non-functional differences such as [APM.1139 94] (Figure 6.3):

- authority: contractual/institutional relation
- administration: management relation
- remuneration: accounting, billing and paying relation
- infrastructure: engineering and technology relation
- application (client-server): object (construction) and interface (interaction) descriptions relation[2].
  The reasons for including the object description are:
  - as an object may support multiple interfaces it may not be sufficient to describe interfaces only
  - to provide evidence to show that claims concerning quality of service, for example, can be supported by the design/construction of the object
  - to provide information necessary to determine the portability and migratability of an object
  - the language in which the application is written is important to determine the language mappings necessary for stubs in the case of a remote invocation.
- modelling: semantics and naming relation.

Check list

- ***What information is required from the client in order to establish successful co-operation?***
- ***Can the available framework help in describing the information?***

6.3.1 The need for a framework for deriving trading information

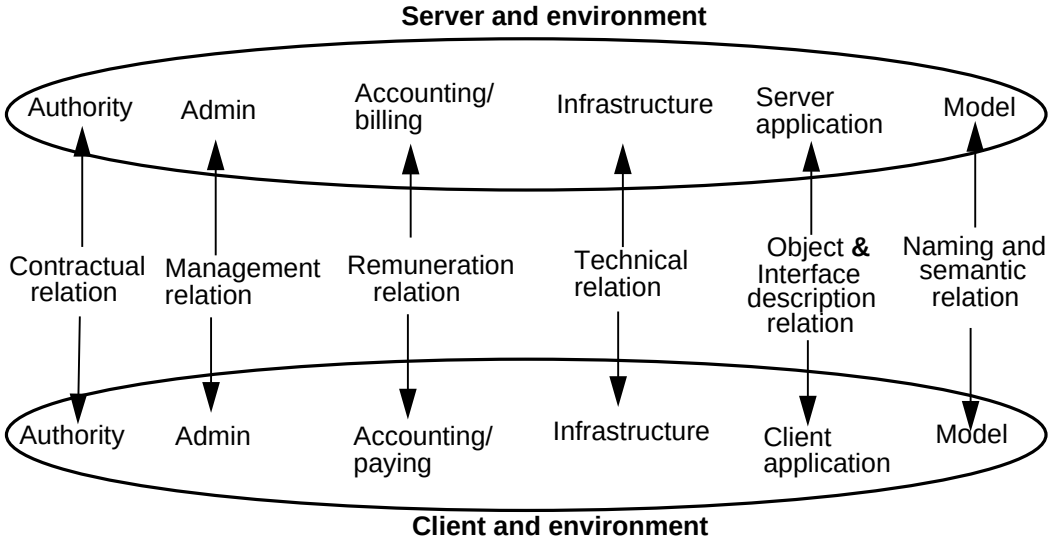
To provide a trading service it is necessary to do the following:

- determine the categories of information necessary for the client and the server in different application domains and environments

1. An increasingly important problem in large, federated systems, such as the public telephone network, is adverse feature interaction. A necessary step toward detecting and avoiding feature interaction problems is to have available a sufficiently complete specification of the services involved.

2. Editorial: It may be expedient to separate these two issues at a later stage.

Figure 6.3: Categories of information necessary for successful co-operation



- develop information models within these categories
- use the models to:
  - define the properties which are needed to specify objects
  - decide how to implement the models using available database technology.

The above list of categories (shown in Figure 6.3) can be used as a starting point for such a framework. Attempts to develop the framework are described in [APM.1139 94] and [APM.1229 94].

Although each application domain and possibly each specific application will have some differences requiring additional information, it is expected that most applications will have sufficient similarities to make such a framework useful. In any case, the framework should be flexible and extendable to accommodate variations.

6.4 Client and server requirements/conditions and offers/guarantees

The relation between a client and a server in terms of the guarantees and conditions they impose on each other is shown in Figure 6.4.

The specification of clients consists of:

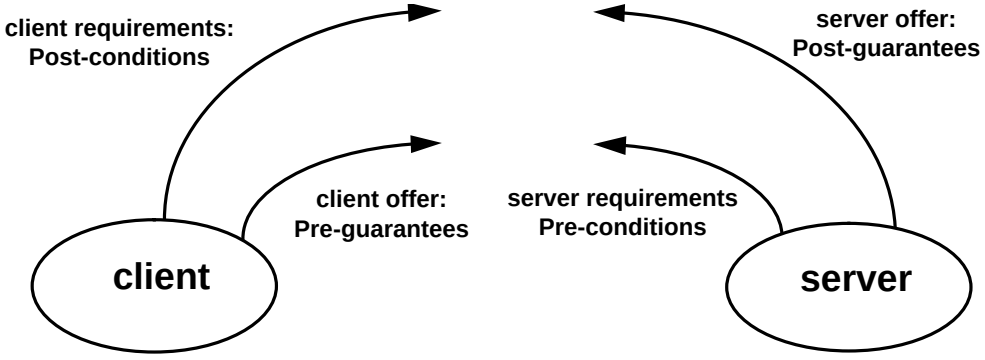
- client *post-condition*: client requirements of the server
- client *pre-guarantee*: client offer to the server in way of behaviour - how the server will be invoked, validity of the data passed to it, etc.

The specification of servers consists of:

- server *post-guarantee*: the promises which the server makes about its behaviour and how it will respond to invocations
- server *pre-condition*: the requirements the server has with regard to the client behaviour.

The client's *post-condition* must be fulfilled by the server's *post-guarantee*. The servers *pre-condition* must be fulfilled by the clients *pre-guarantee* (Figure 6.4).

Figure 6.4: Matching of client and server respective requirements and offers



Check list

- ***Is your compatibility/match maker capable of conducting a two way compatibility test between the client and server?***

6.5 Servers register offers - clients register requests

In addition to service providers advertizing their offers, clients may wish to advertize their request for a service as an on-going concern. This requires an information base of clients requests in addition to that of service offers. The process of trading then shifts its emphasis from searching for a service when a client request comes in, to searching for a potential client when an offer is registered.

Check list

- ***Is there a need for a client to advertize its requirements in a similar manner to a server but on a more permanent basis?***

6.6 Browsing and matching on partial information

Match-making on **incomplete information** is necessary to support the "shopping" model [APM.1162 94] [APM.1193 94]). By incomplete information we mean that a request for a service will not have to specify every aspect of the service it is interested in, only those it knows and is confident about. The more incomplete the specification, the larger the set of services given back.

Two different types of situations may arise with respect to what an object may want from the trading process. These cases revolve around:

- how complete and exact the specification of the required object is
- what information one object wishes to have about another.

Two examples demonstrate the difference:

- reference seeking: the client provides the trader with a specification of the required service, and the trader provides to the client an interface reference to a matching service provider
- information seeking: the client provides the trader with a specification of the required service, together with a specification of what type of information about the service is needed, and the trader provides to the client the specified information about all service providers which meet the partial service specification.

The dialogue in the trading process will enable successive refinement of requirements: a designer of an object may start with a vague notion of the services her objects may require and refine them as more information is obtained through trading

Check list

- ***Do you see your requirements as a client designer changing as you discover more about available services?***
- ***Do you need information other than binding information when trading?***

6.7 Incomplete match between client and server

Trading need not look for a **complete match** between clients and servers, but only for a “sufficiently close match”, where the difference can be bridged by interception [APM.1042 93].

What constitutes a “sufficiently close match” is an open question at the moment.

Check list

- ***What interception technology is at available to you which can be used in conjunction with trading when detecting incomplete matches between clients and servers?***

6.8 Trading at different epochs

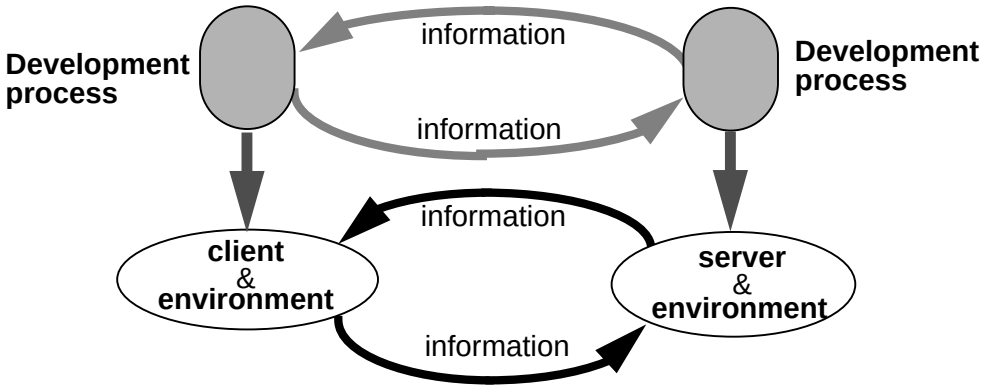
Trading takes place throughout the life cycle of clients and servers, not just at run-time (Figure 6.5).

The two extreme ends of the spectrum are carrying out all the entire trading process before run-time as opposed to carrying it out entirely during run-time.

Check list

- ***At what epochs do you require trading information and in what capacity/role?***
- ***What are the trade-offs for early versus late trading in your application domain?***

Figure 6.5: Trading takes place throughout the life cycle of clients and servers



6.8.1 Different epochs of information exchange

Information is not necessarily conveyed from client to server and vice versa at the same time. For example, a designer of a client may get information about the signature of a service from a design repository during the design phase of the client, so that the client can be designed to match that signature. Later the client may provide information about the interface it needs at bind time (of course, some information about the interface the service offers would have to be available again at this time to enable binding). From an information point of view, it is immaterial exactly when the information is sent.

Information from client to the server can be passed after a client has been given an interface reference to the server. For example, with the first invocation.

7 Configuration view of co-operative trading

This chapter forms part of the co-operative trading model description. It presents a description of the different ways of configuring the process of trading. The need for the different configurations arises as a result of:

- the need to link existing trading processes and trading information bases
- issues of trust, efficiency and optimization which motivate the distribution of the trading process in time and space.

The chapter presents a number of trading configurations:

- surrogate trading
- cascaded trading
- peer trading
- centralized trading
- intra-trading chain
- multi-level trading
- layered trading.

7.1 Distributing trading information and process

A useful notion in developing the model of co-operative trading is the separation between the information required for the trading process and the trading process itself. This may subsequently manifest itself in the separation of the agent(s) implementing the trading process and the database containing the object description.

Given the distinction, there are two major ways in which trading can be distributed:

- the information required for the process can be distributed
- the process can be distributed and/or replicated.

The following sections outline the different ways in which trading can be configured.

7.2 Surrogate trading configuration (distributing the trading information)

The distribution of the trading information can be done at two different levels:

- the information repositories can be distributed so that a single trading process may have to use several information repositories (Figure 7.1). This aspect of trading is not discussed in this paper
- often, to each trading information repository there is attached a trading process who is responsible for its management, providing the only form of access to it. The connection between the trading processes means their

ability to interact with each other on behalf of their clients in order to broaden their scope of trading. This is called **surrogate trading** and done by using Export/Import[1] operations to obtain information about clients and servers outside their scope (Figure 7.2). In this case one trading process performs trading on behalf of another trading process.

Figure 7.1: Distributing the trading information to several repositories

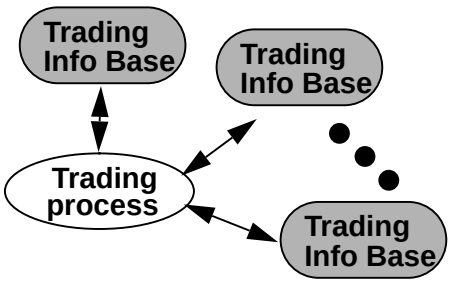
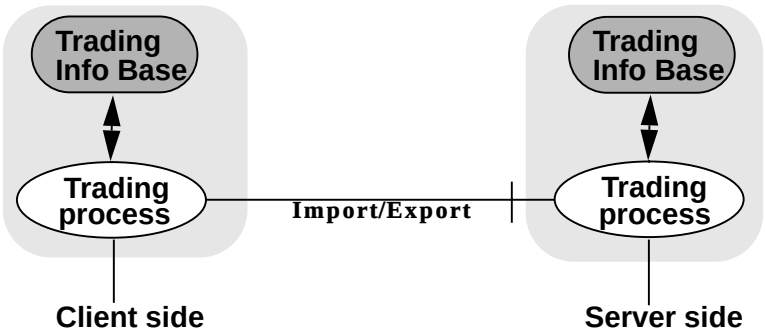


Figure 7.2: Surrogate trading - providing access to distributed trading information



Surrogate trading is likely to be one of the most common forms of trading in large systems. This will be due to the fact that many trading information repositories are likely to emerge for different purposes and it would be useful to be able to use them in different combinations and ways. It is also possible that some of the information in these repositories will be used for things they were not necessarily designed for in the first place.

Check list

- ***How likely is your trader to interact with other traders?***
- ***Are these traders likely to be in different management domains?***

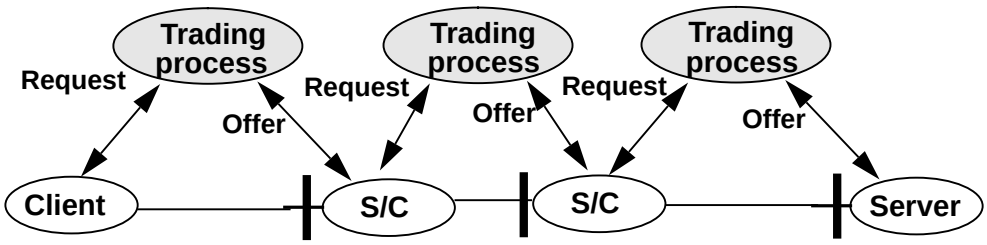
7.3 Cascaded trading configuration

Where service provision involves multiple clients and servers (Figure 7.8), a complex trading process may have to be carried out. There are different ways of approaching this problem:

1. The Import() operation initiates a search on the space of service offers previously Export()'ed by service providers.

- the trader of the original client may supervise the chain of trading processes
- the trader of the original client will initiate the sequence without being directly involved in the stages themselves.

Figure 7.3: The cascaded trading configuration



Several questions arise:

- how are possible delays in the response of each stage to the previous trading stage dealt with?
- should stages in the chain be aware of the fact that they are part of a larger chain?

Check list

- ***Does your client-server configuration consist of complex chains of clients and servers?***
- ***What is the client/server - trader relationship needed to facilitate the delays when each stage sets-up its next link?***
- ***Is the overall quality of the service provided have to be maintained in spite of the multiple stages involved?***

7.4 Peer trading (distributing the trading process)

7.4.1 Issues of trust

There are two problems involved with trading where trust is an issue:

- no trust exists for the information on objects given by another trading process
- no trust exists of the trading process carried out by another agent. This may manifest itself in not trusting any or some of the stages taken by the trading process. For example there may be a lack of trust in:
  - — the compatibility test
  - — the search and selection stages carried out according to the trading policy as requested by clients and servers. There may also be a lack of trust in the policy of the trader itself (for self promotion reasons, commercial competition)
  - — the information provided for resolution of barriers to co-operation or to configuring and setting-up the necessary mechanisms (interception or binding), as derived from the trading information

— not wishing to expose an interest in particular information.

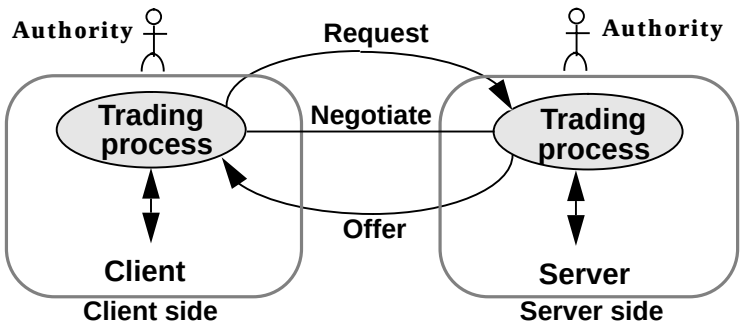
The first problem requires means for authenticating the information. This can be tackled by going to a trusted third party to check the information or to communicate with the management domain of the advertising object. This may ultimately involve interaction between people outside the system.

The distributed approach seems at first sight unreasonable. However when two companies try to reach an agreement they will not hire one lawyer to represent both but two lawyers, who will interact with each other on their behalf. Much of the work carried out by the lawyers will be duplicated but with a bias towards their client. Similarly, the two trading processes in figure 7.4 duplicate some of the effort but provide their respective clients the confidence they seek.

7.4.2 The peer trading process scenario

The trading process can be carried out concurrently in the two domains by two trading processes where each trading process is associated with a domain (Figure 7.4).

Figure 7.4: Peer trading involves co-operation between independent trading processes



Combining Figure 7.4 with the stages of the trading process described in §5.2 results in Figure 7.5.

The trading processes know and interact with each other to pass service offers and requests between them. For that it is necessary to have a prior stage in which the two trading processes are brought together by a co-operation set-up process as described in Chapter 4.

Check list

- ***Do you have sensitive applications which require a high level of security and have to rely on trusted traders?***

7.5 Centralized trading configuration

It is sometimes possible to carry out trading separately by a trusted third party process (Figure 7.6). The result is the trading configuration which appears in many past ANSA documents such as [APM.1005 93] and [ARM 93].

Figure 7.5: The stages of the peer-trading process

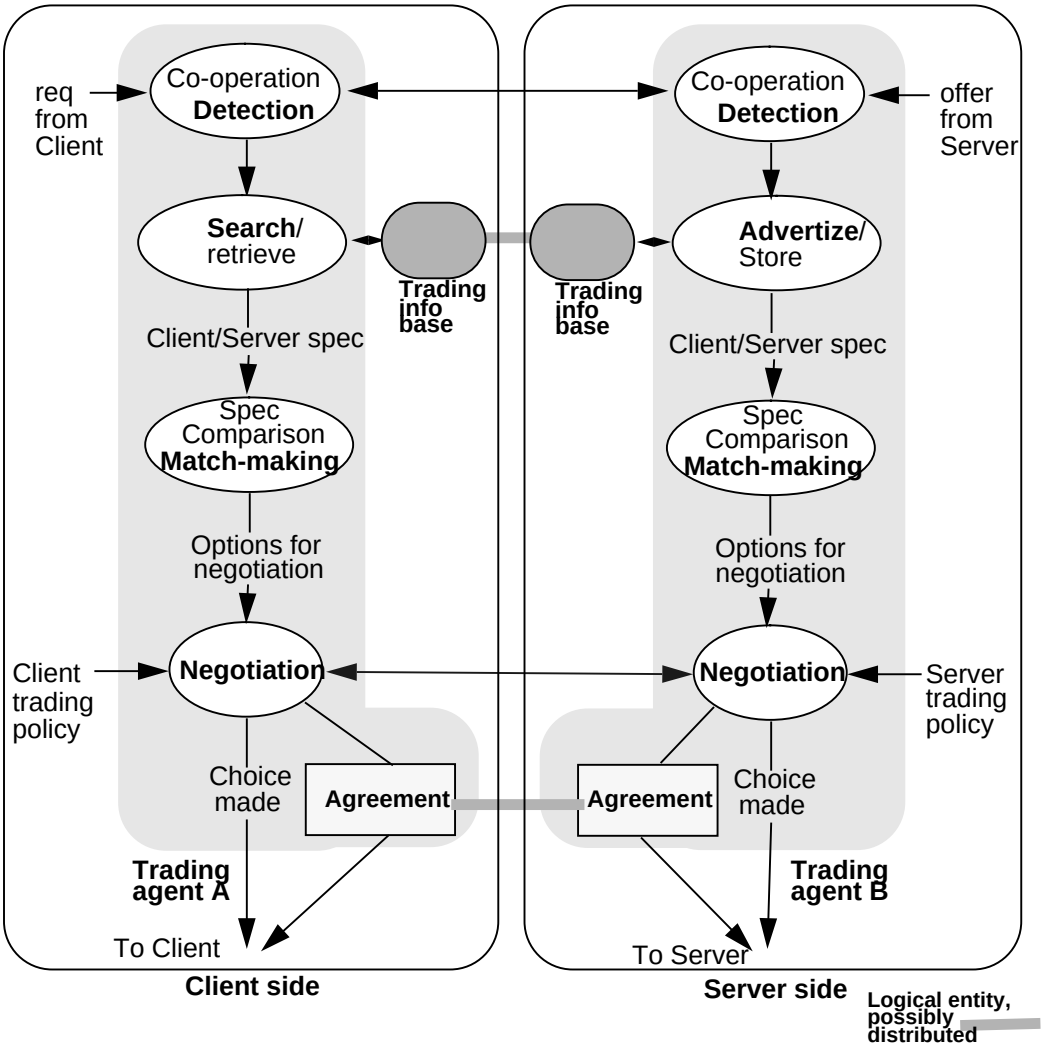
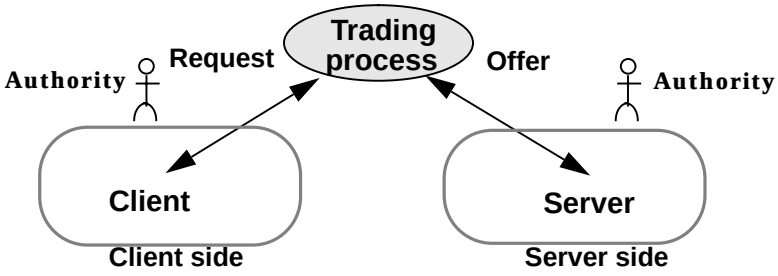


Figure 7.6: The centralized trading process



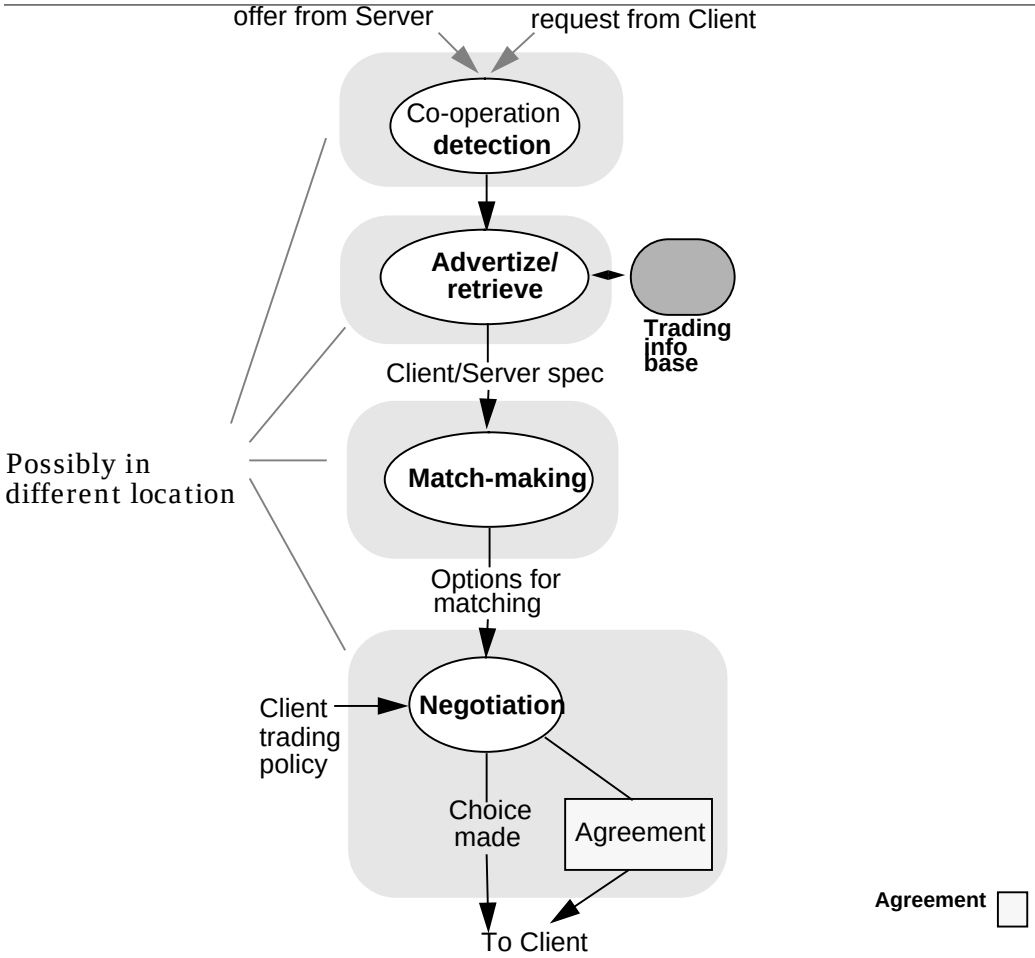
7.6 Intra-trading chain configuration (distributing the trading process)

The sub-stages of the trading process described in §5.2 can be distributed by performing them by different agents in different places and times. The intra-trading principle can be applied to the distributed or centralized model of trading. An intra-trading chain configuration process which is distributed internally is shown in Figure 7.7.

Check list

- ***Can some of your existing tools detect attempts to trade at compilation and linking stages and carry out some trading at that time?***

Figure 7.7: Intra-trading chain



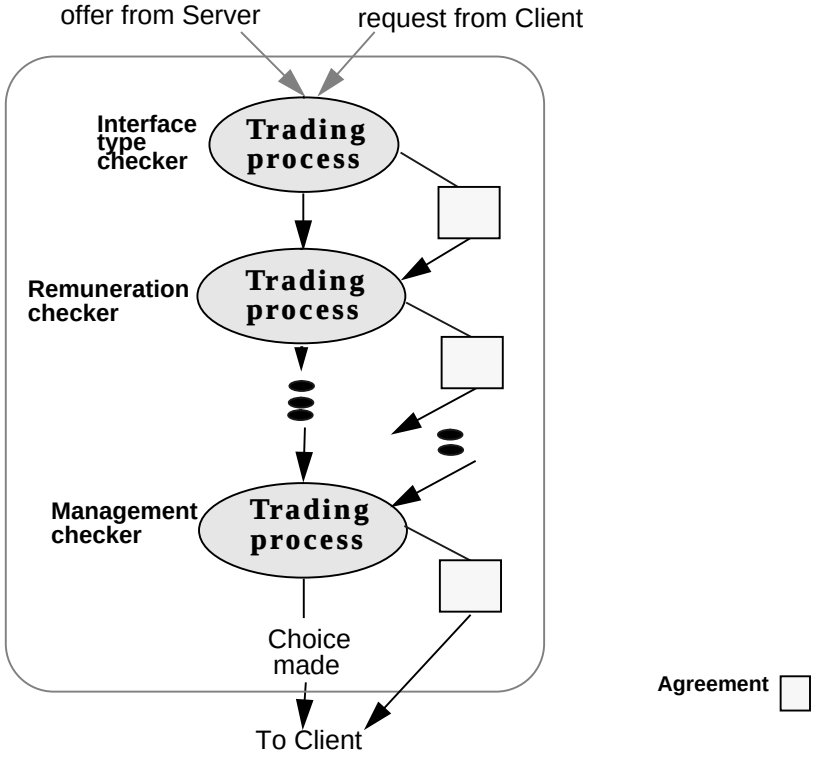
7.7 Multi-level trading configuration (distributing the trading process)

Different aspects of the trading process with respect to different information categories (as described in section 6.3 and Figure 6.3) can be teased apart and carried out by different agents at different epochs. The result is the multi-level trading configuration shown in Figure 7.8.

This configuration has the advantage that it allows the development of trading processes with varying degrees of intelligence built into them. For example, a trading process specializing in trading for remuneration can be programmed to understand different remuneration models and be able to compare them.

One of the advantages of such a configuration is that it is possible to have specialized trading processes for negotiating remuneration or authority agreements.

Figure 7.8: The Multi-level trading configuration



Check list

- ***Can you agree with prospective servers/clients on certain issues before the specific client and server attempt to trade and bind?***
- ***For example, can remuneration and management issues be resolved on a per organization rather than application basis?***

7.7.1 Confidence in information passed between the stages

As stated in §4.4, the stages of the trading process (shown in Figure 7.5) or parts thereof can be carried out by different agents at different points in the application life cycle. This raises several issues:

- how to find trusted agents throughout the development process who can give guarantees, e.g. pre-processors, compilers, traders
- in what form should these guarantees be given
- how to keep the information passed between the stages inviolate.

This problem also applies to the information passed between the trading and binding processes.

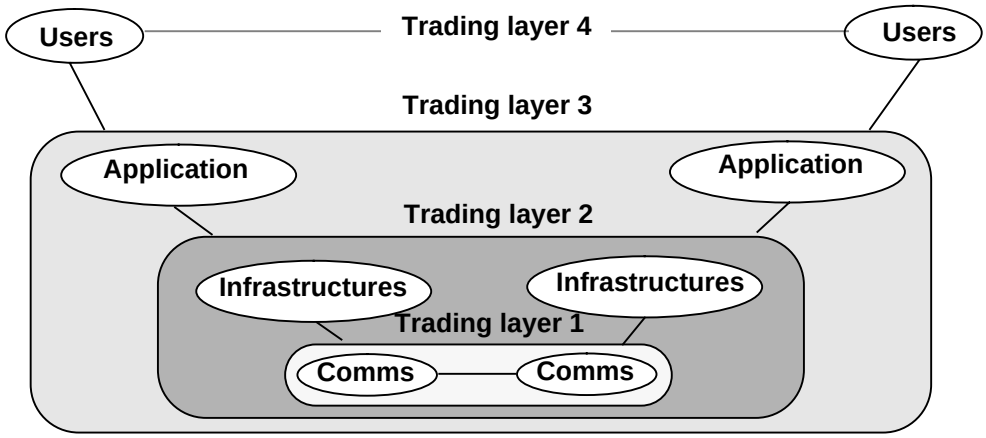
7.8 Layered trading

Many future applications using large scale distributed technology will support human activities carried out remotely. Electronic shopping, multi-media meetings and touring for example, involve people as the drivers of the technology. They therefore require match-making of humans and of their

intentions, of applications as well as of technology requirements. Many applications are therefore likely to need a layered trading process, for example:

- technology layer:
  - communication platforms establishing the basic data transfer channel compatibility
  - infrastructures establishing their compatibility
- application layer: client and server establishing the compatibility of parts of the application
- user layer: establishing the compatibility between the user and the goods/ services provided by the application.

Figure 7.9: A layered trading sequence



8 Implementation view of co-operative trading

8.1 Introduction

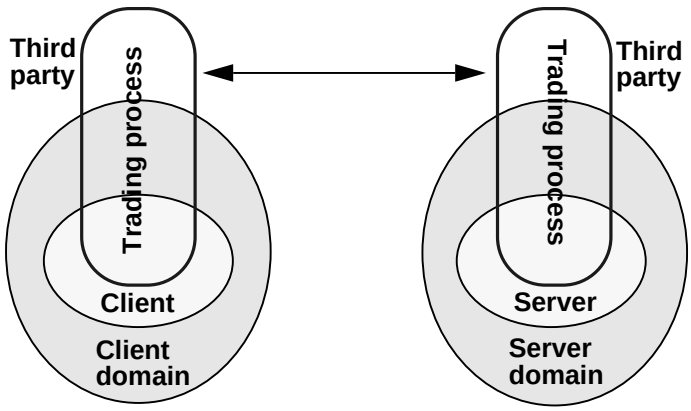
Previous chapters have shown how the trading process can be broken into different parts and how the necessary information can be distributed. This outlines a space of possible implementations of the trading process.

This chapter describes the link between enterprise type considerations and the choice of implementation from those outlined in earlier sections.

8.2 Locality and distribution of trading information and processing

The way in which the trading process can be distributed over a client (or server) and a client (or server) domains is shown in Figure 8.1. In this figure different parts of the trading process are divided between the objects requiring trading, their management domains and external third party agents.

Figure 8.1: Including or excluding parts of the trading process within the boundary of encapsulation of clients, servers and their domains



Note that in different systems the boundaries of encapsulation may change. For example the trading process can be implemented entirely inside the management domain of an object. Alternatively, more of the trading process can be externalized and may also be shared with the respective process in the other domain (Figure 8.2).

The trading processes can also be assigned to an external third party where such a trusted agent exists (Figure 8.3). Examples of such an implementation are the ANSAware Trader [ARM 93] and the Orbix based trader [APM.1177 94].

8.3 Enterprise considerations

Enterprise concerns are the driving force behind the implementation choices made. For example, where the boundary of encapsulation around the trading

Figure 8.2: Where trust exists it is possible to make the trading processes external

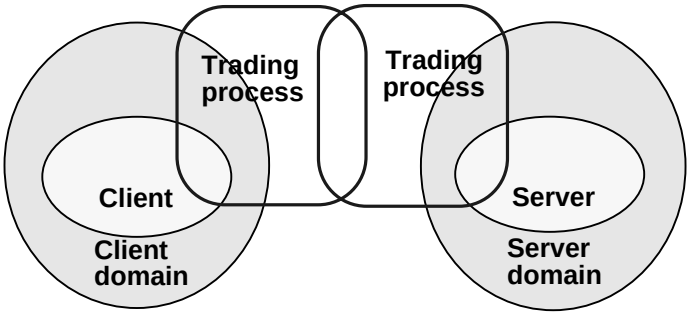
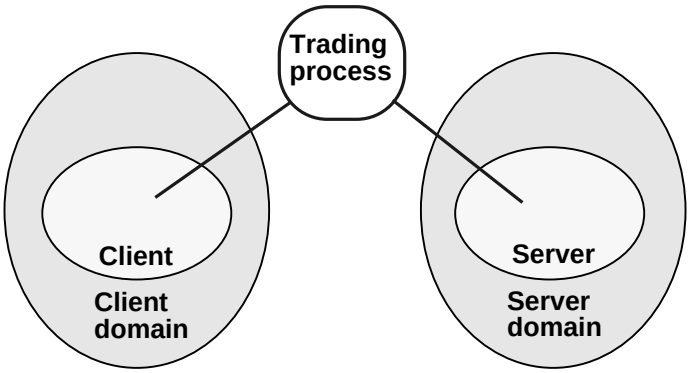


Figure 8.3: Third party trading agents are possible



process should lie, i.e. which parts of the trading process should be implemented inside or outside the client and server and their management domains, is an implementation decision which depends on issues such as:

- locality of information
- performance and scaling
- security and trust
- dependability.

8.4 Implementation strategies

Different trading implementation strategies allow putting the encapsulation boundary of trading process in different places. For example:

- in-line code
- library
- third party agents.

References

[ANSI X3.135 92]

ANS X3.135-1992, American National Standard for Information Systems - Database Language - SQL, American National Standards Institute, October 1992.

[ALMGREN 91]

Almgren, G. and Anderson, M., “Open System Trader using X500”, ISA Project, Televerket, September 1991.

[APM.1001 93]

Rees, R.T.O., “The ANSA Computational Model”, APM.1001, Architecture Projects Management, APM Ltd.,Architecture Projects Management, APM Ltd., Cambridge UK, 1993.

[APM.1003 93]

van der Linden, R., “The ANSA Naming Model”, APM.1003, APM Ltd, Cambridge UK, February 1993.

[APM.1005 93]

Deschrevel, J-P., “The ANSA Model for Trading and Federation”, APM.1005, Architecture Projects Management, APM Ltd., Cambridge UK, 1993.

[APM.1042 93]

Hoffner, Y., Beasley, M.D.R., “Type Conformance and IDLs”, APM.1042, Architecture Projects Management, APM Ltd., Cambridge UK, 1993.

[APM.1064 94]

Cameron, J., “Scenario”, APM.1093, Architecture Projects Management, Cambridge, 1993.

[APM.1091 93]

Beasley, M.D.R., “Advanced Trading”, APM.1093, Architecture Projects Management, Cambridge, 1993.

[APM.1095 94]

Gomer, T. Editor, “Micro-Scenarios for Federation”, APM.1095, Architecture Projects Management, Cambridge, 1994.

[APM.1110 94]

Beasley, M.D.R., “New Trader - Analysis and Design”, APM.1110, APM Ltd,. Cambridge UK, 1994.

[APM.1139 94]

Hoffner, Y., and Gray, C. G. “Boundaries and Domains in Distributed Systems”, APM.1139, Architecture Projects Management, APM Ltd., Cambridge UK, 1994.

[APM.1142 94]

Hoffner, Y., “A Model of Interception”, APM.1142, Architecture Projects Management, APM Ltd., Cambridge UK, 1994.

[APM.1162 94]

Thomas, C.G., Beasley, M.D.R. & Hoffner Y., “Data Management for an Enhanced Trader”, APM.1162, Architecture Projects Management, APM Ltd., Cambridge UK, 1994.

[APM.1163 94]

Thomas, C.G. & Linden, R.V.D., “Remote Database Queries in Open Distributed Systems”, APM.1163, Architecture Projects Management, APM Ltd., Cambridge UK, 1994.

[APM.1177 94]

Girling, C.G., “Characteristic Repository Specification”, APM.1177, Architecture Projects Management, APM Ltd., Cambridge UK, 1994.

[APM.1193 94]

Beasley, Cameron, Gray, Hoffner, Linden, Thomas, “Federation Manifesto”, APM.1100, Architecture Projects Management, APM Ltd., Cambridge UK, 1994

[APM.1229 94]

Cameron, E.J. & Linden, R.V.D., “Information Moedl for Federatiion”, APM.1129, Architecture Projects Management, APM Ltd., Cambridge UK, 1994.

[APM.1314 94]

Otway, D. J., “The ANSA Binding Model”, APM.1314, Architecture Projects Management, APM Ltd., Cambridge UK, 1994.

[ARM 93]

The ANSAware 4.1 manual set, Architecture Projects Management, Cambridge, 1993.

[DUDET 90]

Dudet, M., “X.500 Directory for those familiar with Trading Concepts”, ISA Project, SEPT, 42 rue des Coutures, BP 6243, 14066 Caen cedex, December 1990.

[GRIFFETH 92]

Griffeth, N.D. & Velthuijsen, H., “Negotiations in Telecommunications systems”, Report TM-ARH-021228, Bellcore, 445 South St. Morristown, NJ, USA, 1992.

[HAWKINS 79]

Hawkins, J.M., “The Oxford Paperback Dictionary”, Oxford University Press, 1979.

[HEIMBIGNER 81]

Heimbigner, D. and McLeod, D., “Federated Information Bases - A Premilinary Report”, In *Infotech State of the Art Report: Database*, Vol. 9, Pergamon Infotech Limited, Maidenhead, , 1981, pp. 383-410.

[HEIMBIGNER 85]

Heimbigner, D. and McLeod, D., “A Federated Architecture for Information Management”, ACM Transaction on Office Information Systems 3(3), pp. 253-278, (july 85).

[IONA 93a]

Orbix: Programmer’s Guide, Iona Technologies Ltd., Dublin, Republic of Ireland, 1993.

[IONA 93b]

Orbix: Advanced Programmer’s Guide, Iona Technologies Ltd., Dublin, Republic of Ireland, 1993.

[LITWIN 90]

W. Litwin, L. Mark and N. Roussopoulos, “Interoperability of Multiple Autonomous Databases”, ACM Computing Surveys, Vol. 22(3), pp. 267-293, September 1990.

[ODP 93]

“Basic Reference Model of Open Distributed Processing - Part 3: Prescriptive Model”, ISO/IEC CD 10746-3 (Committee Draft), ISO/IEC JTC1/SC21/WG7, June 1993.

[OMG 92]

“The Common Object Request Broker: Architecture and Specification”, Document Number 91.12.1, Object Management Group and X/Open, 1992.

[OSF 92]

OSF, DCE Application Development Guide, Open Software Foundation, 11 Cambridge Centre, Cambridge, MA 02142, USA, 1992.

[RC.101 90]

Linden van der, R. & Sventek, J.S., “Trading in the Five Projections”, RC.101, Architecture Projects Management, APM Ltd., Cambridge UK, 1990.

[RC.358 92]

Watson A.J., “Types and Projections”, APM/RC.358.03, APM Ltd. Cambridge UK, April 1992.

[RUMBAUGH 93]

Rumbaugh, J, et al., “Object-Oriented Modelling”, Prentice Hall International, 1991.