Poseidon House
Castle Park
Cambridge CB3 0RD
United Kingdom

TELEPHONE: **Cambridge (0223) 323010**
INTERNATIONAL: **+44 223 323010**
FAX: **+44 223 359779**
E-MAIL: **apm@ansa.co.uk**

ANSA Phase III

Remuneration in an ODP Environment

Gray Girling, Yigal Hoffner and Rob van der Linden

Abstract

The commercial exploitation of large scale computer-based systems inevitably requires support for the elicitation of payment (remuneration) from the consumers of the goods and services it provides. Unless service providers can benefit economically from the use which is made of the products they make available in an open distributed system there will be no service provision.The ability to support remuneration is thus fundemental to the creation of an electronic services market place.

The roles, responsibilities, processes that constitute remuneration must be stated before the detailed mechanisms can be placed in a distributed system to support it, are are also required in order to derive the issues to be overcome when considering the federation of different remuneration regimes.

This document presents an initial model outlining the roles, processes and information relevant to computer-based systems that incorporate a requirement for remuneration in exchange for goods and services and goes on to outline some possible consequences on the engineering infrastructure supporting the use of remunerated services including an outline proposal for the support of electronic payment.

APM.1153.00.02 **Draft** 10 May 1994

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

Distribution:
Supersedes:
Superseded by:

Remuneration in an ODP Environment

Remuneration in an ODP Environment

Gray Girling, Yigal Hoffner and Rob van der Linden

APM.1153.00.02

10 May 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	(0223) 323010
INTERNATIONAL	+44 223 323010
FAX	+44 223 359779
E-MAIL	apm@ansa.co.uk

Contents

1 Introduction

Note: This document is still a draft. A number of editorial issues have been noted in the margins.

1.1 Motivation

ANSA enables the development of large scale computer-based systems that potentially provide a very wide range of goods (primarily information) and services. In principal the quality and availability of services could benefit from a clear market structure: the high value goods and services will get paid for whereas poor value ones will tend to cease operation on financial grounds. Given an appropriately supported market place demand will select in favour of popular goods and services

The commercial exploitation of such systems inevitably requires support for eliciting payment (remuneration) from the consumers of these goods and services. Unless service providers can benefit economically from the use which is made of the products they make available in an open distributed system there will be no service provision. This document scopes what such support might entail by describing a number of models that identify relevant concepts.

Although required by commerce, remuneration (and more generally the association of a *cost* with electronic goods and services) is also useful in an environment in which arbitrary mechanisms may be called into play transparently since it provides a means to control the use of the resources involved, and potentially a means to penalise their misuse.

The aim of this paper is:

- to develop a model of remuneration to complement and extend the client-server model of service consumption and provision;
- to document some of the problems that the current approach to remuneration in distributed systems causes;
- to outline the impact of remuneration on distributed systems that support goods or services that must be paid for; and,
- to provide a proposal for an abstract service and protocol that might be used to support payment in distributed systems.

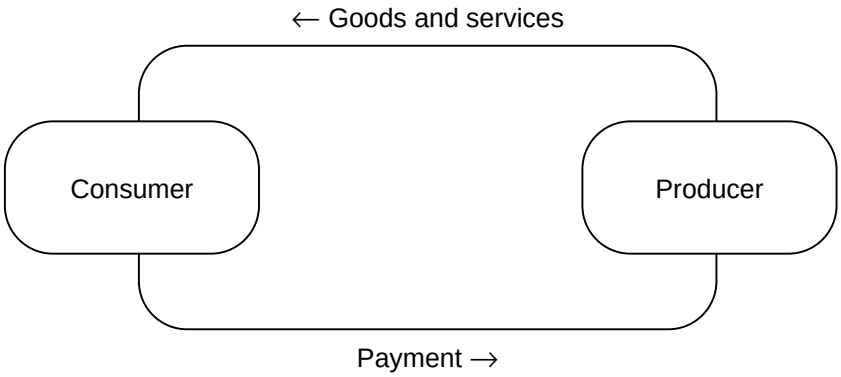
1.2 Basic model of remuneration

It is often the case that within small scale homogeneous systems such as those which serve single organizations or departments, computer goods services are offered free of any charge. This is the case with most of the present installations where personal computers are connected together, usually at a departmental or organizational level. This, however, is not the case with most goods and services provided in markets that cross organizational or national

boundaries, where some form of remuneration is expected. The difficulty of supporting remuneration electronically often means that software based products are sold as goods outside an electronic system, and rarely as services within it.

The basic remuneration model reflects the reality of the commercial world and is shown in Figure 1.1.

Figure 1.1: The basic remuneration model



Although it may be inferred from Figure 1.1 that this model applies only to the delivery of independent goods and services, it is also intended to describe the case in which the goods or service involved is the ownership of the producer (that is, to the case of the sale of the producer).

1.3 Commercial world analogues

There are problems demonstrated by the infrastructure supporting remuneration that already exists in the commercial world. Many of these are associated with issues of assurance and security. Established institutions, agreed procedures as well as trusted agents are essential in order to deal with these problems. Examples of such institutions are: shops; consumer support agencies; and banks. In addition, a considerable amount of effort is spent within the legal and executive system in an attempt to foil or recover from transgressions of legally acceptable procedures.

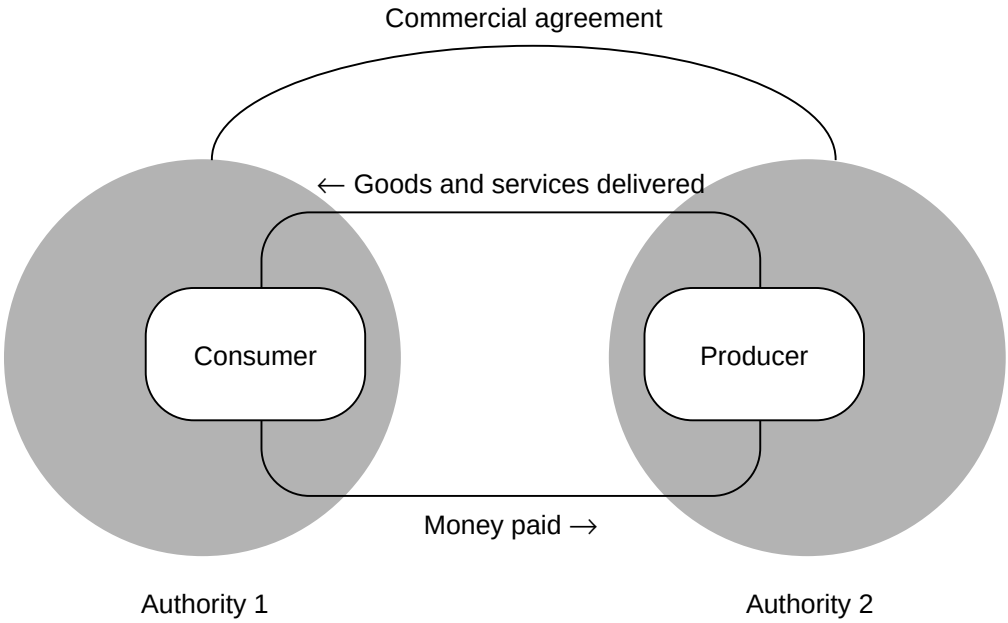
Many of the remuneration orientated problems and their solutions that are found in the commercial world have analogues in distributed systems. Some problems are exacerbated – for example, the necessity to represent bills, payment and receipts in an electronic form causes severe difficulties with regard to copying, forgery and traceability of payment. This is one reason for the strong link between a remuneration infrastructure and security.

1.4 Commercial agreements

In most client-server interactions, prior to service consumption and remuneration activities, there is an implicit or explicit process of agreeing to provide and consume a service, and to pay for it (Figure 1.2). This agreement is established between an authority responsible for the consumer and a corresponding one representing the producer. Commercial agreements can be

identified even in cases where the agreement is based on a common understanding and not an explicit agreement – for example, the implicit agreement that takes place when people buy goods in shops. Problems arise when breaches of the common understanding take place.

Figure 1.2: Commercial agreement to provide service and pay for it



A commercial agreement may represent consensus regarding:

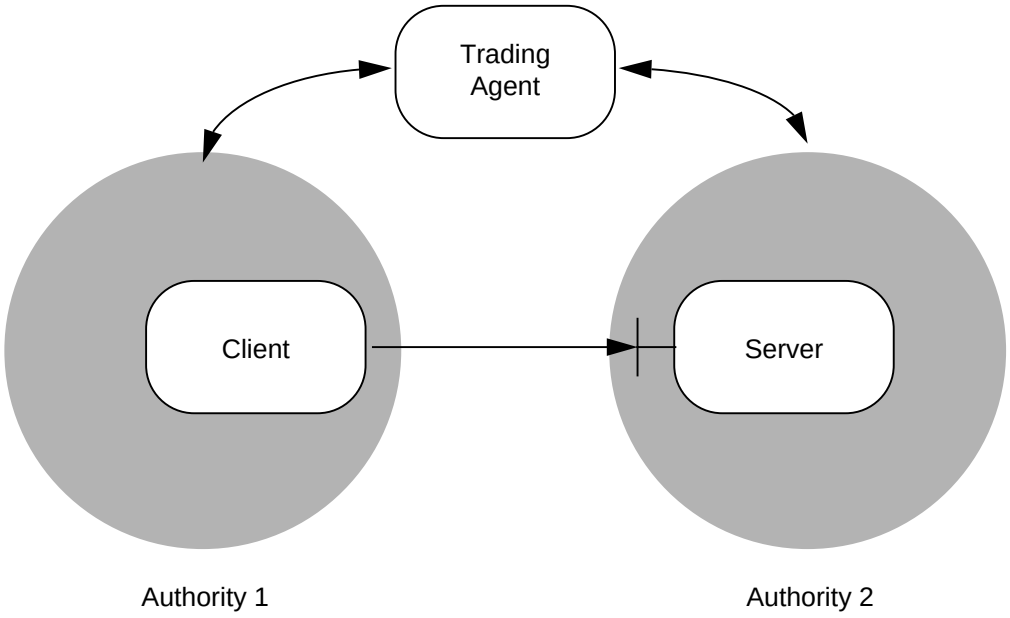
- the way in which payment relates to the consumption of goods and services (the accounting strategy);
- what mechanisms may be used to signal that payment is required (the billing strategy);
- what mechanisms may be used for the transfer of payment (the payment strategy);
- the specification and minimum quality of goods and services; and,
- what mechanisms may be used for the delivery of goods and services.

1.5 Client-server co-operation and remuneration

Remuneration should be discussed in the larger context of the process of setting up the co-operation between the client and server (Figure 1.3) as outlined in [APM 1140 94]. Information required in this process is supplied by a co-operative trading agent. The process entails:

- ensuring that the client and server match not only from the functional point of view but share a commercial agreement;
- setting up the facilities for the necessary interactions between the remuneration agencies; and,
- using the facilities.

Figure 1.3: The process of setting up the co-operation between objects of different authorities involves setting up the required remuneration structures



The remuneration model developed in this document should be applicable to:

- application specific remuneration facilities; and,
- application independent (i.e. system provided) remuneration facilities (such as those supported by a common financial infrastructure);

and continue to be relevant as the boundary between the processes of remuneration automated within a computer system and those that are carried out outside it change.

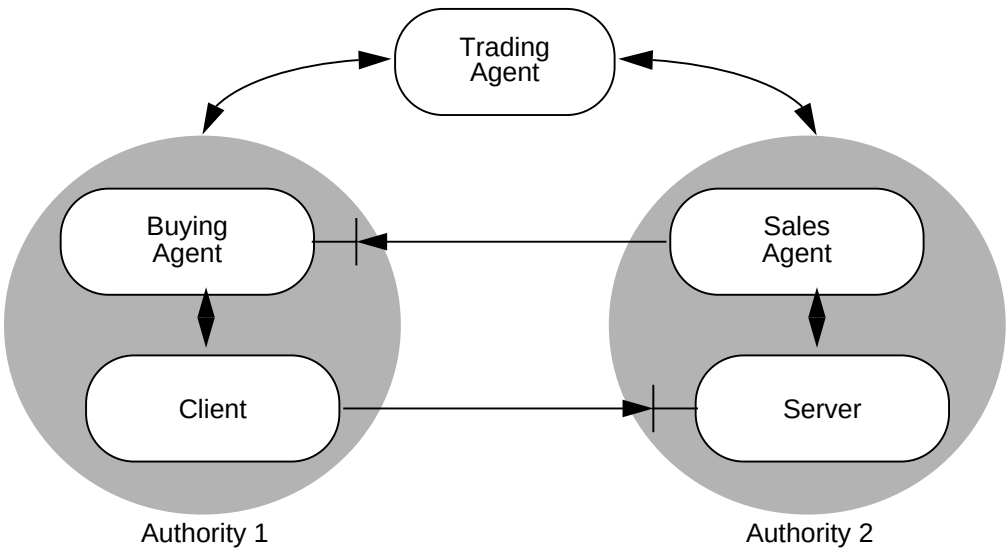
2 Remuneration

This chapter describes remuneration in terms of a responsibility, process, information and dynamic model.

2.1 The Remuneration Responsibility Model

This model is expressed in terms of a set of agents and their responsibilities in the process of establishing co-operation between a client and a server that involves remuneration. Figure 2.1 identifies the agents involved.

Figure 2.1: Agents relevant to the remuneration process and their inter-relation



Note: (Nigel) add system or application examples in this section - how about DIPS

Their responsibilities are as follows.

Authorities are responsible for:

- the fulfilment of the guarantees offered in the specification of the object;
- payment in the case of the client authority; and,
- billing in the case of the server authority.

Trading agents are responsible for:

- passing on accurately the advertised information that correctly represents clients and servers; and,
- match-making between advertised information and requested requirements.

Clients are responsible for:

- carrying out the guarantees in its specification; and,
- passing estimate of work requested from server to the buying agent.

Servers are responsible for:

- carrying out the guarantees in its specification; and,
- passing estimate of work performed for client to the sales agent.

Sales agents are responsible for:

- estimating the amount of work performed and converting it to the cost of the service;
- sending a bill to the buying agent; and,
- (if required) sending a receipt to the buying agent once payment has been received.

Billing agents are responsible for:

- checking the bill sent by the sales agent;
- paying the sales agent;
- checking any receipt returned.

2.2 The Remuneration Process Model

This model identifies the processes that agents undertake during remuneration and outlines the range of strategies that may be associated with them.

Remuneration processes are identified as relevant to two main phases:

1. enabling remuneration; and,
2. remuneration.

2.2.1 Enabling remuneration

Issues of trust.

Note: (Yigal) Get material condensed from [RFA.002]. - TBD

- agreement – establishing commercial agreement
- advertise
- match-make
- bind
- invoke service

2.2.2 Remuneration

Remuneration includes four different processes (Figure 2.2):

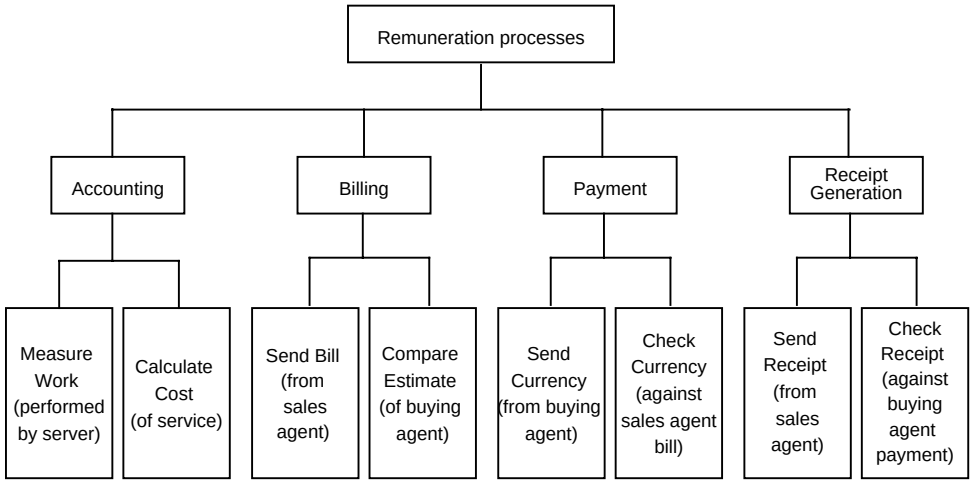
- accounting:

  to measure the amount of work carried out by a server and calculate the price of the service;

- billing:

  to charge the client for services provided;

Figure 2.2: The processes associated with remuneration



- paying:

  to transfer some currency from the client to the server; and,

- receipt generation:

  (optionally) the sending of a receipt from the server sales agent to the client buying agent.

2.2.3 Remuneration strategies

Remuneration processes are partly determined by the different strategies used for accounting, billing, payment and receipt generation. These strategies are also candidate elements of a commercial agreement.

There are different units of a service to which accounting, billing and payment may apply and which must be identified in a commercial agreement. For example the unit sold may be:

- the possession of goods;
- unlimited use of a service;
- limited use of goods or services: limited by date, number of users, number of times used etc.; or,
- the amount of work performed during an instance of service provision.

Examples of these in the current market place include:

- possession of goods (paid for)

  Products like MicroSoft Windows, Lotus 123 and FrameMaker are bought and installed on one or several machines. The number of simultaneous users is regulated by a license server and subject to a maximum.

- possession of goods (free)

  Products like Emacs (editor), MH (mail handler), LaTeX (document preparation), and gcc (compiler) are obtained from well known repositories (FTP servers) and can be used subject to very few if any restrictions.

- limited use of goods or services (licensed)

Products like ANSAware are subject to a paid for license which legally restricts the period during which the product may be used (in addition to the machines on which it may be run and the maximum number of users).

- the amount of work performed

  Telecommunication services are accounted on a basis of use. Telephone services are accounted both in terms of the duration for which the service is available for use and the extent to which it is used, with the use related part being the largest for most people. Database and electronic mail services such as CompuServe and UK Gold are accounted on the basis of usage. The same is true for many IT departments that account on the basis of CPU time or particular processing runs.

2.2.3.1 Accounting strategies

There are many accounting strategies each of which measure work differently.

The measurement of work performed by a service may be based on resources consumed in terms of storage, processing time, communication channels or bandwidth. It may be based on the rate of access to a database, or the load which already exists in a system.

The price of the work may additionally vary depending on factors that are less related to the work performed such as the current level of demand for goods or a service, the priority required, or the quality of service requested.

Accounting involves issues of trust since it determines the bill sent on behalf of the server to the client for services consumed. Both parties must trust the accounting process to measure the work carried out by the server in the way determined by the commercial agreement between the client and server authorities. Since the accounting process is often carried out by the service this is why some clients will keep a record of the requests they made of a server in order to be able to estimate the cost of using the service. Such estimates can then be compared with the bill presented by the server's sales agent.

Accounting has a strong link to monitoring [HOFFNER 93], as both attempt to observe and register some activities that take place in the system.

2.2.3.2 Billing strategies

The relationship between the amount of work performed by the server (determined by the accounting process) and the cost of the service can take a number of different forms. These can be categorized into the use of a direct functional relationship; the use of no such functional relationship (special deals), e.g. freeware; or, a combination of the two: some functional relationship with special deals.

Factors independent of the amount of work performed include establishing a set-price (e.g. none) for all work done (e.g. to a service's creator); providing a cost based on software maintenance and updates (e.g. paid up front or separately); and establishing a price based on a client's history of previous use.

A billing strategy will also determine at what stage, with respect to the time of delivery of a unit of service, payment first becomes due.

Bills may be sent inside or outside the system either in traditional or in electronic form. A billing strategy will determine which mechanisms are to be used.

The billing strategy may depend on the cost being billed.

2.2.3.3 *Payment strategies*

A payment strategy may determine the currency paid:

- cash,
- credit,
- electronic currency,
- other units of pay such as inter-departmental credit, or
- reciprocal services.

A payment strategy will also determine at what stage, with respect to the time of delivery of a unit of service, payment must be received. The main choices are:

- payment before use;
- payment at time of use; and,
- payment after use.

2.2.3.4 *Receipt generation strategies*

Since the use of receipts are optional the involvement of a receipt generation strategy can also be regarded as optional (although not to use receipts could be regarded as a receipt generation strategy).

An important component of a receipt generation strategy determines whether or not a receipt is required for payment. The information required (e.g. to identify the relevant agents, the client and server and the transaction involved) must also be specified. A requirement for assured identification of the sales agent may also be present.

Note that a receipt is a common element that a commercial proposal may specify but that, in general this process may involve a set of exchanges to conclude the sale.

2.3 The Remuneration Information Model

When reading skip this section, it is incomplete and unworked.

The implementation of a specification of the remuneration process (e.g. as determined by a commercial agreement) to be used in a particular instance of remuneration will require specification of:

- accounting, billing, payment and receipt information models –

  such a model should be the basis for developing procedures, structures and mechanisms for accounting, billing, payment and receipt generation; and,
- the setting up of these mechanisms; and,
- their management.

Note: (Yigal) Additional text needed

This model describes the information involved in the remuneration processes described in the process model:

- measurement during service provision

- expressing a measurement
- how is it measured

Note: (Gray) I don't understand the scope of this clause - why is the above here?

- representation of agreements
- representation of bills
- representation of receipts
- representation of currency

Issues:

- contracts
- trading
- measurement of work

2.4 The Remuneration Dynamic Model

The dynamic model consists of two parts: the set-up and the use of the remuneration structures.

2.4.1 Setting up remuneration co-operation

Note: (Gray) Text required

2.4.2 Using the remuneration structures

Note: (Gray) to a certain extent this reads as if it is merely a repeat of the responsibility and process models.

Note: (Nigel) this is a repeat of what has gone before

The general model of remuneration (Figure 2.3), associates an accounting process with the server that measures the use of the server by the client. A charge is calculated, often as a function of the work measured and sent as a bill to the client's buying agent. The bill is then sent to the buying agent of the client.

The buying agent then sends the payment to the sales agent in whatever currency was agreed in the commercial agreement between the authorities of the client and server.

The payment is often followed by the sending of a receipt from the sales agent to the buying agent, notifying it that the payment has been received and enabling it to demonstrate its use of currency to others. (Figure 1.3).

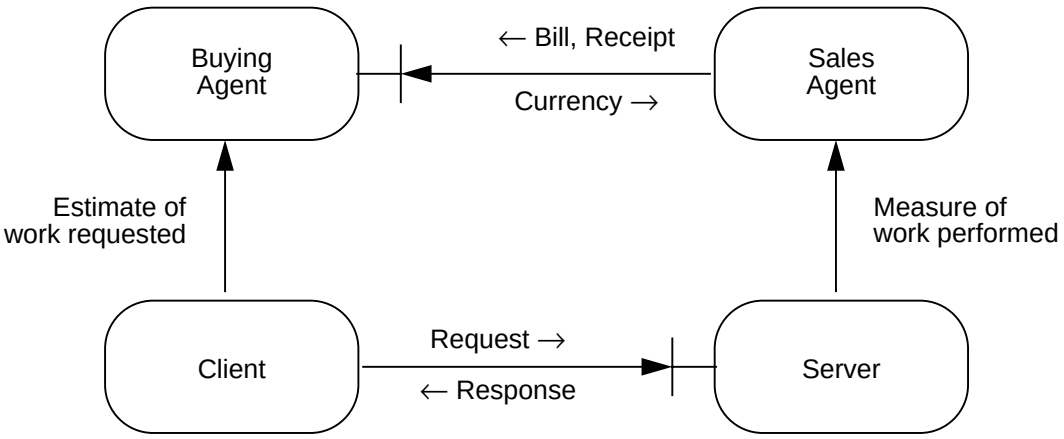
1. authority boundaries may shift as a result of the sale since the client may have new authority over purchased goods or services;
2. service can be provided under auspices of another authority.

Note: (Gray) I thought fig. 2.1 meant to imply that the server providing the service was always under the same authority as the sales agent.

Note: (Yigal) The following points need embellishment here:

1. partial order
2. transfer of ownership

Figure 2.3: A general model of client-server accounting, billing and payment.

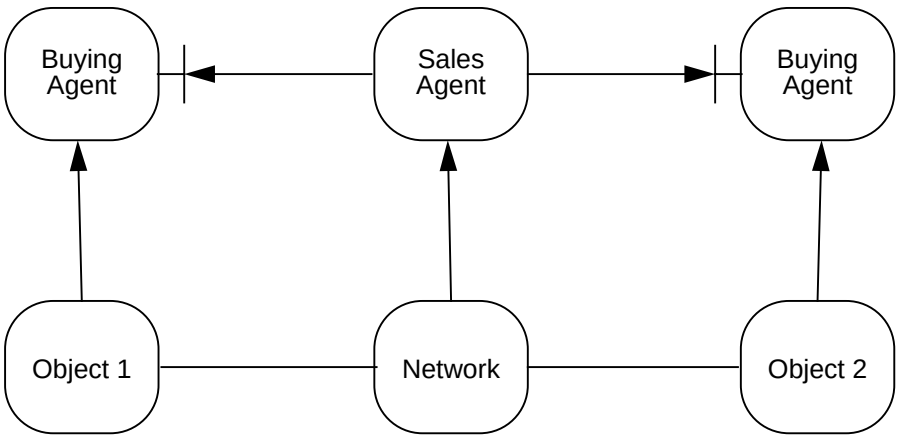


3. optimizations:
   - request and payment at same time (no bill)
   - bill with response

2.4.2.1 Remuneration and peer to peer interaction

Another aspect of remuneration is based on the observation that in a distributed system third parties providing interaction infrastructure (such as communication channels) may require payment. Thus the remuneration requirements of peer-to-peer interaction may not depend only on a client-server relationship (Figure 2.4).

Figure 2.4: A practical configuration of peer-to-peer services requiring an infrastructure support service such as a network



2.4.2.2 Remuneration problems in complex configurations (Information requirements)

There are a number of system configurations that complicate the processes involved in remuneration:

- a single client with a chain of servers (Figure 2.5);
- multiple clients with a single server (Figure 2.6);
- remuneration processes that take place outside the system and the interface between the system and people;
- the use of system remuneration facilities as opposed to application specific facilities (and the interface between them)

Note: (Gray) what is the point above?

Note: (Nigel) couldn't see the point of this (both sections above) – do you need buying and sales agents?

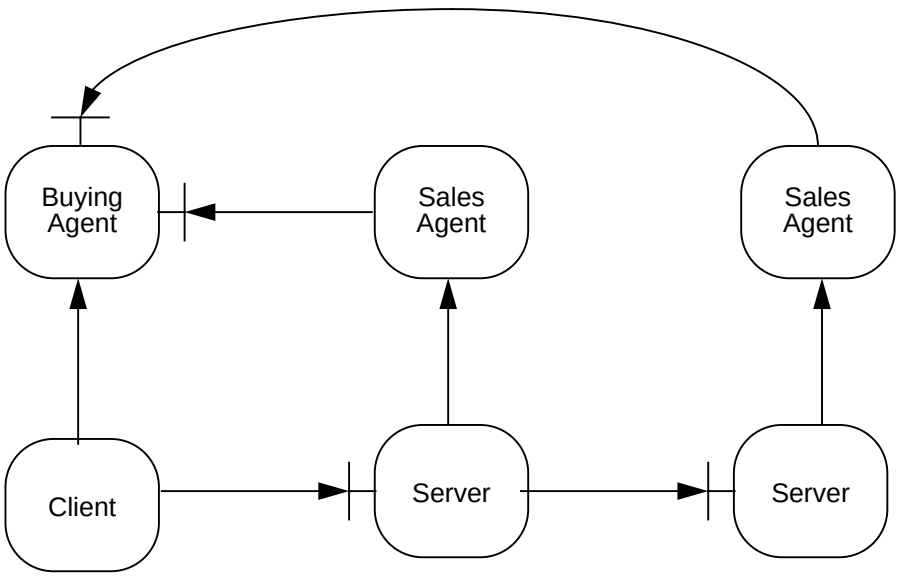
In addition to remuneration information specific to the work carried out locally on behalf of a client, information may be involved that is a consequence of distribution. In particular the configuration may be complicated by the following issues.

- Multiple objects:

  A server may be able to bill its client in either of the following ways:

  — by finding out the original client's billing destination (Figure 2.5): the server can find out who the original caller is in spite of possible multiple intermediate stages; or,

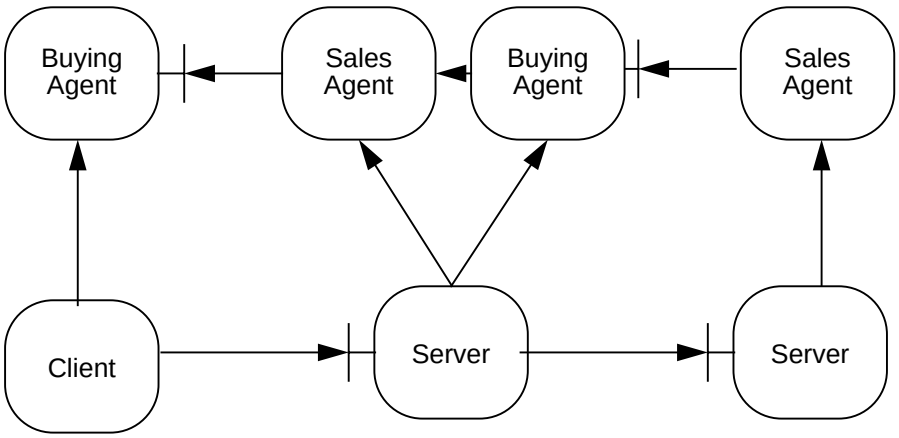
Figure 2.5: The server should be able to charge the original client



  — by billing the first intermediate stage it has contact with: the intermediate stage will pass the bill, possibly incorporating a charge for any added value, to its immediate client and so on until it reaches the initial client (Figure 2.6).

  In the former case clients require a means to authorize all the intermediate objects that may bill them whilst using a server. In the latter case the use of remote servers may be transparent to the client.

Figure 2.6: Servers may pass their bills on to their clients



Note: (Nigel) do you need tot talk about multiple levels of charging? Would there be any impact? If I contact a service provider I expect to pay only them, not the people they have subcontracted to – that would violate the ANSA encapsulation principle.

Note: (Nigel) add an example?

- Remuneration across boundaries:

Note: missing text - TBD

2.4.3 Remuneration and trading

The information necessary for the trading process which encompasses remuneration includes:

- accounting requirements

  indicating the way the cost is expected to be calculated, and thus what the cost is liable to be;

- billing requirements

  for example, how and when is the bill sent to the client;

- paying requirements

  indicating when and how is the client supposed to pay;

- receipt generation requirements

  for determining the type of receipt required (if any);

Note: (Nigel) the above is repetition of what has already been said

- agent references

  for authentication of agents; and,

- client and server references

  for peer authentication of invocations.

3 Problems of Current Remunerative Practice

Accounting on the basis of *use* fits very well with the notion of service provision by encapsulated software components. Billing can take place at all levels: where software components use other components and where components deliver service to a (human) user. Use-based accounting recognises that software (and other information products) can be copied in nanoseconds and transported at the speed of light [Cox 93]. Superdistribution [Mori 90] is based on the idea that it is impossible to control the copy process (because it is external to a software component) but trivial to control its use (because this is intrinsic). Current economic, legal and technical mechanisms prevent the copying of software, protecting the producer who bills for software on the basis of *possession*. Several technical questions, set out in this document, need to be is to become fully practical.

Distributed systems technology can act as an enabler for an environment in which:

- the software producer will be rewarded for creating reusable software components (e.g. by increased revenue and profit); and,
- the consumer will be rewarded by added value in the way software is provided (e.g. by shortened lead times and reduced costs).

In this chapter, however, we examine some of the existing problems that current practice presents.

3.1 Legal protection is required

For most commodities it takes time, specialised knowledge and tooling to produce them. Customers are happy to pay those who possess the appropriate skills and equipment to produce products they need. Software, however, can be copied without special knowledge and by almost anyone.

The shareware paradigm relies on peoples conscience to pay a sum of money for their use of products distributed as goods. Although some may disagree, this is not generally regarded as an acceptable basis on which to do business.

Legal structures are needed to safeguard the interests of the software writer (e.g. under what circumstances is the writer liable for the functioning of a copy?), the service provider (e.g. under what circumstances is it liable for lapsed quality of service) and the consumer (e.g. what constitutes evidence that a service was contracted but not delivered). This aspect is different in nature in that it requires changes in the commercial contracts that govern the way in which software goods and services are marketed. Although important, these contracts are negotiated between consumer and producer and are thus outside the scope of the ANSA Phase III programme.

3.2 Use-based and possession-based accounting pose security problems

The tradition in an important sector of the software industry is one of producing software, then packaging and selling it. When the package is sold, ownership and control over its use (subject to restrictions) is transferred to the consumer. Revenue is generated directly by the volume of sales and indirectly by the rate of innovation (e.g. an improved version of MS-Windows or a C++ compiler can generate a surge in sales).

The tradition in the telecommunications industry is one of providing services and billing for the use of those services. Revenue is generated on the basis of volume of use. Until recently, innovation played a much smaller role. Up to the mid 1980's the basic telephone functionality had not changed much and its volume of use had increased almost to the point of saturation.

The telecommunications industry is now more innovative and offers more diverse and flexible services. Flexibility is achieved by increasing the size of the software component in the service on offer. Revenue, however, is still expected from use of diverse services, not the transfer of ownership.

The increased flexibility of services places further constraints on the extent to which software producers must respond in terms of reuse and reconfiguration of their products.

Typically the value of goods are accounted on the basis of possession and the value of services are accounted on the basis of use. Software, however, can be marketed as either goods or services.

3.2.1 Transfer of possession

The possession of all goods, including software, can be transferred from one party to another. Having to posses a software component before being able to (re)use it limits the extent to which the component will actually be reused. The challenges in representing the concept of ownership by possession in an environment in which something can easily be copied and moved give rise to security problems specifically relevant to software.

3.2.2 Transfer of use

The potential to use a service can also be transferred (for example the usage card associated with some satellite television stations).

The ANSA Computational Model is based on a parameter and result passing mechanism whereby each parameter and each result represents an interface (a place of service provision). Thus the transfer of parameters and results carry with them the implication that the interface they represent can be used to obtain a service.

The implementation of this model in such a way as to prevent use of an interface that has not been explicitly delivered to the potential user has a number of security implications.

3.3 Existing forms of software protection present barriers to service re-use

Third party software cannot generally be used in a product or service without some sort of explicit legally binding commercial agreement (e.g. a licence). In business terms, such agreements are often expensive to obtain.

This is not so true in other engineering disciplines. A hardware component, such as a chip or even a complete board, can be used anywhere and for any purpose once it has been bought. If software is burned into the chips the same applies: buy it and use it anywhere and for any task[1]. The resulting value added product can normally be sold on for a profit, without the component sources requiring a share[2]. This state of affairs allows the industry to build on what others produce quickly, and to progress in terms of productivity.

The software industry has not progressed to this state of affairs, perhaps because software represents both *design* and *implementation* (a commodity). The economics of design mean that if software is delivered on a tape or disc or via file transfer services, it cannot be reused in a product or a service, unless the software component provider is involved financially. The resulting business structure frequently makes it easier to rewrite the required functionality.

Rewriting software is wasteful: the more complex the software, the more difficult and costly it is to test. Reinventing algorithms also results in a proliferation of components, many of which perform similar functions. Components which perform exactly the same function often do so through different interfaces. There is no sense of substitutability, an essential aspect of any engineering discipline.

Note: (Nigel) the above two paragraphs are about the problems of software reuse not remuneration

Software producers who want to sell or licence their products protect themselves against imitators who produce “illegal” software copies. They put in place elaborate mechanisms, both legal and electronic. Whilst these measures are often ineffective, (aptly demonstrated by widespread software piracy) they do inhibit reuse, as access to the product by new parties must first be negotiated with the producers or owners of the IPR.

3.4 Simple components are poorly marketed

Complex software, such as word processors, spreadsheet, database packages, and compilers are generally marketed as goods. Smaller or less complex software components (matrix inverters, sorting algorithms) are not treated as neither goods nor services because they tend not to be owned by someone and therefore are not marketed at all.

Large components often possess functionality which is not required. The buyer potentially pays for things which are not used.

Smaller components are harder to obtain and are often reimplemented rather than reused. Compilers for object oriented languages such as Smalltalk now

1. There may be restrictions such as US Government export restrictions.

2. When a patent applies, the patent holder receives a cut. The purchaser however does not see this.

include a library of useful components. Such libraries encourage reuse of simple software components (but not large ones).Currently, however:

- different suppliers offer different libraries, even if the same language is considered;
- finding a required function in the library can be a problem because the libraries are so large; and,
- the use of a library cannot be enforced, even within a single department or company.

4 Some Existing Remuneration Requirements

Electronic Funds Transfer (EFT) is a general term used to cover a range of remuneration scenarios each of which address different requirements and reveal different implementation issues. This chapter describes some of these scenarios and compares the requirement they address with those of an electronic marketplace.

The reader is referred to [APM 1140 94] for a more detailed exposition of these (and other) scenarios.

4.1 Inter-bank Billing

Banks and other financial institutions provide facilities for transfer of payment between accounts they hold on behalf of their customers. These include both debit transfers (as when a cheque is cashed at its payee's bank) and credit transfers (as when a previously authorized standing order to a payee's account operates).

Banks use electronic funds transfer in the sense that they exchange information indicating the value by which the customer accounts at source and destination banks are to be altered. This flow of information is often achieved through a central clearing bank (partly to avoid the alternative n-squared communications problem).

Many financial transactions are normally processed in this way at once. Each transaction describes a movement of money from one bank to another – it does not implement this movement of money. When a large number of transactions have been cleared a balancing of payment will be due from one bank to another, and this is not accomplished using the inter-bank electronic funds transfer mechanism. In that sense this type of electronic funds transfer is a billing and not a payment mechanism.

Nonetheless there are important requirements of the billing mechanism involved because the transactions exchanged will, eventually, result in real payments. Participants in the mechanism:

- authenticate each other –

  banks know the names of the other banks involved in the clearing scheme, effectively they trust the clearing house to apply each transaction to the bank named (by its sorting code) in the transaction;

- trust each other –

  knowing the names of other banks enables a bank to select which others it trusts to engage in financial transactions with, it is able to black-list banks; and,

- protect the integrity of the transactions –

  whilst in transit between two banks a transaction must not be subject to manipulation or deletion and the forgery of new transactions should be impossible.

Banks need to trust each other not to falsify transactions to their own advantage (particularly in respect of debit transfers). To a certain extent they must also trust each other to provide the relevant payment, but this is ameliorated by the practice of updating a credited account only once the payment relating to the credit has been received (thus it is the bank's customers who need to trust that banks provide due payment). This practice also means that debit transfer takes longer than credit transfer.

Because participants in this EFT scheme need to trust each other it would be difficult to scale this mechanism to cope with large number of customers and providers (simply because of the effort in keeping track of trustworthiness). Because the trust represents a requirement not to do something which is in the trusted entities interest there will be relatively few instances where this mechanism can be used safely.

This scenario is not one in which a producer and consumer exchange goods and services for a consideration.

4.2 Automatic Teller Machine Payment

An Automatic Teller Machine (ATM) enables a bank's client to retrieve cash from his account (amongst other things). The ATM engages the customer in dialogue and engages the ATM's bank in another.

The ATM authenticates the customer (normally using a Personal Identification Number (PIN)) and identifies his account, receives a request for cash, checks the customer's account for creditworthiness, dispenses the cash and instructs the bank that it has done so (which then adjusts the customer's account accordingly).

EFT is used between the ATM and its bank effectively to elicit payment for the cash received.Participants in this mechanism:

- authenticate each other –

  the ATM and the bank implicitly know each other;

- trust each other –

  the ATM trusts the bank to provide a correct credit check and to debit the customer's account correctly, the bank trusts the ATM to authenticate the customer to which it has given money; and,

- protect the integrity of the transactions –

  whilst in transit between two banks a transaction must not be subject to manipulation or deletion.

Normally an ATM is the property of the bank with which it interacts. It is implicitly trusted to handle the transactions (and indeed the cash) in which it deals. The EFT mechanism needs to provide authentication and data integrity for protection from third parties, not from the parties using the mechanism.

The same EFT scheme could be applied between an ATM and another (trusted) bank without major change. However, even then, its use in

supporting general commercial transactions is limited just as the inter-bank billing example above.

This scenario is not one in which a producer and consumer exchange goods and services for a consideration.

4.3 Point of Sale Payment

EFT can be used by an “EFTPOS” terminal at the Point Of Sale (POS) to accomplish the direct transfer of funds between a customer’s bank and the merchant’s bank. These terminals are normally installed on the merchant’s premises and process a series of transactions on its behalf in liaison with the merchant’s bank.

Initially a merchant’s representative (e.g.a shop assistant) will authenticate himself to the terminal (e.g. by inserting the correct key) which has a connection to the merchant’s bank. A customer will, in exchange for goods or services, authenticate himself to his bank (e.g. using a PIN) via the terminal and the merchant’s bank. An agreed sum of money will be entered at the terminal and the client’s credit at his bank will be ascertained by the merchant’s bank before a credit transfer is effected between the two banks for that amount. Once this is done the terminal is informed of the success of the transfer and the customer is regarded as the new owner or user of the goods or services.

EFT is used between the two banks in a similar way to that described above in inter-bank billing. In addition there are the following requirements regarding the other participants.

- Authentication –

  the merchant’s bank must identify the merchant (or his representative or his POS terminal), the merchant”s bank must identify the customer’s bank and the customer’s bank must identify the customer.

- Trust –

  the customer’s bank trusts the POS terminal to authenticate the customer on its behalf and the merchant’s bank to relay this information correctly, the merchant’s bank trusts the POS terminal to authenticate the merchant’s representative and to identify the customer’s bank. Both banks trust the POS terminal to relay the amount agreed (by the customer and merchant) for transfer (especially if the ability to grant a refund is present).

- Integrity –

  whilst in transit between a POS terminal and the merchant’s bank a transaction must not be subject to manipulation or deletion and the forgery of new transactions should be impossible (especially if refunds can be represented).

This scenario is one in which a producer and consumer exchange goods and services for a consideration but neither the consumer nor the producer are intended to be represented electronically. In effect the scenario describes a financial infrastructure that would require further detail adding in order to be applicable to an entirely electronic marketplace.

Abstracting the scenario to a three party activity (in which the POS terminal and the banks are regarded as the financial infrastructure – and the customer

and the merchant are the other two parties) there are the following requirements.

- Authentication –

  both the customer and the merchant must be authenticated to the financial infrastructure;

- Trust –

  the infrastructure is owned by the merchant, the customer must trust him to use it transfer the agreed payment; and,

- Integrity –

  the customer provides a token (card) to authenticate himself and as a reference to his account. This information must correctly be bound together and must be visibly authorized by an authority associated with the account. It must not be subject to forgery (including duplication) or to modification.

These requirements could, in principle, be satisfied by an electronic system (in which the token is represented electronically), however the EFTPOS system does not seek to do this.

4.4 Conclusion

The above scenarios are important in suggesting the kinds of requirement for assured identification, trusted function and integrity that are implicit in existing financial infrastructures. However, none of them are intended to address the support of payment in a marketplace populated entirely by electronic representatives of consumer and providers.

The simple transfer of billing information between consumer and provider using authentication and integrity protection has the disadvantage that the management required of the set of trusted partners in this mechanism means that it will not scale and that it is not applicable to the (common) situation in which the consumer and producer wish to minimize their trust in each other.

These mechanisms are intended to support transactions in officially recognized currencies. This represents a restriction (effectively dictated by national banks) on the way money is manipulated and represented in the financial infrastructure. Greater freedom and autonomy can be obtained through the use of independent currencies.

5 Engineering Implications of Remuneration

This chapter considers some of the basic engineering requirements remuneration implies in and distributed processing environment. In particular it addresses requirements of the means of interacting with remunerated services and infrastructure support for electronic currency.

5.1 Interaction with remunerated services

Remuneration, and in particular payment, confers a real value on access to goods and services in an open distributed processing environment. This value is visible to both a consumer (which must pay something) and a producer (which receives payment). In order for this value to be realized these goods and services must not be available to those that do not pay. Mechanisms are required to ensure this state of affairs and the provision of those mechanisms represent constraints on service implementation.[1]

The new problem that remuneration introduces is to ensure that payment and service are exchanged in equal measure. Variants of the problem occur depending on the payment strategy (e.g. before, during or after service) and the level of assurance required. In terms of the temporal aspect of the payment policy the following are possible solutions.

They are described below in terms of requirements to safeguard a consumer when payment is accepted from it, and requirements to safeguard a producer when goods and services are delivered. The latter is meaningful only when a client is able to verify that a requested service has been performed. Since this is not always easy to achieve in general in a processing environment some preference for payment strategies that require fewer safeguards for the consumer on acceptance of goods and services might generally be preferred.

5.1.1 Payment after service delivery (“cash on delivery”)

5.1.1.1 Acceptance of payment

At minimum some evidence that an agreement to pay for services (in a commercial agreement) must be established by a third party before service is provided.

5.1.1.2 Acceptance of goods and services

If goods and services were not provided payment is not made.

1. Note however that sometimes these constraints are so great that remuneration services are simply abandoned. Telecommunication companies have sacrificed billing information in order to obtain high availability for example.

5.1.2 Payment during service delivery

At a fine grain this case can usually be identified as payment after service delivery or payment before service delivery. The implication of the separation of this category is really that payment is synchronized with relatively small units of services delivery for which non-payment is less of an issue than would be the case for the larger quantity of service that repeated use represents.

5.1.2.1 Acceptance of payment

Access control (e.g. to individual operations) can be exercised based directly on the provision of electronic currency. This is potentially the method allowing the greatest degree of client anonymity (which may or may not be an important aspect of an implicit commercial agreement).

Alternatively if access to a service or use of goods is repeated client authentication can be used in order to “black list” commercial agreement violators and future access can be denied to them. Other methods of censure might also be available.

5.1.2.2 Acceptance of goods and services

Provision of payment can be dependent upon receipt of goods or services in an interaction.

If access to a service or use of goods is repeated server authentication can be used in order to “black list” commercial agreement violators and future uses of their service (and thus payment for their service) can be denied to them. Other methods of censure might also be available.

5.1.3 Payment before service delivery

5.1.3.1 Acceptance of payment

There are a spectrum of applicable mechanisms as follows.

1. Represent a good or a service as an object for which the concept of ownership can be enforced, for example using network capabilities to represent interface references and references to goods. For services this would require the engineering infrastructure that supports the use of interfaces (in general) to make interface use dependent on the provision of an authorized interface reference. For goods it may require services accepting goods to insist on an appropriate reference to them.
2. Trade payment for an unforgeable “ticket” (e.g. a network capability) enabling access to a service that is checked by services themselves (as opposed to their supporting infrastructure).
3. In return for payment register the client with its entitlement for service provision within the service and provide access control based on authentication of clients and their registered entitlements.

Note that the first suggestion implies that the unit of service provision must be an interface whereas the latter enables access to each operation in an interface to be controlled separately. The solutions also vary in terms of their potential support for non-repudiation (i.e. the opportunity they give for the later availability of evidence that payment was exchanged for services, that perhaps were not provided).

5.1.3.2 Acceptance of goods and services

At minimum some evidence that an agreement to provide goods and services (in a commercial agreement) must be established by a third party before service is provided.

5.1.4 Preferred payment strategy

Given the above preference the payment strategy that minimizes risks to both parties to the commercial agreement is probably payment “during” service delivery in which small units of service are paid for after delivery.

5.2 Electronic currency

The current trader provides mechanisms through which services can be selected and used. However this is not the only implication of “trading” in the vernacular. Normally something must be given in exchange for use of a service. Consideration of what a client provides for a server, as opposed to what a server provides a client, is currently highlighted by work on federation of service offers and the symmetry of the trading relationship. [APM.1140?]

In the business world services are normally (but not exclusively) traded for money which has wide currency because its recipient can, in turn, use it to obtain other services for its own benefit.

5.2.1 Financial Currency

Some fundamental aspects of any financial currency are as follows:

1. it is negotiable

   that is, its ownership can change; and,

2. it is unforgeable

   that is, it can be minted only by a specific authority and, more generally, no financial transaction, other than minting and recalling, can increase or decrease the amount of currency in circulation.

Furthermore many national currencies are required to have the following aspect:

3. it is traceable

   that is, its use in illegal financial transactions can be ascertained.

5.2.2 The Requirements

In addition to the above aspects of currency some potentially important aspects of money are:

1. the wide range of services that will accept it;
2. the existence of different authorities associated with different currencies;
3. the network of exchange rates through which money is effectively “federated”; and,
4. units of currency are anonymous with respect their ownership.

New currencies which are required in the open distributed processing environment whose units share these aspects but most importantly can:

- be exchanged for the use of (electronic) services; and,

- be exchanged (e.g. bought and sold) for “real” units of currency.

The latter requirement is a long term aim that requires a good quality technical solution to the former requirement well before it can be realized.

Exchange of a currency for the use of services would effectively associate real value with it (because a service has value). Exchange for “real” units of currency would do so even more obviously. The potential for misuse of “electronic currency” must be no greater than that associated with real money.

5.2.3 The Problems

The most obvious use of money is its exchange. Electronic currency must have some “electronic” representation in order to be exchanged. This poses one or two problems that are particularly relevant in an electronic medium.

- Copying the representation of electronic currency must not lead to the multiplication of its value.
- It must not be possible to forge new electronic currency units.
- The infrastructure required to support electronic currency should ensure a very low rate of accidental loss.
- It should not be necessary to require honesty and trustworthiness of all money holders in order to maintain its correct value.

In an open distributed processing environment money holders will include clients and servers, many supported by nodes that do not provide guarantees to whatever infrastructure is required for the support of electronic currency. Choosing to represent electronic currency simply as an encoding (however complex) of its value would not prevent the duplication of the value when such a representation was duplicated. Some unique record of units of electronic currency needs to be kept by the financial infrastructure capable of preserving the total value of the electronic currency in circulation.

If electronic currency is represented by a reference to this unique record, the unauthorized copying of the representation (i.e. a reference) will not result in a multiplication of the record (and thus will still preserve the total value of the electronic currency in circulation).

The ownership of “real” money is represented by physical possession. The main feature of real money that makes this representation feasible is its unforgeability. When a pound coin is exchanged there is no time during which both the donor and the receiver possess the same representation. This is less easy to achieve using electronic media. These are two candidate mechanisms that attempt to achieve “ownership”:

1. financial infrastructure associates an owner with units of electronic currency; or,
2. ownership is represented by possession of an electronic representation.

Requiring the financial infrastructure to associate an owner with money has the disadvantage that those who require anonymity of electronic currency ownership will be disinclined to use this type of implementation. A lesser disadvantage is that the implementation will probably require the allocation of infrastructure resources for owner references.

Possession of an electronic representation is possible when the representation can be made confidential and unforgeable (i.e. when the representation is protected from being “stolen” by other potential owners and when it can not be

duplicated simply using knowledge of its structure). Using electronic media to exchange a representation, however, will result in both donor and receiver appearing to own the money simultaneously. Nothing for example, can prevent them both attempting to "spend" the money simultaneously. Note that, should this happen, the financial infrastructure should invalidate one of the "spend" attempts if it has the above mentioned ability to preserve the total value of the electronic currency it is responsible for. However the threat of a client spending the money that a server has just been paid may be sufficient to disincline the server from using the financial infrastructure. A means of making money confidential again, having been shared, is required.

Because it is not possible, in general, to remove an electronic representation from a client or server without its permission (or possibly without relying on locally trusted infrastructure) "making money confidential again" might better be achieved by invalidating the previously shared representation and creating a new one to be kept by the new owner. This has the advantage of requiring no particular support from the donor, and thus supporting its autonomy. This revocation of old representations is thus a fundamental feature that should be required of electronic currency.

The "keep" operation proposed here would revoke an old representation and generate a new one for the same value to be held by the "keep" invoker.

The transfer of money from a client to server would thus involve the sharing of the money reference representation, then the "keeping" of the money by the server. It would be necessary for the server somehow to check that the kept representation has the value expected before acknowledging receipt of the money (in case the client had spent the money after initially sharing it and before the server kept it).

Such a transfer would be an example of a means to implement a "use once" representation.

5.2.4 Financial infrastructure functions

The above effectively proposes the implementation of units of electronic currency by using revocable, certified, confidential and unforgeable references to objects representing financial values in which each object supports (in abstract) at least the operations:

```
share(moneyrep, sharingobject)

newmoneyrep = keep(oldmoneyrep)

value = valueof(moneyrep)
```

Furthermore the following two operations are proposed to support the management of the value of money:

```
newmoneyrep = accumulate(oldmoneyrep1, oldmoneyrep2)

newvaluerep,newchangerep = splitout(oldmoneyrep1, value)
```

The first, `accumulate`, revokes two financial representations and returns a new representation for a reference to a financial value of the sum of those revoked. The second, `splitout`, revokes one representation and returns two new ones, one for the value requested and another for "the change".

The design choice implied here is to model as objects, and thus refer to, variable amounts of money, as opposed to having a fixed number of fixed value units of money (as most “real” currencies do). The reasoning is that the resulting additional convenience is likely to outweigh the difficulty in implementing these two abstract primitives. This should, however, can only be validated by implementation.

5.2.5 Currency transfer protocol

The following summarizes the protocol proposed for the transfer of an electronic currency of value V from a donor object C to a recipient object S.

Initially C must possess a representation, R(W), for currency of value W greater than V.

1. C uses splitout(R(W), V) to obtain a representation for the currency it wishes to spend, $R_1(V)$;
2. C then uses share($R_1(V)$,S) in order that S may also possess $R_1(V)$;
3. S then uses keep($R_1(V)$) to revoke $R_1(V)$ and create $R_2(V)$ which it keeps;
4. S may then use valueof($R_2(V)$) to verify that it has currency of value V;
5. S must then indicate to C whether or not it has accepted payment (e.g. with a receipt), and if not should then return currency in a separate transfer.

If C receives no acceptance from S it may abandon the transfer and itself perform keep($R_1(V)$).

If the transfer of currency to S is a pre-requisite to its provision of goods or services and its does not accept payment S should not provide the goods or services.

If, for any reason, S neither retrieves the currency using keep nor has currency of equivalent value returned with a negative acceptance and C does not provide the expected good or service this is grounds for reference of the event to an external authority. This authority may require the transfer to be protected using a non-repudiation service in order that it has access to third party evidence of the (inevitably) disputed transaction.

5.2.6 More advanced financial instruments

It should be noted that more advanced financial instruments could be based on this infrastructure. For those who are less concerned with anonymity “bank” services can provide account objects with a variety of added value services, including electronic credit card/cheque books that effectively hide the transfer of electronic currency from account to account. However the use of electronic currency in the implementation of these services would preserve a separation of the relatively sensitive and simple financial infrastructure from the implementation of complex (but possibly less sensitive) bank services.

There are issues in the implementation of electronic credit cards and cheque books that need careful design. In particular “use once” mechanisms may still be required (it may be possible to bind them to specific unique transactions, for example).

5.2.7 Federation

As elsewhere in an open distributed processing environment, electronic currency must be capable of implementation in separate federated domains, in a fashion analogous to the currencies of different countries. The authority guaranteeing a financial representation will be different in different financial domains and the mechanisms used to support the revocation, certifiability, confidentiality and unforgeability of the representations may also be different.

The number of domains could perhaps be expected to be initially very large (e.g. one per service or one per bank) but the convenience of a common representation and a common authority can be expected to decrease their number over time.

Highly trusted interceptors between different financial domains could be built, each interceptor capable of revoking and minting new money in two or more domains and managed using a system of exchange rates. Alternatively the interceptors could simply trade in the different currencies, transferring funds that they possess in one currency in exchange for funds in the other (again according to its own exchange rates).

If federation mechanisms are well integrated into distributed processing platforms the use of these interceptors will pose a relatively small barrier to trade. However, these interceptors could also be delegated certain trade-preventing functions by different financial authorities.

5.2.8 Some Implementation Notes

Note: these are distinctly first thoughts, some improvement of these implementation details is likely to be possible. It is anticipated that more than one set of implementations may be necessary to meet different detailed requirements.

5.2.8.1 Revocation

In effect a set of current references needs to be maintained in each financial domain. Replication could be used to increase the longevity and reliability of this information, but this would increase the time before revocation can be guaranteed to have taken effect. The information could also be conveniently decentralized by the use of disjoint subsets of the reference name spaces.

Depending on other design choices it may not be necessary to maintain any other information local to the financial infrastructure. The main design choice that implies this is the inclusion of the value of the money along with the reference. This, however, would additionally require the representation to be integrity protected (in order to prevent the alteration of the value).

Only certification authorities must be able to perform this function.

5.2.8.2 Certifiability, unforgeability and integrity

This could be achieved by asymmetric (public key) encipherment of the reference and value using a private key associated with the certifying authority. The representation would consist of the result of this encipherment and the value and reference. This would enable any user of the representation to decipher the certificate with the corresponding public key and check the result against the unenciphered information. Such a check would not succeed if the representation were altered (i.e. if integrity were violated) and the fact that the decipherment key corresponds to an enciphering key only the certification authority possesses indicates that it must have been produced with the knowledge of that authority (i.e. it has not been forged).

5.2.8.3 *Confidentiality*

This could be achieved by enforcing the engineering encapsulation of objects, protecting any state associated with the money representations using access controls and confidentiality protecting the communications in which the representations are used using encipherment.

5.2.8.4 *Implementation of financial infrastructure functions*

The `share`, `keep`, `valueof`, `accumulate` and `splitout` operations are expressed here as operations provided by the "financial infrastructure" and the electronic currency objects need not map on to basic engineering objects. The money object references can occupy any (sufficiently large) name space and need not correspond to interface references. Naturally it is also possible to express these operations as operations on an electronic currency computational object, although that type of specification might be less appropriate for a literal implementation.

```
Operation share(moneyrep, sharingobject)
```

This is achieved simply by sending `moneyref` to the object quoted.

```
Operation newmoneyrep = keep(oldmoneyrep)
```

This needs to be provided by (a representative of) the certification authority. It could be achieved by extracting the reference and value from the `oldmoneyrep`, removing the reference from the set of current references, allocating a new one and producing a certificate for the new money representation with the same value but new reference.

```
Operation value = valueof(moneyrep)
```

If the representation includes the value and reference associated with it "in clear" (unenciphered by a certification authority). The representation can be checked by deciphering it and comparing the values. A check that the financial object reference has not been revoked is then required. If no error is present the value can be returned.

```
Operation newmoneyrep = accumulate(oldmoneyrep1, oldmoneyrep2)

Operation newvaluerep,newchangerep =
                        splitout(oldmoneyrep1, value)
```

These two operations are provided in a similar way to `keep`, except the new representations are constructed with values derived from the primitive arguments.

This needs to be provided by (a representative of) the certification authority.

5.2.9 Conclusion

The provision of a financial infrastructure is a logical requirement in the support of trading in a wide sense. Some requirements are proposed together with some design considerations which emphasize minimizing state that the infrastructure needs to keep (but unfortunately does not eliminate it) whilst maintaining anonymity of financial ownership. Many of the requirements for such a system depend on various security services. Some initial thoughts about an implementation are proposed.

References

[APM 1140 94]

Hoffner Y, *A Model of Co-Operative Trading*, APM Ltd, Cambridge, UK, April 1994

[DaviesPrice 89]

Davies D W and Price W L, "*Security for Computer Networks*" chapter 10: "Electronic fundes transfer and the intelligent token", John Wiley & Sons, 1989

[Sloman 88]

Sloman M, A Framework for Distributed Systems Management, Eds. Barton, H.M., Dagless, E.L. Reijns, G.L., IFIP conference 1988, Elsevir Science Publishers B.V. 1988.

[Warner 93a]

Warner M. Computer Science Department, Cambridge University, Cambridge, UK.

[Warner 93b]

Warner M. Computer Science Department, Cambridge University, Cambridge, UK.

[Warner 93c]

Warner M. Computer Science Department, Cambridge University, Cambridge, UK.

[Cox 93]

Brad Cox on Developing Software for Large-Scale Reuse, OOPSLA 93, p142-143

[Mori 90]

R. Mori, M. Kawahara, "Superdistribution: The Concept and the Architecture", Transactions of the IEICE; Vol. E-73#7 July 1990; Special Issue on Cryptography and Information Security.